BlobTree Trees

Callum Galbraith Peter MacMurchy Brian Wyvill

callum/peterm/blob@cpsc.ucalgary.ca
Department of Computer Science
University of Calgary

Abstract

In recent years several methods for modelling botanical trees have been proposed. The geometry and topology of tree skeletons can be well described by L-systems, however there are several approaches to modelling smooth surfaces to represent branches, and not all of the observed phenomena can be represented by current methods. Many tree types exhibit non-smooth features such as branch bark ridges and collars.

In this paper we present a botanical tree modelling approach that uses a hierarchical implicit modelling system. The *BlobTree* provides several techniques to control the combination of primitives, and allows both smooth and non-smooth effects to be combined in a single blend volume in an easily controlled fashion. We use the *BlobTree* to produce models of botanical tree branching structures that capture smooth branching, branch collars and branch bark ridges in a single model. The *BlobTree* is used as a procedural surface modelling system, taking input from L-system botanical tree descriptions. We show that smooth, C1 continuous blends can be obtained without noticable bulging, using summation blending. We implement irregular effects using Precise Contact Modelling, Constructive Solid Geometry and space warping.

Key words: implicit surfaces, BlobTree.

Figure 1: Photograph of a poplar tree showing the branch bark ridge, and resulting rings around branching points

1 Introduction

Modeling of branching topologies for botanical structures is a well-studied problem, as is their subsequent visualization. However, generating a realistic and smoothly connected surface around an arbitrary branching skeleton remains difficult. Current methods do not capture the subtleties of the blends at the branches, in particular the branch bark ridge [37], shown in Figure 1, and collars shown in Figure 2.

Previous work has focused on the construction of smoothly connected surfaces using schemes such as parametric surface patches, implicit surfaces, and subdivision surfaces. In this paper, we propose a method for the next step in generating realistic models of trees. Our approach uses skeletal implicit surfaces in the context of a hierarchical data structure called the *BlobTree* [45], which combines blending operations with other non-blending operations such as Constructive Solid Geometry (CSG), spacial warping, and Precise Contact Modeling (PCM). Using the *BlobTree* we are able to model non-smooth qualities of trees, such as scars formed where branches have been discarded. By using PCM, the method allows the limbs to both blend smoothly at branching points, and to form the branch bark ridge as appropriate. Additionally, it is resolution independent, providing a well-defined surface at all levels of detail.

A branching hierarchy is obtained using the well-studied computer graphics technique *L-systems* [26, 31]. An L-system is used to generate a string that encodes a branching structure. The string is then used to generate

Figure 2: Photograph of a poplar tree showing a collar around the branch.

an implicit surface retaining the overall structure of the model described by the string rewriting system.

Figure 3: Photograph of an arbutus tree, showing smooth junctions between branches.

2 Background

Research into methods for modeling botanical structures initially focused on the skeletal branching structure of trees and herbaceous plants [3, 15, 23, 38, 43]. The simplest step in the quest for realism was the representation of internodes as 3D cylinders [11, 32, 33]. A visually more advanced technique was the use of generalized cylinders [5, 13, 22, 24, 28, 29, 34]. Unfortunately, generalized cylinders do not properly capture the smooth geometry of branching points, such as that shown in Figure 3. Additionally, the resulting self-intersecting meshes create problems for some non-photorealistic rendering methods [39].

Modeling of the geometry around branching points was first addressed by Bloomenthal, who initially proposed to solve this problem by crafting parametric patches [5]. Unfortunately, it is difficult to properly fit patches at branching points so that they smoothly connect with the surfaces representing branches, and the method could not be easily extended to allow for tri-furcation.

Bloomenthal also proposed the use of implicit surfaces [7], and convolution surfaces [8], which address the problem of fitting a surface to a branching structure in a general way. A common side effect of these methods is bulging, an unnatural-looking increase in girth where two or more branches are blending together [6]. To solve this Bloomenthal introduced combination surfaces [7] which eliminate bulging, but introduce small C^1 discontinuities and reduce the radius of blending. Combination blending was used by Maritaud [27] in combination with procedural bark texturing methods. However, the results suffer from the reduced radius of blending and the model fails to capture the branch bark ridge. Hart [21] also modeled trees using implicit methods and a procedural bark texture, but the method was incomplete and suffered the same drawbacks as Maritaud's approach.

A different solution to the modeling of branching points is to make use of subdivision surfaces. This technique was pioneered by Tobler et. al. [42], who procedurally grew a mesh by repetitively adding simple predefined blocks to the initial mesh, then subdivided the resulting control mesh. Related techniques, using subdivision to model branches, were introduced by Weta Digital in the film "Lord of the Rings: The Two Towers"[1] and by Akleman et. al. [2]. These techniques are globally smooth, and fail to capture the smaller rough details. Additionally, it is problematic to create the initial subdivision mesh for arbitrarily complex branching structures.

2.1 Implicit Surfaces

An implicit surface S may be derived from a field function $F(x, y, z)$, and is defined as the set of points $P = (x, y, z)$ at which the value of F equals some threshold value T, as follows:

$$S = \{P = (x, y, z) \in \mathbb{R}^3, F(x, y, z) = T\} \qquad (1)$$

Implicit modeling is a natural method for representing solid objects, particularly when blending between primitives is desired. All that is required for visualization is a field value for each sample point, which can be obtained by using a blending function to combine the field values from different primitives. Skeletal implicit surfaces have the advantage that the skeleton can be articulated, allowing realistic natural models to be built and animated.

