Erschienen in: Computer Graphics Forum ; 34 (2015), 2. - S. 361-372
https://dx.doi.org/10.1111/cgf.12566

Woodification: User-Controlled Cambial Growth Modeling

J. Kratt[1], M. Spicker[1], A. Guayaquil[2], M. Fiser[2], S. Pirk[4], O. Deussen[1], J. C. Hart[3], B. Benes[2]

[1]University of Konstanz, Germany [2]Purdue University, USA
[3]University of Illinois at Urbana-Champaign, USA [4]Esri R&D Center, Switzerland

Figure 1: *A polygonal object developed by using cambial growth against an obstacle.*

Abstract

We present a botanical simulation of secondary (cambial) tree growth coupled to a physical cracking simulation of its bark. Whereas level set growth would use a fixed resolution voxel grid, our system extends the deformable simplicial complex (DSC), supporting new biological growth functions robustly on any surface polygonal mesh with adaptive subdivision, collision detection and topological control. We extend the DSC with temporally coherent texturing, and surface cracking with a user-controllable biological model coupled to the stresses introduced by the cambial growth model.

1. Introduction

D'Arcy Thompson [Tho42] demonstrated that the history of a natural object is revealed by its form. This is well-known for trees in temperate geographic regions, as the history of their development is represented by their growth rings. Much of the shape of trees is thus dictated by the growth documented by these rings.

The field of computer graphics includes many explorations in the study of primary "apical" tree growth, most notably through L-systems [PL91] and recently through environmental factors [MP96, PHL*09]. These works simulate the entire tree, focusing on the introduction and elongation of branches as opposed to their thickening. Fewer studies can be found in computer graphics (though many exist in the botanical literature) about the secondary "cambial" growth. Cambial growth is the process that increases a tree's girth so that it can support more branches and leaves, transport more water and nutrients, and produce wood and its tree-ring structure. Cambial growth also produces familiar tree features such as crotches, bulges, burls and knobs. A simulation of cambial growth would provide designers and animators with a time-saving tool for producing such effects on a large forest scenery scale, as well as "enchanting" any computer graphics shape model by synthesizing such familiar woodgrown features. To the best of our knowledge an automatic tool that would allow simulation of wood-like appearance of general polygonal mesh objects does not exist.

A new cambial growth model (Sec. 4) based on recent advances in robust meshed surface propagation is introduced. This new model derives biologically motivated growth functions that transfer a variety of natural botanical processes (Sec. 3.1) into the shape morphology of offsetting. We also incorporate obstacle collision detection with the novel method of embedding obstacles directly into the surface

propagation data structure (Sec. 4.5), along with botanically motivated collision response. We also incorporate user control into the growth model to support species-dependent behavior simulation, environmental effects and natural image synthesis. Unlike previous work, we do not focus exclusively on tree modeling, but also on small wood-like features that enhance ordinary polygonal meshes.

We combine this cambial growth model with an existing surface fracture model in Sec. 5, extended with specific botanical stress computations and responses, to simulate the realistic appearance of bark. A key observation motivating our coupling of cambial growth and bark cracking is that cambial growth causes lateral expansion which produces stress that we use in our cracking simulation to produce a visually plausible bark model. We also introduce the notion of *woodification,* a process of converting ordinary everyday meshed geometric objects into a tree-like appearance, with familiar woodgrown features and cracked bark textures. Our cambial growth simulation on trees and other objects can also be reversed (Sec. 4.3), to infer the growth history and therefor the growth rings of a scanned real-world tree or any other object.

Figure 1 shows an example produced by our framework. The input tree trunk is a simple polygonal-mesh cylinder, along with a user-positioned obstacle (a sign) also embedded in the simulation propagation data structure. The simulation computes the cambial growth of the tree as it grows and adapts around the obstacle, maintaining a coherent texture parameterization (Sec. 4.6) and supporting bark that cracks based on growth stress. This result compares well to real-world examples, such as Fig. 2 (left).

Figure 2: *Two real-world examples of wood grown around obstacles.*

Our system, summarized in Fig. 3, operates directly on polygonal meshes. Our growth model is based on the *deformable simplicial complex* (DSC) [Mis10, MB12] reviewed in Sec. 3.2. The DSC simulates surface evolution directly and robustly on a polygonal mesh lying along the boundary between an interior and a exterior tetrahedral meshes. The DSC evolves the shape model over time using a biological growth function, which creates stress that our

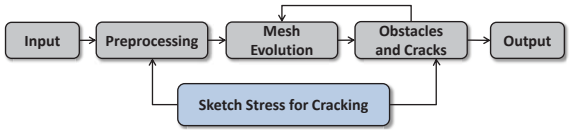

Figure 3: *System Overview. A polygonal model is converted to a tetrahedral mesh during preprocessing. The mesh is evolved by using our developmental model and, at the same time, is covered by cracks and checked for collisions in the surrounding environment. The user can control the definition of the cracks and the growth function.*

cracking model uses to simulate bark fractures on the surface. The user can control the simulation by sketching areas to increase, reduce, or modify the growth direction (Sec. 4.4) and by sketching stresses during the simulaton to guide bark cracks (Sec. 5).

In summary, this work includes the following novel contributions:

- a robust polygonal cambial growth model based on the DSC,
- new biological user-controlled growth functions,
- a reversible growth function for inferring tree rings from the shape of an existing tree,
- persistent texturing of a DSC evolving mesh, and
- a physical cracking model coupled to lateral growth stress.

2. Related Work

Woodification relates to the results in simulation of natural phenomena, fracture simulation and surface propagation.

Bark modeling has evolved from phenomenological approximations to more botanically faithful simulations. Bloomenthal [Blo85] parameterized a branching tree model to support realistic bark texturing. Hart and Baker [HB96] parameterized an implicit tree model using particle flows from the trunk to the branches. Lefebvre and Neyret [LN02] parameterized the surface of a branching tree model using horizontal strips, which were procedurally fractured and propagated to generate a bark texture.

A variety of procedural, implicit, and texture synthesis methods have been used to model bark and branching junctions geometrically [Opp86, Har97, GMW04, WWL*03]. Federl and Prusinkiewicz [FP96] simulated a bark cracking texture by using a mass-spring network and associated finite-element model [FP04], which they applied to a layered model [FP02] for capturing the stresses and strains while growing differential cambial layers. Hirota *et al.* [HTK98] similarly applied a layered mass spring model to simulate cracked appearance of bark. Pfaff *et al.* [PNdJO14] restructure a surface mesh to mimic tearing and cracking of thin sheets. Our approach is based on a surface model as well, however it does not focus on thin layers.

