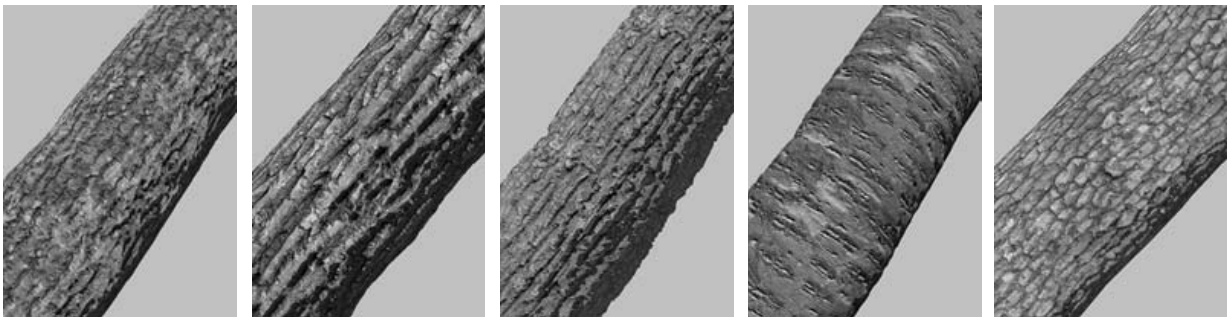
Interactive Modeling of Tree Bark

Xi Wang[‡] Lifeng Wang Ligang Liu Shimin Hu[‡] Baining Guo
[‡]Tsinghua University Microsoft Research Asia



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

Figure 1. Different bark features modeled with our techniques: (a) fracture, (b) furrowed cork, (c) ironbark, (d) lenticel, and (e) tessellation.

Abstract

There exist many computer graphics techniques which could achieve high quality tree generation. However, only a few works focus on realistic modeling of tree bark. Difficulties lie in the complex appearance of the bark surfaces which are largely determined by their mesostructures. Unlike traditional physical based methods, in this paper, we present an appearance-based method to model tree bark surfaces from a single image. We address three main issues here: feature specification, height field assignment and texture correction. For feature specification, we use texton channel analysis to specify a variant of common bark features, including ironbark, vertical and horizontal fractures, tessellation, furrowed cork, and lenticels. For height field assignment, we develop an intuitive and easy-to-use user interface (UI). Here similarity-based texture editing is used for assigning height fields within a texton channel mask. For texture correction, we use the modeled height fields to eliminate the underlying lighting effects in a captured texture. Our modeling system is image-based: it takes as input a bark image and produces as output a textured height field representing a bark sample. We demonstrate that our method is an effective and easy-to-use technique to interactively model a variety of photo realistic bark surfaces.

Keywords: mesostructure, natural phenomena, tree rendering, segmentation

1 Introduction

In computer graphics, surface geometry is typically represented in multiple levels of detail. At the highest level, meshes provide an efficient representation of gross shape, while bi-directional reflectance functions (BRDFs) describe the light scattering effects of micro-scale material structure. In between these two levels is the mesostructure [11, 3], which contains the visible high-frequency geometric details that may not be efficiently processed within a mesh representation. Bark is a surface with a complex appearance which is largely determined by mesostructures. Many papers such as [4, 14, 20] focus on tree modeling in the highest level and ignore the mesostructure. In this paper, we present an approach for realistic modeling of bark's mesostructures, thus provide the possibilities to realistically render the associated lighting effects.

Images of bark features are shown in Fig. 3. For modeling these surfaces, an immediate idea is to recover the 3D information of its surfaces with vision technologies. Shape from shading is an apparent choice. But these algorithms do not work because of the surface discontinuity and lighting estimation, which results in difficulties to recover shapes with luxuriant texture colors. Another choice is the physical based method. But the biological and physical mechanisms behind these bark features are enormously complex and probably too difficult to simulate on today's computer.