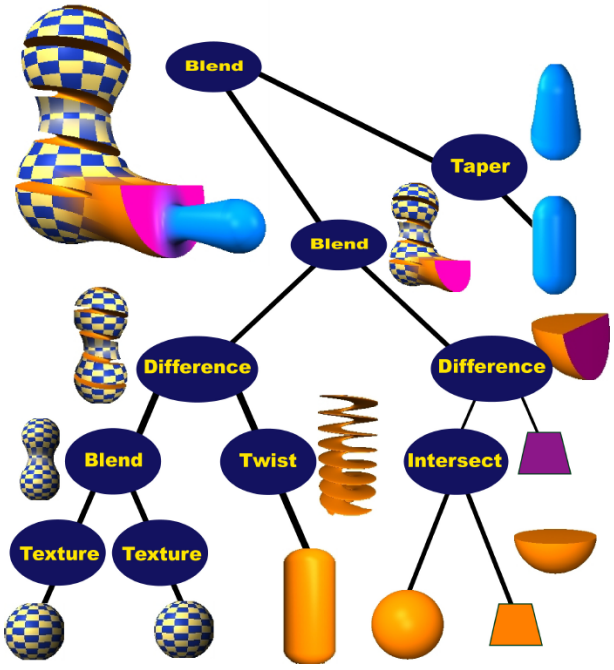

Figure 4: A model built from the BlobTree.

Implicit surfaces are gaining popularity because of recent advances in skeletal implicit models, improved implicit animation techniques, and the suitability of this methodology for collision detection and response [9]. The traditional bottleneck of implicit surface modeling is that visualization of implicit models requires considerable computing power. Continuing research in this area, combined with increasing computing power, has improved performance to the point where complex implicit surface models can be visualized in a reasonable time frame [44, 46, 20, 36, 14].

2.2 The *BlobTree*

The *BlobTree* provides a hierarchical data structure for the definition of complex models based on implicit surfaces. The *BlobTree* uses implicit surface primitives as leaf nodes and operations on arbitrary implicit surfaces as internal nodes (see Figure 4). The *BlobTree* is an extensible data structure, currently supporting the following modeling techniques: CSG, space warping, 2D texture mapping [40], controlled blending [19], precise contact modeling [18], local refinement [25] and R-functions [30]. As Pasko [16] points out, there are problems with using the max/min functions for CSG in certain circumstances. However, these are not problems in the case of the tree examples presented in this paper.

When visualizing the *BlobTree*, the threshold value $T = 0.5$ is typically used when evaluating Equation 1.

Figure 5: Example implicit model, Murex cabritii seashell.

This choice supports definition of primitives with a finite range of influence, which allows for accelerated visualization techniques [14]. Operations on the *BlobTree*, such as visualization of a surface, depend only on the ability to evaluate the field function at any point in space, which is performed by a traversal of the *BlobTree*. For example, a super elliptic blend of m primitives can be expressed functionally as a blend of arbitrarily defined child functions F_i as follows [35]:

$$F(x,y,z) = (\sum_{i=0}^{m} F_i(x,y,z)^n)^{1/n} \qquad (2)$$

The power of the *BlobTree* comes from the fact that the large number of operations mentioned above are easily implemented as operations performed in the context of the tree traversal [45]. Using the *BlobTree* is is possible to create models of complex biological objects (see Figure 5).

Specification of *BlobTrees* can be done through a specialized interactive user interface, or using a procedural approach either through the interpreted language Python [41], or the C++ programming language. This approach, together with the intuitive appeal of combining geometric skeletal elements to form complex models, makes implicit modeling an appropriate choice for solving many problems [17].

2.3 Precise Contact Modeling

Precise Contact Modeling (PCM) is a method of deforming implicit surfaces in contact situations to maintain a precise contact surface with C^1 continuity. Originally described by Gascuel [18], a more efficient approach was introduced by Desbrun [12]. The method is only an ap-

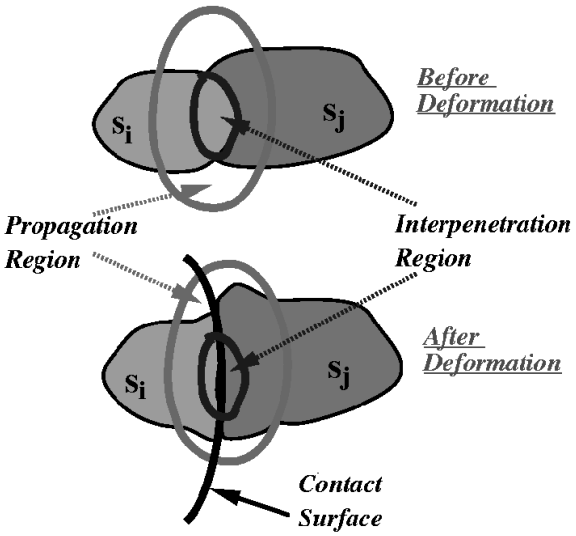

Figure 6: Precise contact modeling objects before and after deformation.

proximation to a properly deformed surface, but is an attractive algorithm due to its simplicity.