Biological growth modeling includes methods constructed from statistical models on field measurements, e.g., [PR95, SDPLV06, CMF*10] and pure simulation-based approaches. Cambial growth is part of such growth processes and can be described by an offsetting operation that is supported by the level set method [KB93, Set99, OF02]. Here, a growth function is defined over a surface; typically, however, only on voxel grid models with fixed resolution that is much too low for our approach. Nevertheless, there have been attempts to simulate such growth: Buchanan [Buc98] visually simulated woodgrain and growth rings with a user-controlled growth function. Mann et al. [MPW07] computed cambial growth with level sets that encompassed growth around obstacles and curvature-dependent growth functions. Sellier *et al.* [SPH11] coupled a level-set cambial growth model with an apical (e.g., L-system) growth model, and used the result to investigate various botanical hypotheses. Similarly, Mizoguchi *et al.* [MM11] used a combination of L-systems and cell subdivision to simulate the development of trees by using polygonal meshes.

Surface propagation is commonly implemented using the level set method, which operates on a volumetric representation for robustness, but suffer appearance artifacts and additional volumetric work for a surface simulation, which can be reduced by adaptive and compressed level set methods, (e.g., [HNB*06]). Surface meshes can also be robustly propagated, (e.g., [Jia07]). We utilize the deformable simplicial complex (DSC) [Mis10, MB12] which also maintains an interior and exterior tetrahedral mesh. The DSC allows us to operate directly, robustly and adaptively on a surface mesh, which avoids cuberille voxel grid artifacts.

Our interactive interface supports shape modeling by painting a growth function on a polygonal surface model. Previous work simply propagated mesh vertices without concern for mesh topology, and limited growth rates to amounts small enough to avoid shocks, collisions, and foldovers [Mat02, CN06]. We show that the interface now runs robustly using the DSC method. Furthermore, we are able to maintain a parametrization on the deforming shape. This is done by using weighted primitive coordinate systems [WWM87, Ped95]. Rasmussen *et al.* [REN*04] creates textures with level sets by attaching texture coordinates to advected particles. The method fails when the topology of the surface changes, e.g., if wood grows around an obstacle.

Cracking and fracture modeling includes meshless [PKA*05], tetrahedral [SDF07], and real-time [PO09] methods. We utilize a surface-based method [IO06] that relies on a stress field over a surface mesh and extends a previous tetrahedral FEM approach [OH99]. This method is capable of simulating ceramic glaze, mud, or glass. We introduce a new way to initialize the stress field interactively and evolve it during the growth process.

3. Background

This work simulates cambial growth using the deformable simplicial complex. In this section both are reviewed.

3.1. Cambial Growth

Wood and bark grow primarily because of new cells that are added to the plant surface at its cambium layer [Bai23, Lar94], demonstrated by trees when they grow around obstacles as shown Fig. 2. The cambium generates new interior xylem wood cells that transport water and exterior phloem living tissue bark cells that transport sugar, both in vertical "fusiform" and horizontal "ray" configurations. The interior "heartwood" eventually dies but continues to serve as a structural support for the tree.

Cambial growth forms visible growth rings in temperate geographic regions due to differences between lighter thin-walled cell springtime earlywood and darker thick-walled cell autumnal latewood [Wil84]. Trees must support the mass of extensive branching structures designed to gather photosynthate. A growing tree is able to regulate its cambial growth to support limbs through the production of "reaction wood". Softwood species generate compression wood that supports from below, whereas hardwood species generate tension wood that suspends from above [Mat91].

The (cork) cambium also produces exterior vertical fusiform cork cells that form the protective layer of the bark. As the cambium circumference grows, it stresses the outer smaller-circumference cork-cell layers, which then shed, peel or crack vertically along the fusiform cell walls [Wil84].

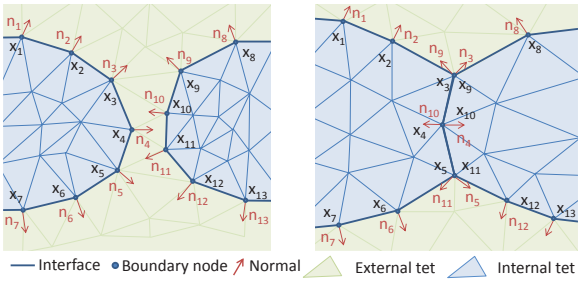

Figure 4: *The DSC method propagates surface mesh vertices, using interior/exterior tetrahedral meshes to detect collisions, and can maintain contact between components.*

3.2. The Deformable Simplicial Complex

We simulate such cambial growth with a configurable offsetting deformation using the deformable simplicial complex (DSC). The DSC [Mis10, MB12] deforms a surface mesh directly by moving its vertices $\mathbf{x}_i$, usually along the surface normal $\mathbf{n}_i$ controlled by a speed function $s(\mathbf{x}_i, t)$ evaluated at the vertex position at different time steps

$$\dot{\mathbf{x}}_i = s(\mathbf{x}_i, t) \cdot \mathbf{n}_i. \tag{1}$$

The function can control the speed function to orient the direction of reaction wood growth and the resulting eccentricity of the growth rings. Whereas level sets only grow in their normal direction, the DSC can grow in any user defined direction and can support both the lateral growth of the cambium in the normal direction and the apical growth of meristems in the tangential direction. Hence, the DSC provides the flexibility to simulate all aspects of tree growth with a single method.

The DSC also overcomes the common roadblocks of Lagrangian surface mesh deformation caused by self-intersections (e.g., fold-overs) by maintaining interior and exterior tetrahedral meshes that conform to the surface triangle mesh. Both local and global self-intersections of the surface mesh are detected by inverted tetrahedra in these internal and external tetrahedral meshes.

The method robustly grows a surface mesh by moving each of its vertices (in arbitrary order) to its destination. If this motion would invert a tetrahedron in the internal or external mesh, then the vertex motion is temporarily limited to the point where it yields a degenerate (flat) tetrahedron, which is removed by a local retesselation allowing the vertex motion to continue. The amount of re-tesselation needed can be controlled by the time step, which determines how far a vertex moves relative to its neighbors, furthermore smoothing of the interior/exterior meshes improves performance. We can adaptively refine and coarsen the surface mesh to improve mesh quality without undue computation.

