Available online at www.sciencedirect.com
SCIENCE DIRECT®
Graphical Models 68 (2006) 333–342
Graphical Models
www.elsevier.com/locate/gmod

Modeling hairy plants

Martin Fuhrer [a,*], Henrik Wann Jensen [b], Przemyslaw Prusinkiewicz [a]

[a] *University of Calgary, Canada*
[b] *University of California San Diego, USA*

Received 21 March 2005; accepted 30 November 2005
Available online 10 March 2006

Abstract

The appearance of computer generated plants has improved significantly due to recent advances in both modeling and rendering. In this paper, we describe a system that further improves the appearance of CG plants by including the tiny hairs that cover many plant organs. A plant skeleton is generated using an L-system and graphically interpreted using generalized cylinders. The individual hairs are then mapped onto the surfaces and boundary edges of the mesh. Hair properties are specified and adjusted according to positional information. Sample images included in the paper illustrate the impact of hairs on the appearance of rendered plants.
© 2006 Elsevier Inc. All rights reserved.

Keywords: L-system; Plant modeling; Photorealism; Hair; Generalized cylinder; Positional information

1. Introduction

Plant hairs are an outgrowth from the surface (epidermis) of leaves, shoots and roots, and are a common element of many different plants. From a biological perspective, they are representative of *trichomes*, which also include such structures as scales, warts, and spines [21]. The appearance of trichomes varies from the pronounced scales covering young fern fronds to the fine hairs covering some leaves. These fine hairs are often perceived as "fuzz," and give rise to a characteristic soft glow around backlit surfaces, as illustrated in Fig. 1. This phenomenon has been termed *asperity scattering* [10].

The mechanisms of trichome distribution on their underlying surfaces are an interesting problem in developmental biology, and have recently been studied from a molecular perspective (e.g., [12,20]). Trichome placement is controlled by a signaling mechanism that prevents the formation of new organs too close to each other, and in this respect it is related to the placement of organs in phyllotactic patterns around plant stems [12]. Functionally, plant hairs and related structures play a variety of roles. For example, they may shade the plant surface from excessive exposure to sunlight, decrease the loss of water through transpiration, or ward off hungry insects or herbivores [21].

Modeling of hairs has been extensively investigated in the context of human hair and animal fur. Kajiya and Kay [8] approached fur rendering using texels, three dimensional arrays of parameters describing visual properties of the fur. Goldman [7]

[*] Corresponding author.
E-mail address: fuhrer@cpsc.ucalgary.ca (M. Fuhrer).

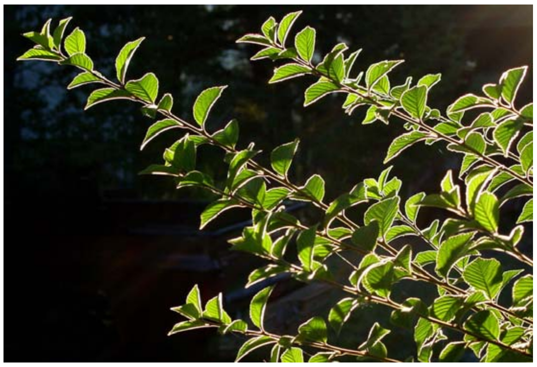
Fig. 1. Fuzz on the leaves of Nankin cherry boughs glows brilliantly when illuminated from the rear.

proposed a probabilistic method for rendering fur from a distance. The texel-based and probabilistic approaches make it possible to synthesize images of furry objects while keeping the geometry simple, but they provide only a limited control over the individual hairs. In contrast, hairs modeled as individual geometric objects allow for better interactive [3] and physically-based [2] control of individual strands. These strands can be grouped together and controlled as tufts or wisps [15,22,9] for performance reasons. The trade-off between level of hair representation and final appearance is further exemplified by the work of Lengyel et al. [11].

