COMPUTER ANIMATION AND VIRTUAL WORLDS
Comp. Anim. Virtual Worlds 2015; **26**:423–432
Published online 29 April 2015 in Wiley Online Library (wileyonlinelibrary.com). DOI: 10.1002/cav.1661

SPECIAL ISSUE PAPER

Interactive tree modeling and deformation with collision detection and avoidance

Yong-Joon Kim[1], Jong-Hwa Woo[2], Myung-Soo Kim[2]* and Gershon Elber[1]

[1] Computer Science Department, Technion, Haifa, Israel
[2] School of Computer Science and Engineering, College of Engineering, Seoul National University, Seoul, Korea

ABSTRACT

We present an interactive tree modeling and deformation system that supports an efficient collision detection and avoidance using a bounding volume hierarchy of sweep surfaces. Starting with conventional tree models (given as meshes), we convert them into sweep surfaces and deform their branches interactively while detecting and avoiding collisions with many other branches. Multiple tree models (sharing the same topology) can be generated with great ease using this sweep-based approach, and they can serve as a basis for the generation of a multiparameter family of trees. We demonstrate the effectiveness of our approach in an automatic generation of similar trees, the colonization of trees to form a forest, and the tree growth, aging, and withering simulations. Copyright © 2015 John Wiley & Sons, Ltd.

KEYWORDS
interactive tree modeling and deformation; collision detection and avoidance; sweep surfaces; bounding volume hierarchy (BVH); convex combination; growth simulation; aging; withering

***Correspondence**
Myung-Soo Kim, School of Computer Science and Engineering, College of Engineering, Seoul National University, Seoul 151-744, Korea.
E-mail: mskim@snu.ac.kr

1. INTRODUCTION

Collision-based plant modeling is a powerful scheme for the synthesis of static scenes as demonstrated in the seminal work of Fowler *et al.* [1] (for a realistic modeling of botanical garden). Nevertheless, it has been a great challenge to extend the collision-based approach to dynamic scenes including a large number of trees interacting with each other. It is only a recent development that one can handle nontrivial interference among nearby trees and obstacles such as buildings and other environments [2–6]. There are new research directions one can further explore to extend these results. An important issue is the development of geometric tools that can assist collision-based methods in the dynamic simulation and interactive design of plant models. In this paper, we present an efficient sweep-based algorithm for the detection and avoidance of nontrivial intersections among trees and their branches.

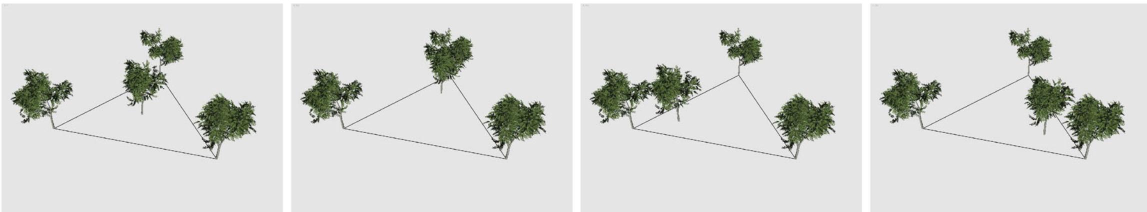
Image-based methods for tree modeling developed handy tools for the reconstruction of real-world trees as three-dimensional (3D) models and for the synthesis of similar trees possibly with some modifications. In practice, it is often needed to generate a large number of similar trees from a small set of template 3D models. Although there are many possible ways of automating the generation of similar trees by randomly perturbing their design/shape parameters, the main difficulty is in the automatic detection and avoidance of collisions among the tree branches interacting with each other in a highly complex manner.

From the shape design point of view, a direct control of tree shape must be included in the design toolbox. The inherent recursive nature of trees and their branches/leaves, however, immediately impose a serious computational bottleneck to the detection and avoidance of self-intersections that may occur in the controlled deformation of various components of the tree structures. It is thus very important to develop efficient algorithms and data structures for resolving this nontrivial problem.

Assuming the skeleton representation of trees, which can be extracted from their mesh models using conventional techniques [7,8], we attack the collision detection problem for a large number of tree branches under deformation, where the tube-like branches interact with each other in a highly complex manner. Essential to the development of such an algorithm is the design of an optimal spatial data structure for the tube-like branches represented as B-spline sweep surfaces. We generate a dynamic bounding volume hierarchy (BVH) for trees in a way similar to

(a) (b) (c) (d)

Figure 1. Collision detection and avoidance in an interactive editing of sweep-based three-dimensional tree models: (a) and (b) interactive editing of sweep-based three-dimensional tree models, and (c) and (d) collision detection and avoidance.



(a) (b) (c) (d)

Figure 2. Generating a multiparameter family of trees from a set of template tree models with the same topology. The self-intersections in the intermediate trees are automatically detected and resolved while the user drags the design tree around in the triangular design space: (a)–(d) four different convex combinations of the three template three-dimensional tree models and the resulting trees complete with their leaves.

the BVH generation for space curves. For an efficient generation of the dynamic BVH, we employ Filip *et al.* [9] for an efficient estimation of the error bounds in the approximation of space curves by polylines, which requires only the second derivative information from the skeleton space curves. Computationally speaking, we are only dealing with curve networks instead of the interference among 3D volumes.