Fig. 4 demonstrates DSC propagation, and how it provides control over geometry and topology. Geometric control is provided by adaptive remeshing options. A big advantage of the DSC over voxel level set approaches is that it provides complete control of the surface topology. When the two interfaces in Figure 4 meet, they preserve their shared boundary instead of merging as they would with a level set approach. The DSC can support both merging and precise-contact surfaces, but the latter is needed for simulating cambial growth, e.g., when different limbs overlap.

4. Simulation of Cambial Growth

To simulate cambial growth, we embed our input watertight polygonal surface mesh in a tetrahedral mesh, generated by TetGen [Si07]. The DSC evolves this representation using a supplied surface growth speed function in the direction of the surface normal. This section describes how we include biological growth properties into the speed function, how we can reverse growth to predict a tree's prior development, and how we extend DSC to manage obstacles and texture. The stress due to cambial expansion is significant enough to produce cracking in the cork, but not enough to affect tree shape or growth rate, and so is not included in our macroscopic growth function. The stresses that affect the growth rate have been accounted for in our reaction wood approximation. Unlike some previous work (e.g., [CN06, MPW07]), we did not incorporate curvature into our growth model. Curvature has been used to regulate growth in e.g., the level set method to smooth shocks it but is not needed for the DSC. We did incorporate curvature in the reversed growth function to predict growth history where stems emerge from branches, instead of appearing out of thin air and joining branches, as described in Section 4.3.

4.1. Allometric Growth

Allometric growth simulates how the rate of tree growth varies with the age of the tree. The rules of allometry and the field of forest mensuration produce data-driven models of tree growth based on field measurements. They have modeled cambial growth as a single-parameter Weibull distribution function [LS11]

$$s(t) = ct^{c-1}\exp(-t^c), \tag{2}$$

where t represents time of growth and $c > 0$ is a species-dependent parameter (we use $c = 2$ in our experiments). The parameter represents the stretching strength associated to the surface growth.

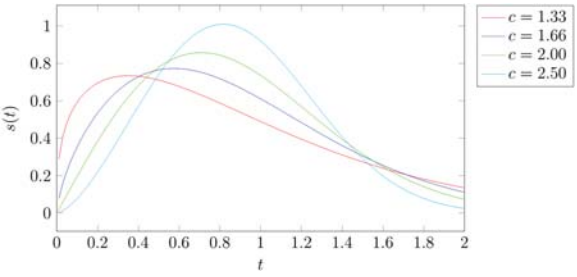

Figure 5: *Graph of the allometric growth function defined in Eq. (2) over value t (proportional to age) for different values of species parameter c.*

Figure 5 shows the plot of this function for varying c, indicating a peak "prime of life" growth rate which marks the transition from a juvenile period of increasing growth rates into a mature period of declining growth rates. Use of this function is an approximation, as the function accounts for all tree growth instead of just cambial growth.

4.2. Structural Growth

We approximate the growth of reaction wood through an eccentricity parameter e to define the function:

$$s_e(t,\mathbf{x}) = (1 + e \cdot \hat{\mathbf{n}}(\mathbf{x}) \cdot \hat{\mathbf{y}}) \cdot s(t), \tag{3}$$

where $\hat{\mathbf{y}}$ is a unit vector that points upwards in vertical direction and $\hat{\mathbf{n}}$ a unit vector in the direction of the normal. The compression wood of softwood trees is modeled by negative eccentricity, whereas the tension wood of hardwood species is modeled by positive eccentricity.

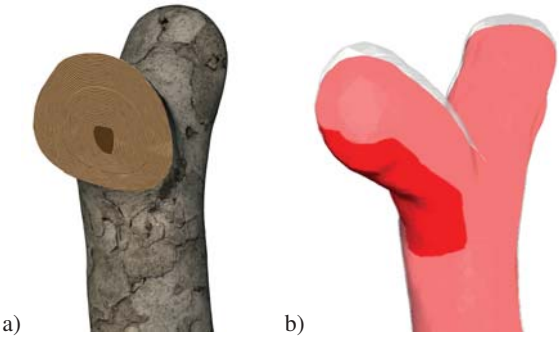

Figure 6: *(a) Annual growth rings generated by intermediate cambial meshes demonstrate positive eccentricity to simulate the growth of tension wood. (b) Positive eccentricity (transparent white) overlays non-eccentric growth (red).*

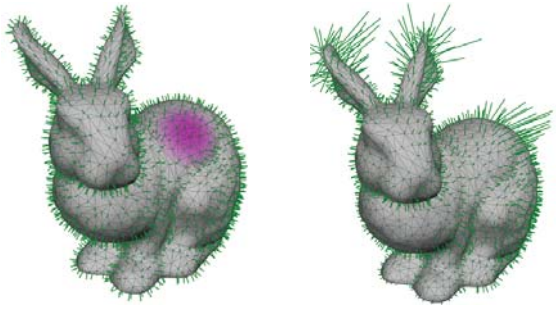

Figure 7: *Visualization of the brush tool. Left: The purple region shows growth directions to be modified interactively. Right: the growth magnitude and direction of the ears and back have been modified by dragging the green normal vectors.*

Figure 6(a) shows the inner structure of the wood with growth rings that were generated by compositing successive cambial meshes of the growth model. The positive eccentricity (Eq. (3)) simulates the growth of extra wood at places with high tension. Figure 6(b) shows the growth of such reaction wood, in this case positive eccentric tension wood (transparent white), in comparision to the ordinary growth (in red). Note that a complete growth model would couple cambial growth for reaction wood with the apical growth of e.g., an L-system to better estimate the structural needs of the branching skeleton and hence the growth rate of the reaction wood [MBH03]. These effects are omitted here since we concentrate on the details of the cambial wood production.We furthermore neglect factors such as light, water concentration, and wood density for our model.