An important aspect of the appearance of hair and fur is the modulation of a hair's characteristics according to its position on the underlying surface. To this end, Miller [13] aligned the orientation of individual hairs according to the texture coordinates of the mesh. Gelder and Wilhelms [6] recognized the need to control fur properties of animals according to the orientation of body segments. They employed orientation vectors for various limbs, but encountered difficulties describing orientation in areas such as the head and thoracic region. Lengyel et al. [11] oriented fur according to vector fields placed on meshes, and made use of an interactive combing tool to further style fur according to the modeler's taste. Fleisher et al. [4] explored the problem of positioning and orienting small elements ('cellular particles') across surfaces of different topologies. They described a number of strategies to tackle the problem, and illustrated their techniques by covering surfaces with scales and thorns.

Fowler et al. [5] proposed the first method aimed at placing elements visually related to hairs on the surfaces of plants. This method exploited phyllotactic patterns to position and orient spines on cacti and achenes (fruits) on the receptacle of goatsbeard.

The modeling of plant hairs must take into consideration several phenomena. Properties of plant hairs may vary continuously between different locations on a plant and developmental stages of plant organs. For example, hair density may be greater in young leaves, and drop off as the leaves become older. Fully mature leaves often lose all the hairs on their front and back surfaces, but maintain hairs along their edges. Hairs may be aligned in specific directions, depending on their position. Hairs on opposite sides of the same surface may be different from each other. In addition, there is much diversity in the shape of individual hairs: given any patch of hairy surface, it is common to see a mixture of hairs that are straight and curly, or long and short.

To account for these phenomena, we propose to model plants with hairs using a method with the following characteristics.

Geometric representation of hairs. We specify individual hairs as generalized cylinders, which are simplified to connected line segments during interactive modeling. This makes it possible to control the shape of hairs and their distribution across a surface in fine detail. This geometric approach is computationally efficient (e.g., plants with hairs can be previewed at interactive rates), because the density of hairs on plants is typically low in comparison to human hair or animal fur. In addition, fewer line segments are required since plant hairs are often relatively short.

The use of positional information. Many aspects of plant architecture can conveniently be characterized using functions of position along plant axes [18]. We extend this technique to plant hairs, changing hair parameters along the length and breadth of plant organs.

The framework of L-systems with the turtle interpretation. In addition to being a general method for modeling plant architecture, L-systems lend themselves well to the local control of hair attributes such as length, radius, curvature, and density of placement. We use for this purpose L-system *modules* (symbols with parameters) that are analogous to those used to control width and color of limbs in the standard L-system interpretation [17]. The 'turtle' used to interpret L-system strings also provides a convenient frame of reference [18] for orienting hairs along plant organs such as stems, leaves, and petals.

2. L-systems and generalized cylinders

L-systems represent branching structures as sequences (strings) of *modules* that describe the paths taken by the branches as well as various drawing parameters [17]. The sequence of modules is generated by applying various *productions*. A production P takes a *predecessor* module M_p, evaluates a boolean *condition* β, and if the condition turns out true, replaces the predecessor with a *successor* consisting of a string of modules N_s:

$$P : M_p\{block_1\}\beta\{block_2\} \rightarrow N_s$$

The $block_n$ statements are optional lines of C code for calculating variables that can be used in the condition or as parameters for modules. The first block is always executed during evaluation of a production, while the second block is executed only if the condition turns out true. An *axiom* is the starting point of an L-system operation, and represents the initial string of modules for the process of production evaluation and string derivation.

The string of modules generated by an L-system is interpreted graphically by considering specific modules as commands to a LOGO-style *turtle* [1,18]. The turtle is oriented in a local coordinate frame consisting of three orthogonal vectors: heading, left, and up. The turtle moves forward in discrete steps along its heading vector. By reorienting the coordinate frame at every step, the turtle can trace a curve through space [18].