Based on the collision detection and avoidance tools, we have developed an interactive freeform tree editing system that can enforce the 3D tree models to satisfy certain geometric constraints including nonpenetration to other tree branches and the maintenance of a certain separation distance from interacting branches. Figure 1 shows some snapshots from an interactive editing of a 3D tree model where collisions are automatically avoided by pushing the other branches away from the detected collisions. In Figure 1(a), tree branches are represented as B-spline sweep surfaces. Figure 1(b) is a snapshot from an interactive editing session where the branches colliding with other branches are shown in red. In the close-up view of Figure 1(c), one can notice that the branches in red are intersecting with each other. The collisions are automatically avoided in Figure 1(d).

Starting with a small set of trees with the same topology, we can combine them (serving as a basis) component-wise (using a convex combination of the corresponding branches) and generate a multiparameter family of trees with the same topology. In this process, there may occur many intricate intersections among different branches under continuous deformation, which can be avoided automatically using the same geometric tools that we propose in this work. Figure 2 shows four snapshots from an interactive design session for a family of trees generated from a set of three template trees shown at the three corners of a triangle representing the design space.

2. RELATED WORK

There are many previous results on tree modeling [10–12], including variants of L-systems [2,4]. Comprehensive surveys on conventional approaches are very well documented in recent publications [5,6,13]. Thus, in this section, we focus mainly on the previous results directly related to the interference of trees and their branches, which is the main subject of the current work.

Fowler *et al.* [1] demonstrated the effectiveness of collision-based plant modeling by synthesizing convincing examples of botanical garden. The result is quite impressive even though the collisions tested there are extremely simple. The collision-based approach can be extended to more general types of plant models including those for dynamic environments. Nevertheless, the main computational bottleneck has been in the development of efficient and reliable algorithms for collision detection and avoidance among 3D tree models with highly complex branch structures. In this paper, we address this important issue by employing techniques from collision detection and distance computation [14,15] and sweep surface modeling [16].

DOI: 10.1002/cav

Interference among competing branches for resources such as light and space has been handled somewhat indirectly based on growth simulations [3,17]. To accelerate the interference handling process, Pirk *et al.* [5] proposed a completely different approach where one can start with fully grown tree models and make them self-adjust depending on various conditions of light distribution, proximity, and interference. Our work is motivated by the view of plastic tree [5] and proposes a purely geometric tool that can be effectively used in handling geometric aspects of the plastic tree model. Although originally motivated by the collision-based and interference handling approaches of plant modeling, our work may also be employed as a post-processor to enrich the reconstruction results from image-based tree modeling techniques [18,19] and even the growth simulation of trees [6].

The performance of collision detection is heavily dependent on the type of bounding volume(s) employed, in particular, when the objects change shapes dynamically [20]. For polygonal meshes under deformation, spheres and axis-aligned bounding boxes are among the most popular ones in recent algorithms. For a dynamic BVH generation of Non Uniform Rational B-Spline (NURBS) surfaces under deformation, Krishnamurthy *et al.* [21] employed the condition of Filip *et al.* [9] for an efficient bounding of the maximum error in the approximation of NURBS surfaces using triangles.

Our collision detection algorithm for trees under deformation is mainly based on the distance computation. This is because we need to keep interfering branches to a certain separation distance away from each other. For distance computation, Larsen *et al.* [15] proposed the line swept sphere (LSS) and the rectangle swept sphere as better choices than the oriented bounding box [14]. For sweep surfaces, the LSS volume is the most suitable type for distance computation.

Hyun *et al.* [22] and Lee and Kim [23] employed sweep surfaces as a useful tool for human body and hand deformation. Using a dynamic BVH similar to ours, a recent work of Park *et al.* [7] demonstrated a sweep-based real-time collision detection and avoidance system for human blood vessels under deformation. Nevertheless, collisions in human anatomy applications occur mainly in joint areas such as knees and elbows. On the other hand, collisions observed in interactive tree modeling applications are considerably more complex, a resolution of which is a far more challenging research problem.

3. SWEEP-BASED BOUNDING VOLUME HIERARCHY FOR TREES

Trees are usually represented as generalized cylinders or sweep surfaces. The sweep surface representation of tree branches greatly simplifies the construction of their BVH as it is essentially reduced to the construction of BVH for their skeleton curves.

3.1. Sweep Surface Representation

Sweep surface is generated by moving a cross-sectional curve $C(u)$ along a trajectory $T(t)$. For example, given a cross-sectional curve $C(u)$ and a trajectory $T(t)$ and a time-varying linear transformation $L(t)$, we can define a bivariate sweep surface $S(u,t)$, for $0 \leq u, t \leq 1$, as follows:

$$S(u,t) = L(t) \cdot C(u) + T(t),$$

where the linear transformation $L(t)$ usually represents the rotation and scaling of a cross-sectional curve.

Tree branches are represented as sweep surfaces by interpolating the position, orientation, and radius of each branch and using the circle of radius $r(t)$ as a cross-sectional curve.

3.2. Bounding Volume Generation

Tree branch can be bounded by a hierarchy of LSS volumes [15]. As the distance computation for LSS volumes can be reduced to that for their skeleton line segments, we can greatly accelerate the collision detection and avoidance for the tree branches using the LSS tree.