4.3. Reversed Growth

Our cambial lateral growth model operates on an initial polygonal surface mesh. A full apical growth model, e.g., an L-system, would grow tree limbs from their sapling radii, but would also require significant growth of the tangential meristem. Since we want to concentrate on lateral growth, we model an adult tree surface, and simulate its growth both forward and backward in time. We simulate forward growth with varying dilation using the above-mentioned growth speed functions. To obtain a tree's growing history we use backward simulation by applying a number of successive erosion operations on the adult tree. We thus negate the speed function and in this way model the interior annual growth rings by inverse growth (Figure 6). Note that such negative growth should include a small proportion of curvature in its speed function to avoid unrealistic sharp "shock" features [Set99].

4.4. User-Defined Growth

The above-described growth functions work in a fully automatic way and can be controlled only by their input parameters and the user interface at runtime. To provide extra control, we allow the user to interactively modify the growth function on the growing surface. Such user-defined growth can be defined when the object is created or applied during each iteration by letting the user sketch changes to the growth function on the surface of the object.

A ray-casting technique is used to change the growth by user interaction with the value and direction of the growth function depicted as length and direction of vectors at the vertices as shown in Figure 7. A Gaussian filter defines the spatial decay of the brush, which is used to alter the values on the surface.

4.5. Growth Around Obstacles

Trees grow around obstacles, such as nearby fences or sign posts (Figure 2). When the cambium and the obstacle collide, the cambium continues to grow around the obstruction, eventually absorbing it into the interior of the tree.

We utilize the DSC data structures to simplify the collision detection and response within our cambial growth simulation. Recall that DSC maintains an external tetrahedral mesh that conforms to the growing surface mesh used to simulate our cambial growth. This external tetrahedral mesh also conforms to the boundaries of any obstacles (which is also how DSC fluids handle obstacles [EMB11, MEB*12]).

The DSC propagates each moving front vertex until it reaches a tetrahedral face. If this face is also the face of an obstacle embedded in the external mesh, then that vertex is "frozen" at the point of collision. The DSC will do the same for the associated tetrahedrons in the external mesh, such

that the front propagation process cannot alter these vertices any longer. Figure 8 shows the reaction of the simulation to an obstacle. The volume of the obstacle is shown in red. As soon as the front hits the volume, its vertices are frozen and the rest of the mesh continues to grow. Thanks to the subdivision feature of DSC, the interaction with an obstacle can be modulated to create a smooth wrapping.

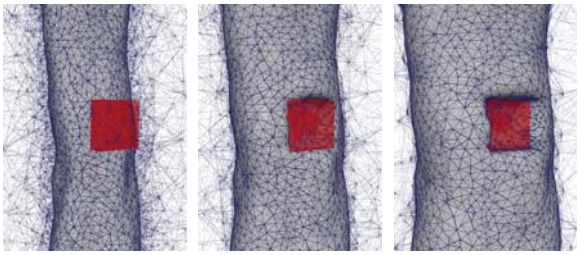

Figure 8: *Different frames for the exterior tetrahedral mesh (blue) for a tree surface (gray) along with a box obstacle (red) during the growth.*

Even though vertices move only in their normal direction, differences in growth rate due to obstacle collision affect the normal direction since they change the relative positioning of neighboring vertices. Hence normal growth produces growth components tangential to the original normal direction, as seen in Figure 1, where the tree grows around the sign and begins to cover it.

If the growth continues the front will eventually surround the obstruction and intersect with itself. Since the DSC can maintain precise contact after self collision (Figure 4) by a similar "freezing" procedure than when fronts collide, the front in this case creates a crease wound that realistically persists through future growth iterations. Figure 1 shows a result of such a simulation.

4.6. Persistent Texture Coordinates

Our growth simulation is initialized with a mesh that is parameterized to support texture mapping. As the mesh expands due to its growth, the original parametrization is also propagated. However, the DSC approach moves front vertices until they create an degenerated external tetraedral mesh with subsequent inversion. At this point DSC performs a local remeshing, which can result in vertex insertions and deletions, and various edge swaps. Such adaptations require an update of the parametrization.

When an existing vertex is deleted, its texture coordinates are removed. When a new vertex is inserted, we interpolate its texture coordinates from its neighborhood. When a vertex moves, we decompose the motion into the tangent and the bi-normal components, by using the Gram-Schmidt process, with respect to the vertex normal. Normal motion preserves texture coordinates, whereas tangential space spatial motion

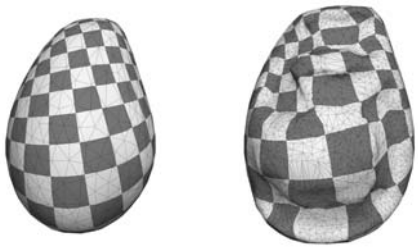

Figure 9: *Texture coordinates are preserved after DSC propagation and numerous remeshing operations.*

requires a proportional movement of the corresponding texture coordinates. The amount of this motion can be either computed from the spatial motion using the texture Jacobian $\partial\mathbf{u}/\partial\mathbf{x}$, or reinterpolated from the neighborhood. In all of our experiments no visible numerical dissipation on the texture maps were visible.

As the cambial surface grows, so will the texture Jacobian, which magnifies the texture. This growth is primarily due to the insertion of new cells of the same size, so cellular details in the base texture should not greatly expand. We realize this phenomenon with a mip-map of textures of increasing resolution that maintain a constant pixel resolution of cellular details. Figure 9 shows how texture coordinates are updated after remeshing and growth by using DSC.

5. Surface Cracking Model

The cambial growth model from Section 4 expands the surface over time, creating tension which is eventually relieved by creating cracks. We simulate such behavior by using the physically-based approach of Iben et al. [IO06]. This algorithm uses a stress field defined over a triangle mesh and evolves it over time. Stress is distributed from regions of higher stress to regions of lower stress. Each simulation step computes the separation tensor to represent the forces acting on the vertices of the mesh. If the stress exceeds a user-given threshold, a new crack edge is inserted perpendicular to the separation tensor's maximum eigenvector. We encourage crack propagation by adding additional stress at the tips of crack edges, which makes them prone to grow in a subsequent steps, instead of creating and merging nearby cracks. In comparison to related methods it offers a large amount of user control and the possibility to initialize the stress field heuristically. This allows to create different crack patterns appearing on bark of different species. Patterns emerge due to the initialization of the stress and develop during the simulation. If no stress is specified, the stress field is initialized as an equilibrium. Working directly on surface meshes, in contrast to some previous work [LN02], allows for an easier coupling between the growth modeling and crack simulation.