In the simplest case, the curve consists of a sequence of line segments. However, by sweeping out a contour and controlling its width, the moving turtle can produce a *generalized cylinder* [18]. Closed contours are used to produce closed generalized cylinders that represent volumetric organs such as stems or branches. Other plant organs such as leaves and petals resemble thin surfaces, and have cross sections described by open contours, so are represented by open generalized cylinders. Closed generalized cylinders expose only their front surface, while open generalized cylinders can expose both front and back. In addition, open generalized cylinders have edges that may represent the rim of a leaf or petal. The distinction between front, back, and edge is significant from a hair modeling perspective, since hairs existing along these locations may exhibit different properties according to their location.

We polygonize generalized cylinders into sequences of connected *segments*, where each segment corresponds to a step taken by the turtle. These segments consist in turn of a number of quadrilateral *faces* that approximate the mathematical definition of the contour. In the actual implementation, each quadrilateral face may consist of two joined triangles, but for the purpose of discussion in this paper, we will assume that each face is a quadrilateral.

We make use of the software environment CPFG [16] to generate strings of modules with L-systems and to graphically interpret them. CPFG defines several modules to manipulate generalized cylinders. The module *@Gs* begins a generalized cylinder, *@Gc* instructs the turtle to sweep out a cylinder segment during its next step, and *@Ge* ends the cylinder. The generalized cylinder segments are the main building blocks for our models, and it is at this level that we manipulate and apply plant hairs.

3. Hair generation

The process of hair generation involves several steps. First, we determine the hair distribution on the underlying surface. We also model individual hairs, and place an instance of a pre-determined hair at every attachment point.

3.1. Distribution of the attachment points

In the extensively studied case of trichome distribution on the leaves of *Arabidopsis thaliana*, the individual trichomes are spaced in a semi-regular manner, preserving a minimum distance from each other. The underlying morphogenetic process is believed to be of the reaction-diffusion type [12]. Instead of simulating this process, we approximate its outcome as a Poisson-disk pattern generated using the point diffusion algorithm proposed by Mitchell [14]. Comparisons of images indicate that trichome distribution patterns (shown, for example, in [12]) are visually indistinguishable from Poisson-disk patterns (as illustrated in [14]).

Point diffusion can be viewed as a texture generation algorithm that computes a texture value (on or off) and a diffusion value at every grid element. This computation is based on previously computed diffusion values stored in a neighborhood of four adjacent elements, one to the immediate left and three in the row above. Point diffusion can be applied in two ways for distributing hairs: on-the-fly or using texture mapping. The former approach involves continuously generating the distribution pattern as the surfaces are produced. The resulting seamless pattern is ideal for offline rendering purposes, but too slow to compute during interactive modeling. We therefore preferably use the texture-mapping method, whereby a single point-diffusion texture is repeatedly tiled along the surface of the plant organ. As the neighborhood used to compute the points in Mitchell's algorithm wraps around the vertical edges of the texture, vertical seams are barely visible. Horizontal seams are more conspicuous; in practice, however, once hairs have been placed and oriented, the seams are not noticeable.

The point-diffusion texture is a function $\tau(u,v) \rightarrow \{0,1\}$, where a returned value of 1 indicates an attachment point for a hair. We have found that a texture of size 100×100 results in visually acceptable hair distribution patterns. The texture-mapping technique is illustrated in Fig. 2. Although it introduces distortions (shear of the texture) when applied to generalized cylinders with a rapidly changing circumference or width, we have found the resulting artifacts visually negligible.

3.2. Hair modeling and placement

Hairs on a single plant come in many different shapes and sizes. While it is not feasible to model the shape of every hair individually, it is possible to capture the overall diversity of a hairy surface by modeling several different *template hairs* and instancing them multiple times. We model our hairs as curved generalized cylinders constructed using simple L-systems. These hair L-systems are independent from the L-system used to generate the whole plant. We control the hair curvature with functions for tilt, twist, and pitch to rotate the turtle along its up, left, and heading vectors, respectively [18]. The number of segments used to approximate each hair depends on its curvature: relatively straight hairs can be reasonably modeled with three turtle steps, while twisted hairs may require seven or more steps. The number of individual hairs to model depends on the nature of plant's hairy surface: for most of our models we found that three to five different hairs properly capture the varied appearance of a hairy surface.

