Spring 2014

Woodification of polygonal meshes

Gustavo Alejandro Guayaquiul Sosa
Purdue University

Graduate School ETD Form 9
(Revised 01/14)

PURDUE UNIVERSITY
GRADUATE SCHOOL
Thesis/Dissertation Acceptance

This is to certify that the thesis/dissertation prepared

By GUSTAVO ALEJANDRO GUAYAQUIL SOSA

Entitled

WOODIFICATION OF POLYGONAL MESHES

For the degree of Master of Science

Is approved by the final examining committee:

Bedrich Benes	
Xavier M. Tricoche	
Yingjie Chen	

To the best of my knowledge and as understood by the student in the *Thesis/Dissertation Agreement. Publication Delay, and Certification/Disclaimer (Graduate School Form 32)*, this thesis/dissertation adheres to the provisions of Purdue University's "Policy on Integrity in Research" and the use of copyrighted material.

Approved by Major Professor(s): Bedrich Benes

Approved by: Patrick E. Connolly 04/18/2014

Head of the Department Graduate Program Date

WOODIFICATION OF POLYGONAL MESHES

A Thesis

Submitted to the Faculty

of

Purdue University

by

Gustavo Alejandro Guayaquil Sosa

In Partial Fulfillment of the

Requirements for the Degree

of

Master of Science

May 2014

Purdue University

West Lafayette, Indiana

Dedicated to my family.

“Para Candy, Sofia y Elena. Voy a recordarlas con mucho amor mis pequeñas.”

ACKNOWLEDGMENTS

I would like to thank the *High Performance Computer Graphics Laboratory*, Hansoo, Illia, Innfarn, Jay, Jorge, Juraj, Marek, Michel, and its leader, Dr. Bedrich Benes. They have been my family the last three years making my stay at Purdue something I will remember and talk about with affection.

To Bedrich, who showed me a different type of advisor, the one who is in constant care of students and let them to figure out the solution of problems, providing excellent (and sometimes more) guidance. I will be in debt with him and I hope to continue the colleague relationship we have.

The final application was made on top of the graphics framework of Dr. Soeren Pirk, who was the bridge between the programmers and the professors. He, and part of the computer graphics group of Kontstanz University, Julian Kratt, Marc Spicker, and Dr. Oliver Deussen, were very helpful implementing and solving cracking part. They also add a lot of ideas during meetings.

Dr. John C. Hart was a key member pointing to the entire group the underlying data structure for the growth model. He adds ideas and possible results to be obtained once the system was implemented. The Figure 5.1(a) is his work with the data set provided from the simulation of the thesis.

I have to acknowledge to Marek Fiser for the help in the development, theoretical and practical, of the code associated to the growth model. Half of the tools and images were made in conjunction with him. A funny case is the sketch tool which was used to create the *kitty punks* of Chapter 5.

There are three other people I would also like to express gratitude. Maria Yakuskina, Michel Abdul, and Mao Konishi. Maria, it is nice to meet people outside of the geeky-nerd world of a programmer-scientist, Michel, it is true you had become one of the most reliable person one can meet, and Mao, thanks for the time we spent together. Michel and Mao, thanks both for helping me reading and fixing the grammar.

Finally I wish to gratefully acknowledge my thesis committee for their insightful comments and paths to follow regarding this thesis.

PREFACE

The following thesis is an extension of the paper submission *Woodification: User-Controlled Cambial Growth Modeling*. The so-called *Deformable Simplicial Complex* (DSC) for evolving meshes, (Christiansen, Nobel-Jrgensen, Aage, Sigmund, & Brentzen, 2013), is a novel approach which removes some implementation issues, such as trade-off of accuracy for the Semi-Lagrangian method or precise reconstruction of meshes via Marching-Cubes (Lorensen & Cline, 1987), when an advecting Level Set is used. Although DSC brings its own problems (such as time consuming simulation), it is used as underlying representation of a polygonal mesh. With the mesh in hand, a biologically-growth model to evolve the surface is proposed. The DSC representation allows to have a temporal coherence of the evolving mesh and an automatic subdivision to increase the mesh resolution in defined regions, such as the locality of a collision with obstacles. The grown mesh is coupled with a cracking simulation based on tensors, defining a stress threshold and a relaxation method to compile different cracks behavior. A refresh in manipulating indexes to represent linear transformations is recommended for the cracking part of the present work. Lastly, the results are, in the majority of cases, graphical output through the OpenGL Application Programmable Interface (API). It is advised to check the article for DSC, O'Brien and Hodgins (1999), and a book regarding OpenGL, for example Angel and Shreiner (2011).

TABLE OF CONTENTS

LIST OF FIGURES

ABSTRACT

Guayaquil Sosa, Gustavo Alejandro M.S., Purdue University, May 2014. Woodification of polygonal meshes. Major Professor: Bedrich Benes.

An evolving polygonal mesh based on stem's tree growth coupled with a physical simulation of bark's cracking is presented. This process is denominated *woodification*. Whereas previous approaches use a fixed resolution voxel grid, woodification is built on the deformable simplicial complex representation, which robustly simulates growth with adaptive subdivision. The approach allows any meshed object to be grown and textured. Features, such as interaction with obstacles, attributes interpolation, and sketching tools, are added to provide control during the woodifible process.

CHAPTER 1. INTRODUCTION

Defining surface characteristics and material properties is an inherently important problem in Computer Graphics research. Due to the steadily increasing demand for realism this has been addressed by the Computer Graphics community with a large quantity of methods over the last two decades. While early approaches focused on modeling materials based on assumptions and simplifications, modern techniques more closely mimic the actual characteristics of materials at different scales using science-based approaches. A topic covering such stages is the evolution of a surface mesh, where initially the direction of growth was decided to be based in the surface normal whereas nowadays such direction is defined procedural with rules from observations. The evolving nature of growing objects make them hard to simulate, conveying special problems to reproduce material properties. An example of growth based on a surface mesh is the upgrowth of a tree.

The growth of trees and the development of branching structures has been addressed by a multitude of approaches over the years (Deussen & Lintermann, 2004; Edelstein-Keshet, 1988; Kaandorp, 1994; J. Landsberg & Sands, 2011b). However, apart from the faithful generation of branching systems, the surface behavior and appearance plays an important role for the realistic rendering of trees. A number of techniques have addressed the modeling of bark structures and patterns (Kopf et al., 2007; Lefebvre & Neyret, 2002a; Mann, Plank, & Wilkins, 2006; Wang, Wang, Liu, Hu, & Guo, 2003). However, only a limited set of techniques approach the lateral growth behavior.

Several methods exist that allow modeling fractures and cracks for rigid bodies (Hirota, Tanoue, & Kaneko, 1998; O'Brien & Hodgins, 1999). A large set of these techniques focuses on visualizing physically correct cracking to lower the burden on content creators when modeling or animating these effects. These techniques are predominantly known as *meshfree* (Rabczuk & Belytschko, 2004) or as *meshless* (Pauly et al., 2005) methods. They offer many features that make them well-suited for modeling fractures and cracks. However, most of the existing techniques focus on cracking rigid bodies. To the best of the author's knowledge, there is no method that addresses the cracking of living objects or their surfaces. So far, the modeling of bark and the simulation of cracking of solid objects have mostly been treated separately. Both domains focus on deforming and manipulating surface meshes. However, due to complicated processes within the entire volume of an object, the growth of wood and the development of bark cracking patterns need to be addressed jointly.

The proposed model is an interactive approach for growing arbitrary polygonal meshes in a wood-like fashion, i.e., the transformation of an object appearance by emulating girth growth over the surface.

The following are the contributions of this thesis:

1. Usage of detailed growth functions encoding botanical process.
2. A new cracking model based on stress propagation with feedback from the lateral growth
3. The introduction of the term *woodification* as the process of converting water-tight triangular meshes to a grown appearance with bulge and crack features

In addition to the above contributions the work incorporates collision detection with obstacles and a sketching tool for manipulating the direction of the growth and/or the cracking generation.

1.1 Scope

A three step process is used to *woodify* a polygonal mesh:

1. Drive the evolution of the surface by a biologically-motivated function.

2. Check surface information by the Deformable Simplicial Complex tetrahedral mesh (tet-mesh) representation.

3. Generate cracks through a physical process of stress and tension.

The thesis does not consider branching as a simulated feature, and misses the comparison with botanical tables of different species of trees. Also, the transformation from a triangular mesh to a tet-mesh is based on the third-party library TetGen (Si, 2013). A list of tet-meshes compatible with DSC and used during the experiments can be downloaded from DSCCompatibleMeshes (HPCG, 2014).

Lastly, the evaluation will consist on providing a plausible visual appearance of bulge and cracked wood with comparison of photos with images generated by the simulation.

1.2 Significance

Modeling trees and *surface modeling* are two relevant topics to Computer Graphics . The first has been researched during the last decade (Deussen et al., 1998) using biologically-based (Chiba, 1990b) or procedural (Chen, Neubert, Xu, Deussen, & Kang, 2008; Stava et al., 2014) techniques, including the study of lateral growth, branching, cracks formation, interactions with objects, and others. The second is a common method for visualizing three-dimensional objects in Computer Graphics (Lorensen & Cline, 1987). The representation varies from explicit equations, such a parametrized sphere or patches constructions, to implicit forms, such as the Level Set formulation. Recently the DSC method has been introduced to address the surface manipulation (Misztal & Bærentzen, 2012).

Combining a tree lateral growth, a force-based cracking, and the DSC method has not been used to simulate growth of polygonal meshes. Furthermore, merging these pieces provides a technique to define the process of transform meshes to resemble wood.

1.3 Research Question

We would like to answer if using an evolving mesh method, DSC, driven by a biologically motivated equation and coupled with a cracking technique is possible to transform a polygonal mesh to resemble wood.

1.4 Assumptions

The assumptions for this study include:

1. The biologically-motivated growth function derived from a probabilistic distribution function is correct and resemble the expansion of a stem's tree.
2. Any water-tight polygonal mesh can be converted to a tet-mesh representation.

1.5 Limitations

The limitations for this study include:

1. The growth direction is defined per triangle and quasi-constant to have computational performance (using the up direction to consider some preferred orientation) losing control over the symmetry of the simulation.
2. No external agents, such as weather or human interaction, are considered during the simulations.

1.6 Delimitations

The delimitations for this study include:

1. The number of vertices of an object is proportional to the memory machine used for the simulation. Below hundred thousand a current computer with a modern graphics card can perform the simulation in real time.

2. The values in the growth speed or relaxation time step are in the range $[0, 0.1]$ to avoid inaccuracy, due floating point machine precision, of the numerical methods used.

3. No branching growth of a tree.

1.7 Thesis Overview

The present document is divided as follow: A paragraph introducing the proposal and key observations of the work is presented at the end of this chapter. Next, relevant previous work regarding *Biologically growth*, *Bark*, and *Crack-Fracture* modeling is presented. Also, in Chapter 2 some concepts respecting the manipulation of polygonal meshes in a computer, giving aperture to precise definition of a DSC algorithm, are introduced. Chapter 3 shows the study design, the unit and sampling, and the pipeline for the thesis. Chapter 4 covers in detail the necessary equations and physics assumptions to reproduce the simulations. The results, subdivided as lateral growth and surface cracking, are shown in Chapter 5. This chapter also covers the difficulties experienced during the simulations. Lastly, Chapter 6 contains conclusion, the possible future worked, and the lessons learned combining science-based techniques in Computer Graphics.

1.8 Summary

A method is presented for transforming polygonal meshes into a tree-looking shapes with wood appearance (*woodify*). The process is done by transforming the original shape to a tree-looking outline which has bulge and crack features. The original mesh is transformed to a tet-mesh for mimicking lateral growth using the DSC method. We utilize a probabilistic distribution function, commonly known in the field of theoretical biology, to approximate the object growth. Other growth functions are also introduced by modifying some aspects of the distribution, such as preferred direction of growth. The evolving surface is coupled with a forced-based cracking technique to generate bark. The DSC and cracking are extended by adding attributes directly to the surface evolution, such as texturing wrapping, local speed adjustment, and interaction with solid objects providing flexibility to define material properties, cracking patterns, and growth rate.

CHAPTER 2. PREVIOUS WORK

This chapter discusses previous work of different methods for growing trees in Computer Graphics. Special attributes to simulates over a mesh, such as bark and cracking, are also covered.

2.1 Tree Growth

A desired feature of a tree growth simulation is to provide enough control while a natural aspect is maintained. The usage of equations coupled with sketching tools allows to fulfill these characteristics as shown by Lawrence and Funkhouser (2003). With the idea of providing an explanation to the related work, the stem's growth of a tree can be consulted in the literature as *biologically-based* or *procedurally-based*. Additionally, an extensive description of different equations and tools of growth methods can be consulted in Amar, Goriely, and Müller (2011) or Mann et al. (2006).

2.1.1 Biologically-based approaches

A tree's physiology is defined as *"the facts related to the dimensions of a tree"* (Huber, 1956), such as *metabolism*, *growth*, and *reproduction*. An important result regarding the stem's growth says that an asymmetrical curve becomes symmetrical as time increases. One of such curve is generated by the lateral growth variable from the *forest growth* model (Dale, Doyle, & Shugart, 1985), which depends on a collection of parameters to assess an individual production. The forest model does not consider species characteristics or environment interaction.