Let $l(t)$ denote the line segment connecting the endpoints of the trajectory curve $T(t)$. Assume that the maximum difference for $\|T(t) - l(t)\|$ can be bounded by a certain value $\bar{d}$

$$\max \|T(t) - l(t)\| \leq \bar{d},$$

and the radius $r(t)$ of the cross-sectional circle $C(u)$ is also bounded by $\bar{r}$: $\max \|r(t)\| \leq \bar{r}$. Then the sweep surface $S(u,t)$ is completely contained in the LSS volume that is obtained by sweeping a ball of radius $\bar{r} + \bar{d}$ along the line segment $l(t)$.

We can repeat the same procedure by recursively subdividing the trajectory curve $T(t)$ and the radius function $r(t)$ at the mid-parameter value of t and then bounding the two half sweep surfaces by their respective LSS volumes. The result is an LSS tree [15], using which we can greatly accelerate the distance computation between two sweep surfaces.

When the radius function $r(t)$ is given as a spline function $r(t) = \sum r_i B_i^n(t)$, an upper bound $\bar{r}$ can be taken simply as

$$\bar{r} = \max |r_i|,$$

as we have

$$\begin{aligned} |r(t)| &= \left|\sum r_i B_i^n(t)\right| \\ &\leq \sum |r_i| B_i^n(t) \leq \bar{r} \sum B_i^n(t) = \bar{r}. \end{aligned} \tag{1}$$

Using the condition of Filip *et al.* [9], we can efficiently bound the maximum error between the trajectory curve $T(t)$, for $0 \leq t \leq 1$, and the polyline approximation

$l(t)$ that connects $(N+1)$ uniform samples $T(i/N)$, for $i = 0, \dots, N$, as follows:

$$\max ||T(t) - l(t)|| \le \frac{1}{8N^2} \max ||T''(t)||.$$

This means that an upper bound can be estimated in an a priori fashion even without computing the sample points $T(i/N)$. The second derivative information for the term, $\max ||T''(t)||$, is computed only once for the whole trajectory curve $T(t)$, for $0 \le t \le 1$. Then we can bound the maximum error in the polyline approximation. Although the upper bound is not quite tight, we can greatly accelerate the LSS tree construction using this simple approach.

3.3. Convex Combination of Trees

Several trajectory curves $T_k(t) = \sum_i \mathbf{b}_i^k B_i^n(t)$ sharing the same knot sequence can be added in a convex combination to generate a multiparameter family of similar curves defined on the same knot sequence

$$T(t) = \sum_k w_k T_k(t),$$

where $0 \le w_k \le 1$ and $\sum w_k = 1$. Each control point of $T(t)$ is given by the same convex combination of the corresponding control points for $T_k(t)$

$$\sum_k w_k T_k(t) = \sum_k w_k \left(\sum_i \mathbf{b}_i^k B_i^n(t) \right) = \sum_i \left(\sum_k w_k \mathbf{b}_i^k \right) B_i^n(t).$$

Similarly, we can add several radius functions $r_k(t)$ in a convex combination.

The convex combination of rotations $R_k(t)$ is tricky – there is no guarantee that a component-wise convex combination of rotation matrices will produce a rotation matrix. Thus, in this work, we take an approach based on the rotation minimizing frame [24], which determines the rotations $R_k(t)$ from the translational curves $T_k(t)$.

3.4. Sweep-based Leaf Modeling

In conventional methods, tree leaves are usually texture mapped to triangles. To improve the shape flexibility of leaves, we map the textures of leaves to translational surfaces (Figure 3). As a special type of B-spline sweep surfaces, a translational surface is defined by two space curves [25], where one space curve $C(u)$ is translated along the other curve $D(v)$ to generate a bivariate surface $S(u, v) = C(u) + D(v)$.

The tessellation of a translational surface is quite straightforward: (i) tessellate the curve $C(u_i)$, $i = 0, \dots, m$, as a polyline with m edges, and (ii) translate the polyline by

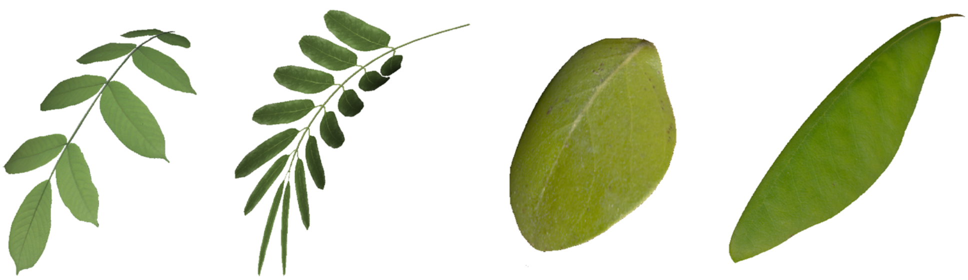


Figure 3. Texture mapping of leaves on translational sweep surfaces.

Figure 4. Withering effects of autumn leaves using translational sweep surfaces.

$D(v_j)$, $j = 0, \ldots, n$. The result will be n quad strips, each with m quads.