In this paper, the proposed modeling system is interactive and image based: it takes as input a bark image and produces as output a textured height field representing a bark

sample. Our bark modeling system is developed with two main objectives in mind. First, we want a general system that can capture a variety of bark features. Second, we want the system to be easy to use. The user should be able to create a bark sample with a few minutes of interactive editing, and the editing process ought to give the user intuitive control of bark's appearance. To design a system not restricted to a particular bark feature, we choose an image-based approach. This is an approach pioneered by Bloomenthal [1], who built a bump map from an x-ray image of real bark. Unlike [1], we only require a photograph, which is much easier to obtain than an x-ray image. In our system, the bark image serves as a guide to the user for creating a bark sample of desired appearance. We do not attempt to reconstruct the precise height field of the bark in the input image. In fact, such a reconstruction is not possible since we only have a single image with unknown lighting [10].

With our system, creating a tree bark sample is easy. The user first segments the bark image into different features and then applies appropriate editing operations to model the height field of each feature. A key ingredient of our system is the texton channel analysis proposed by Malik et al. [16]. With texton analysis, our system can automatically segment the bark image into a small number (≈ 25) of channels corresponding to different visual phenomena. The user then merges these channels into a few channels representing the bark features to be modeled. The merging is easy and can be done with a few seconds of user interaction.

The reason that texton channel analysis is so effective with bark mesostructures is that, at the local scale, there is only a small number of perceptually distinguishable mesostructure prototypes [16]. Surface features such as ridges and grooves could occur at a continuum of orientations and heights, but perceptually we can only distinguish them up to an equivalent class. The number of prototypes is further reduced by the fact that we are working with textures, which are spatially repeating by definition.

Once the bark image is segmented into texton channels, we need to construct the height field within each texton channel respectively. To make this construction easy, we adopt an interactive texture editing technique proposed by Brooks and Dodgson [2]. Their technique makes use of the self-similarity within a texture to replicate local editing operations globally. With this technique incorporated, our system allows the user to interactively construct the height field of a bark sample with a few mouse clicks.

2 Related Work

There exist two approaches to bark modeling. One approach uses textures, which can be either procedural or image based. Oppenheimer textured tree surfaces with a bump map made from a simulated bark texture, which he obtained by adding noise to a ramp and passing the result through a sawtooth function [18]. Hart and Baker built tree surfaces using implicit surfaces with geometric bark textures generated by particle flow approximation [8]. These procedural techniques are usually restricted to particular tree species and much parameter tweaking is necessary to get a desired effect. Image-based techniques (e.g., [1]) often have superior realism.

The other approach to bark modeling is physics-based simulation. Researchers following this approach mostly focused on fractures. Federl and Prusinkiewicz simulated bark with a mass-spring network placed on a tree surface [6]. Their method rely on mass-spring networks mapped onto the tree surface. When a string breaks and the crack propagates through the surface, a fracture is generated. Hirota et al. extended [6] to model cracks with a multi-layer mass-spring network [9]. Both their methods provide textures with small-scale cracks, but cannot represent open fractures. Moreover, their methods require lots of springs to achieve their requirements, as details are only created by the simulation. In principle, their methods are also applicable to bark simulation. Recently, Lefebvre and Neyret proposed a model for simulating vertical fractures, which are generated as either texture or geometry [13]. This method is a simplified physical based method, which only handles fractures. Terzopoulos and Fleisher first introduced an early representation of fractures together with a model of elastic, plastic, and visco-elastic continuous material [22]. Fractures were represented as discontinuous in their paper.

Graphics researchers have investigated techniques for recovering mesostructure from images and these techniques are applicable to image-based bark modeling. Liu et al. reconstructed mesotructures using shape from shading [15]. Rushmeier et al. applied photometric stereo to bump map capturing [21]. Both [15] and [21] require multiple images with known illumination. Recently, Dischler et al. proposed a technique for recovering a bump map from a single image [5]. A technique with similar functionality is described in [12]. Our bark modeling system could potentially use [12, 5] for estimating an initial height field, which can be then interactively edited to get the desired result. However, [12, 5] come with their restrictions. For example, [5] requires that the large-scale relief in the bark can be characterized by a single curve and that the user must supply a training set for segmenting large-scale relief.