Chiba (1990b) uses the statistical model of Oohata and Shinozaki (1979) to derive an equation for the lateral growth. For a given cutting plane z, he defines the stem density as a function of tree's height, denoted by $S(z)$, and computes the lateral growth by assuming a power function relationship between the total weight and $S(z)$, Figure 2.1. His solution is a linear combination of exponential functions outlining fast increment of the stem at lower heights. The model also suggests a relationship for branching development. An extension of the model, taking into account ramification and crown evolution, was later presented in Chiba (1990a, 1991) and summarized by Chiba and Shinozaki (1994) as a linear partial differential equation for the relationship between the stem's increment and its density.

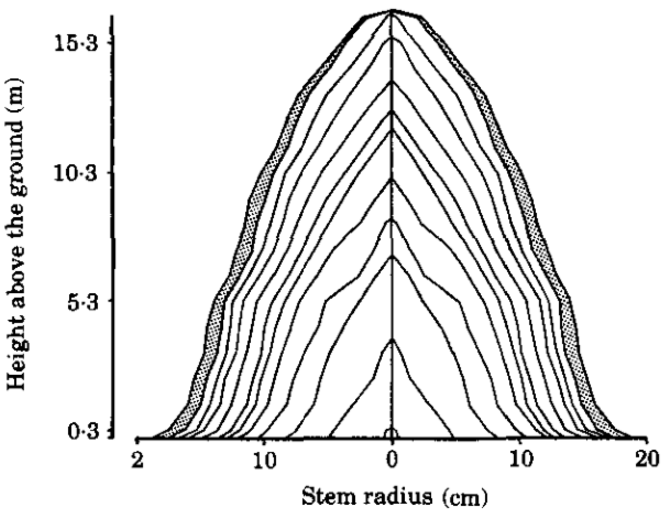


Figure 2.1. Profile curves, of stem's growth, every five years. Image from Chiba and Shinozaki (1994)

Tree's dynamics can be studied through a biological mechanism, such as *photosynthesis* or *water consumption*, a chemical processes such as *carbon balance*, a geometrical assumption such as *stand structure*, a physical equation such as *mechanical models of stem*, or by statistical models such as *height-diameter relation distribution* (J. J. Landsberg, 2011). One of the models related to stem's growth is the so-called *Weibull distribution* (Weibull, 1951), which combined with Chiba's derivation of stem growth is the biologically-motivated equation used for our mesh growth. Other biologically-based approaches are: a *dynamic flow* and *storage model* shown in Steppe, De Pauw, Lemeur, and Vanrolleghem (2006), a combination of an architectural model with a forest model driven by a level set (Sellier, Plank, & Harrington, 2011).

2.1.2 Procedurally-based approaches

A common approach for simulating a tree is the so-called *L-system* (Lindenmayer, 1968), Figure 2.2. The procedure can be coupled with a terrain generator and ecosystem rules to recreate diverse landscapes as demonstrated by Deussen et al. (1998). An *L-system* for trees has been also extended to consider environmental conditions such as *weight support* or *space tropism* (Lam & King, 2005).

A more naive approach is looking of lateral growth as an offsetting operation controlled by defining the speed function over the surface (Buchanan, 1998), and can be extended by coupling with an L-system as demonstrated by Sellier et al. (2011).

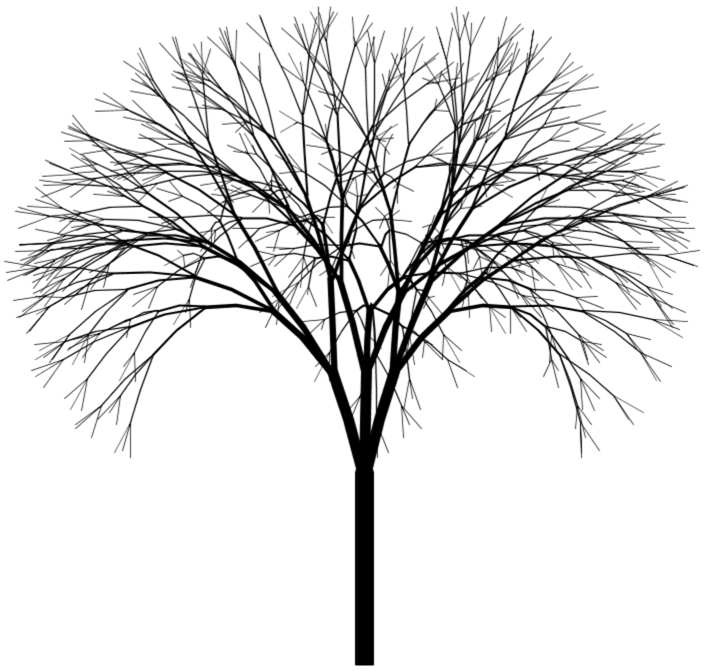


Figure 2.2. Example of a tree branching structure generated by an L-system. Image obtained from the software Malsys (Fišer, 2012).

Other authors have used *fractals* to model trees (Oppenheimer, 1986; Zhou, Chen, Liu, Li, & Rao, 2010). Fractals have the property of sensitivity to their initial parameter values but they provide straightforward simulations for two-dimensional and three-dimensional fractal trees (Kaandorp, 1994). A third way is to use an inverse modeling technique to describe plant organs in terms of their global position with respect to the tree (Galbraith, Mndermann, & Wyvill, 2004). Alternative methods are: converting a sketch into a 3D surface model by a *Markov* random field (Chen et al., 2008), tracking methodology, for branch knots and lateral variations, based on computer scanned images (Colin et al., 2010), space colonization algorithm (Runions, Lane, & Prusinkiewicz, 2007), and branch inference based on tree silhouette (Wither, Boudon, Cani, & Godin, 2009).

2.2 Bark Modeling

The simulation of bark has also evolved from initial heuristic approaches to a more botanical techniques.

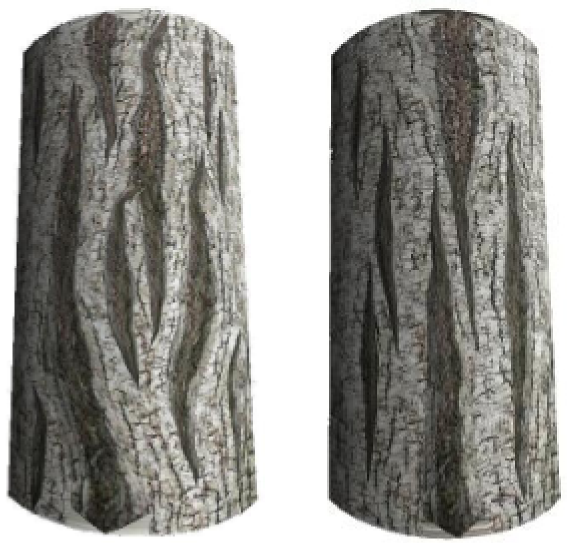


Figure 2.3. Two instances of simulation of bark. Image from Lefebvre and Neyret (2002b)

Bloomenthal (1985) uses a parametric texture model for the branching of a tree while Hart and Baker (1996) extended using particles to track the flow from the trunk to the ramifications. A method using horizontal strips with procedural fractures is shown by Lefebvre and Neyret (2002b), Figure 2.3.

Along the texturing map a variety of procedural, and implicit synthesis, methods have been used to generate bark in a geometrical fashion to naturally jointly to a branching structure. Approaches based on spring-mass model, finite element method, and layer-based technique are used, (Federl & Prusinkiewicz, 2004), to capture differential over stress and strain of a tree. Coupled with a surface representation a bark is shown as an extra attribute of a texture. Kirota, Kato, and Kaneko (1998) also uses a spring-mass model for the bark modeling.

2.3 Cracking

Cracking is an important aspect to represent the destruction of natural and artificial objects. The general approach is to use equations to describe the *stress* and *strain* of a material. Previous models are based on a *finite element method* (Federl & Prusinkiewicz, 2002; Glondu et al., 2012; O'Brien & Hodgins, 1999), a *spring network technique* (Hirota et al., 1998; Hirota, Tanoue, & Kaneko, 2000), or *cellular automata* (Gobron & Chiba, 2001).

One of the first models simulated the connection between vertexes as shear-springs (Hirota et al., 1998). A change in the stress between two different points generates a fracture. The model uses two layers for representing a material, the exterior having all the possible fractures and the interior plain. In this method, the pipeline used is: (1) set up parameters, (2) shrink or enlarge a spring, (3) compute forces checking values in nodes, (4) translate nodes, and (5) cut springs. The same authors later presented an extension for three-dimensional objects (Hirota et al., 2000), considering an adaptive subdivision of the spring to solve jaggedness in the results.

Paquette, Poulin, and Drettakis (2002) show weathered results such as cracking and peeling of layers by defining paint strength and tensile stress. The first computes the adhesion between layers in order to determine the curl of a peeling, while the second considers elasticity factors to generate a crack. The simulation is summarized as: (1) new crack, (2) propagation, (3) relaxation, and (4) adhesion. The data representation is similar to the spring simulation using two layer definition: a base layer and a paint layer.

Gobron and Chiba (2001) employ a spectrum stress model to simulate cracks over surfaces. They define a crack as a result of internal stresses becoming greater than the material resistance and the stress as a tension in a predefined direction. Depending on the angle between the tension and the principal axis of the object, the stress has aliases such as *compression*, *expansion*, *torsion*, etc. The object's geometry restricted to a surface made the authors consider only compression and expansion.

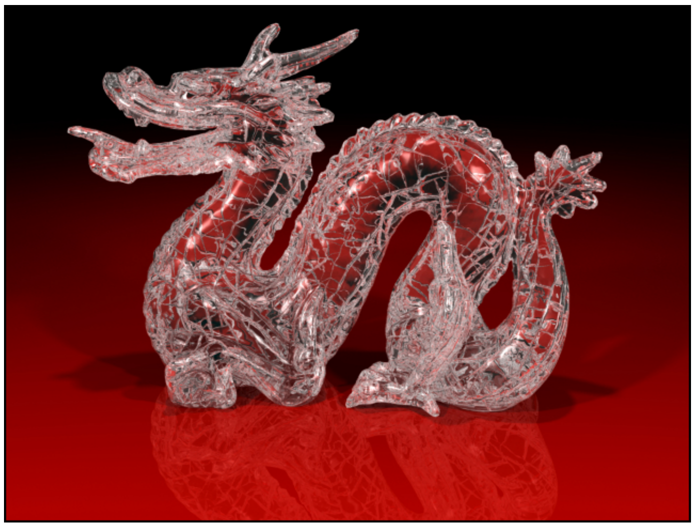


Figure 2.4. Glass dragon with a cracking model applied on the surface. Image from Iben and O'Brien (2006)

Cracks over surface models using a *stress field* are generated by Iben and O'Brien (2006), Figure 2.4. The computation of the field is defined heuristically and applied to each node of the triangulated surface. The steps of the algorithm are: (1)

initialization of the field, (2) compute the failure criteria, (3) during a failure: (a) crack the mesh, (b) evolve stress field, (c) update mesh, (4) display cracks by direct rendering of edges or by doing a filling representation. The stress tensor can be defined as zero, representing an equilibrium surface, store constant tension, simulating a drying or shrinking, being filled according to the curvature of the mesh, or using random noise to model material inhomogeneities. To determine the direction of the crack, the authors decompose the forces into the tensile part and the compressive part, building a *separation tensor* whose *eigenvalue* determines the existence of a crack. The associated *eigenvector* provides the plane for the crack.

Using a mesh to characterize the stress motivates this thesis to use in conjugation with the growing-bark modeling. The connection can be done by different techniques such as voxelization defining properties to carry in each voxel, level set to define an isosurface, or directly to the triangular mesh. An extension of the latest if the novel DSC method.

2.4 Deformable Simplicial Complex

Moving vertices along normals provide advantages of non-reconstruction of surface and immediate overload of properties to each vertex. However an important issue of topology change, when two connected component intersect, arises. To address this Christiansen et al. (2013) extend the explicit moving of vertices by introducing a tet-mesh representation of a surface. Maintaining volumetric information to resolve topology conflicts such as collisions or unstitching.

Also, the DSC underlying representation maintains the triangular information making it suitable to use *Finite Element Method* as one attribute.

Other usages are fluid simulation Misztal, Bridson, Erleben, Bærentzen, and Anton (2010) and topology optimization Maute and Ramm (1995).

2.5 Summary

This chapter provided a review of the literature relevant to growing trees and their cracking and bark modeling attributes. The next chapter provides the framework and methodology to be used in the development of an implementation for a simulation of the woodification process.

CHAPTER 3. FRAMEWORK AND METHODOLOGY

3.1 Study Design

The present study is about transforming polygonal meshes into a tree-looking objects. To perform the transformation, three steps are proposed: (1) Development of a biologically-motivated equation which will provide an evolving mechanism for the mesh, (2) Representation of the propagation by the DSC method, and (3) Application of a cracking and bark model to emulate a realistic surface appearance of a tree. The coupling of each step is derived as interrelated attributes of the mesh.

The key observations of the proposed approach are:

1. An equation for the growth resembling biological results can be derived from a probability distribution function.
2. The equation can be added to the direction of growth of a mesh.
3. Coupling a force-based cracking and bark texturing.

3.2 Unit & Sampling

In the following sections the hypothesis, population, sample, variables, and the evaluation, are discussed.

3.2.1 Hypothesis

The simulation of wood objects through the coupling of a biologically-motivated growth equation and a physics-based cracking model in a surface propagation algorithm produce results of good visual quality

3.2.2 Population

The population are polygonal meshes with sufficient properties to be represented as a tet-mesh to be fed to the proposed pipeline, Figure 3.1. The quantification is done showing the number of vertices at the beginning and at the end of the simulation. Also, the number of iterations required for different meshes. The qualification is provided by the visual feedback of visually plausible cracks in geometry and texture (bark).