Using translational surfaces, it is straightforward to simulate the withering effects for autumn leaves by simply bending the two component curves $C(u)$ and $D(v)$. Figure 4 shows two examples of generating the withering effects based on this simple technique [26,27].

4. DYNAMIC COLLISION AVOIDANCE

Using the LSS tree as discussed earlier, we can develop an efficient algorithm for the detection and avoidance of collisions among sweep surfaces representing the tree branches under deformation.

4.1. Dynamic Bounding Volume Hierarchy Generation

The recursive binary subdivision of the trajectory curve $T(t)$ and the radius function $r(t)$ at the mid-parameter of each subinterval $[t_i, t_{i+1}]$ generates a binary tree where each node contains information for the construction of an LSS volume. Because this simple binary tree can be generated on the fly as needed, there is no need of a preprocessing step for the BVH construction. As we change the trajectory $T(t)$ interactively, we recompute $T''(t)$ for a dynamic update of the radii for all LSS nodes. The radius function $r(t)$ seldom changes in the sweep surface deformation. Thus, the update of the term, max $\|T''(t)\|$, is the main computational overhead in this approach.

4.2. Collision Detection

Following the basic approach of Larsen *et al.* [15], we can detect the collision between two sweep surfaces by checking if their minimum distance drops below a certain threshold $\epsilon > 0$. For this purpose, we sample two surface points, one from each surface, and measure the distance between the two samples, which is an upper bound for the minimum distance between the two sweep surfaces. If the upper bound is smaller than the threshold ϵ, the two surfaces should be within distance ϵ, and they are classified as a colliding pair.

On the other hand, the distance between two LSS bounding volumes can serve as a lower bound for the distance between two sweep surfaces. If two LSS bounding volumes are separated by more than a distance $\delta > 0$, the two sweep surfaces must be separated at least by δ. The distance computation between two LSS volumes is essentially reduced to computing the distance between their skeleton line segments, which can be carried out in a straightforward manner.

4.3. Collision Avoidance

When two branches are in collision or within a certain minimum separation distance, we try to push away the younger (thinner) branch from the older one or push both branches simultaneously when the two are of similar ages. There are many possible ways of resolving the collision status. Yoon and Kim [16] introduced some techniques for freeform deformation of sweep surfaces. However, the deformation results are not physically plausible.

A simple way of handling the collision avoidance problem is to consider the control polygon for the trajectory curve $T(t)$ as a semi-physical multi-link body, where each link has a fixed length. By placing a virtual spring to each joint connecting two adjacent links, we can emulate the effect of physically pushing away the control polygon. We continue the push-way action until the colliding branches are separated by a certain distance $\delta > 0$.

Table I. Storage requirement for the three-dimensional (3D) tree models.

3D trees	(a)	(b)	(c)
Number of vertices	313 644	77 761	188 087
Number of triangles	321 552	124 742	304 556
Size (KB)	11 386	3363	8169
Number of sweeps	2377	63	2236
Number of leaves	28 277	699	1532
Size (KB)	1475	46	369

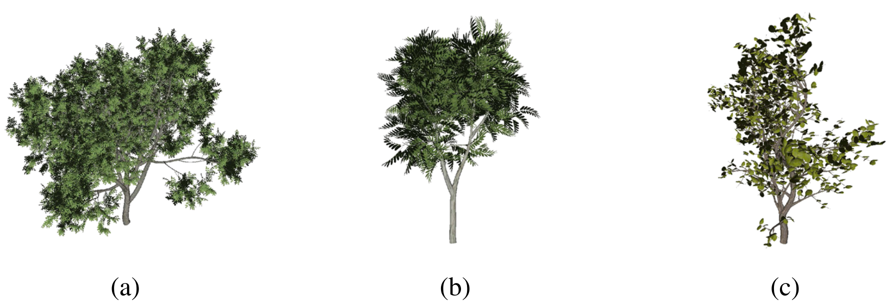
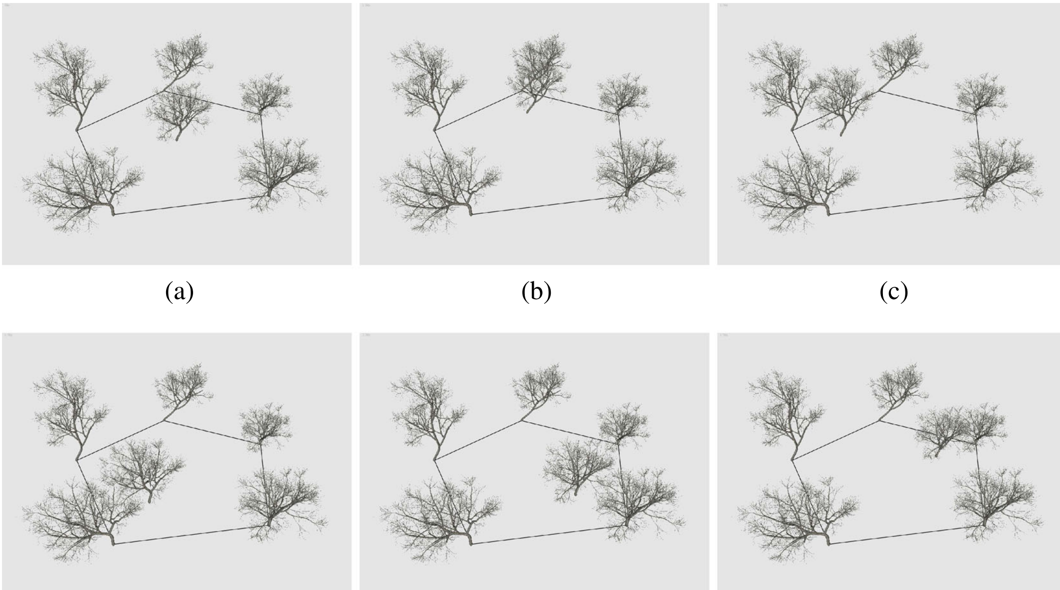