3 Bark Modeling

In this section, we describe a two-step process for generating the mesostructure of a given tree surface. The first step is the specification of the bark features, which segments a variety of bark features. The second step is the bark sample construction, which produces a textured height field repre-

senting a bark sample. Fig.2 illustrates these 2 step with correspondent components: specification component and editing component.

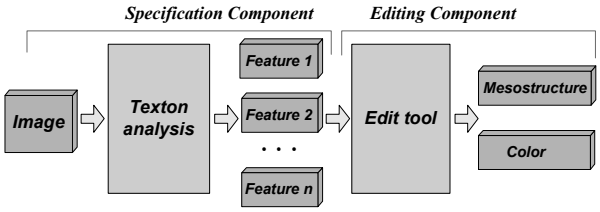


Figure 2. Framework of our approach.

Our approach builds a bark sample as a height field H_0 with texture T_0 from an input bark image I_0. Note that our goal is not to reconstruct the true height field from I_0 but to create a height field H_0 using I_0 as a guide, which is adequate (good enough) for rendering. Our challenge is to design an interactive modeling system that is easy to use yet powerful enough to capture a large variety of bark features.

3.1 Bark Features

Bark is a living cylinder of tissue which, as the tree grows, periodically produces a new inner layer with water and sugar transport as well as formation of wood fibers, and a new outer protective layer [19]. The previously formed outer protective layer gradually dies. Two things may happen to this new dead outer protective layer: it may be shed or accumulate on the trunk of the tree. In this paper, we use the word "bark" to refer to the dead outer protective layer that is easily seen by the human eye. Some of the common features that we wish to capture are as follows.

ironbark Perhaps the easiest bark to recognize is that of the traditional ironbark shown in Fig. 3 (a). In these species, the rough bark becomes hard and compacted.

fracture As a tree increases in girth, great tension on the bark can cause fractures, which can be either vertical or horizontal. Fig. 3 (b) shows vertical fractures.

tessellation The bark fractures to form flakes or plates as illustrated in Fig. 3 (c). Some bark has large plates with deep furrows.

furrowed cork The cork oak and a number of other trees have deeply furrowed bark with thick accumulation of cork cells. Fig. 3 (d) shows an example.

lenticel One of the small, oval, rounded spots upon the trunk or branch a tree, from which the underlying tissues may be protrudent or cuppy. Lenticels are usually horizontal as shown in Fig. 3 (e).

Our system will have difficulties while modeling a bark sample that cannot be represented by a height field. An example is the so-called "stringy bark", which has many dangling fiber strips as shown in Fig. 3 (f).

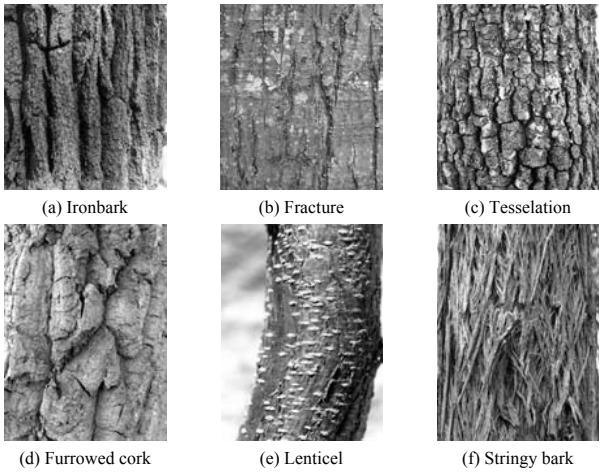


Figure 3. Bark features. Our system can model (a) through (e).