3.2.3 Sample

The techniques to be used are:

1. A biologically-based process to simulate lateral growth by the so-called Weibull distribution (expanded in the next chapter) as a structural growth to represent eccentricity, a reversed growth, a user defined-growth by a sketching tool, and interaction around obstacles.

2. A DSC representation for geometrical representation at each step of the simulation.

3. A stress-based crack by storing tensors in each triangle of the geometrical representation.

3.2.4 Variables

The set of variables can be divided into:

1. The biologic process: *magnitude of the underlying velocity field for the growth* bounded to be less than one, *eccentricity factor* in the opposite direction of gravity, *shape parameter*, for the Weibull distribution, with value equal to two, *minimum length edge* for a subdivision of the mesh during the growth to be less than one.

2. Mesh representation: *water-tight* polygonal mesh and *tetrahedralized* for volume information, *vertex-attributes* to define growth direction and texture mapping.

3. Cracking-bark modeling: *stress direction* as horizontal, vertical, or loaded by a predefined path, *relaxation steps* for the emergence of a crack in the range of

one to ten, *vertex-id-attribute* for coupling with the texture map and grow direction making the crack evolve coherently.

3.2.5 Evaluation

The thesis evaluation will be:

1. Simulation of interaction with obstacles, holes, and bark appearance as shown in images of Figure. 3.2.
2. Interactive response.

3.3 Pipeline

The following is a graphical representation of the pipeline.

3.4 Summary

This chapter provided the framework and methodology to be used in the woodification process. The process can be decomposed in two main events: (1) mesh operations, and (2) stress definitions. The next chapter provides implementation details for each step of the pipeline, adding some preliminary results as example images of the woodification application.

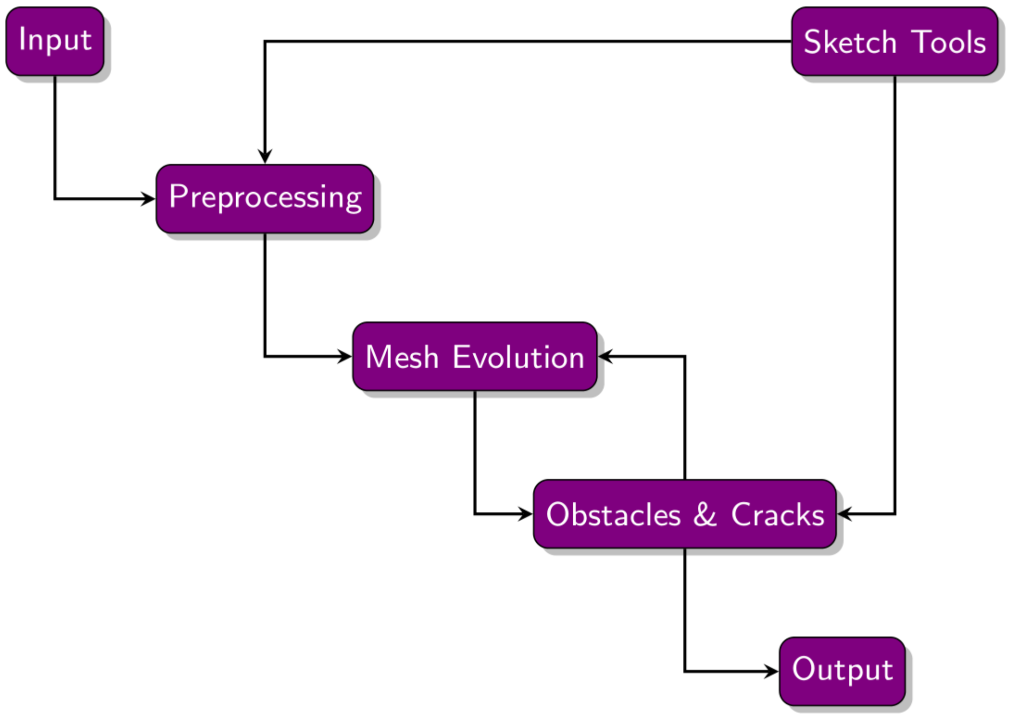
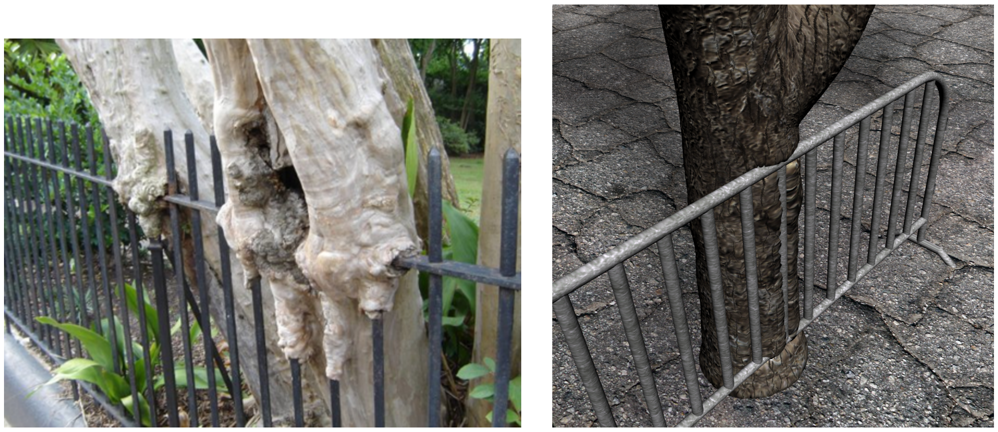
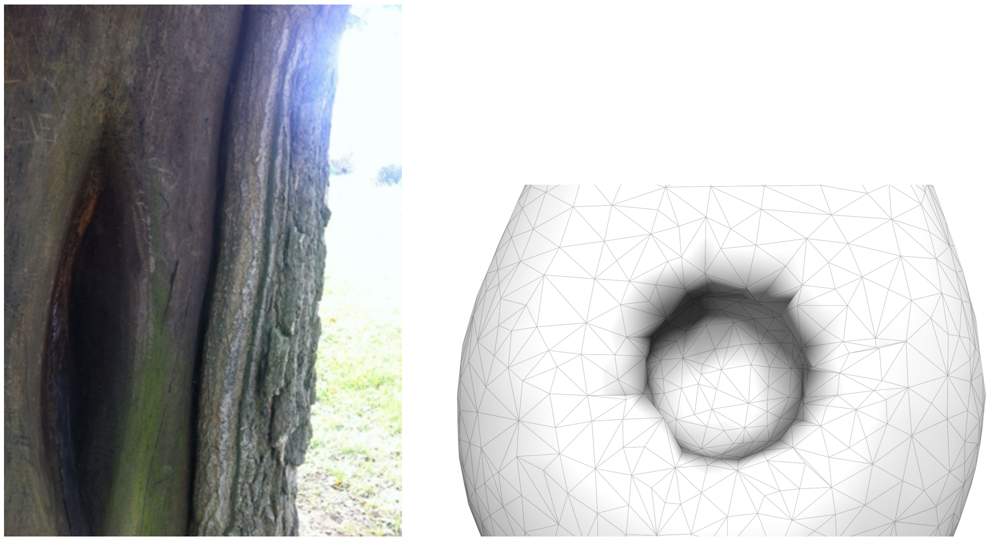


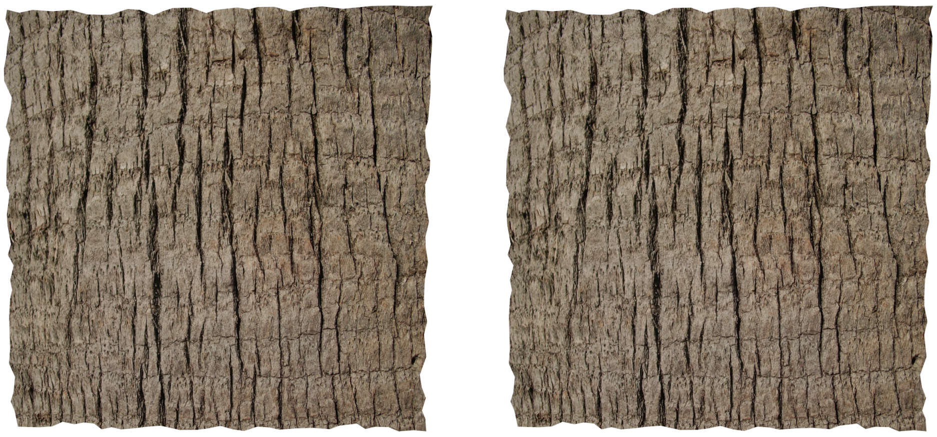
Figure 3.1. A polygonal model is converted to a tetrahedral mesh during pre-processing. The mesh is evolved by using the DSC and growth model while, at the same time, is covered by cracks and checked for collisions in the surrounding environment. The user can control, by a brush tool attached to mouse events, the definition of the cracks and the growth function. The output can be selected to be a polygonal mesh or the full tetrahedral mesh.



(a) Tree growing into obstacles.



(b) Tree affecting itself.



(c) Tree bark.

Figure 3.2. Phenomena to simulate. To the right a photo of a real tree, to the left mesh result from the simulation.

CHAPTER 4. IMPLEMENTATION

The following describes the mathematical and physical assumptions, for the proposed pipeline of *Woodification of polygonal meshes*, see Figure 3.1.

4.1 Deformable Simplicial Complex

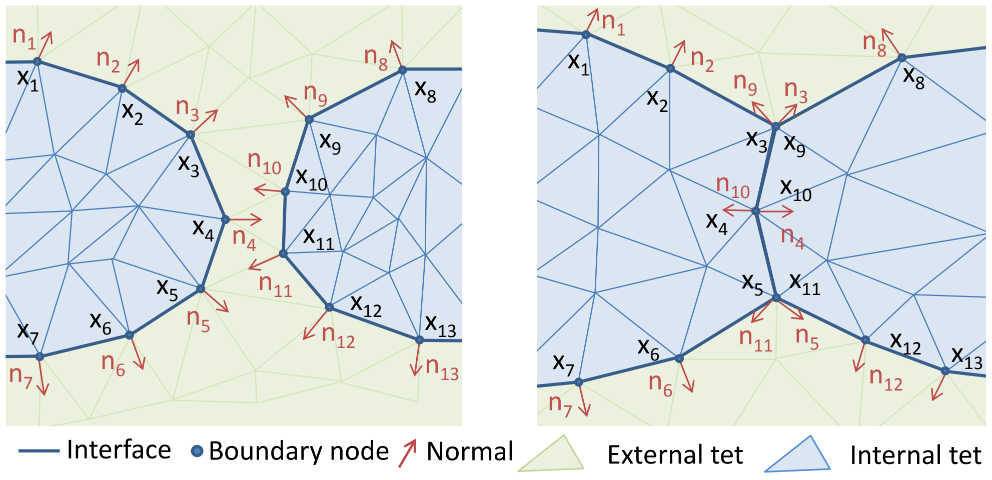
The method is initialized by converting the polygonal mesh into a single layer of tetrahedral mesh. Afterwards, each vertex, $\mathbf{x}_i$, is moved, from t_n to $t_{n+1} := t_n + \Delta t$, by an Eulerian integration along the normal $\mathbf{n}_i$, i.e.

$$\mathbf{x}_i^{n+1} = \mathbf{x}_i^n + s\left(\mathbf{x_i}, t_n\right)\mathbf{n}_i^t$$

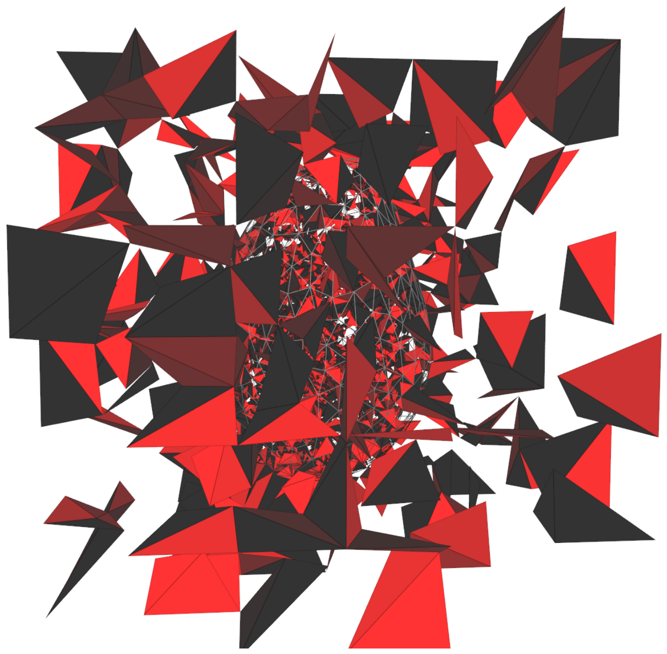
where $\mathbf{x}_i^n := \mathbf{x}_i\left(t_n\right)$. The function s represents the growth speed. Depending on the control required by the user, the speed and direction can be modified to produce more appealing, although not physically accurate, results (see Figure 4.4).

The apparent issue of topological changes during the mesh deformation, generated by local and global self-intersection, is solved maintaining an interior and exterior tet-mesh to generate a flag once a tet crosses the surface boundary.