Figure 5. Three-dimensional tree models: (a) Indian sandalwood, (b) African mahogany, and (c) Egyptian carissa.

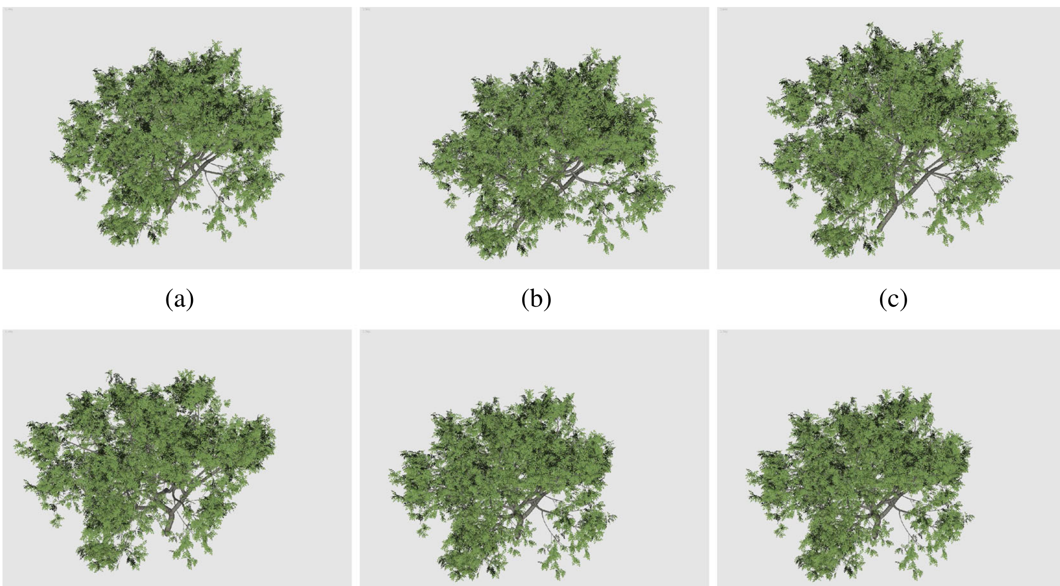
5. EXPERIMENTAL RESULTS

We have implemented the proposed algorithm in C++ on an Intel Core i7 3.4 GHz personal computer with a 3.25 GB main memory and an NVIDIA GeForce GTX 680 GPU. To test the feasibility of our algorithm in various application scenarios for deformable 3D tree models of nontrivial complexity, we have purchased commercial 3D tree models from Xfrog (http://xfrog.com) and converted the polygonal meshes for tree trunks, branches, and leaves into sweep-surface representations. Figure 5 shows the 3D tree models that have been employed for this experiment. Employing the algorithm of Park *et al.* [7], we extract the skeleton of tree and generated the trajectory curve $T(s)$.



(a) (b) (c) (d) (e) (f)

Figure 6. Generating a family of trees (free of self-intersection) using a convex combination of five template models, given at each corner of the pentagonal design space.



(a) (b) (c) (d) (e) (f)

Figure 7. The same trees as in Figure 6 but complete with leaves.

Comp. Anim. Virtual Worlds 2015; **26**:423–432 © 2015 John Wiley & Sons, Ltd.
DOI: 10.1002/cav

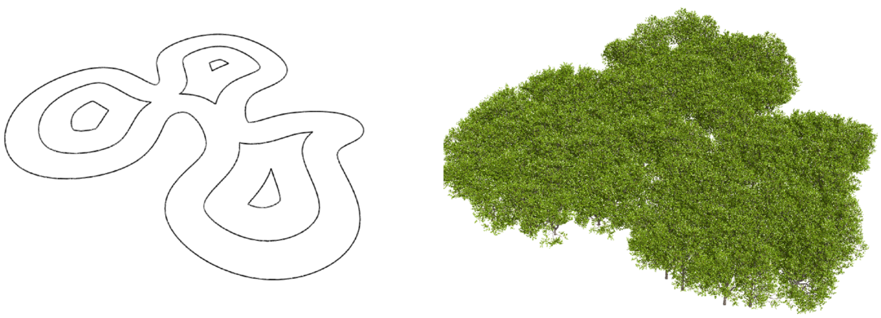
We recorded the relative position and orientation of each leaf with respect to the moving frame along $T(s)$, and the leaves are then smoothly transformed according to the deformation of each branch to which they belong.