A hair is stored as a sequence of vertices describing the position of the turtle at every step. Whenever a hair is instanced, its vertices are connected together. During real-time rendering in CPFG, the vertices are connected by antialiased line segments, which are stored in a display list to improve performance. During high-quality offline rendering, the vertices are joined by cylindrical primitives. We assume that each hair has a constant radius.

When placing a hair on the front or back side of a generalized cylinder, we set the hair's orientation by aligning the axis of its first segment with the *hair direction vector* $\widehat{d}$. By default, hair direction vectors are aligned with the surface's shading normals $\widehat{n}$, so that hairs have the appearance of emerging perpendicularly to the surface of the plant. In the same way that shading normals are computed by linearly interpolating between the four vertex normals of a face, hair direction vectors are computed by linearly interpolating between the four *hair vertex vectors* of a face.

An open generalized cylinder also includes a specification of *hair edge vectors* for the outer edges of the first and last face. A hair edge vector is defined by the cross product of the height vector $\widehat{h}$ (Fig. 2) and the shading normal $\widehat{n}$, so that it points away from the cylinder segment. By default, hair direction vectors for edge hairs are aligned with the hair edge vectors.

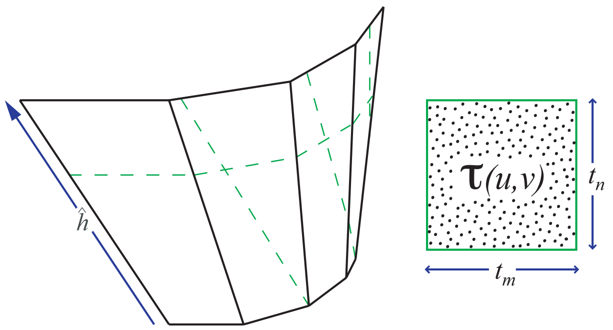

Fig. 2. The point-diffusion texture is mapped onto the generalized cylinder segment (unfolded, curling into the page) in scan-line fashion, starting at the lower left corner and ending at the upper right corner. The mapping of the $t_m \times t_n$ texture is indicated by the dashed lines. Density of the hair is controlled by scaling the texture according to positional information.

During hair placement, we transform the vertices of the hair into the local coordinate frame formed by the shading normal $\widehat{n}$, the vector $\widehat{n} \times \widehat{h}$, and a tangential vector $\widehat{n} \times (\widehat{n} \times \widehat{h})$. This default hair orientation can be modified as needed (Section 4.4).

We currently do not check for collisions between hair and surface, a likely occurrence if the hair direction vector for a curly hair is nearly tangential to the surface.

4. Control of hair parameters

4.1. Overview

We have defined a set of modules for controlling hair properties within the L-system of the plant model (Table 1). The module names follow the convention used by other modules in CPFG, but can be adapted for any L-system software. The basic modules for turning hair on and off accept one boolean parameter. These modules are: *@Hf* for hairs on the front surface, *@Hb* for hairs on the back surface, and *@He* for edge hairs. The remaining modules accept additional parameters. The optional parameters *loc* sets the property for only those hairs in a particular location—front, back, or edge; ignoring this parameter results in the property being set for all hair types. The properties of individual hairs can be varied by specifying the *rng* parameter. This adds a random offset to the value of the property, such that the resulting value lies within the range *value* ± *rng*. We also allow the specification of a *modulating function*. This function modifies the property value around the girth of a cylinder segment. The module parameter *fun* is the index of this function.