The deformation calculation is applied by adding a deformation term d to the implicit function F. There are three cases when calculating the deformation term d as shown in Figure 6:

1. **Within the interpenetration region:** In this region the deformation should model the compression so that a contact surface between objects can be generated. As a result, the deformation value is negative.

2. **Within the propagation region:** This region defines a smooth junction between the compressed portion of the surface (the contact surface) and the non-deformed portion of the surface. In this region the deformation value is positive, modeling local expansion due to compression in the interpenetration region.

3. **Elsewhere:** The surface isn't deformed beyond the propagation region. In this area, the deformation value is zero.

Given a set of n primitives L, with elements $l_i \in L, i \in [0, n-1]$, and corresponding field functions $f_{l_i} : \mathbb{R}^3 \rightarrow \mathbb{R}$, where $l_{max} \in L$ is the primitive with maximal field value at evaluation point p, F is determined as follows:

$$\begin{aligned} F_0 &= f_{l_{max}} \\ F_i &= \begin{cases} F_{i-1} + d(F_{i-1}, f_{i-1}) & l_{i-1} \neq l_{max} \\ F_{i-1} & \text{otherwise} \end{cases} \\ F &= F_{n-1} \end{aligned} \tag{3}$$

The field value is first determined by the maximum of the primitives' field values. It is then iteratively deformed by the field values of all the other primitives. The deformation term defined by d is dependent upon the field value being deformed and the field value which is deforming it. A complete description of this function is given by Cani [10].

3 BlobTree Trees

3.1 Overview

A botanical tree is first described using L-systems. An instance of a *BlobTree* data structure is then built by interpreting the L-system string. To specify a branching hierarchy, the input must define length, base width, tip width, and position of each internode (branch segment). In addition, connectivity must be specified. The L-system description provides all of the necessary information, and thus botanically based simulations may be used to construct the *BlobTree* trees. Other methods can also be used to define the skeleton.

From the L-system specification, a *BlobTree* is constructed in which each internode is represented by an individual skeletal implicit primitive. The *BlobTree* hierarchy is used to define which primitives will blend with each other, and which will not. We specify additional non-blending *BlobTree* operations between primitives depending on their relationship within the branching hierarchy obtained from the L-system. The final step is the application of a bark texture using a skeletal implicit surfaces 2-D texturing algorithm.

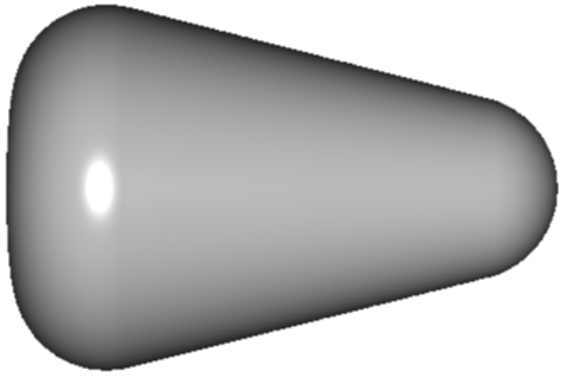

Figure 7: A BlobTree *cone primitive.*

3.2 Modeling Internodes

To model individual internodes a cone primitive is used, as seen in Figure 7. The radius varies linearly from top to bottom to create a gradually tapering branch. The radius of the 0.5 iso-surface surrounding the primitive is equal

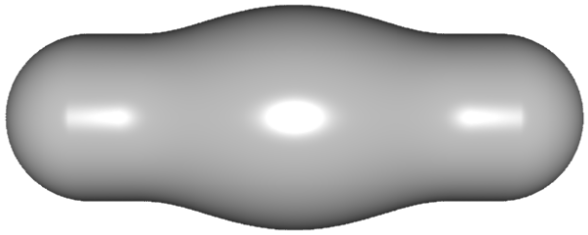

Figure 8: Bulge when two line segments with coincident end points are blended.

to the radius at the tip of the internode.

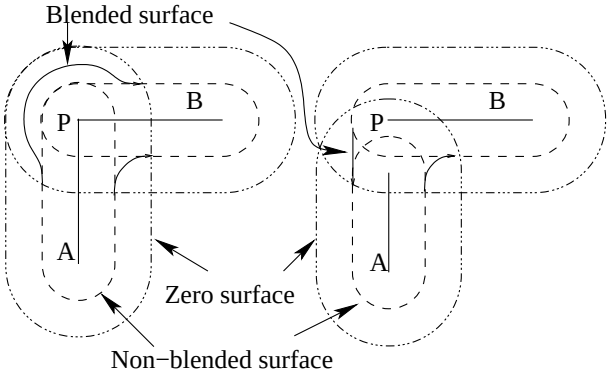

Figure 9: Left: the two skeletal primitives A and B intersect at branching point P, resulting in bulging. Right: the length of primitive A is shortened to reduce bulging.

A common problem when using skeletal implicit surfaces is the unwanted bulging which can occur when blending skeletal elements (Figure 8). Bloomenthal noted [7] that the amount of bulge when using convolution surfaces may be reduced by offsetting the skeletal elements. The same approach may be used with skeletal implicit surfaces. With appropriate choices for size and position of the skeletal elements, the amount of bulging can be reduced to an acceptable level for the modeling of trees, while retaining smooth blends where desired.