Table I shows the data size required for storing the 3D tree models when represented as sweep surfaces as well as polygonal meshes. As one can notice in Table I, compared with the mesh model, there is a significant size reduction in storing the sweep-based representation of the tree models. In the sweep model, the leaves in the same tree require only a small set of translational surfaces. Thus, only a few copies of leaf surfaces are needed for the whole set of leaves.

To demonstrate the performance of our algorithm, we have tested the collision detection and avoidance algorithm in the generation of multiparameter family of trees using a convex combination of template models of the same topology. Figure 6 shows six instances of the sweep-based tree generation (without leaves) as a convex combination of the five templates, given at each corner of the pentagonal design space. (Each template tree is first generated by editing and deforming the original input tree model using our sweep surface editing interface.) The corresponding six trees thus generated are shown in more detail with leaves in Figure 7. For each of the tree models shown in Figure 5, we have randomly generated a total of 100 trees using a convex combination and measured the computing time in (i) the tree generation using a convex combination; (ii) the collision detection; and (iii) the collision avoidance in resolving all collisions occurring in the intersecting branches. The average computing time is reported in Table II. Also reported in the table is the average number of colliding pairs of branches detected in this process.

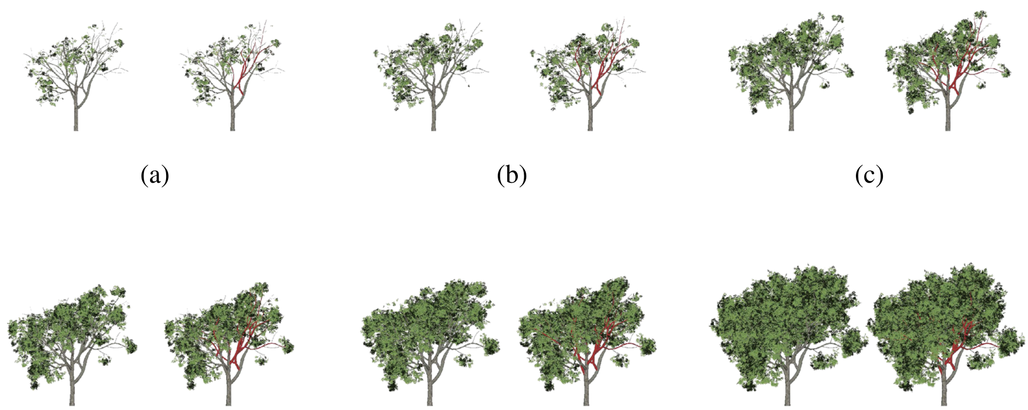
Table II. Computing time for the generation of collision-free trees as a convex combination of five template tree models of the same topology.

Computing time	(a)	(b)	(c)
Tree generation	8	0.15	7
Collision detection	94	0.95	53
Collision avoidance	7	0.02	11
Total (ms)	109	1.12	71
Number of colliding pairs	221	1.06	257



(a) (b)

Figure 8. Colonization of 144 trees that form a forest: (a) the pattern of forest and (b) the result of collision-based tree colonization.



(a) (b) (c)

(d) (e) (f)

Figure 9. Tree growth simulation: (left) the tree with all collisions avoided and (right) the colliding branches in red.

DOI: 10.1002/cav

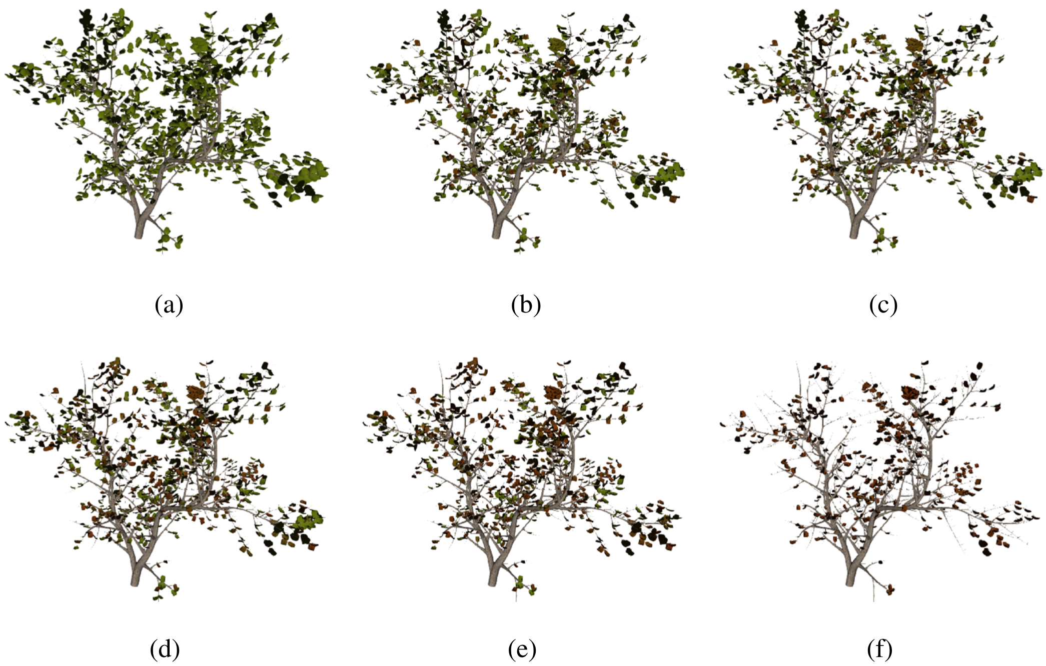


Figure 10. Snapshots from a simulation of withering effects for autumn falling leaves with changing color.