DSC will propagates each vertex to a possible destination considering to not invert a tet during the path. In case of a hitting a flag for inverting a tet, the tet geometry is locally re-meshed to remove the degeneracy and continue with the motion, see Figure 4.1.



(a) DSC surface propagation. Topological issues are resolved when fronts collided.



(b) Tetrahedral visualization. Not all the mesh is represented. Instead, a minimum size for a tet is defined as threshold for rendering.

Figure 4.1. 2D sketch and 3D graphical representation of the Deformable Simplicial Complex.

In addition, surface re-meshing, tet-smoothing, and adaptive refinement can be applied to improve the quality of the mesh.

4.2 Growth Model

The surface is then evolved in time using the built-in evolution method of the DSC providing the direction and speed of growth for each vertex. The default direction considers the vertex *normal* while the speed can be selected from: (1) biologically-motivated allometric equations, (2) reversed growth, or (3) user-defined by a brush-tool.

4.2.1 Lateral

As stated by J. Landsberg and Sands (2011a), the standard *Weibull* function, with shape parameter $c > 0$, is a plausible growth model deduced from field-measurements:

$$s\left(t\right) = ct^{c-1}e^{-t^{c}} \tag{4.1}$$

The plot of the function, Figure 4.2, indicates a growth peak showing a transition from early to mature stages of development. Parameter c represents the stretching strength associated to the surface growth.

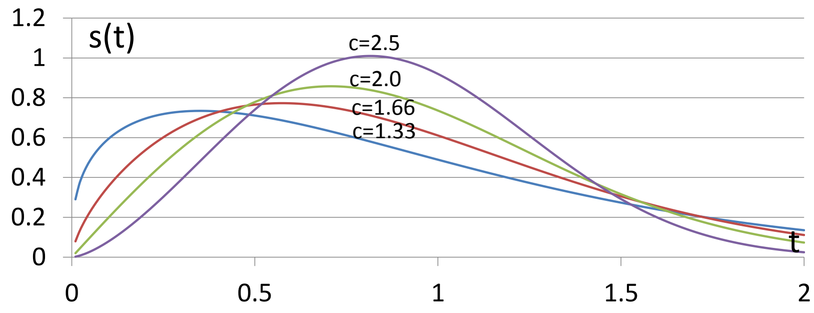


Figure 4.2. Graph of Weibull distribution, Eq. (4.1), for different values of shape parameter c in the range $[0, 2]$.

A modification to consider eccentricity, denoted as a real number δ, defines the speed as

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

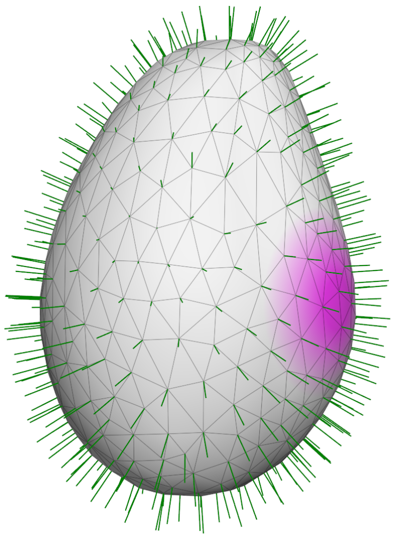
where $\hat{\mathbf{y}}$ represents the unit vertical direction of the coordinate system. For the results shown in next chapter the frame is defined by the canonical orthonormal basis $\hat{\mathbf{x}} = (1, 0, 0)$, $\hat{\mathbf{y}} = (0, 1, 0)$, $\hat{\mathbf{z}} = (0, 0, 1)$, and origin at $(0, 0, 0)$. The provided dot product, $\mathbf{n} \cdot \hat{\mathbf{y}}$, is measured using the Euclidean distance in $\mathbb{R}^3$. A compression growth is modeled by negative eccentricity whereas tension is defined by positive eccentricity.

4.2.2 User-Defined Growth

The above description of growth is fully automatic and parametric. To provide extra control a brush-tool for changing the growth direction is also supported.

To allow a procedural change in the growth direction the normals are updated at each time step with the set of new selected directions to evolve. Thus, at each iteration the user can select surface triangles to be modified via Ray-casting. Once the ray hits the surface a Gaussian filter distance decay is used to modify the neighborhood, Figure 4.3.

Another feature of the growth selection is to change the speed allowing velocity gradients. As with the normal, once the region is selected the growth direction can be increased or decreased. To allow smooth transition in the gradient, a factor which measures the frames per second is used as multiplier in the growth re-sizing, see Figure 4.4.


(a) Brush represented as the illuminated pink region. Growth directions shown as green.

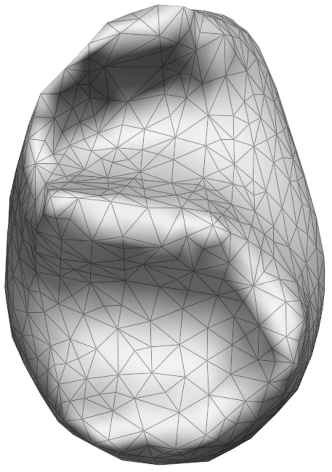

(b) Effect of growing by (4.1) with brushed directions.

Figure 4.3. User can define the growth direction with a brushing tool attached to mouse events.

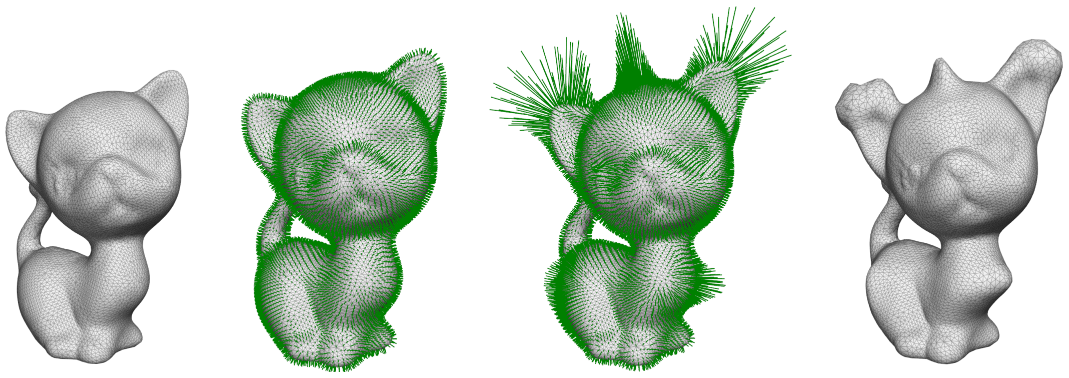


Figure 4.4. User defined growth. Left, the initial polygonal mesh. Middle left, the initial growth directions as green. Middle right, editing the ears, head, back, and chest of the kitten. Right, the resulting mesh after edition.

4.2.3 Obstacles

Realistic growth requires interaction with obstacles such as fences or sign posts. After the stem collides with the obstruction, the bark continues growing around it to eventually be absorbed into the interior of the tree.

DSC simplifies the detection of an obstacle to its internal data structure. Since the tet-mesh maintains the current polygonal mesh as one of the layers, it is possible to test if the movement of a vertex will fall inside an obstacle at each step. When a collision occurs, then the growth of such a vertex is frozen. Once a vertex is stopped, the underlying exterior and interior tet-mesh will change the possible space in which a vertex can move. The self-check of topology by DSC provides a robust method to avoid penetration inside the obstacle.

Furthermore, the extraction of normals from DSC can be overloaded to consider re-computation of face or vertex normal after modification of the surface mesh. Coupled with the update of normals to re-define the directions of growth, the front eventually surrounds the obstruction creating a wounded-region which persists through future iterations.

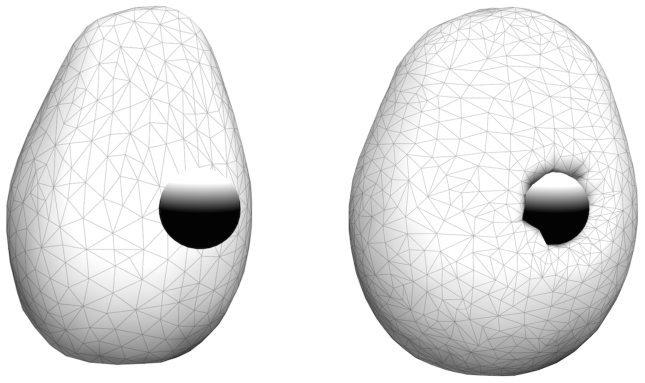


Figure 4.5. Growth around obstacle continues surrounding the obstruction.

4.3 Bark Model

Faithful representation of bark requires the addition of texturing to reproduce visual impact. DSC allows to add attributes to each vertex to consider material properties while the growth occurs. Thus, a texture mapping is added to the surface mesh. With this in hand, it is possible to represent a static bark at any given time. In the following section we will cover the automatic extension provided by DSC to preserve coherent texture mapping while the expansion takes place. Even more, the next chapter will cover the application of texturing to the cracking model reproducing in total fractured and bulged tree image.

4.3.1 Persistent Texturing

Initially, during the model loading, each vertex of the surface mesh propagates and carries the parametric information for texturing. As DSC supports local re-meshing the texture coordinate requires to also update. If a new vertex is added during the subdivision its texture attribute is acquired by averaging the ends where it is inserted. For deleted vertices, their texture attribute is also removed.

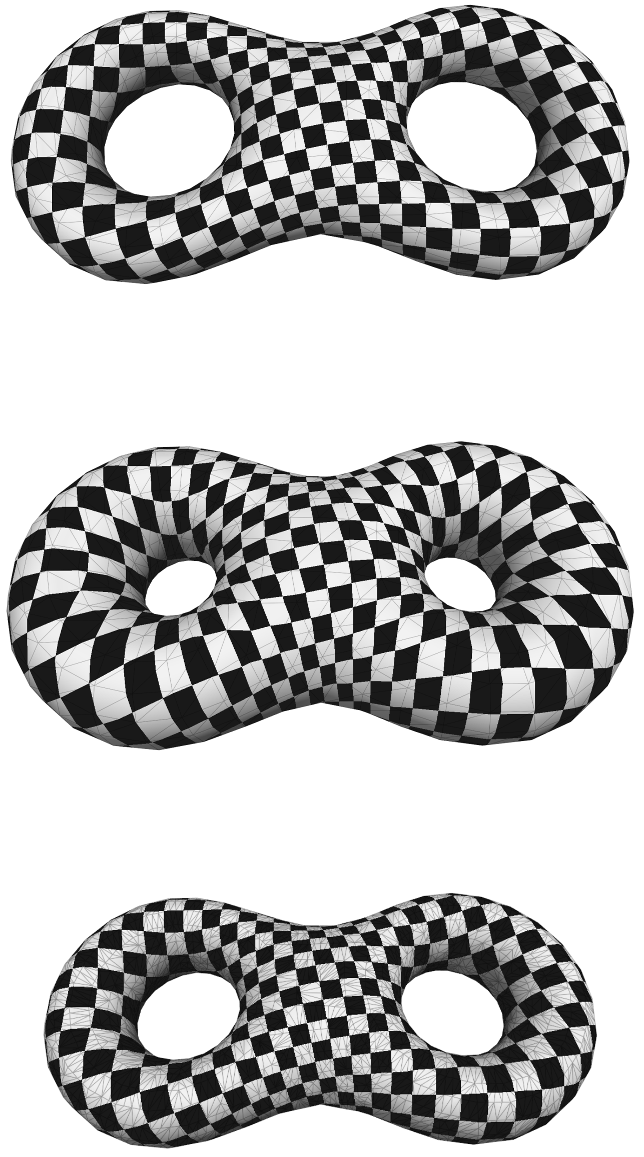


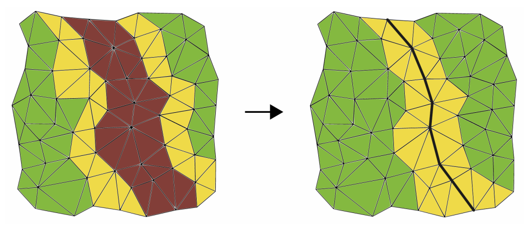
Figure 4.6. Top, texture mapping added to one of the DSC vertex attributes. Middle, the image is maintained over the growth. Bottom, subdivision is also supported by interpolation.

4.4 Cracking model

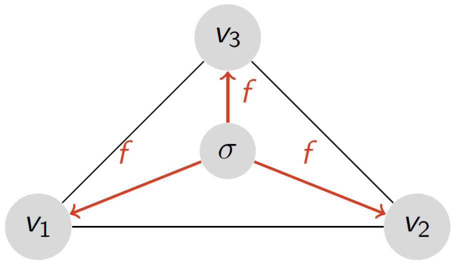
A stress field is defined over the polygonal mesh to recreate a crack once a surface stress threshold is surpassed. Defining the fractures as a consequence of a force field allows to transfer the tension generated by the stretching of the growth model over the surface.

4.4.1 Stress Field

As shown in Figure 4.7 the stress is defined per triangle adding to it a tensor σ of rank-2. The tensor encapsulates forces acting per area and their stiffness. The forces, per vertex, are then computed distributing the tensor along median rays.



(a) Color encodes stress magnitude. A crack will appear in the region of high stress alleviating the magnitude by a relaxation step.



(b) Force, f, distributed along the vertices.

Figure 4.7. Cracking construction. To the left stress field representation, to the right stress tensor, σ, stored per triangle face.