The length of the skeletal primitive representing the basal internode is reduced by an amount δ_A, defined in equation 4, where r_B is the radius of internode B and is equal to the distance from B to its 0.5 iso-surface.

$$\delta_A = 1.75 * r_b \tag{4}$$

Figure 9 shows the resulting iso-surfaces when skeletal line primitives A and B are blended. On the left, A and B are connected at the branching point P, resulting in a large bulge. In contrast, the image on the right shows the result of reducing the length of A by δ_A. This method

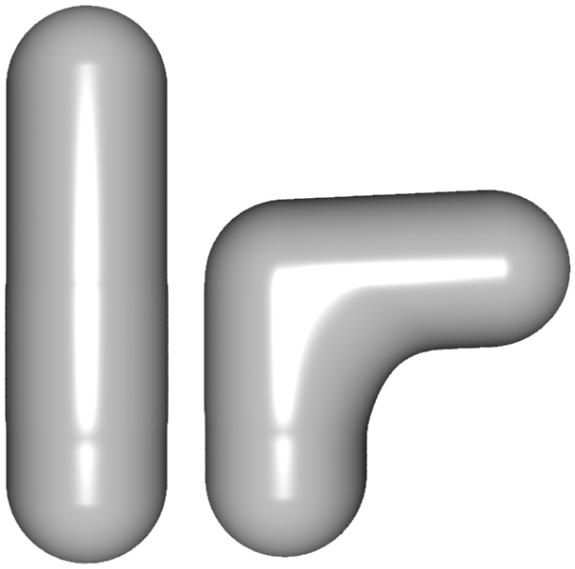

Figure 10: Two cone primitives with their end points offset by a specified distance are blended using a summation blend. Left 180 degrees separation, right 90 degrees.

does not completely remove the bulging between internodes, but makes it practically unnoticeable while maintaining a large blending radius in the interior angle between adjoining internodes, as shown in Figure 10. It was found that the scale factor 1.75 used in Equation 4 could not be varied by more than 3% without affecting the bulge appreciably.

Further to the method introduced above, it is possible to reduce the length of internode B, and displace it away from the branching point P along its axis, while keeping the length of A fixed. It was found that this results in less desirable blends when the radius decreases noticeably from A to B. Additionally, if the lengths of both A and B are reduced so that neither of the skeletal elements actually intersects the branching point P, then an undesirable indentation on the elbow of the blend occurs. Thus the base of B is fixed to the branching point P.

Previous implicit methods for generating branching structures include convolution surfaces [7] and combination surfaces [7]. Convolution surfaces are computationally complex and bulge at branching points. Combination surfaces only have C^0 continuity and are limited to low-radius blends. Summation blending allows for large radius blends with negligible bulging, while maintaining higher continuity than combination blending. Additionally, it does not have the computational complexity inherent in convolution.

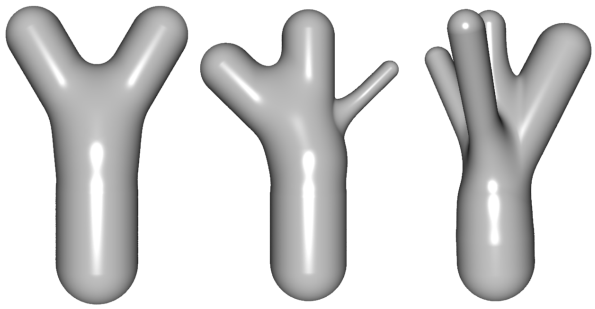

Figure 11: Three examples of branching structures using summation blend.

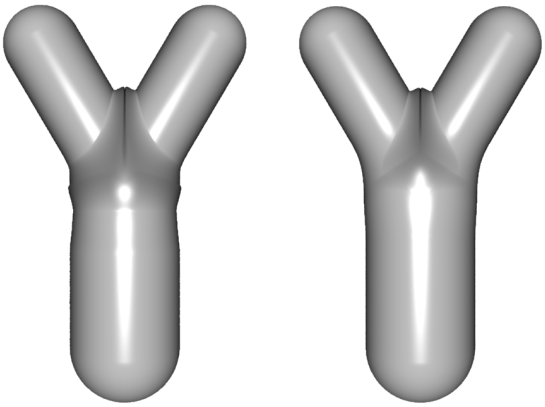

Figure 12: PCM combination of branches blended with basal primitive. Left: Bulging is observed due to branch primitives intersecting at the branching point. Right: bulging removed by offsetting branches from the branching point.

3.3 Branching Points

One of the advantages of using implicit surfaces to model branching structures is the automatic way in which any number of branches may be combined using *BlobTree* operators. The operators used to combine the branches are commutative, and thus the branches may be defined in any order. Additionally, there is no limit to the number of branches that may be combined at a single location. Thus the method is directly applicable to branching structures of arbitrary complexity and configuration. The only problem is to ensure that no additional bulging occurs.