Stress is initialized on the input mesh either by a predefined pattern [IO06] or interactively sketched on the mesh or texture as seen in Figure 11. The direction in which the user has placed strokes is used to initialize the stress orientation. In Figure 10 we compare our results for the cracks with photographs of tree bark from different species. We removed existing cracks in the photographs with the technique presented by [BSFG09] and used the remainder as input texture. The stress was initialized by a set of predefined patterns.

During the growth simulation the DSC method can perform local changes to the mesh. The cracking and stress information stored in the vertices and faces of the mesh has to be transferred to the evolved mesh in order to achieve coherence in the final results. To track those local changes, every vertex within the DSC is assigned an unique identifier. If new vertices are added or the identifier is lost due to vertex deletion, a local proximity-based similarity model is used to identify corresponding vertices in subsequent evolution steps of the meshes. This also allows us to track faces and their stored stress information. With this, cracks and stress tensors can be transferred between evolved meshes at consecutive time steps. If a corresponding face cannot be found in the previous mesh, the stress in this face is interpolated from the local neighborhood. In case the cracking model introduces new edges and vertices, these have to be created in the subsequent mesh as well. We set time steps for the evolution and cracking small enough that only a few local remeshing operations are necessary and tracking is most of the time possible.

The growth process introduces additional stress, measured as the displacement of a triangle barycenter between subsequent DSC time steps. The user can freely define the mapping between the displacement and the stress increment. In each step, this increment is added to the stress that was transferred from the previous time step of the evolution. Additionally, the stress in regions with high curvature can be increased, resulting in more cracks in strongly curved regions. The curvature tensor is determined [CSM03] and added to the stress tensor.

The edges associated with cracks are divided into connected components for rendering. Each connected component of edges represents a single crack. The width of the crack is determined per-vertex based on the distance to the nearest endpoint of its crack component. Hence cracks open more in the middle. We use displacement mapping to create the depth of the cracks. The depth also correlates with its width: The closer to the crack edge, the larger the displacement, creating a V-shaped profile. Both the depth and the width of the cracks are modulated by noise to add a realistic natural variance. The mesh is also refined near crack ridges to improve displacement mapping quality. The properties of a crack including its appearance can change, so we use full dynamic displacement mapping instead of static displacement maps.

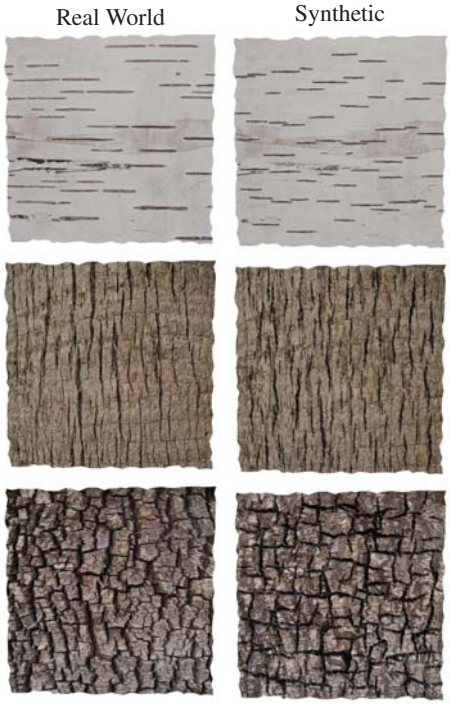

Figure 10: *Comparison between photographs of tree bark (left) and synthetic results rendered with the cracking model (right). Stress tensor initialization creates different patterns. Top: vertical stress. Middle: horizontal stress. Bottom: mix.*

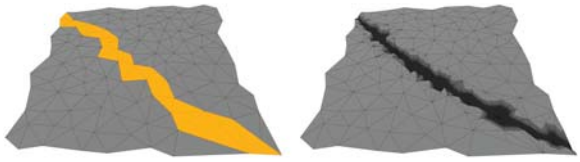

Figure 11: *An example of user-defined cracks (left) and the resulting cracks on a surface (right).*

Figure 11 shows an example of user-defined cracks. A user sketched stroke influences the formation of cracks along the corresponding triangles of the mesh, shown in yellow on the left. The direction of the stroke is used to initialize the orientation and magnitude of the stress tensors: The outer product of the vector orthogonal to the stroke direction through the triangle is used as the stress tensor at the center of this triangle. Thereby the main direction of the stress tensor in the selected triangles is oriented to produce the painted crack path. The resulting crack pattern follows the location and direction of the user input, as shown on the right.

6. Implementation and Results

Our system is implemented in C++, using OpenGL and GLSL for rendering, and the 3D DSC library [Bær14,Chr14] for surface propagation. All results were generated on a 2.0 GHz Intel Xeon E5 2620 CPU with 32GB RAM, and rendered with a 2GB Nvidia Quadro 4000 GPU.

Tables 1, 2, and 3 list parameters used in the simulations. The DSC parameters tradeoff speed with quality. Values that increase simulation speed created deformed tetrahedrons whereas values that increase precision slowed down the simulation. In all simulations the time step was set to one. Cracking parameters were chosen in a range proposed by the original authors. Fig. 1 shows frames from the animation of successive growth of a tree trunk model against a sign obstacle. The DSC method smoothly and robustly adapts the cambium around the sign geometry embedded in its external mesh. Figures 12, 13, 14 and 15 show a variety of other cambial growth examples.

Model (Figure)	v_i	v_f	S	t [s]	t/S [ms]
Tree Close-up (Fig. 1)	10k	39k	200	32,4	162
Tree Rings (Fig. 6)	6k	5k	100	3,6	36
Hand (Fig. 12)	7.3k	7.9k	200	5,4	27
Kitten (Fig. 12)	5.5k	8.7k	200	5,4	27
Bunny (Fig. 15)	4.0k	6.6k	200	3,0	15
Knot (Fig. 15)	8k	13k	200	4,5	22.5
Memento (Fig. 15)	7.6k	8.1k	200	7,2	36

Table 1: *Performance statistics showing the number of initial vertices v_i final vertices v_f and iterations S, as well as the computational time t and average time per time step t/S.*

Model (Figure)	Speed Growth	Subdivision Length
Tree Close-up (Fig. 1)	0.01	0.3
Tree Rings (Fig. 6)	0.01	0.2
Hand (Fig. 12)	0.01	0.3
Kitten (Fig. 12)	0.02	0.4
Bunny (Fig. 15)	0.01	0.4
Knot (Fig. 15)	0.05	0.3
Memento (Fig. 15)	0.02	0.2

Table 2: *Growth parameters, selected to best tradeoff performance with mesh quality.*

Table 1 shows the computation time of our method. The table shows the number of the initial vertices, number of the vertices at the end of the simulation, number of iterations, overall time of simulation, and the average time per step. The most time-demanding operation of the simulation is incurred by the DSC library when it checks propagated vertices against tetrahedra. The number of vertices has increased for the forward growth simulations because of the adaptive nature of the DSC method. The example in Figure 6 was generated by using inverse growth and therefore it ends with fewer vertices.