Suppose a triangle has vertices p_i, $i = \{1, 2, 3\}$. The associated orthonormal basis can be computed as $e_1 = \hat{u}$, $e_2 = e_1 \times e_3$, and $e_3 = \hat{u} \times \hat{v}$, where $\{u, v\}$ are the associated vectors to the triangle, i.e. $u = p_{\Pi(i)} - p_{\Pi(j)}$, $v = p_{\Pi(i)} - p_{\Pi(k)}$, $i \neq j \neq k$, and Π is a permutation of indices to create counter-clockwise vectors. The definitions of e_i are built to be a right-handed coordinate system if the set $\{u, v\}$ is linearly independent and rotates in the positive direction.

Once the basis is defined, a force vector f can be decomposed in tensile f^+, parallel to e_1, and compressive f^-, parallel to e_2, for each vertex. Hence, it is possible to define a *separation tensor* as

$$\zeta = \frac{1}{2}\left(\sum_{h\in\{f^+\}} h^T \otimes h \; + \sum_{l\in\{f^-\}} l^T \otimes l\right)$$

where $\otimes$ denotes the outer product of a vector. Σ represents a square matrix for which the largest positive eigenvalue is obtained to determine the possibility of a crack in a vertex.

4.4.2 Mesh Update

A priority queue is used to store the largest positive eigenvalue of each vertex computed in the previous section. When the eigenvalue exceeds a given threshold, a new edge (crack) is inserted using the vertex position and the orthogonal direction associated to the eigenvector. A local re-mesh is required for the newly created crack to preserve triangulation and a well-suited surface mesh. To avoid a poor quality mesh, if the crack generated is too close to an edge, the crack is replaced by the edge via a merging operation.

Once a crack appears, a change in stress is performed to alleviate the tensor in adjacent faces. The factor to relief varies from 0 to 1 indicating a completely removing stress or a zero change in value, respectively.

Additionally, a virtual displacement is computed and stored each time step , i.e. no actual movement of vertices caused by the forces. The main reason for using virtual storing is to avoid numerical instabilities of time integration. However, this value is used for distributing the stress to resemble a diffusion process during the relaxation step.

4.4.3 Relaxation

The relaxation step transfers stress from regions of high magnitude to regions of low magnitude. The distribution avoids concentration of cracks over a region, providing better visual appearance, see Figure 4.8.

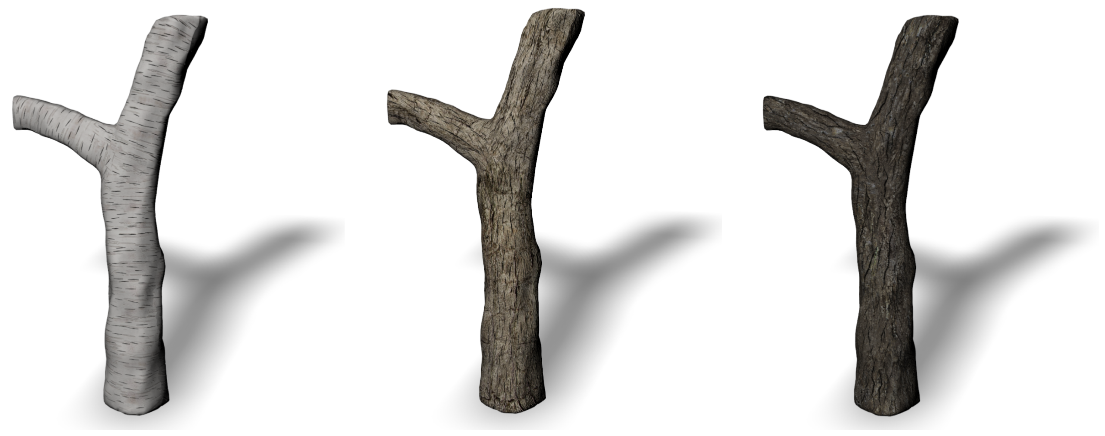


Figure 4.8. Different crack patterns. Left, horizontal pattern. Middle, vertical pattern. Right, random pattern.

The propagation of stress is simulated by a diffusion process. Thus, defining the deformation as

$$\Delta\sigma = \eta\Delta p \tag{4.2}$$

for η the Euclidean symmetric strain tensor

$$\eta_{ij} = \frac{1}{2}\left(p\beta_{ik}\beta_{kj} + p\left[\beta_{jk}\beta_{ki}\right]^T\right)$$

where p is the world space coordinate of a vertex and β_{ij} is the matrix formed by the barycentric coordinates of a triangle. In Eq.(4.2) the quantity Δp is associated to the virtual displacement computed during the mesh update. The diffusion direction can be also controlled to follow a predefined direction.

4.5 Level of detail

Once a region of crack is generated by the relaxation step, a tessellation shader is used to provide a level of detail in the region as shown in Figure 4.9

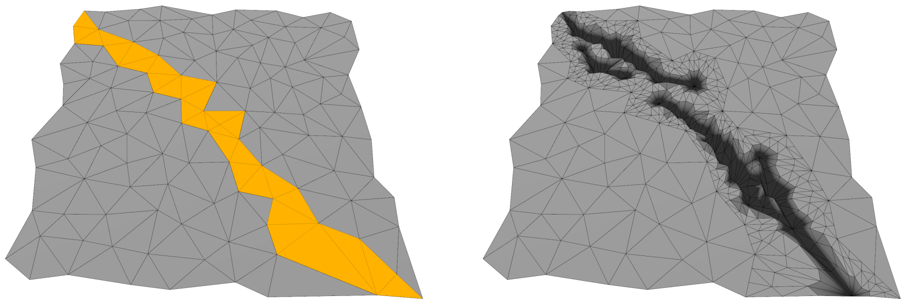


Figure 4.9. Left, shaded triangles as region where a crack will occur. Right, a level of detail using a tessellation shader is used to provide a compelling crack.

4.6 Coupling

The generated meshes at each time step from the surface expansion are coupled with the cracks transferring the set of cracks and the stress tensor between time steps. Moving the attributes of the cracks between meshes maintains the consistency in the simulation and allows to reproduce living wood over an object, see Figure 4.10.

To transfer attributes, an unique vertex identifier is used in case a new triangle is inserted. Similar to the texture, and interpolation is needed to get the stress value for the new triangles.

4.7 Rendering

The visualization of the simulation is done using OpenGL. The DSC is being rendered by a checker-board shader to recognize when a texture has been loaded.

The cracking exploits the geometry and tessellation capabilities of the shader language to do small scale detail when a triangle is being fractured.

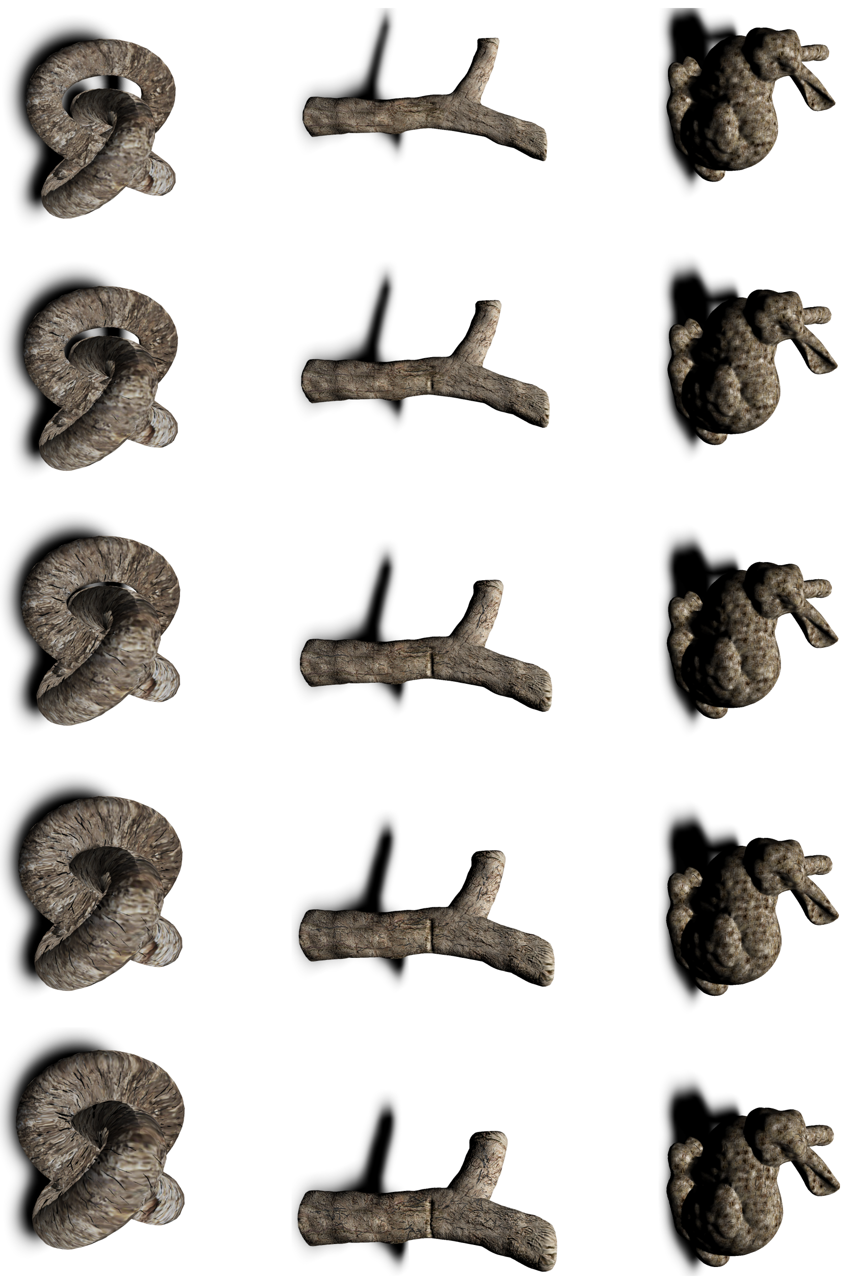


Figure 4.10. Growth and cracking patterns coupling. The surface growth according to Eq.(4.1) generating a polygonal mesh each time. The cracks and attributes of a triangle are transferred between temporal meshes to maintain coherence in the appearance. Top of images. Initial step of simulation. Bottom images. Cracks added as mesh grows.

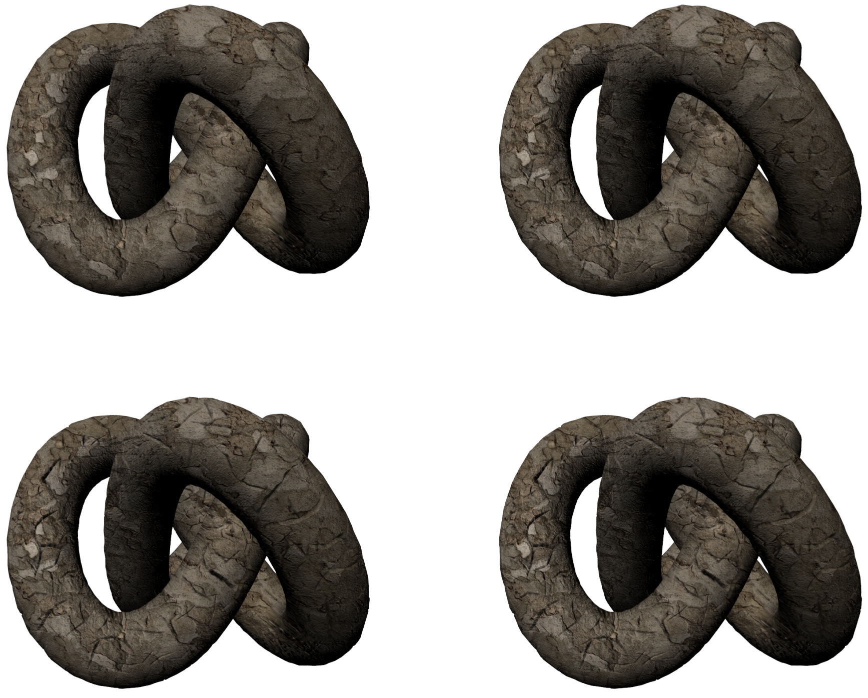


Figure 4.11. Cracking rendering pipeline. The tessellation shader controls the level of detail and local dimensions of a crack as shown in Figure 4.9.