3.2 Texton Channels

The first task in the bark sample construction is to segment the bark image I_0 into regions corresponding to different features. To accomplish this we use the texton channels proposed in [16]. Similar with [16], we filter I_0 with a bank of 36 Gaussian derivative filters. This filter bank consists of two phases (even and odd), three scales (separated by half-octaves), and six orientations (equally spaced from 0 to π). For each pixel of I_0, the filter response is a 36-dimensional data vector. The data vectors of all pixels are clustered using the K-means algorithm [7]. The resulting K-means centers are the textons of I_0. In this paper, we set the number of K-means centers to $K = 25$. Fig. 4 (a) shows pixel-to-texton mapping for the bark image in Fig. 4 (b). Each subimage in Fig. 4 (a) shows the pixels in the image that are mapped to a given texton. The pixel-to-texton mapping gives us a collection of discrete points and this collection of points is the texton channel of the given texton. The K texton channels constitute a partition of the image I_0 because each pixel in I_0 belongs to exactly one texton.

From Fig. 4 (a) and (b) we can make a few interesting observations about the correspondence between the texton channels and bark features. On the one hand, a texton channel usually contains activities in specific regions and nowhere else. As pointed out in [16], the pixel-to-texton mapping provides us with a set of discrete points where we had continuous-valued filter response vectors. The result is a much cleaner representation than filter response vectors, which are generally non-zero at every pixel of I_0. On the other hand, the same bark features (e.g., vertical fractures in Fig. 4 (b)) are usually captured across several channels. This is not surprising since texton channels are results of low-level vision processing. We call such texton channels raw channels.

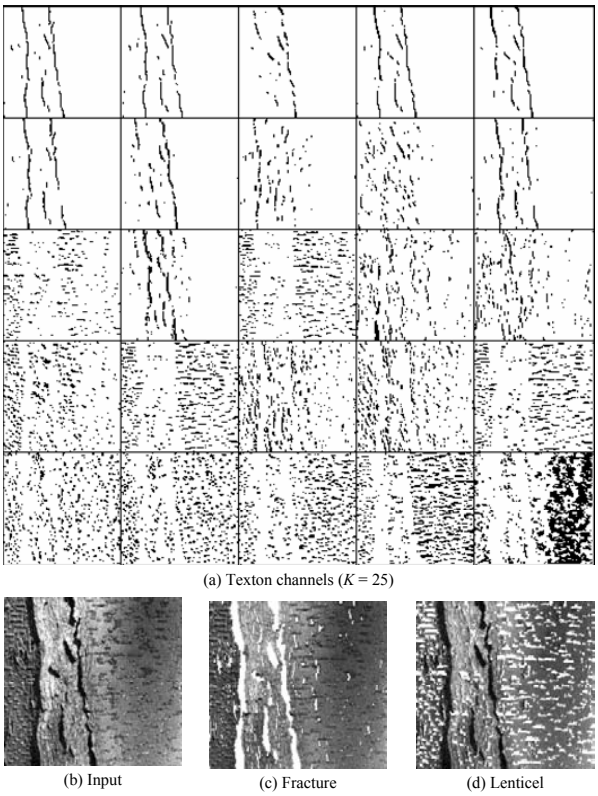


Figure 4. Texton channels. (a) All 25 channels. (b) The corresponding bark image. (c) Channels for fractures. (d) Channels for lenticels. In (c) and (d), texton channels are shown in white.

In order to obtain texton channels suitable for constructing the height field of a bark sample, the user needs to manually merge raw channels. Fig. 5 shows an example of merging the channels corresponding to vertical and horizontal fractures. Here the channel in (b) corresponds to vertical fractures whereas the channel in (c) corresponds to horizontal fractures. Generally speaking, individual raw channels correspond to different visual phenomena at some level but not necessarily the level the user wants to work at. By merging channels, the user can arrive at the level he wants to work at. When merging raw channels, the user makes use of his high-level understanding of the visual information in I_0. Thus the merged channels incorporate both low-level vision processing and high-level human knowledge.