Properties may be modified after every turtle step. In the simplest case, properties are kept constant for all hairs on a generalized cylinder segment. If the step sizes are sufficiently small, the sudden transition of properties from segment to segment is nearly invisible, but if the step sizes increase, visual discontinuities may become apparent. To eliminate this artifact, properties may be interpolated across segments.

Table 1
L-system modules for specifying hair attributes

Hair module	Purpose
@Hf(*bool*)	Enable/disable hairs on front side
@Hb(*bool*)	Enable/disable hairs on back side
@He(*bool*)	Enable/disable hairs along edges
@Hd(*value*,{*rng*,*fun*,*loc*})	Specify density
@Hl(*value*,{*rng*,*fun*,*loc*})	Specify length
@Hr(*value*,{*rng*,*fun*,*loc*})	Specify radius
@Hi(*value*,{*rng*,*fun*,*loc*})	Specify angle of incline
@Ht(*value*,{*rng*,*fun*,*loc*})	Specify twist angle
@Hw(*value*,{*rng*,*fun*,*loc*})	Specify wrapping angle
@Hp($P_1, \ldots, P_n$,{*loc*})	Specify placement probabilities

Module parameters are explained in the text.

Algorithm 1. This L-system describes the change in hair properties for length, incline, and density along the length of a generalized cylinder. The probability of two types of hairs is also being set at every step, and changes as we move along the cylinder (the straight hairs become more likely near the right end). The resulting image is displayed in Fig. 4.

```
1.  #define $l$ 10 /* length of axis */
2.  #define $\Delta s$ 1 /* turtle step */
3.  Axiom: @Gs @Hf(1) B(0)
4.  B($s$): $s \leqslant l$
5.  {
6.      $relativeDistance = s/l$;
7.      $len = \mathscr{F}_1(relativeDistance)$
8.      $inc = \mathscr{F}_2(relativeDistance)$;
9.      $den = \mathscr{F}_3(relativeDistance)$;
10. } $\rightarrow$
11. $@Hl(len)$ $@Hi(inc)$ $@Hd(den)$
12. $@Hp(relativeDistance, 1 - relativeDistance)$
13. $f(\Delta s)$ $@Gc$ $B(s + \Delta s)$
14. $B(s)$: $s \geqslant l \rightarrow @Ge$
```

Algorithm 1 presents a sample L-system used to generate the cylinder pictured in Fig. 4. We set the total length and step size in lines 1 and 2, and begin the generalized cylinder with module *@Gs* in the axiom of line 3. The module *@Hf* turns on the hairs. The production *B* repeatedly advances the turtle and draws a generalized cylinder segment *@Gc* (line 13) as long as the current position is less than the total axis length (line 4). For every step, we calculate the relative distance traveled by the turtle (line 6). In lines 7–9, parameters for length, incline, and density are computed as a functions of relative distance. The L-system modules in line 11 assign these properties to the hair being rendered. The actual hair shapes are selected at random from two previously defined template hairs (cf. Section 3.2), with the probabilities set in line 12. In our example, the first hair type is straight and it occurs with increasing probability as we move along the cylinder. The second hair type, a curly hair, occurs frequently at the start of the cylinder but gradually disappears toward the end.