With a summation blend the unwanted bulging is mitigated by the fact that, in trees, the sum of the cross sectional areas of the branching internodes B_i is less than or equal to the cross sectional area of the basal internode A [32]. With the cone primitives being used to model internodes, this implies that the radius of the field due to A is larger than that due to the branches, and this larger field will tend to smooth out any unwanted bulging. Figure 11 shows three examples of branching, and the amount of bulging is minimal in all three. The branching primitives all intersect at the branching point, while the basal internode has been reduced in length by δ_A from Equation 4.

3.4 Precise Contact Modeling at Branching points

While smoothly blending branching points can be generated using a simple summation blend, the resulting models do not model the appearance of the branch bark ridge. To model this phenomenon, Precise Contact Modeling (PCM) is used [18]. PCM provides a good simulation of the growth of bark at branching junctions. Each primitive modifies the field of neighboring primitives' fields by adding to them in the propagation region. Although the mechanism used by PCM differs from the forces involved in the growth of tree bark, similar visual effects are observed. The advantages of using PCM in this way are that it is easy to apply, and that the branch bark ridge can be generated without having to perform a physically-based simulation.

The field F_J of a branching junction J with basal internode A and m branching internodes $B_i, i \in [0, m-1]$ is formed by first combining the branch primitives B_i with each other using PCM to produce the field F_B. F_J is then defined as the blend of F_B and the field F_A due to A:

$$F_B = PCM(F_{B_0}, F_{B_1}, ..., F_{B_{m-1}}) \quad (5)$$

$$F_J = F_B + F_A \quad (6)$$

The left image in Figure 12 shows the result. Unwanted bulging is observed in the region just below the branching point. This is caused by the PCM operation adding material not only in the space between branches, but also on the outer part of the branching junction due to the branching skeletal primitives intersecting, and thus overlapping, at the branching point.

In Section 3.2 it was required that the base of branching skeletal elements B intersect with the branching point P to produce natural-looking blends between internodes. To apply PCM at a branching junction without bulging, this requirement must be altered. To reduce the bulging, each branch B_i is displaced away from the branching point P along its axis. The amount of displacement δ_{B_i} is given by Equation 7, where r_A is the radius of the basal internode A, and r_i is the radius of branch i.

$$\delta_{B_i} = r_A - r_i \quad (7)$$

To avoid a loss of blending between the basal primitive A and the branches, the length of A is increased by $r_A - max(r_i)$, where $max(r_i)$ is the maximum of the radii of the child branches. This results in two equations for

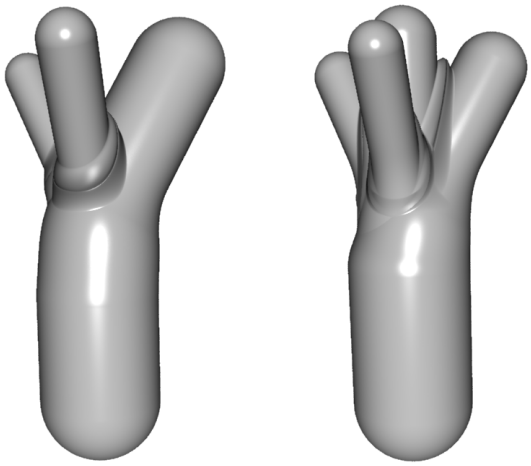

Figure 13: Use of PCM in higher order branching situations.

the value of δ_A, which is used to reduce the length of the basal primitive. Equation 4 is used when the basal internode has a single branch at its tip, and Equation 8 is used when multiple branches are present.

$$\delta_A = 2.75 \times max(r_i) - r_A \tag{8}$$

The resulting blend when using this formulation is shown in the right image of Figure 12.

This method extends easily to higher order branching nodes as shown in Figure 13. Three effects of this formulation are observed: the first observation is the formation of ridges between branches; the second is that large branches continue to blend smoothly with the basal branch; the third is that smaller branches exhibit a collar where they blend with the main trunk. As a lateral branch is increased in diameter, the collar will gradually become a smooth blend with the main branch while maintaining a branch bark ridge with the other branches present. All three of these effects are present in real trees.

3.5 Modeling Missing Organs

Trees, and indeed all plants in nature, often lose lateral organs for a variety of reasons. Shedding organs can be intentional, as in the case of deciduous leaves, or unintentional, as when a branch is broken. We would like to capture these effects in our model.

The surface left behind when a branch is broken is by no means simple to define, and this phenomenon is not present in the method presented. We speculate that CSG may be used to capture these effects. As a simple example CSG is used to simulate the effect of pruning a branch by cutting it, shown in Figure 14.

To model the scars left behind when a plant organ is lost, a negative potential field is used to make an indentation into the structure of the tree. This negative potential field is blended with the *BlobTree* in the same location at which a shortened branch stub has been placed. By varying the radius of influence of the branch stub and the negative potential field a variety of different sized scars can be achieved, shown in Figure 15.

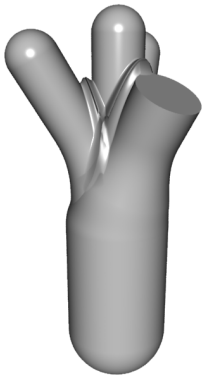

Figure 14: Use of CSG to prune branches as if they were cut off with a saw.