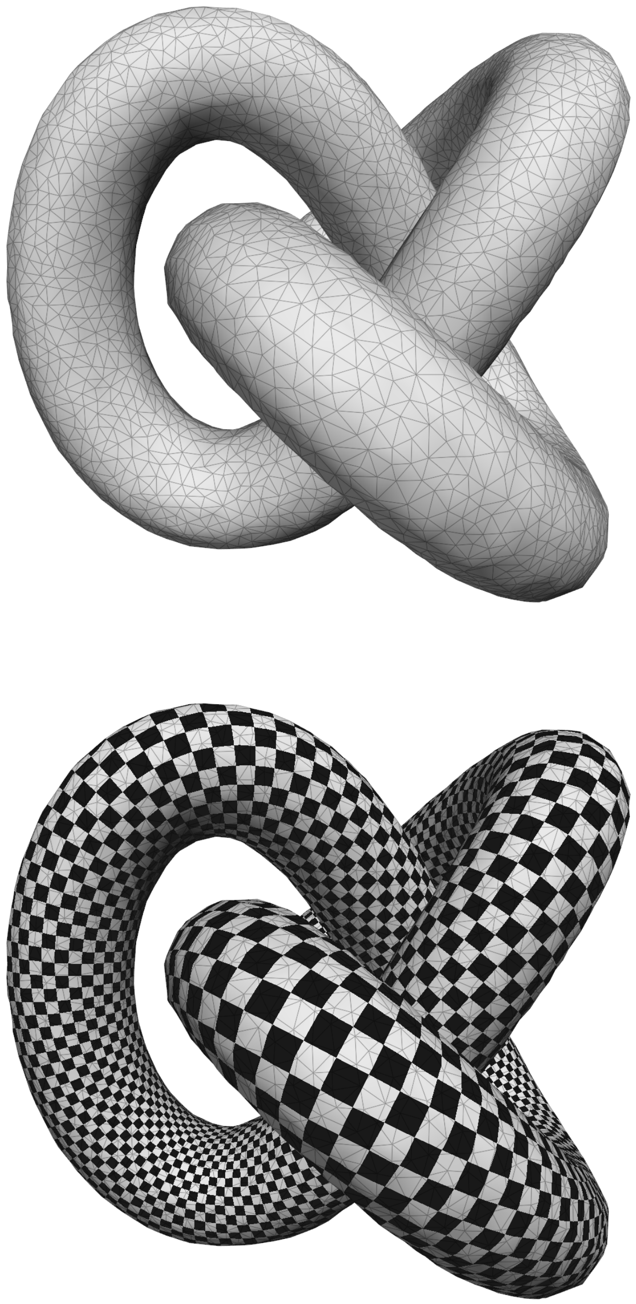


Figure 4.12. Polygonal mesh rendering. Top, knot mesh without texture information. Bottom, knot mesh as checkerboard pattern to represent texture information load

The system of the simulation is implemented using C++ for the computation and management of client code. The window system is managed by Qt, version 4.8. The third party libraries used are: DSC for the simplex representation of a mesh, TetGen for the preprocessing of polygonal meshes to a tet-mesh, GLEW for the extensions of OpenGL required in the core profile, and GLUT for the creation of a minimal graphical user interface during the experiments.

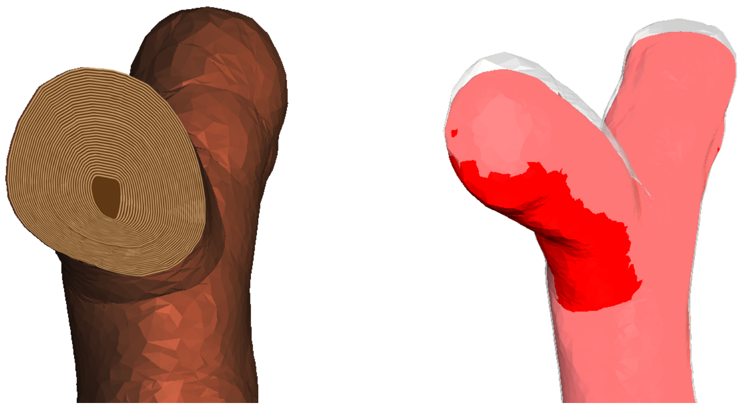


Figure 5.1. Left, growth rings. Right, positive eccentricity (white) overlaying non eccentric growth (red)

The DSC library needs some functionality from C++11, such as constructor by a list initializer, requiring a current compiler of C++. The following results are generated on a desktop computer equipped with an Intel Core i7-2600K processor clocked at 2.0Ghz with 16GB of RAM and a NVIDIA GeForce GTX 480 GPU with 1.5GB of memory.

Figure 5.1(a) shows the inner structure of a tree with rings generated by the lateral growth model. Figure 5.1(b) represents the usage of positive eccentricity during the growth (white layer) and the comparison with ordinary growth (red layer).

Figure 5.2 shows an example of selecting some polygons of the mesh to generate cracks in that region.

Figure 5.3 has two woodified objects. After objects start to expand, sharp edges are being smoothed by bloating, a feature provided when combining lateral growth in DSC. The boundaries eventually close two fronts.

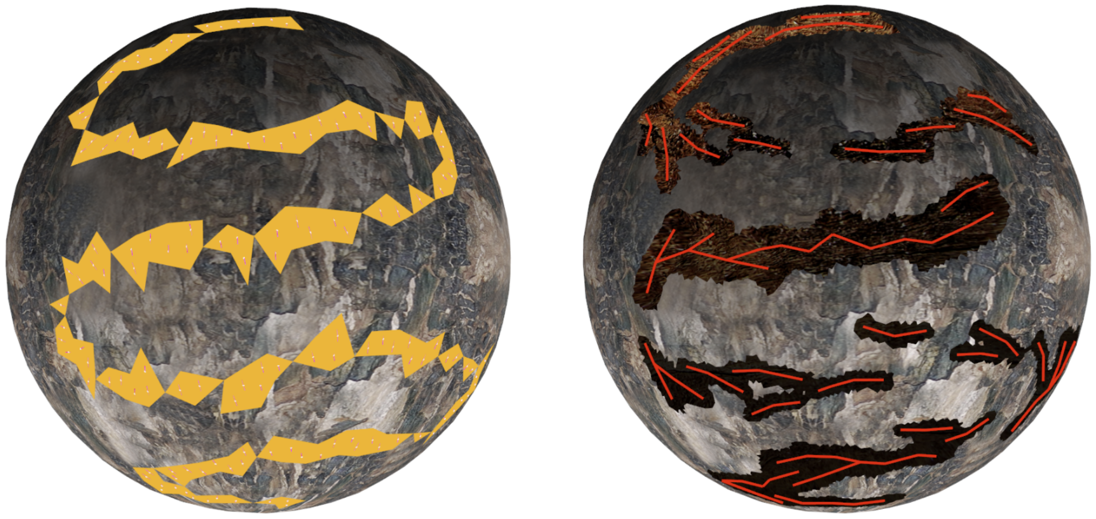


Figure 5.2. An example of selecting a region for cracking. Left, initial mesh with selected triangles. Right, generation of cracks in selected regions shown as lines and different texture.

The material for a crack can also be manipulated as shown in Figure 5.4. The image is transformed to a crack render via texture mapping. Chapter 3, Figure 3.2 shows different frames of a growing trunk which respond to the fence obstacle. The DSC adaptivity allows a smooth transition when the object hits the obstacle, wrapping around it.

Topological changes of an object, Figure 5.5. The last example, Figure **??**, shows another set of polygonal meshes woodified.

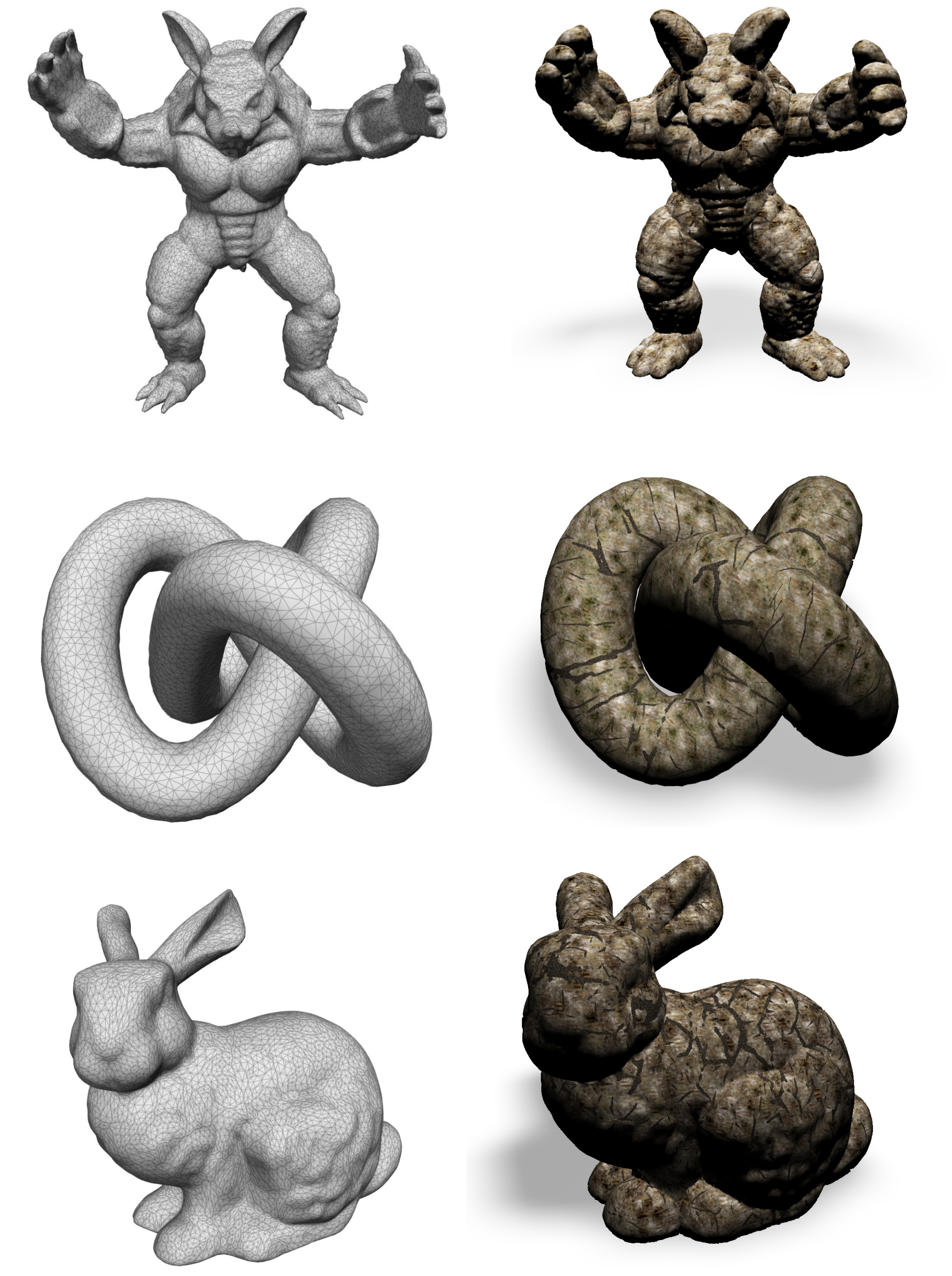


Figure 5.3. Three polygonal meshes woodified. Left, input mesh as solid polygons with wireframe overlay and no texturing. Right, output mesh as solid polygons with texture, mesh growth, and cracks.

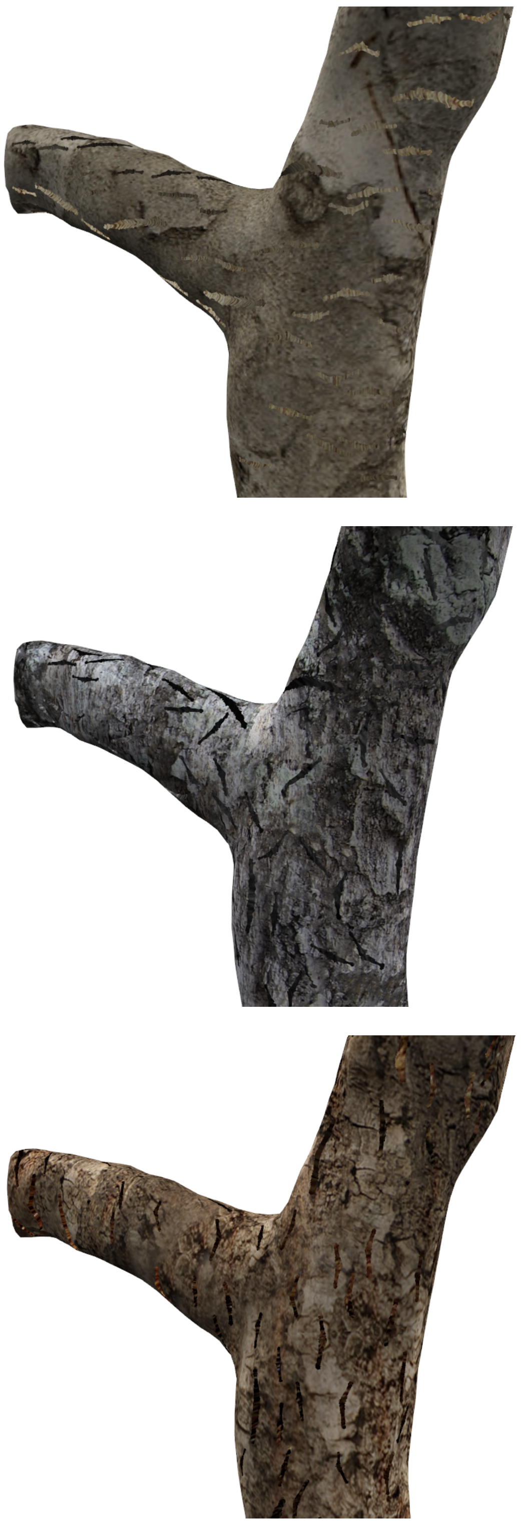


Figure 5.4. Different texture mapping of object material and crack material.