Merging texton channels is easy and can be done with a few seconds of user interaction. As shown in Fig. 5, a texton channel is visualized as points drawn on top of the bark image I_0. We found this to be a good way to provide the user with immediate visual feedbacks during merging.

3.3 Height Field Construction

Having segmented the bark image I_0 into texton channels, we are now ready for the bark sample construction.

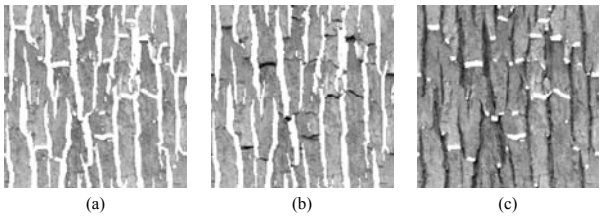


Figure 5. Merging texton channels, which are highlighted in white. (a) is the result of merging (b) and (c).

The UI for building the height field H_0 is illustrated in Fig. 6.

Starting with an H_0 that is zero everywhere, the user interactively edits the height values of H_0 to get the desired final result. The editing is done separately for each texton channel. When the user selects a pixel p_0 and changes its height value, the change is automatically propagated to other pixels within the current channel. Because of this propagation, the user can build the height field within each texton channel by editing just a few height values. As shown in Fig. 6, window (1) is the control panel for setting various parameters. Window (2) displays the bark image I_0 with a texton channel highlighted in white. Window (3) shows the weight mask for height field editing. Window (4) provides a thumbnail selection of the (merged) texton channels, from which the user can select a channel for editing. Window (5) is an interactive 3D rendering of the height field H_0, with the lighting condition of the rendering being controlled by window (6). The key to propagate the change at p_0 to other pixels is the weight mask M_{p_0} shown in window (3) of Fig. 6. Specifically, the height value change at a pixel p_1 is the height value change at p_0 weighted by the value of M_{p_0} at p_1. In window (3), darker pixels indicates smaller weights.

We compute the weight mask using self-similarity of the bark image I_0 [2]. Self-similarity-based editing is an interactive texture editing technique that uses self-similarity within a texture to replicate local operations globally [2]. We adapt this technique for editing the height field within a texton channel, as follows. To compute the weight mask M_{p_0} for the selected pixel p_0, we compare the local circular neighborhood of p_0 against that of every other pixel's neighborhood. The neighborhood of a pixel is defined as those pixels bounded by an immediate circle of pixel diameter d. The weight is one at p_0 and decreases linearly according to the neighborhood distance between the neighborhood of the current pixel and that of p_0. The distance between two neighborhoods is measured by L_2 norm. When the neighborhood distance is beyond a prescribed threshold the weight is set to zero. The weight mask in window (3) of Fig. 6 is computed this way. To avoid sluggish response times with large textures, we use multi-scale neighborhoods

as in [2].

We also experimented other ways to compute the weight mask. The simplest weight mask is the one with the same weight for all pixels. Not surprisingly, this mask leads to artificial-looking height fields. An alternative is to variate the weight according to the grey-scale value of I_0. Through experiments we found out that this technique usually introduces unwanted high-frequency detail in the height field H_0. Our findings are consistent with similar experiments conducted by other researchers (e.g., [5]). For this reason, we only variate the weight according to the grey-scale value of I_0 if we want to add small perturbations to H_0 after we have determined the overall shape of H_0 using self-similarity based editing.

Editing individual channels separately is important. Different channels correspond to different visual phenomena, and the user needs to apply different editing operations based on his understanding of the visual phenomena. Since the texton channels of I_0 form a partition of the image plane, a complete H_0 is constructed when the user finishes editing all channels.