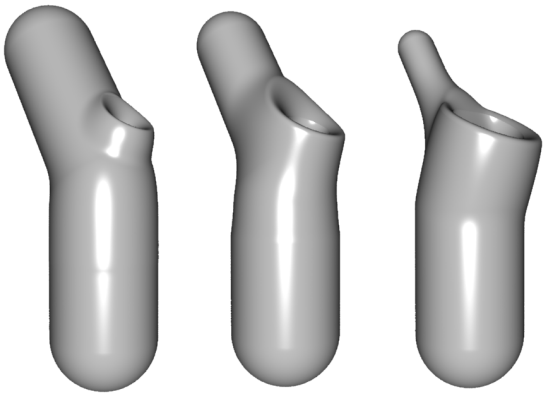

Figure 15: Use of negative potential fields to create scars where organs have been lost.

3.6 Bending the Branches

Previous work involving tree modeling has focused on generating perfect versions of real world plant structures. As a result, many models of plant structures tend to be too rigid, and lack a feeling of life. The *BlobTree* provides a set of spacial warping operators which are used to reduce the regularity present in the model. A good example is to use the Bend operator [4] to alter the perfectly straight individual internodes.

Randomly applying bend results in a very artificial look. Better results can be obtained by applying bend to individual skeletal primitives based on their position

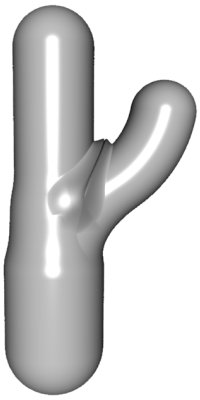

Figure 16: Use of spacial warping to bend internodes, the bend is applied in the plane defined by the bent internode and its basal internode.

in the hierarchy and their relationship to adjacent internodes.

The direction is chosen so that the result of the bend will cause the bent internode to grow more directly away from the axis of its basal internode, and slowly bend back toward its defined termination point. This choice is based purely on aesthetics and is not based on biological principles.

The bend is parameterized so that regardless of what angle is chosen, or what plane the bend occurs in, the bent internode will have its initial and terminal points at the same location as if it were not bent. This allows individual internodes to be bent without altering the connectivity of the surface or modifying the global position of any internodes. In order to achieve this effect, rotations are applied, and the length of the internode is increased depending on the amount of bend.

Figure 16 shows a simple example of bend. The branch segment which is collinear with the basal internode is not bent at all. The lateral branch is bent in the plane defined by its original position and its basal internode, and its direction of bend causes it to grow more directly away from the axis of the basal internode, then bend back toward its original terminal point.

3.7 Texturing *BlobTree* Trees

The final step in building the tree model is to apply texture to give a more realistic appearance. The *BlobTree* defines several methods that can be used to apply 2D texture; the most straightforward of them is used [40]. Individual *BlobTree* primitives can be described parametrically, and thus have their own natural 2D coordinate systems. The global surface attributes at any point will be determined by a linear combination of the attributes defined by the contributing primitives at that point. Each contributing primitive uses its native 2D coordinates to determine its contribution. This method results in an automatic blending of textures across branching points. The effects are good when viewing the whole tree. However, when viewed from close up the method gives a slightly unnatural look to the texture within regions of blend if the texture contains an anisotropic pattern. The poplar texture used in Section 4 does not have this property and gives good results for all levels of detail.

Figure 17: A poplar tree with smoothly blending branches.

4 Results

Figures 17 and 18 show the results of our model with smoothly blending branches and with the use of PCM. PCM allows us to more accurately model the phenomenon of the bark branch ridge shown in Figure 2.

The final images were rendered on an Intel(R) Xeon(TM) CPU 2.66GHz with 1 Gb of RAM using direct ray tracing methods. It required just under 1 hour to render Figures 17 and 18 at a resolution of 1536x2048 pixels while super sampling with up to 9 samples per pixel. The *BlobTree* may also be viewed using OpenGL with an approximate polygon mesh. However, it requires at least 20 seconds to generate enough polygons to be use-

Figure 18: A poplar tree with PCM to model the branch bark ridges.

ful, and the resulting discretization of the implicit surface does not accurately represent the topology of the underlying surface. This prohibits the use of the method for interactive modeling, but does make it feasible as a visualization method for procedural modeling. The lengthy visualization times are offset by the power of the modeling paradigm.

Current problems with the method are as follows: the branches are currently tipped by hemispherical caps, and more work needs to be done to model the branch tips correctly; also, it is difficult to control the appearance of branch bark ridges and collars using the current formulation. As a result of the latter problem, producing models of arbitrary trees with varying properties at the branch bark junction is not currently possible.

5 Conclusions and Future Work

An outstanding problem in the modeling of trees is the construction of a surface which captures both the smooth and non-smooth attributes of their branching structure. Previous methods have focused on smoothly blending branching junctions and simulated rough bark texture, but have failed to model the branch bark ridge. The method presented combines smoothly blending branching junctions with those that exhibit the branch bark ridge and collars. The method relies upon the *BlobTree* to provide both smooth and non-smooth operators to combine skeletal implicit surface primitives.

A method was introduced to specify a set of skeletal implicit primitives which define a non bulging and smoothly blending surface for an arbitrary branching structure. The method compares favorably to two other implicit methods used to construct similar surfaces. It is easier to implement and less computationally expensive to visualize than convolution surfaces, and provides a larger blending radius than combination blending, while retaining C^1 continuity.