Model (Figure)	CPI	RPI	α	ESA [π]
Tree Close-up (Fig. 1)	10	0	0.6	0.4
Hand (Fig. 12)	8	1	0.4	0.5
Kitten (Fig. 12)	15	5	0.3	0.3
Bunny (Fig. 15)	3	1	0.2	0.5
Knot (Fig. 15)	10	5	0.54	0.3
Memento (Fig. 15)	15	5	0.6	0.3

Table 3: *Cracking parameters. CPI: Cracks introduced into the mesh per iteration. RPI: Relaxation steps per iteration. α: Amount of stress added to the tips of the crack edges ($[0,1]$). ESA: Angle under which a newly created crack edge snaps onto an existing edge.*

Our single-core implementation was designed primarily to prove the concepts and image quality, and not for speed and interactive rendering. Similarly, our simulation is built on a beta DSC library that focuses on robust Lagrangian mesh propagation, and has not yet been optimized for speed. We expect DSC to eventually run in linear time, given a constant bound on the number of tetrahedra traversed by a moving vertex, and a constant bound on the number of tetrahedra involved in a local remeshing operation. Similar performance concerns were reported when DSC was introduced [Mis10] and used for fluid simulation [EMB11, MEB*12].

We found the method ran in acceptable computational times on meshes consisting of several thousands of vertices. The cambial growth simulation can handle even larger meshes, given that the DSC can control the introduction of new vertices. We tuned the growth speed to acquire a stable vertex advection and the subdivision length threshold to introduce few new triangles, which reduce the number of topology checks by the DSC in each iteration. All parameters used for the growth of the models shown in this paper are depicted in Table 2.

The cracking model, on the other hand, can introduce many more vertices depending on its parameters which could result in significantly longer computation times if not properly tuned. Parameters used for the cracking simulation of the models are shown in Table 3. For a more detailed description of the cracking parameters see [IO06].

7. Conclusions

We have introduced a framework for user-controlled woodification of polygonal meshes. A polygonal mesh is converted to a tetrahedral mesh and is evolved by using deformable simplicial complexes. The mesh development can be controlled by a biologically-motivated growth function based on the Weibull function or by a user-defined growth function sketched on the mesh surface. The evolving mesh can collide with objects in the scene and with itself while geometrical and topological changes are consistently maintained. The growth generates stress on the surface that is alleviated by

Figure 12: *Images from the sequence of two evolving objects with their initial mesh on the left and the final evolved mesh on the right.*

Figure 13: *A tree growing around a fence obstacle at different evolution steps. The model adapts to the existing obstacle.*

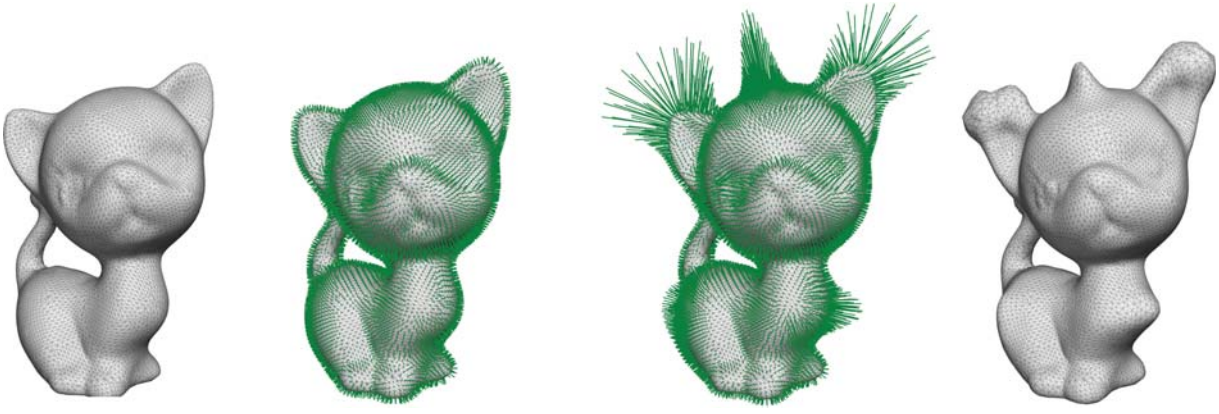

Figure 14: *User-defined growth: Left: The initial mesh and a uniform growth function, indicated by the uniform length of the normal vectors. Right: the user altered the growth function, which causes a significant distortion of the mesh during growth.*

Figure 15: *Growth sequences of complex geometries.*

cracks and the cracks can also be painted on the surface by modifying the stress map. We have shown that our method is capable of growing objects around obstacles, creating complex woodified geometries, predicting growth rings, and reproducing different bark structures. These tools also support interactive control to enable artists to intuitively influence these processes. One of the main limitations of this method is that the growth and cracking parameters depend on mesh resolution. A finer mesh requires a different set of parameters than a coarser mesh to produce similar results. The transfer of cracks between meshes of different simulation steps can also create errors due to local remeshing operations. This is especially conspicuous when using an uneven growth speed over the mesh.

Another limitation is the time needed for robust DSC growth, and so an obvious avenue for future work is optimizing the DSC method. The retesselation that occurs as each vertex propagates through a tetrahedral mesh face makes the approach robust, but also time consuming.

The DSC also includes an internal tetrahedral mesh that a future simulation could use to simulate interior cracking.

An interesting problem to explore would be incorporating different surface features, such as knots, defects, or cellular-based surface features such as moss or lichen to the simulation. Our cambial growth is always in the normal direction, but a more detailed simulation would represent the orientation of the fusiform cells, to more accurately simulate cambial growth around obstacles and wounds.