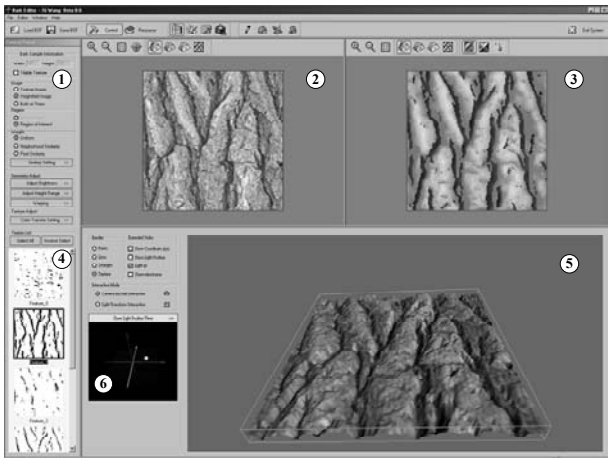


Figure 6. The user interface for building a height field H_0 from a bark image I_0.

3.4 Texture Map Construction

After constructing the height field H_0, we can compute the texture map T_0 from H_0. A simple way to do so is to assign the pixel color of I_0 to the corresponding pixel of T_0. The problem with this method is that the pixel color of I_0 combines both texture and illumination. Factoring the image I_0 into a texture component and illumination component is a challenging task. Oh et al. proposed a decoupling filter in [17]. Unfortunately, their filter works under the assumption that large-scale luminance variations are due to lighting whereas small-scale detail is due to texture. This assumption is not valid for bark images, which are usually full of small-scale detail caused by shading and shadowing of fine geometric features.

We construct the texture T_0 by dividing the image I_0 by $\mathbf{N} \cdot \mathbf{L}$ on a per-pixel basis, where $\mathbf{N}$ is the unit normal of the height field surface and $\mathbf{L}$ is the local lighting direction. $\mathbf{L}$ is obtained by having the user interactively adjusting the light source position to achieve an illumination similar to that of the bark image I_0. The computation could be illustrated as:

$$T_0 = \frac{I_0}{\mathbf{N} \cdot \mathbf{L}}$$

In our experience, this heuristic technique eliminates a significant amount of the illumination component. The elimination would be complete if the following assumptions hold: a) the bark surface is ideally Lambertian, b) the height field H_0 is the true height field of the bark, and c) the only illumination in I_0 is local shading (e.g., no shadows). In practice, none of these assumptions holds strictly and some illumination component remains. Fortunately, this is not a serious problem for bark.

4 Experimental Results

We implement our algorithm in a Pentium IV computer with 1.4G Hz CPU and 256M memory. All input images were captured with a CANNON digital camera. We completed the whole process for a texture in a few minutes.

Fig.7 is the modeled height fields and texture with our algorithm. Fig.7 (a) shows two input images with size of 256×256, Fig.7(b) shows the correspondent constructed height fields represented by gray scale bitmap. Fig.7(c) illustrates the corrected textures respectively, and Fig.7(d) shows the rendering results with OpenGL Phong shading.

The constructed height fields and textures could be mapped or synthesized to rough models to create models with rich mesostructure. Fig.1 shows examples of different modeled bark features (height fields and textures) mapped to a tree trunk model. Fig.1(a)-(e) show fracture, furrowed cork, ironbark, lenticel and tessellation respectively. The rendering was achieved by view-dependent displacement mapping (VDM) in realtime [23]. We could find we obtained high quality modeling by using our approach.

Fig.8 shows similar modeling results for different bark types. In these examples, different constructed bark surfaces were mapped to a original branch model with 5,888 triangles.

Fig.9 shows modeling and mapping results with a whole tree model with 15,138 triangles. Fig.9(a) and (d) show vertical fractures, Fig.9(b) shows tessellation, and Fig.9(c) shows the ironbark surface. All rendering results were obtained by VDM [23].

5 Discussion and Future Work

Our modeling approach is easy-to-use and effective. A user could easily build height fields and correct textures from a single image with a few mouse clicks in a few minutes. Our approach provides an effective and interactive modeling tool for a variant of bark types.

Our construction of height fields H_0 starts with H_0 being zero everywhere. A potential improvement is to start with a better initial guess of H_0. We tried to obtain a good initial H_0 by extending a shape-from-shading technique [12]. The extended technique accounts for discontinuities, shadows, and uncertainty of the lighting direction. With this technique we succeeded in recovering a height field with small photometric errors ($< 0.01\%$ in most cases), but the recovered height field is of poor quality when viewed in 3D. This is one of our future research.