Precise Contact Modeling was used to model the branch bark ridge and collars, two phenomena which have not been modeled by any previous technique. The resulting surfaces are similar to the phenomena observed in nature. A model of a Poplar tree was presented illustrating the effect. Additional operations, such as spacial warping and Constructive Solid Geometry, can be used to enhance realism in the resulting model.

The method proposed enables the production of complex models not easily obtained by other means. The power of the modeling paradigm thus compensates for the lengthy visualization times.

Input was generated using the L-system formalism, which allows the method to be used for realistic visualization of biological simulations of trees. The input is not limited the L-system formalism, and interactive techniques may be used to specify the underlying branching structure.

Future areas of research are:

- The deformed fields resulting from PCM may be used as input to a procedural texturing algorithm to modify the texture on the branch bark ridge, where a discolouration is observed.
- Providing greater control over the shape of the branch bark ridges.
- Correlating the resulting branch bark ridges with biologically accurate measurements of this phenomena.
- Incorporate information about the development of non-smooth surface features into the underlying L-system model.
- Using the model to create animation of tree growth, including the effects over time resulting from discarded organs.

Acknowledgements

The authors would like to thank the many students, past and present who have contributed toward the implicit modeling research project at the University of Calgary, in particular Dr. Prusinkiewicz, whose L-system tree models and work in teh area of sub-division surfaces inspired this work. Mark Fox was instrumental in the design phase of the current *BlobTree*. We would also like to thank the Department of Computer Science. This work is partially sponsored by the Natural Sciences and Engineering Research Council.

References

[1] M. Aitken and M. Preston. Foliage generation and animation for “The Lord of the Rings: The Two Towers”. SIGGRAPH 2003 DVD-ROM, ACM SIGGRAPH, New York, 2003.

[2] E. Akleman, J. Chenb, and V. Srinivasan. A minimal and complete set of operators for the development of robust manifold mesh modelers. *Graphical Models*, 65(5):286–304, September 2003.

[3] M. Aono and T. L. Kunii. Botanical tree image generation. *IEEE Computer Graphics & Applications*, 4(5):10–34, May 1984.

[4] A. H. Barr. Gloabl and Local Deformations of Solid Primitives. In *Proceedings of SIGGRAPH 84*, volume 18 of *Computer Graphics Proceedings, Annual Conference Series*, pages 21–30, 1984.

[5] J. Bloomenthal. Modeling the Mighty Maple. In *Proceedings of SIGGRAPH 1985*, volume 19, pages 305–311. ACM, July 1985.

[6] J. Bloomenthal. Bulge elimination in implicit surface blends. In *Implicit Surfaces '95*, April 1995.

[7] J. Bloomenthal. *Skeletal Design of Natural Forms*. Ph.D. dissertation, University of Calgary, 1995.

[8] J. Bloomenthal and K. Shoemake. Convolution surfaces. In *Proceedings of SIGGRAPH 1991*, pages 251–256. ACM, 1991.

[9] Jules Bloomenthal. *Introduction to Implicit Surfaces*. Morgan Kaufmann, ISBN 1-55860-233-X, 1997. Edited by Jules Bloomenthal With Chandrajit Bajaj, Jim Blinn, Marie-Paule Cani-Gascuel, Alyn Rockwood, Brian Wyvill, and Geoff Wyvill.

[10] Marie-Paule Cani. Layered models with implicit surfaces. In *Graphics Interface (GI'98) Proceedings*, Vancouver, Canada, Jun 1998. Invited paper, published under the name Marie-Paule Cani-Gascuel.

[11] P. de Reffye, C. Edelin, J. Françon, M. Jaeger, and C. Puech. Plant models faithful to botanical structure and development. In *Proceedings of SIGGRAPH 1988*, pages 151–158. ACM, 1988.

[12] Mathieu Desbrun and Marie-Paule Cani-Gascuel. Active implicit surface for animation. *Graphics Interface '98*, pages 143–150, June 1998. ISBN 0-9695338-6-1.

[13] O. Deussen and B. Lintermann. A modelling method and interface for creating plants. In *Graphics Interface '97*, May 1997.

[14] Mark Fox, Callum Galbraith, and Brian Wyvill. Efficient Implementation of the BlobTree for Rendering Purposes. *Proc. Shape Modelling International, Genova, Italy*, May 2001.

[15] D. Frijters and A. Lindenmayer. A model for the growth and flowering of Aster novae-angliae on the basis of table (1,0)L-systems. In G. Rozenberg and A. Salomaa, editors, *L-systems, Lecture Notes in Computer Science 15*, pages 24–52. Springer-Verlag, Berlin, 1974.

[16] M. Ikeda T. Kunnii title = "Bounded Blending Operations" booktitle = Proceedings of the International Conference on Shape Modeling and Applications (SMI 2002) year = 2002 month = May isbn = 0-7695-1546-0 pages = 95–103 publisher = IEEE Computer Society G. Pasko, A. Pasko.

[17] Callum Galbraith, Przemyslaw Prusinkiewicz, and Brian Wyvill. Modeling a murex cabritii sea shell with a structured implicit surface modeler. *The Visual Computer*, 18(2):70–80, 2002.