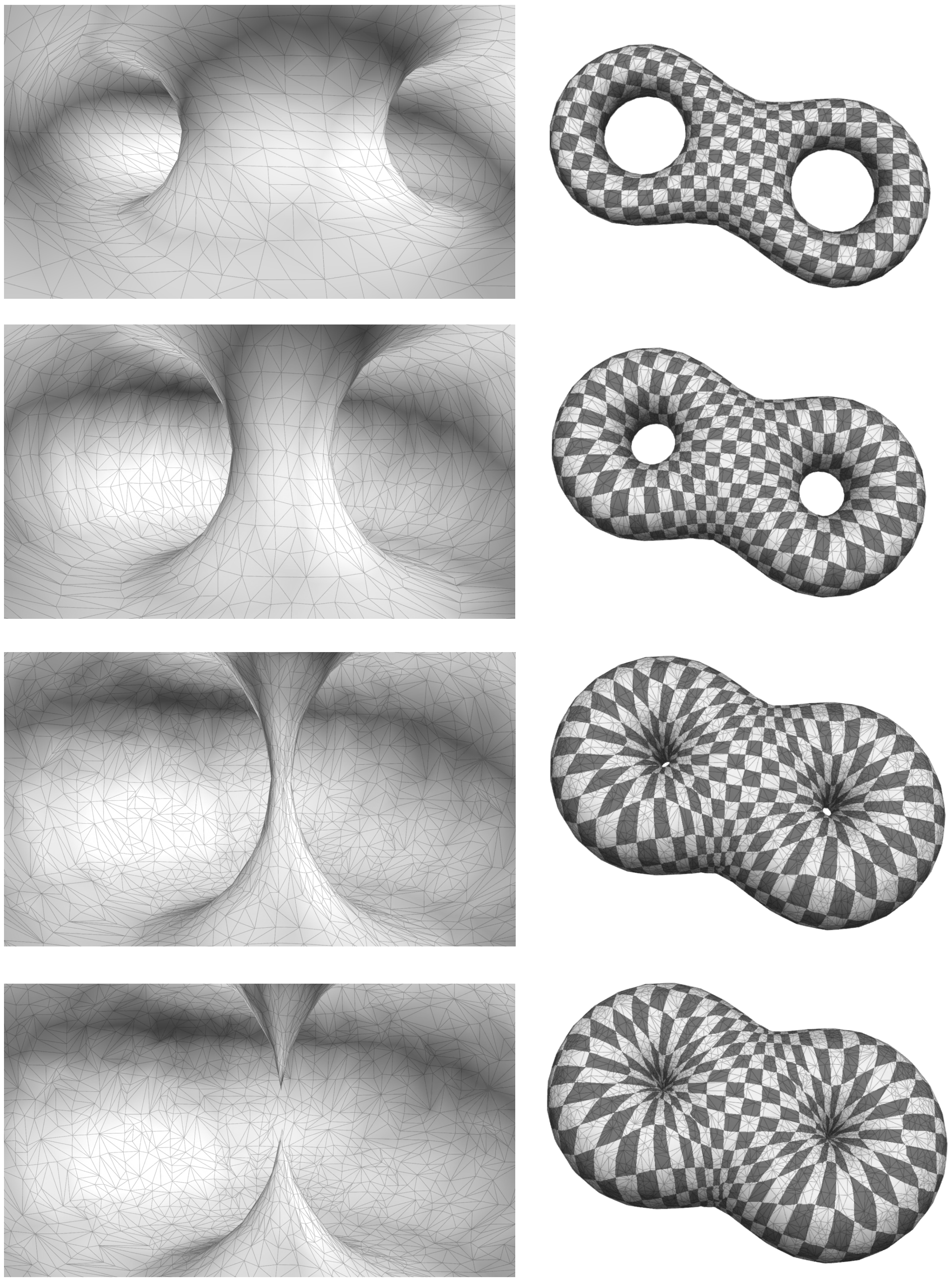


Figure 5.5. The growth model change the mesh's topology. The images shown how DSC handles these changes automatically.

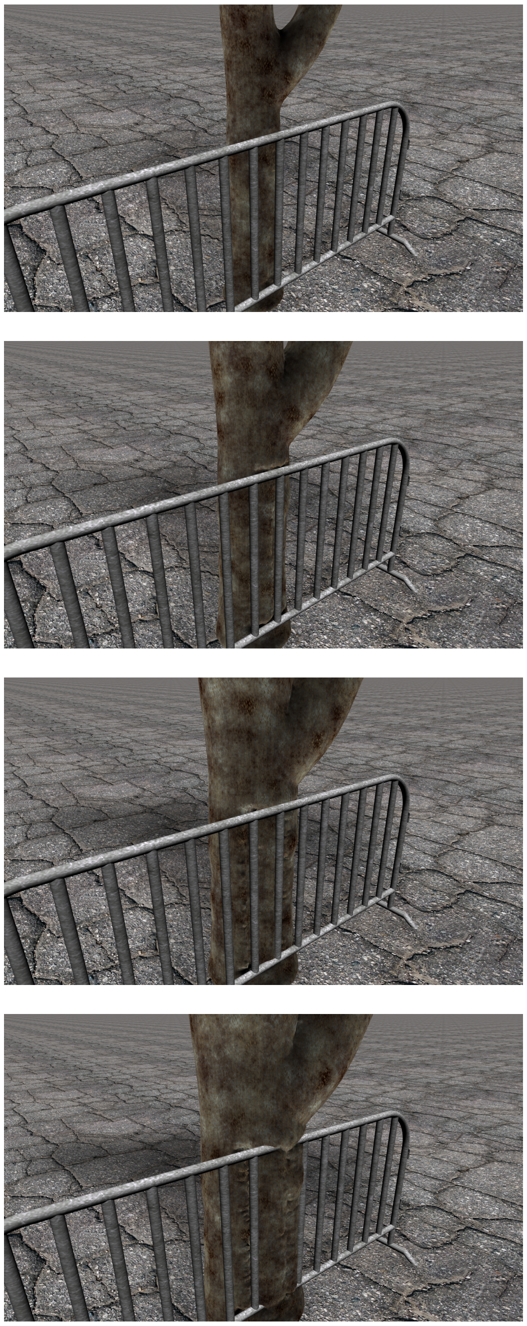


Figure 5.6. Sequence of tree growing around obstacle. The DSC handle the wrapping automatically subdividing the regions of collision.

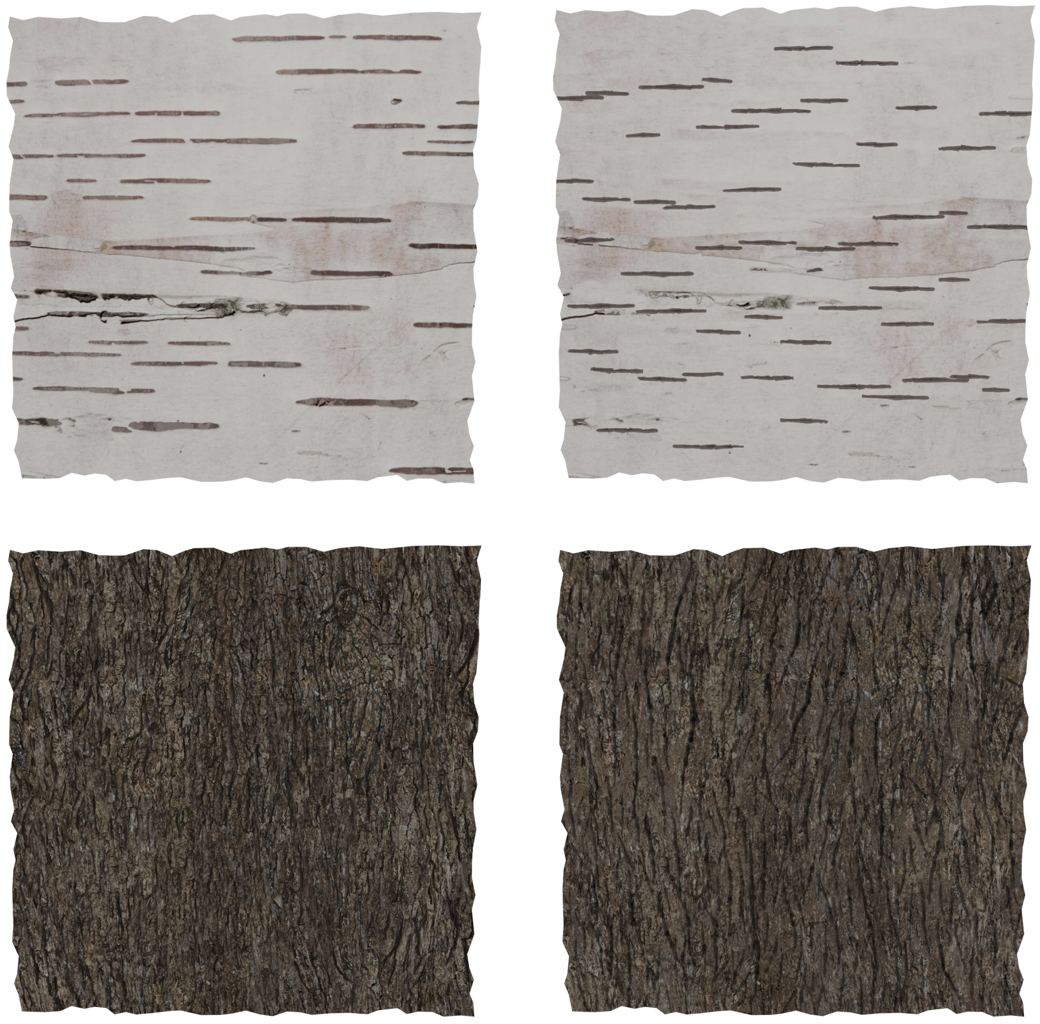


Figure 5.7. Comparison of photos, left, of two types of cracks with the ones generated by the simulation, right.

CHAPTER 6. CONCLUSIONS

A framework for woodifying polygonal meshes has been presented. The process starts by converting the mesh to a tetrahedral representation which provides volumetric information necessary for the Deformable Simplicial Complex method.

The evolution is automated by a growth function based on the Weibull distribution. Using the up direction and a scalar value we modify the growth function to represent eccentricity. Also, a sketch tool for interactively modify direction and speed of expansion is provided. The DSC has been extended to consider collision detection with objects when the mesh is propagating, including the adaptation of the surface to consider topological and geometrical changes, such as merging and subdivision. Such extension also maintains the texture coordinates during the expansion, via an interpolation, allowing to reproduce texture material with no additional effort.

The growth is coupled to a force-based cracking model which accounts for the stress on the face of the polygonal mesh. At each iteration, the cracks are transferred to maintain coherent appearance throughout the simulation. Cracks can be defined following a predefined pattern, such as vertical strips or uniform distribution, or can be applied directly by the user via a sketching tool. Such arrangements are computed by distributing forces along vertices of a triangle and defining a relaxation method to mark a path to be converted into a crack. In previous chapters we have shown the result of applying different texture maps onto a crack to have a more realistic appearance. The level of detail for the crack is control by a tessellation shader, providing quality results.

Since the most time consuming step of the simulation is the expansion of the surface using DSC, caused by adding vertices and edges when subdividing or the collision response between object and obstacle, an optimization of the method is required as future work. An interesting problem to explore will be the addition of other attributes to simulate other surface features, such as knots, imperfections, or cellular-features, like moss or dust. Also, a comparison of the simulation with data from forestry and biology research would give more support to the presented model. In this aspect, the evolution and cracking of the mesh is done in the CPU by a single thread, delegating the GPU for the visualization only. Transferring the methods to a parallel version will provide an easier tool to test and manipulate the woodification idea.

Finally, although some comparison between photographs and images from the simulation are provided, Figure. 3.2, more scenarios to intent to reproduce by the simulation will provide robustness to the method.

LIST OF REFERENCES

LIST OF REFERENCES

Amar, M., Goriely, A., & Müller, M. (2011). *New trends in the physics and mechanics of biological systems: Lecture notes of the les houches summer school: Volume 92, july 2009*. OUP Oxford. Retrieved from `http://books.google.com/books?id=BMIn3Ft9YRQC`

Angel, E., & Shreiner, D. (2011). *Interactive computer graphics: A top-down approach with shader-based opengl*. ADDISON WESLEY Publishing Company Incorporated. Retrieved from `http://books.google.com/books?id=KhPmtgAACAAJ`

Bloomenthal, J. (1985, July). Modeling the mighty maple. *SIGGRAPH Comput. Graph.*, *19*(3), 305–311. Retrieved from `http://doi.acm.org/10.1145/325165.325249` doi: 10.1145/325165.325249

Buchanan, J. W. (1998). Simulating wood using a voxel approach. *Comp. Graph. Forum*, *17*(3), 105–112.