Obviously, our system will have difficulties in modeling a bark sample that cannot be represented by a height field. An example is the so-called "stringy bark", which has many dangling fiber strips (see Fig.3(f)). This should be another future work in our modeling tool.

6 Conclusion

We presented a system for realistic modeling mesostructure of tree bark. Our image-based bark modeling system can successfully model a variety of common bark features including ironbark, vertical and horizontal fractures, tessellation, furrowed cork, and lenticels. Our goal is not to model a precise surfaces but a convenient tool to achieve user-controlled bark modeling. We provide an intuitive and easy-to-use modeling tool for analyzing the bark features and modeling bark surface either in its height field or its texture. Our experimental results verify our approach.

Acknowledgement

This research was partially supported by NSFC (No. 60225016) and National Basic Research Project of China (No. 2002CB312101).

References

[1] J. Bloomenthal. Modeling the mighty maple. *Computer Graphics (SIGGRAPH '85 Proceedings)*, 19:305–311, 1985.

[2] S. Brooks and N. Dodgson. Self-similarity based texture editing. *Computer Graphics (SIGGRAPH '02 Proceedings)*, pages 653–656, 2002.

[3] K. J. Dana, S. K. Nayar, B. van Ginneken, and J. J. Koenderink. Reflectance and texture of real-world surfaces. *ACM Transactions on Graphics*, 18(1):1–34, 1999.

[4] P. de Reffye, C. Edelin, J. Francon, M. Jaeger, and C. Puech. Plant models faithful to botanical structure and development. *Proceedings of SIGGRAPH 1988*, 22:151–158, August 1988.

[5] J.-M. Dischler, K. Maritaud, and D. Ghazanfarpour. Coherent bump map recovery from a single texture image. *Graphics Interface*, 2002.

[6] P. Federl and P. Prusinkiewicz. A texture model for cracked surfaces, with an application to tree bark. *Proceedings of Western Computer Graphics Symposium*, 23–29, 1996.

[7] A. Gersho and R. M. Gray. *Vector Quantization and Signal Compression*. Kluwer Academic Publishers, 1992.

[8] J. C. Hart and B. Baker. Implicit modeling of tree surfaces. *Proceedings of Implicit Surfaces '96*, pages 143–152, October 1996.

[9] K. Hirota, Y. Tanoue, and T. Kaneko. Generation of crack patterns with a physical model. *The Visual Computer*, 14(3):126–137, 1998.

[10] B. K. P. Horn and M. J. Brooks. *Shape from Shading*. MIT Press, 1989.

[11] J. J. Koenderink and A. J. V. Doorn. Illuminance texture due to surface mesostructure. *Journal of the Optical Society of America*, 13(3):452–463, 1996.

[12] Y. G. Leclerc and A. F. Bobick. The direct computation of height from shading. *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 552–558, 1991.

[13] S. Lefebvre and F. Neyret. Synthesizing bark. *Eurographics Workshop on Rendering*, 2002.

[14] B. Lintermann and O. Deussen. A modelling method and user interface for creating plants. *Computer Graphics Forum*, 17(1):73–82, 1998.

[15] X. Liu, Y. Yu, and H.-Y. Shum. Synthesizing bidirectional texture functions for real-world surfaces. *Computer Graphics (SIGGRAPH '01 Proceedings)*, pages 97–106, 2001.

[16] J. Malik, S. Belongie, J. Shi, and T. Leung. Textons, contours and regions: cue integration in image segmentation. In *Proceedings of International Conference on Computer Vision (ICCV)*, pages 918–925, 1999.

[17] B. M. Oh, M. Chen, J. Dorsey, and F. Durand. Image-based modeling and photo editing. *Computer Graphics (SIGGRAPH '01 Proceedings)*, pages 433–442, 2001.