References

[Bær14] BÆRENTZEN J. A.: DSC - https://github.com/janba, 2014. URL: https://github.com/janba. 8

[Bai23] BAILEY I.: The cambium and its derivative tissues. IV. The increase in girth of the cambium. *Amer. J. Botany 10*, 9 (1923), 499–509. 3

[Blo85] BLOOMENTHAL J.: Modeling the mighty maple. *SIGGRAPH Comput. Graph. 19*, 3 (1985), 305–311. URL: http://doi.acm.org/10.1145/325165.325249, doi:10.1145/325165.325249. 2

[BSFG09] BARNES C., SHECHTMAN E., FINKELSTEIN A., GOLDMAN D. B.: Patchmatch: A randomized correspondence algorithm for structural image editing. *ACM Trans. Graph. 28*, 3 (2009), 24:1–24:11. 7

[Buc98] BUCHANAN J. W.: Simulating wood using a voxel approach. *Comp. Graph. Forum 17*, 3 (1998), 105–112. 3

[Chr14] CHRISTENSEN A.: DSC3D - github.com/asny/dsc, 2014. URL: github.com/asny/DSC. 8

[CMF*10] COLIN F., MOTHE F., FREYBURGER C., MORISSET J.-B., LEBAN J.-M., FONTAINE F.: Tracking rameal traces in sessile oak trunks with x-ray computer tomography. *Trees 24*, 5 (2010), 953–967. URL: http://dx.doi.org/10.1007/s00468-010-0466-1, doi:10.1007/s00468-010-0466-1. 3

[CN06] COMBAZ J., NEYRET F.: Semi-interactive morphogenesis. In *Shape Modeling and Applications, 2006. SMI 2006. IEEE International Conference on* (June 2006), pp. 35–35. doi:10.1109/SMI.2006.35. 3, 4

[CSM03] COHEN-STEINER D., MORVAN J.-M.: Restricted delaunay triangulations and normal cycle. In *Proc. of the Nineteenth Annual Symp. on Computational Geometry* (New York, NY, USA, 2003), SCG '03, ACM, pp. 312–321. URL: http://doi.acm.org/10.1145/777792.777839, doi:10.1145/777792.777839. 7

[EMB11] ERLEBEN K., MISZTAL M., BÆRENTZEN J.: *Mathematical foundation of the optimization-based fluid animation method*. ACM, 2011, pp. 101–110. doi:10.1145/2019406.2019420. 5, 8

[FP96] FEDERL P., PRUSINKIEWICZ P.: A texture model for cracked surfaces with an application to tree bark. *Proc. Western Comp. Graph. Symp.* (1996), 23–29. 2

[FP02] FEDERL P., PRUSINKIEWICZ P.: Modelling fracture formation in bi-layered materials with applications to tree bark and drying mud. *Proc. Western Comp. Graph. Symp.* (2002), 29–35. 2

[FP04] FEDERL P., PRUSINKIEWICZ P.: Finite element model of fracture formation on growing surfaces. *Proc. ICCS, (LNCS 3037)* (2004), 138–145. 2

[GMW04] GALBRAITH C., MÜNDERMANN L., WYVILL B.: Implicit visualization and inverse modeling of growing trees. *Comp. Graph. Forum 23*, 3 (2004), 351–360. URL: http://dx.doi.org/10.1111/j.1467-8659.2004.00766.x, doi:10.1111/j.1467-8659.2004.00766.x. 2

[Har97] HART J. C.: Implicit representations of rough surfaces. *Comp. Graph. Forum 16*, 2 (1997), 91–99. 2

[HB96] HART J. C., BAKER B.: Implicit modeling of tree surfaces. *Proc. Implicit Surfaces* (1996), 143–153. 2

[HNB*06] HOUSTON B., NIELSEN M. B., BATTY C., NILSSON O., MUSETH K.: Hierarchical RLE level set: A compact and versatile deformable surface representation. *ACM Trans. Graph. 25*, 1 (2006), 151–175. URL: http://doi.acm.org/10.1145/1122501.1122508, doi:10.1145/1122501.1122508. 3

[HTK98] HIROTA K., TANOUE Y., KANEKO T.: Generation of crack patterns with a physical model. *The Visual Computer 14*, 3 (1998), 126–137. URL: http://dx.doi.org/10.1007/s003710050128, doi:10.1007/s003710050128. 2

[IO06] IBEN H. N., O'BRIEN J. F.: Generating surface crack patterns. In *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation* (Sept 2006), pp. 177–185. URL: http://graphics.cs.berkeley.edu/papers/Iben-GSC-2006-09/. 3, 6, 7, 8

[Jia07] JIAO X.: Face offsetting: A unified framework for explicit moving interfaces. *J. Computational Physics 220*, 2 (2007), 612–625. 3

[KB93] KIMMEL R., BRUCKSTEIN A.: Shape offsets via level sets. *CAD 25*, 3 (1993), 154–161. 3

[Lar94] LARSON P.: *The vascular cambium: Development and structure*. Springer series in wood science. Springer-Verlag, 1994. 3

[LN02] LEFEBVRE S., NEYRET F.: Synthesizing bark. In *Proceedings of the 13th Eurographics Workshop on Rendering* (2002), EGRW '02, Eurographics Association, pp. 105–116. URL: http://dl.acm.org/citation.cfm?id=581896.581911. 2, 6

[LS11] LANDSBERG J., SANDS P.: Chapter 4 - stand structure and dynamics. In *Physiological Ecology of Forest Production*, vol. 4 of *Terrestrial Ecology*. Elsevier, 2011, pp. 81 – 114. doi:10.1016/B978-0-12-374460-9.00004-4. 4

[Mat91] MATTHECK C.: *Trees: The Mechanical Design*. Springer, 1991. 3

[Mat02] MATTHEWS M. J.: *Physically Based Simulation of Growing Surfaces*. Master's thesis, University of Calgary, Canada, 2002. 3

[MB12] MISZTAL M. K., BÆRENTZEN J. A.: Topology-adaptive interface tracking using the deformable simplicial complex. *ACM Trans. Graph. 31*, 3 (2012), 24:1–24:12. URL: http://doi.acm.org/10.1145/2167076.2167082, doi:10.1145/2167076.2167082. 2, 3

[MBH03] MICHAELRAJ J., BAKER B., HART J. C.: Structural simulation of tree growth. *The Visual Computer 19*, 2-3 (2003), 151–163. 5