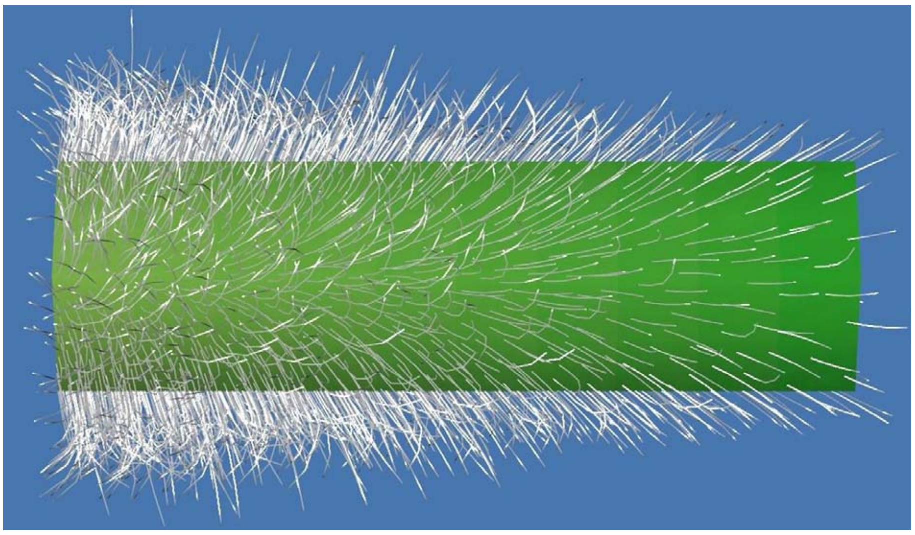

Fig. 3. Cross-sectional view of a leaf. No hair wrapping was specified in the top cross-section. In the lower cross-section, a profile function specified by a hair-wrapping module causes hairs to gradually tilt toward the leaf edges.

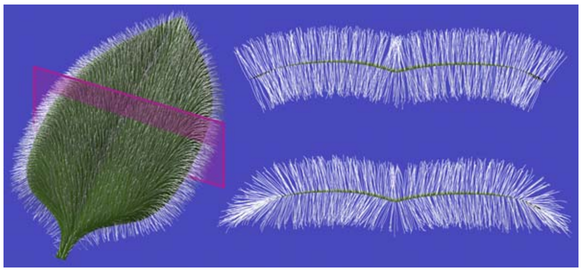

Fig. 4. Hair parameters for length, density, and incline are modified along the generalized cylinder. The distribution of different types of hairs is also altered, as curly hairs near the left end give way to straight hairs near the right end.

The control of hair attributes is discussed in more detail below.

4.2. Density

Density is adjusted with the module *@Hd*. While each generalized cylinder segment is associated with a single density value, the actual density of hairs may be non-uniform if the segment's width or curvature change. In particular, the density of hairs along the inner arc of a bending cylinder will be higher than the density of hairs along the outer arc. This behavior is consistent with our observations of hair densities along bent stems of oriental poppies.

4.3. Size

The relative length of a hair is adjusted with the module *@Hl*. Its parameter controls the hair geometry by scaling the vertex positions prior to placement. Some hairy surfaces consist of both short and long hairs mixed together. In the simplest case, we can randomize the length of the individual hairs by passing a *rng* parameter. Alternatively, we can model several template hairs with different sizes and place them together on the surface. The radius for the hair is specified by the module *@Hr* (as noted in Section 3.2, we assume that each hair has a constant radius).

4.4. Orientation

Hairs do not always grow perpendicularly to a surface, but can potentially be oriented in any direction. We provide modules that modify the hair orientation with three degrees of freedom. For surface hairs, these modules indicate the rotation of a hair direction vector around three axes: the *twist axis* $\widehat{d}$ (the hair direction vector itself), the *incline axis* $\widehat{n} \times \widehat{h}$, and the *wrapping axis* $\widehat{n} \times (\widehat{n} \times \widehat{h})$. (These definitions are slightly modified in the case of edge hairs.) Since hair direction vectors are calculated via interpolation between hair vertex vectors, we simply reorient the hair vertex vectors at the four corners of every face.

The module *@Ht* indicates the *twist angle* and rotates the hair around the twist axis. Similarly, the module *@Hi* indicates the *incline angle* and rotates the hair around the incline axis, and the module *@Hw* indicates the *wrapping angle* and rotates the hair around the wrapping axis. This last rotation is particularly useful when modeling hairy leaves, on which it is commonly observed that hairs running close to the axis are approximately perpendicular to the leaf surface, but hairs near the edges tend to wrap around to the other side (Fig. 3). Hence, the wrapping angle is often associated with a modulating function that increases the amount of wrap as we move toward the edges of an open generalized cylinder.