The collision-based tree modeling can be used as an effective tool for the colonization of many trees to form a forest of special pattern. Figure 8(a) shows such a forest pattern, where the outline is given as a closed curve and the smaller loops inside are the result of erosion by a certain distance related to the radius of tree. Figure 8(b) shows the result of planting 144 trees while inter-tree collisions are automatically resolved.

Using the sweep representation of trees, it is also relatively easy to simulate the growing process of trees – we can control the shape change of component sweep surfaces in a time-dependent manner. Figure 9 shows some snapshots from the growth simulation of a sweep-based tree model. In each subfigure, the tree on the left is the result of collision avoidance, and the tree on the right shows all colliding branches (in red) at the current stage of the growth simulation.

Finally, Figure 10 shows snapshots from a simulation of withering effects for autumn falling leaves with changing color. The falling sequence is determined randomly for each leaf, and the color also changes somewhat randomly but in certain trends towards autumn colors for tree leaves.

6. CONCLUSIONS

We have presented an efficient algorithm for the collision detection and avoidance of sweep-based 3D tree models, which can be used as an interactive design tool for collision-free 3D tree models. Based on this geometric tool, we have introduced a natural user interface for blending several given 3D tree models with the same topology to form a multiparameter convex combination of the base shapes. We have also demonstrated the effectiveness of the collision-based modeling approach in the tree growth and aging simulations. In the future work, we plan to extend our current result (limited to the case of blending trees of the same topology) to a more general case of combining different tree models of arbitrary topology.

ACKNOWLEDGEMENTS

This work was supported in part by the People Programme (Marie Curie Actions) of the European Union's Seventh Framework Programme FP7/2007-2013/ under REA grant agreement PIAP-GA-2011-286426, in part by the Israel Science Foundation (Grant No. 278/13), and also in part by the Korean MCST and KOCCA in the CT R&D Program 2014 (No. R2014060001), and in part by NRF Research Grants (No. 2013R1A1A2010085).

REFERENCES

1. Fowler D, Prusinkiewicz P, Battjes J. A collision-based model of spiral phyllotaxis. *Computer Graphics (SIGGRAPH 1992),* Chicago 1992: 361–368.
2. Měch R, Prusinkiewicz P. Visual models of plants interacting with their environment. *Computer Graphics (SIGGRAPH 1996),* New Orleans 1996: 397–410.
3. Palubicki W, Horel K, Longay S, Runions A, Lane B, Měch R, Prusinkiewicz P. Self-organizing tree models

DOI: 10.1002/cav

for image synthesis. *ACM Transactions on Graphics* 2009; **28**(3): 58:1–58:10.

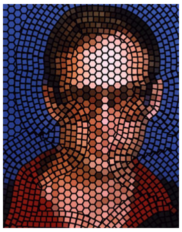
4. Prusinkiewicz P, James M, Měch R. Synthetic topiary. In *Proceedings of the 21st Annual Conference on Computer Graphics and Interactive Techniques*, New York, NY, USA, SIGGRAPH '94, ACM, 1994; 351–358.
5. Pirk S, Stava O, Kratt J, Said MAM, Neubert B, Měch R, Benes B, Deussen O. Plastic trees: interactive self-adapting botanical tree models. *ACM Transactions on Graphics* 2012; **31**(4): 50:1–50:10.
6. Pirk S, Niese T, Deussen O. Capturing and animating the morphogenesis of polygonal tree models. *ACM Transactions on Graphics* 2012; **31**(6): 169:1–169:10.
7. Park J, Shim M, Park S, Kang Y, Kim M. Realistic deformation of 3D human blood vessels. *Computer Animation and Virtual Worlds* 2013; **24**(3–4): 317–325.
8. Zhu X, Jin X, You L. High-quality tree structures modelling using local convolution surface approximation. *The Visual Computer* 2015; **31**(1): 69–82.
9. Filip D, Magedson R, Markot R. Surface approximations using bounds on derivatives. *Computer Aided Geometric Design* 1986; **3**(4): 295–311.
10. Lintermann B, Deussen O. Interactive modeling and plants. *IEEE Computer Graphics and Applications* 1999; **19**(1): 56–65.
11. Prusinkiewicz P, Lindenmayer A. *The Algorithmic Beauty of Plants*. Springer-Verlag: New York, 1990.
12. Weber J, Penn J. Creation and rendering of realistic trees. *Computer Graphics (SIGGRAPH 1995),* Los Angeles 1995: 119–128.
13. Stava O, Pirk S, Kratt J, Chen B, Měch R, Deussen O, Benes B. Inverse procedural modeling of trees. *Computer Graphics Forum* 2014; **33**(6): 118–131.
14. Gottschalk S, Lin M, Manocha D. OBB-tree: a hierarchical structure for rapid interference detection. *Computer Graphics (SIGGRAPH 1996),* New Orleans 1996: 171–180.
15. Larsen E, Gottschalk S, Lin MC, Manocha D. Fast proximity queries using swept sphere volumes. *Technical Report TR99-018*, Department of Computer Science, UNC, Chapel Hill, NC, USA, 1999.
16. Yoon S-H, Kim M-S. Sweep-based freeform deformations. *Computer Graphics Forum* 2006; **25**(3): 487–496.
17. Runions A, Lane B, Prusinkiewicz P. Modeling trees with a space colonization algorithm. In *Eurographics Workshop on Natural Phenomena*, Prague, Czech Republic, 2007; 63-70.
18. Tan P, Fang T, Xiao J, Zhao P, Quan L. Single image tree modeling. *ACM Transactions on Graphics* 2008; **27**(5): 108:1–108:7.
19. Tan P, Zeng G, Wang J, Kang SB, Quan L. Image-based tree modeling. *ACM Transactions on Graphics* 2007; **26**(3): Article 87:1–8.
20. Akenine-Möller T, Hains E, Hoffman N. *Real-time Rendering* 3rd ed. AK Peters, Natick, MA, 2008.
21. Krishnamurthy A, McMains S, Hanniel I. GPU-accelerated Hausdorff distance computation between dynamically deformable NURBS surfaces. *Computer-Aided Design* 2011; **43**(11): 1370–1379.
22. Hyun D, Yoon S, Chang J, Seong J, Kim M, Jüttler B. Sweep-based human deformation. *The Visual Computer* 2005; **20**(8–10): 542–550.
23. Lee J, Kim M. Realistic human hand deformation. *Computer Animation and Virtual Worlds* 2006; **17**(3–4): 479–489.
24. Wang W, Jüttler B, Zheng D, Liu Y. Computation of rotation minimizing frames. *ACM Transactions on Graphics* 2008; **27**(1): Article 2:1–18.
25. Farin G. *Curves and Surfaces for CAGD* 5th ed. Morgan Kaufmann: San Francisco, CA, 2002.
26. Chi X. Simulation of autumn leaves. *ACM SIGGRAPH Posters* 2007: Article 36.
27. Chiba N, Ohshida K, Muraoka K, Saito N. Simulation of leaf arrangement and autumn colors. *Journal of Visualization and Computer Animation* 1996; **7**(2): 79–93.