[MEB*12] MISZTAL M., ERLEBEN K., BARGTEIL A., FURSUND J., CHRISTENSEN B., BÆRENTZEN J., BRIDSON R.: *Multiphase flow of immiscible fluids on unstructured moving meshes*. Eurographics Association, 2012. 5, 8

[Mis10] MISZTAL M. K.: *Deformable Simplicial Complex*. PhD thesis, Technical University of Denmark, 2010. 2, 3, 8

[MM11] MIZOGUCHI A., MIYATA K.: Modeling trees with rugged surfaces. In *Trust, Security and Privacy in Computing and Communications (TrustCom), 2011 IEEE 10th International Conference on* (Nov 2011), pp. 1464–1471. 3

[MP96] MĚCH R., PRUSINKIEWICZ P.: Visual models of plants interacting with their environment. In *SIGGRAPH '96: Proc. of the 23rd annual conference on Comp. graphics and interactive techniques* (New York, NY, USA, 1996), ACM, pp. 397–410. doi:http://doi.acm.org/10.1145/237170.237279. 1

[MPW07] MANN J., PLANK M., WILKINS A.: Tree growth and wood formation — Applications of anisotropic surface growth. *Proc. Math in Industry* (2007), 153–192. 3, 4

[OF02] OSHER S. J., FEDKIW R. P.: *Level Set Methods and Dynamic Implicit Surfaces*. Springer-Verlag, New York, 2002. 3

[OH99] O'BRIEN J. F., HODGINS J. K.: Graphical modeling and animation of brittle fracture. In *Proceedings of the 26th Annual Conference on Computer Graphics and Interactive Techniques* (1999), SIGGRAPH '99, pp. 137–146. URL: http://dx.doi.org/10.1145/311535.311550, doi:10.1145/311535.311550. 3

[Opp86] OPPENHEIMER P. E.: Real time design and animation of fractal plants and trees. *SIGGRAPH Comput. Graph. 20*, 4 (1986), 55–64. URL: http://doi.acm.org/10.1145/15886.15892, doi:10.1145/15886.15892. 2

[Ped95] PEDERSEN H. K.: Decorating implicit surfaces. In *Proceedings of the 22Nd Annual Conference on Comp. Graph. and Interactive Techniques* (1995), SIGGRAPH '95, ACM, pp. 291–300. doi:10.1145/218380.218458. 3

[PHL*09] PALUBICKI W., HOREL K., LONGAY S., RUNIONS A., LANE B., MĚCH R., PRUSINKIEWICZ P.: Self-organizing tree models for image synthesis. *ACM Trans. Graph. 28*, 3 (2009), 1–10. doi:http://doi.acm.org/10.1145/1531326.1531364. 1

[PKA*05] PAULY M., KEISER R., ADAMS B., DUTRÉ P., GROSS M., GUIBAS L. J.: Meshless animation of fracturing solids. *ACM Trans. Graph. 24*, 3 (2005), 957–964. doi:10.1145/1073204.1073296. 3

[PL91] PRUSINKIEWICZ P., LINDENMAYER A.: *The Algorithmic Beauty of Plants*. Springer, 1991. 1

[PNdJO14] PFAFF T., NARAIN R., DE JOYA J. M., O'BRIEN J. F.: Adaptive tearing and cracking of thin sheets. *ACM Trans. Graph. 33*, 4 (July 2014), 110:1–110:9. URL: http://doi.acm.org/10.1145/2601097.2601132, doi:10.1145/2601097.2601132. 2

[PO09] PARKER E. G., O'BRIEN J. F.: Real-time deformation and fracture in a game environment. In *Proc. of the ACM SIGGRAPH/Eurographics Symposium on Comp. Animation* (Aug. 2009), pp. 156–166. URL: http://graphics.berkeley.edu/papers/Parker-RTD-2009-08/. 3

[PR95] PAN Y., RAYNAL D. J.: Decomposing tree annual volume increments and constructing a system dynamic model of tree growth. *Ecological Modelling 82*, 3 (1995), 299 – 312. URL: http://www.sciencedirect.com/science/article/pii/030438009400096Z, doi:10.1016/0304-3800(94)00096-Z. 3

[REN*04] RASMUSSEN N., ENRIGHT D., NGUYEN D., MARINO S., SUMNER N., GEIGER W., HOON S., FEDKIW R.: Directable photorealistic liquids. *Proc. SCA* (2004), 193–202. 3

[SDF07] SIFAKIS E., DER K. G., FEDKIW R.: Arbitrary cutting of deformable tetrahedralized objects. In *Proceedings SCA* (2007), SCA '07, Eurographics Association, pp. 73–80. URL: http://dl.acm.org/citation.cfm?id=1272690.1272701. 3

[SDPLV06] STEPPE K., DE PAUW D., LEMEUR R., VANROLLEGHEM P. A.: A mathematical model linking tree sap flow dynamics to daily stem diameter fluctuations and radial stem growth. In *Tree Physiol* (2006), 26(3), pp. 257–73. 3

[Set99] SETHIAN J. A.: *Level Set Methods and Fast Marching Methods*. Cambridge University Press, 1999. 3, 5

[Si07] SI H.: TetGen. Tetrahedral mesh generator and three-dimensional delaunay triangulator., 2007. URL: http://tetgen.berlios.de. 4

[SPH11] SELLIER D., PLANK M. J., HARRINGTON J. J.: A mathematical framework for modelling cambial surface evolution using a level set method. *Annals of Botany 108*, 6 (2011), 1001–1011. doi:10.1093/aob/mcr067. 3

[Tho42] THOMPSON D. W.: *On Growth and Form: A New Edition*. Cambridge U. Press, 1942. 1

[Wil84] WILSON B. F.: *The Growing Tree*. U. Mass. Press, 1984. 3

[WWL*03] WANG X., WANG L., LIU L., HU S., GUO B.: Interactive modeling of tree bark. In *Proceedings of the 11th Pacific Conference on Computer Graphics and Applications* (Washington, DC, USA, 2003), PG '03, IEEE Computer Society, pp. 83–. URL: http://dl.acm.org/citation.cfm?id=946250.946979. 2

[WWM87] WYVILL G., WYVILL B., MCPHEETERS C.: Solid texturing of soft objects. *Proc CGI* (1987), 129–141. 3