As with most other properties, the orientation angles can be randomized to create hairy surfaces with a disheveled appearance. The hairs in the plant models presented in Section 5, for example, have a randomized twist angle.

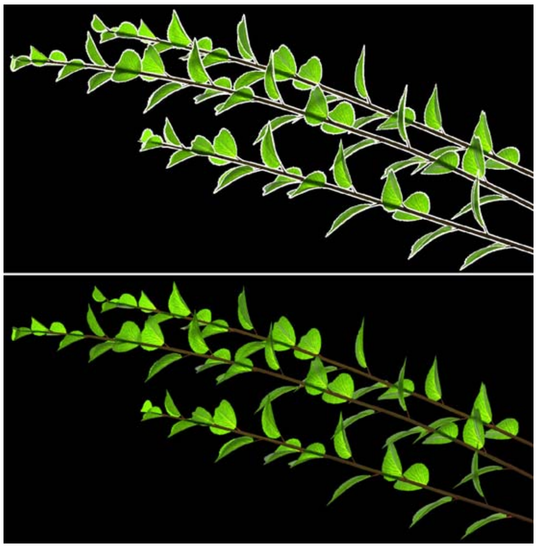
Fig. 5. Nankin cherry branches with and without hairs, illuminated with backlight. Comparison shows the dramatic effect resulting from the inclusion of hairs in the model.

4.5. *Placement probability*

After modeling several template hairs, we must be able to describe in what proportions they are to be distributed across the surface. The *placement probability* indicates the probability that a particular hair will be placed at a given attachment point. Given that we have modeled *n* hairs, the module *@Hp* accepts *n* parameters that set the placement probability for each hair.

5. Results

We have chosen to model several plants that display different types of hairs. All the images were rendered using a Monte Carlo ray-tracer capable of simulating translucency and rendering hairs. The plants are visualized against solid backgrounds to better emphasize the appearance of the hairs. Hair shading is achieved with a slightly modified Kajiya model [8] that accounts for backscattering (similar to the model proposed by Goldman [7]). Each model is illuminated with a single spherical light source and a hemispherical skylight.

The leaves and branches of Nankin cherries are covered in a velvety fuzz that is barely visible under normal frontlighting. Under the influence of direct backlighting, however, the fuzz produces an unmistakable halo effect, as demonstrated by the photograph in Fig. 1. Fig. 5 simulates this effect through the use of surface hairs for the branches and edge hairs for the leaves. The image was rendered with approximately one million tiny hairs. The venation patterns were generated using a particle-based simulation, related to the method by Rodkaew et al. [19], and texture-mapped onto the leaves. Bump

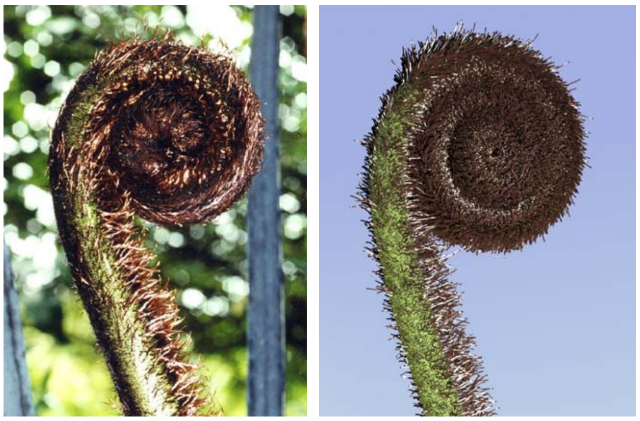
Fig. 6. A photograph and a model of fern croziers.