Chen, X., Neubert, B., Xu, Y.-Q., Deussen, O., & Kang, S. B. (2008, December). Sketch-based tree modeling using markov random field. *ACM Trans. Graph.*, *27*(5), 109:1–109:9. Retrieved from `http://doi.acm.org/10.1145/1409060.1409062` doi: 10.1145/1409060.1409062

Chiba, Y. (1990a). Plant form analysis based on the pipe model theory i. a statical model within the crown. *Ecological Research*, *5*(2), 207-220. Retrieved from `http://dx.doi.org/10.1007/BF02346992` doi: 10.1007/BF02346992

Chiba, Y. (1990b). A quantitative analysis of stem form and crown structure: the s-curve and its application. *Tree Physiology*, *7*(1-2-3-4), 169-182. Retrieved from `http://treephys.oxfordjournals.org/content/7/1-2-3-4/169.abstract` doi: 10.1093/treephys/7.1-2-3-4.169

Chiba, Y. (1991). Plant form based on the pipe model theory ii. quantitative analysis of ramification in morphology. *Ecological Research*, *6*(1), 21-28. Retrieved from `http://dx.doi.org/10.1007/BF02353867` doi: 10.1007/BF02353867

Chiba, Y., & Shinozaki, K. (1994). A simple mathematical model of growth pattern in tree stems. *Annals of Botany*, *73*(1), 91-98. Retrieved from `http://aob.oxfordjournals.org/content/73/1/91.abstract` doi: 10.1006/anbo.1994.1011

Christiansen, A., Nobel-Jrgensen, M., Aage, N., Sigmund, O., & Brentzen, J. (2013). Topology optimization using an explicit interface representation. *Structural and Multidisciplinary Optimization*, 1-13. Retrieved from `http://dx.doi.org/10.1007/s00158-013-0983-9` doi: 10.1007/s00158-013-0983-9

Colin, F., Mothe, F., Freyburger, C., Morisset, J.-B., Leban, J.-M., & Fontaine, F. (2010). Tracking rameal traces in sessile oak trunks with x-ray computer tomography: biological bases, preliminary results and perspectives. *Trees*, *24*(5), 953-967. Retrieved from `http://dx.doi.org/10.1007/s00468-010-0466-1` doi: 10.1007/s00468-010-0466-1

Dale, V., Doyle, T., & Shugart, H. (1985). A comparison of tree growth models. *Ecological Modelling*, *29*(14), 145 - 169. Retrieved from `http://www.sciencedirect.com/science/article/pii/0304380085900511` doi: http://dx.doi.org/10.1016/0304-3800(85)90051-1

Deussen, O., Hanrahan, P., Lintermann, B., Měch, R., Pharr, M., & Prusinkiewicz, P. (1998). Realistic modeling and rendering of plant ecosystems. In *Proceedings of the 25th annual conference on computer graphics and interactive techniques* (pp. 275–286). New York, NY, USA: ACM. doi: http://doi.acm.org/10.1145/280814.280898

Deussen, O., & Lintermann, B. (2004). *Digital design of nature: Computer generated plants and organics.* SpringerVerlag.

Edelstein-Keshet, L. (1988). *Mathematical models in biology.* Random House, New York.

Federl, P., & Prusinkiewicz, P. (2002). *Modelling fracture formation in bi-layered materials, with applications to tree bark and drying mud.*

Federl, P., & Prusinkiewicz, P. (2004). Finite element model of fracture formation on growing surfaces. *Proc. ICCS, (LNCS 3037)*, 138-145.

Fišer, M. (2012). *L-systems online.* Prague, Czech Republic: Charles University, Faculty of Mathematics and Physics.

Galbraith, C., Mndermann, L., & Wyvill, B. (2004). Implicit visualization and inverse modeling of growing trees. *Computer Graphics Forum*, *23*(3), 351–360. Retrieved from `http://dx.doi.org/10.1111/j.1467-8659.2004.00766.x` doi: 10.1111/j.1467-8659.2004.00766.x

Glondu, L., Muguercia, L., Marchal, M., Bosch, C., Rushmeier, H., Dumont, G., & Drettakis, G. (2012, June). Example-based fractured appearance. *Comp. Graph. Forum*, *31*(4), 1547–1556. Retrieved from `http://dx.doi.org/10.1111/j.1467-8659.2012.03151.x` doi: 10.1111/j.1467-8659.2012.03151.x

Gobron, S., & Chiba, N. (2001). Simulation of peeling using 3d-surface cellular automata. In *Proceedings of the 9th pacific conference on computer graphics and applications* (pp. 338–). Washington, DC, USA: IEEE Computer Society. Retrieved from `http://dl.acm.org/citation.cfm?id=882473.883422`

Hart, J. C., & Baker, B. (1996). Implicit modeling of tree surfaces. *Proc. Implicit Surfaces*, 143-153.

Hirota, K., Tanoue, Y., & Kaneko, T. (1998). Generation of crack patterns with a physical model. *The Visual Computer*, *14*, 126-137. Retrieved from `http://dx.doi.org/10.1007/s003710050128` (10.1007/s003710050128)

Hirota, K., Tanoue, Y., & Kaneko, T. (2000). Simulation of three-dimensional cracks. *The Visual Computer*, *16*, 371-378. Retrieved from `http://dx.doi.org/10.1007/s003710000069` (10.1007/s003710000069)

HPCG. (2014, March). *Set of possible meshes for woodification.* Retrieved from `http://alejandroguayaquil.com/branch/tmp/dscmeshes.zip`

Huber, B. (1956). Die gefleitung. In M. Adriani et al. (Eds.), *Pflanze und wasser / water relations of plants* (Vol. 3, p. 541-582). Springer Berlin Heidelberg. Retrieved from `http://dx.doi.org/10.1007/978-3-642-94678-3_30` doi: 10.1007/978-3-642-94678-3_30

Iben, H. N., & O'Brien, J. F. (2006, Sept). Generating surface crack patterns. In *Proceedings of the acm siggraph/eurographics symposium on computer animation* (pp. 177–185). Retrieved from `http://graphics.cs.berkeley.edu/papers/Iben-GSC-2006-09/`

Kaandorp, A. J. (1994). *Fractal modelling: Growth and form in biology.* Springer–Verlag. Retrieved from `http://books.google.com/books/about/Fractal_modelling.html?id=U17JnN7UzZUC`

Kirota, K., Kato, H., & Kaneko, T. (1998). A physically-based simulation model of growing tree bark. *IPSJ Journal*, *39*(11). (in Japanese)

Kopf, J., Fu, C.-W., Cohen-Or, D., Deussen, O., Lischinski, D., & Wong, T.-T. (2007). Solid texture synthesis from 2d exemplars. *ACM Transactions on Graphics (Proceedings of SIGGRAPH 2007)*, *26*(3), 2:1–2:9.

Lam, Z., & King, S. A. (2005). Simulating tree growth based on internal and environmental factors. In *Graphite '05: Proceedings of the 3rd international conference on computer graphics and interactive techniques in australasia and south east asia* (pp. 99–107). New York, NY, USA: ACM Press. doi: http://doi.acm.org/10.1145/1101389.1101406

Landsberg, J., & Sands, P. (2011a). Chapter 4 - stand structure and dynamics. In *Physiological ecology of forest production* (Vol. 4, p. 81 - 114). Elsevier. Retrieved from `http://www.sciencedirect.com/science/article/pii/B9780123744609000044` doi: http://dx.doi.org/10.1016/B978-0-12-374460-9.00004-4

Landsberg, J., & Sands, P. (2011b). Chapter 8 - modelling tree growth: Concepts and review. In *Physiological ecology of forest production* (Vol. 4, p. 221 - 240). Elsevier. Retrieved from `http://www.sciencedirect.com/science/article/pii/B9780123744609000081` doi: http://dx.doi.org/10.1016/B978-0-12-374460-9.00008-1

Landsberg, J. J. (2011). Series editors. In *Physiological ecology of forest production* (Vol. 4, p. ii -). Elsevier. Retrieved from `http://www.sciencedirect.com/science/article/pii/B9780123744609000160` doi: http://dx.doi.org/10.1016/B978-0-12-374460-9.00016-0

Lawrence, J., & Funkhouser, T. (2003). A painting interface for interactive surface deformation. *Proc. Pacific Graphics*, 141-150.

Lefebvre, S., & Neyret, F. (2002a). Synthesizing bark. In *Proceedings of the 13th eurographics workshop on rendering* (pp. 105–116). Aire-la-Ville, Switzerland, Switzerland: Eurographics Association. Retrieved from `http://dl.acm.org/citation.cfm?id=581896.581911`

Lefebvre, S., & Neyret, F. (2002b). Synthesizing bark. In *Proceedings of the 13th eurographics workshop on rendering* (pp. 105–116). Eurographics Association. Retrieved from `http://dl.acm.org/citation.cfm?id=581896.581911`

Lindenmayer, A. (1968). Mathematical models for cellular interactions in development i. filaments with one-sided inputs. *Journal of Theoretical Biology*, *18*(3), 280 - 299. Retrieved from `http://www.sciencedirect.com/science/article/pii/0022519368900799` doi: http://dx.doi.org/10.1016/0022-5193(68)90079-9

Lorensen, W. E., & Cline, H. E. (1987, August). Marching cubes: A high resolution 3d surface construction algorithm. *SIGGRAPH Comput. Graph.*, *21*(4), 163–169. Retrieved from `http://doi.acm.org/10.1145/37402.37422` doi: 10.1145/37402.37422

Mann, J., Plank, M., & Wilkins, A. (2006). Tree growth and wood formation - applications of anisotropic surface growth. *Mathematics in Industry Study Group 2006*, 153–191.

Maute, K., & Ramm, E. (1995). Adaptive topology optimization. *Structural optimization*, *10*(2), 100-112. Retrieved from `http://dx.doi.org/10.1007/BF01743537` doi: 10.1007/BF01743537

Misztal, M. K., & Bærentzen, J. A. (2012, June). Topology-adaptive interface tracking using the deformable simplicial complex. *ACM Trans. Graph.*, *31*(3), 24:1–24:12. Retrieved from `http://doi.acm.org/10.1145/2167076.2167082` doi: 10.1145/2167076.2167082

Misztal, M. K., Bridson, R., Erleben, K., Bærentzen, J. A., & Anton, F. (2010). Optimization-based fluid simulation on unstructured meshes. In *Vriphys* (p. 11-20).

O'Brien, J. F., & Hodgins, J. K. (1999, August). Graphical modeling and animation of brittle fracture. In *Proceedings of acm siggraph 1999* (pp. 137–146). ACM Press/Addison-Wesley Publishing Co. Retrieved from `http://graphics.cs.berkeley.edu/papers/Obrien-GMA-1999-08/` doi: http://doi.acm.org/10.1145/311535.311550

Oohata, S.-i., & Shinozaki, K. (1979, dec). A statical model of plant form-further analysis of the pipe model theory. *Japanese Journal of Ecology*, *29*(4), 323-335. Retrieved from `http://ci.nii.ac.jp/naid/110001883056/en/`

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

Paquette, E., Poulin, P., & Drettakis, G. (2002). The Simulation of Paint Cracking and Peeling. In W. Stuerzlinger & M. McCool (Eds.), *Proceedings of Graphics Interface* (p. 10). Calgary, Canada: Canadian Human-Computer Communications Society. Retrieved from `http://hal.inria.fr/inria-00606725`

Pauly, M., Keiser, R., Adams, B., Dutré, P., Gross, M., & Guibas, L. J. (2005, July). Meshless animation of fracturing solids. *ACM Trans. Graph.*, *24*(3), 957–964. Retrieved from `http://doi.acm.org/10.1145/1073204.1073296` doi: 10.1145/1073204.1073296

Rabczuk, T., & Belytschko, T. (2004). Cracking particles: a simplified meshfree method for arbitrary evolving cracks. *International Journal for Numerical Methods in Engineering*, *61*(13), 2316–2343. Retrieved from `http://dx.doi.org/10.1002/nme.1151` doi: 10.1002/nme.1151

Runions, A., Lane, B., & Prusinkiewicz, P. (2007). Modeling trees with a space colonization algorithm. , 63-70. Retrieved from `http://www.eg.org/EG/DL/WS/NPH/NPH07/063-070.pdf` doi: 10.2312/NPH/NPH07/063-070

Sellier, D., Plank, M. J., & Harrington, J. J. (2011). A mathematical framework for modelling cambial surface evolution using a level set method. *Annals of Botany*, *108*(6), 1001-1011. Retrieved from `http://aob.oxfordjournals.org/content/108/6/1001.abstract` doi: 10.1093/aob/mcr067

Si, H. (2013, November). *Tetgen. a quality tetrahedral mesh generator and a 3d delaunay triangulator.* Retrieved from `http://wias-berlin.de/software/tetgen/`

Stava, O., Pirk, S., Kratt, J., Chen, B., Mch, R., Deussen, O., & Benes, B. (2014). Inverse procedural modelling of trees. *Computer Graphics Forum*, n/a–n/a. Retrieved from `http://dx.doi.org/10.1111/cgf.12282` doi: 10.1111/cgf.12282

Steppe, K., De Pauw, D. J. W., Lemeur, R., & Vanrolleghem, P. A. (2006). A mathematical model linking tree sap flow dynamics to daily stem diameter fluctuations and radial stem growth. *Tree Physiology*, *26*(3), 257-273. Retrieved from `http://treephys.oxfordjournals.org/content/26/3/257.abstract` doi: 10.1093/treephys/26.3.257

Wang, X., Wang, L., Liu, L., Hu, S., & Guo, B. (2003). Interactive modeling of tree bark. In *Proceedings of the 11th pacific conference on computer graphics and applications* (pp. 83–). Washington, DC, USA: IEEE Computer Society. Retrieved from `http://dl.acm.org/citation.cfm?id=946250.946979`

Weibull, W. (1951). A statistical distribution function of wide applicability. *Journal of Applied Mechanics*, *18*, 293–297.

Wither, J., Boudon, F., Cani, M.-P., & Godin, C. (2009). Structure from silhouettes: a new paradigm for fast sketch-based design of trees. *Computer Graphics Forum*, *28*(2), 541–550. Retrieved from `http://dx.doi.org/10.1111/j.1467-8659.2009.01394.x` doi: 10.1111/j.1467-8659.2009.01394.x

Zhou, J., Chen, L.-T., Liu, Q.-H., Li, Y.-M., & Rao, Y.-B. (2010). Fractal-based 3d tree modeling. In *Computer design and applications (iccda), 2010 international conference on* (Vol. 5, p. V5-454-V5-457). doi: 10.1109/ICCDA.2010.5540927