AUTHORS' BIOGRAPHIES

Yong-Joon Kim is a postdoctoral research fellow in the Department of Computer Science, Technion, Israel. His research interests are computer-aided geometric design, real-time 3D geometry processing, and computer graphics. He received a BS degree in Mathematics Education and a PhD degree in Computer Science from Seoul National University in 2008 and 2012, respectively.

Jong-Hwa Woo is a software engineer at INFINITT Healthcare Co., Ltd., Seoul, Korea. He received BS and MS degrees in Computer Science and Engineering from Seoul National University in 2011 and 2013, respectively.

Myung-Soo Kim received BS and MS degrees from Seoul National University in 1980 and 1982, respectively, and an MS degree in Applied Mathematics and MS and PhD degrees in Computer Science from Purdue University in 1985, 1987, and 1988, respectively. He is currently a Professor of the Department of Computer Science and Engineering, Seoul National University. His research interests are in computer graphics and geometric modeling. From 1988 to 1998, he was with the Department of Computer Science, POSTECH, Korea. He serves on the editorial boards of the journals *Computer- Aided Design*, *Computer Aided Geometric Design*, and the *International Journal of Shape Modeling*. He also edited more than 10 special issues of journals including *Computer-Aided Design*, *Computer Aided Geometric Design*, *Graphical Models*, the *Journal of Visualization and Computer Animation*, *The Visual Computer*, and the *International Journal of Shape Modeling*. Together with G. Farin and J. Hoschek, he edited the *Handbook of Computer Aided Geometric Design* (Amsterdam: North-Holland, 2002). In 2010, he served as a technical program co-chair of the ACM Symposium on Solid and Physical Modeling, Haifa, Israel. In 2011, he also served as a program co-chair of the SIAM Conference on Geometric and Physical Modeling, Orlando, USA.



Gershon Elber is a professor in the Computer Science Department, Technion, Israel. His research interests span computer-aided geometric designs and computer graphics. Professor Elber received a BSc in Computer Engineering and an MSc in Computer Science from the Technion, Israel, in 1986 and 1987, respectively, and a PhD in Computer Science from the University of Utah, USA, in 1992. He is a member of SIAM and the ACM. Professor Elber has served on the editorial board of the *Computer Aided Design*, *Computer Graphics Forum*, *The Visual Computer*, *Graphical Models*, and the *International Journal of Computational Geometry & Applications* and has served in many conference program committees including Solid Modeling, Shape Modeling, Geometric Modeling and Processing, Pacific Graphics, Computer Graphics International, and SIGGRAPH. Professor Elber was one of the paper chairs of Solid Modeling 2003 and Solid Modeling 2004, one of the conference chairs of Solid and Physical Modeling 2010, and the chair of GDM 2014. He has published over 200 papers in international conferences and journals and is one of the authors of a book titled *Geometric Modeling with Splines—An Introduction*. Professor Elber received the John Gregory Memorial Award, 2011, in "Appreciation for Outstanding Contributions in Geometric Modeling."

Comp. Anim. Virtual Worlds 2015; **26**:423–432 © 2015 John Wiley & Sons, Ltd.
DOI: 10.1002/cav