[18] P. E. Oppenheimer. Real time design and animation of fractal plants and trees. *Computer Graphics (SIGGRAPH '86 Proceedings)*, 20(4):55–64, 1986.

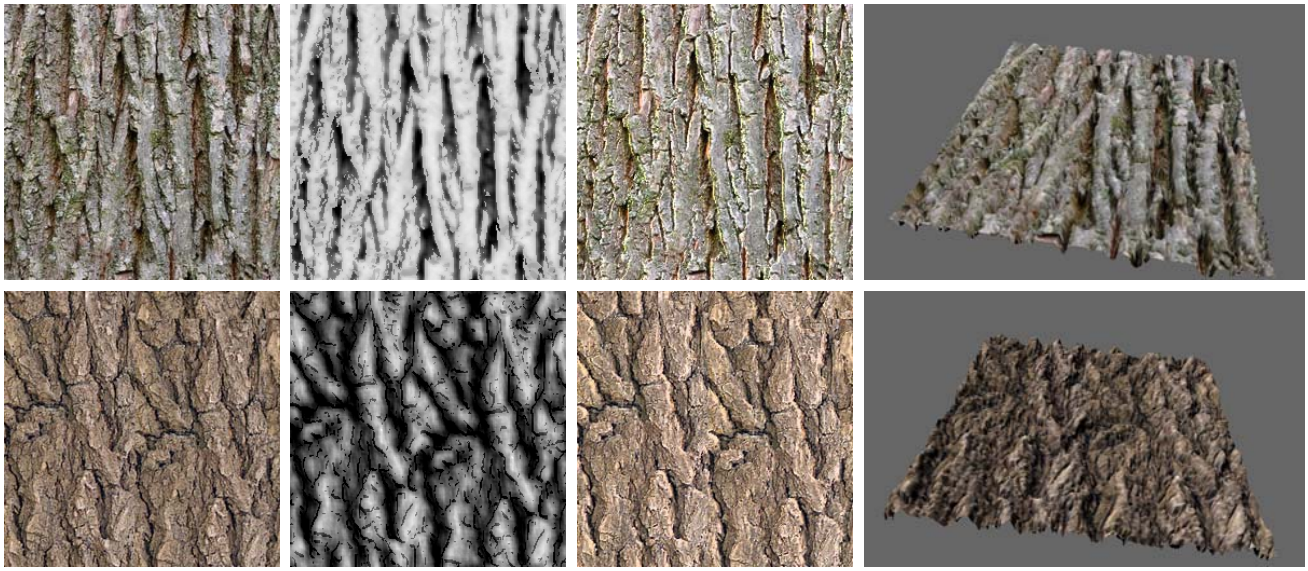
[19] A. E. Prance, K. B. Sandved, and G. T. Prance. *Bark : The Formation, Characteristics, and Uses of Bark Around the World*. Timber Press, 1993.

[20] P. Prusinkiewicz, A. Lindenmayer, and J. Hanan. Development models of herbaceous plants for computer imagery purposes. *Proceedings of SIGGRAPH 1988*, 22:141–150, August 1988.

[21] H. Rushmeier, G. Taubin, and A. Gueziec. Applying shape from lighting variation to bump map capture. In *Eurographics Workshop on Rendering*, pages 35–44, 1997.

[22] D. Terzopoulos and K. Fleisher. Modeling inelastic deformation: Viscoelasticity, plasticity, fracture. *SIGGRAPH88 Conference Proceedings*, pages 269–278, 1988.

[23] L. Wang, X. Wang, X. Tong, S. Hu, B. Guo, and H.-Y. Shum. View-dependent displacement mapping. *To appear in Proceedings of SIGGRAPH 2003*, 2003.



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

Figure 7. Height fields and textures modeled with our approach

(a) (b) (c)

(d) (e) (f)

Figure 8. A tree brunch with different bark features.

(a) (b)

(c) (d)

Figure 9. A complete tree with different bark features.