[18] Marie-Paule Gascuel. An Implicit Formulation for Precise Contact Modeling Between Flexible Solids. *Computer Graphics (Proc. SIGGRAPH 93)*, pages 313–320, August 1993.

[19] Andrew Guy and Brian Wyvill. Controlled blending for implicit surfaces. In *Implicit Surfaces '95*, April 1995.

[20] J.C. Hart. Sphere tracing: A geometric method for the antialiased ray tracing of implicit surfaces. *The Visual Computer (in press)*, December 1996.

[21] John C. Hart and Brent Baker. Implicit modeling of tree surfaces. In *Proceedings of Implicit Surfaces '96*, pages 143–152, October 1996.

[22] M. Holton. Strands, gravity, and botanical tree imagery. *Computer Graphics Forum*, 13(1):57–67, 1994.

[23] H. Honda. Description of the form of trees by the parameters of the tree-like body: Effects of the branching angle and the branch length on the shape of the tree-like body. *Journal of Theoretical Biology*, 31:331–338, 1971.

[24] M. Kitagawa De Leon. Branching object generation and animation system with cubic Hermite interpolation. *Journal of Visualization and Computer Animation*, 2(2):60–67, 1991.

[25] X.K. Liang and Brian Wyvill. Hierarchical implicit surface refinement. *Proc. Computer Graphics International, Hong Kong*, pages 291–298, 2001.

[26] A. Lindenmayer. Mathematical models for cellular interaction in development, Parts I and II. *Journal of Theoretical Biology*, 18:280–315, 1968.

[27] K. Maritaud, J.-M. Dischler, and D. Ghazanfarpour. Rendu réaliste d'arbres à courte distance. *13èmes journées de l'AFIG*, pages 41–51, 2000.

[28] R. Měch. *Modeling and simulation of the interaction of plants with the environment using L-systems and their extensions*. PhD thesis, University of Calgary, 1997.

[29] P. E. Oppenheimer. Real time design and animation of fractal plants and trees. In *Proceedings of SIGGRAPH 1986*, volume 20, pages 55–64, August 1986.

[30] A. Pasko, V. Adzhiev, A. Sourin, and V. Savchenko. Function representation in geometric modeling: concepts, implementation and applications. *The Visual Computer*, 2(8):429–446, 1995.

[31] P. Prusinkiewicz, M. Hammel, J. Hanan, and R. Měch. *Handbook of Formal Languages*, chapter Visual models of plant development. Springer–Verlag, Berlin, 1996.

[32] P. Prusinkiewicz and A. Lindenmayer. *The Algorithmic Beauty of Plants*. Springer-Verlag, New York, 1990.

[33] P. Prusinkiewicz, A. Lindenmayer, and J. Hanan. Developmental models of herbaceous plants for computer imagery purposes. In *Proceedings of SIGGRAPH 1988*, pages 141–150. ACM, 1988.

[34] P. Prusinkiewicz, L. Mündermann, R. Karwowski, and B. Lane. The use of positional information in the modeling of plants. In *Proceedings of SIGGRAPH 2001*, pages 289–300. ACM, August 2001.

[35] Ricci. Constructive Geometry for Computer Graphics. *computer journal*, 16(2):157–160, May 1973.

[36] Andrei Sherstyuk. Fast Ray Tracing of Implicit Surfaces. *Computer Graphics Forum*, 18(2):139–148, June 1999.

[37] Alex Shigo. Tree Autopsy . *Tree Care Industry*, 7(6), 1996.

[38] A. R. Smith. Plants, Fractals and Formal Languages. In *Proceedings of SIGGRAPH 1984*, pages 1–10, New York, 1984. ACM, ACM.

[39] M.C. Sousa and P. Prusinkiewicz. A few good lines: Suggestive drawing of 3d models. *Computer Graphics Forum (Proc. of EuroGraphics '03)*, 22(3), 2003.

[40] Mark Tigges and Brian Wyvill. Texture Mapping the BlobTree. *Implicit Surfaces*, 3, June 1998.

[41] Mark Tigges and Brian Wyvill. Python for scene and model description for computer graphics. *Proc. IPC 2000*, January 2000.

[42] R. F. Tobler, S. Maierhofer, and A. Wilkie. A Multiresolution Mesh Generation Approach for Procedural Definition of Complex Geometry. In *Proceedings of the International Conference on Shape Modeling and Applications (SMI 2002)*, pages 35–42. IEEE Computer Society, May 2002.

[43] X. G. Viennot, G. Eyrolles, N. Janey, and D. Arques. Combinatorial Analysis of Ramified Patterns and Computer Imagery of Trees. In *Proceedings of SIGGRAPH 1989*, pages 31–40. ACM, 1989.

[44] Andrew Witkin and Paul Heckbert. Using particles to sample and control implicit surfaces. *Computer Graphics (Proc. SIGGRAPH 94)*, 28:269–277, July 1994.

[45] Brian Wyvill, Eric Galin, and Andrew Guy. Extending The CSG Tree. Warping, Blending and Boolean Operations in an Implicit Surface Modeling System. *Computer Graphics Forum*, 18(2):149–158, June 1999.

[46] Geoff Wyvill, Craig McPheeters, and Brian Wyvill. Data Structure for Soft Objects. *The Visual Computer*, 2(4):227–234, February 1986.