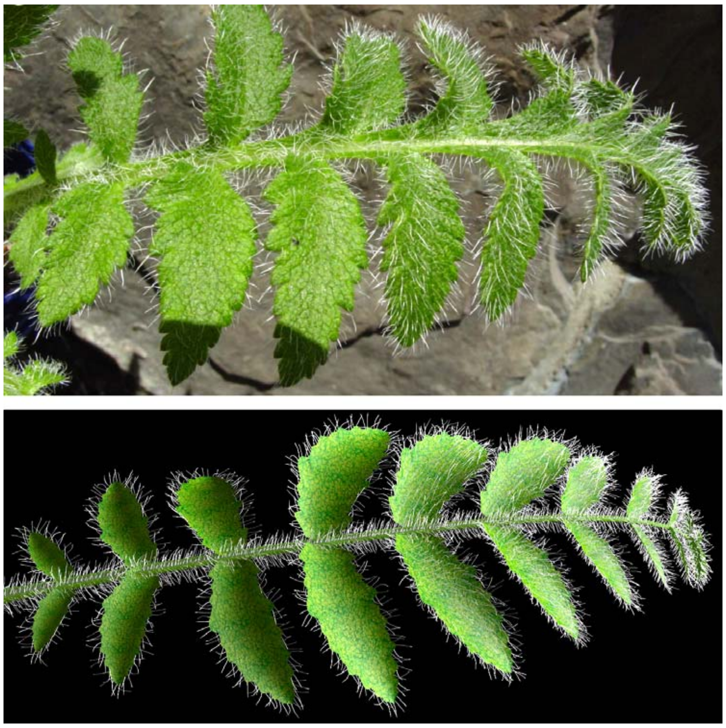
Fig. 7. Comparison of a real and rendered oriental poppy frond. New leaves are covered in dense surface hairs, which fall off as the leaves age. However, edge hairs are retained.

mapping lends the veins a sense of depth, while diffuse transmission is used to simulate the translucency in the leaves.

In contrast to the delicate hairs of the Nankin cherry, the fern croziers shown in Fig. 6 are characterized by relatively large scales. The model captures changes in the length and density of scales along and around the stem.

Oriental poppies are covered in pronounced white hairs that resemble bristles but are soft to the touch. New leaves growing at the tips of fronds have a very dense covering of hairs, but the density decreases as the leaves age. At a certain point, the leaves lose all their front and back hairs, but retain edge hairs (see Fig. 7). A fully grown (but not yet flowering) oriental poppy, with approximately 120,000 hairs, is displayed in Fig. 8. We gradually decreased hair length from the base to the tip of a frond, and used shorter hairs for the backside of leaves. We also used a hair wrapping function around the leaves. While a mixture of straight and curved hairs were used for most of the plant, the placement probability for straight hairs on the bud was set to *1*, and the hairs were tilted such that they became almost tangent to the surface of the bud.

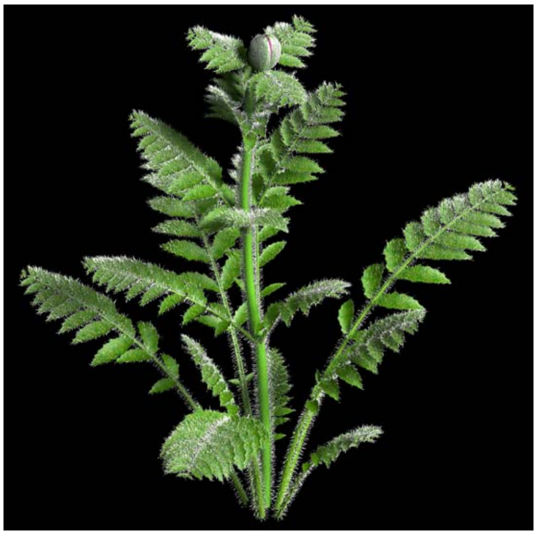
Fig. 8. Oriental poppy.

The wild crocus is an enjoyable modeling and rendering challenge, not only for its wispy hairs, but also for the fuzzy translucency and subtle venation patterns of its petals. The crocus model shown in Fig. 9 contains 65,000 hairs. Hair length and orientation have been varied along the surface

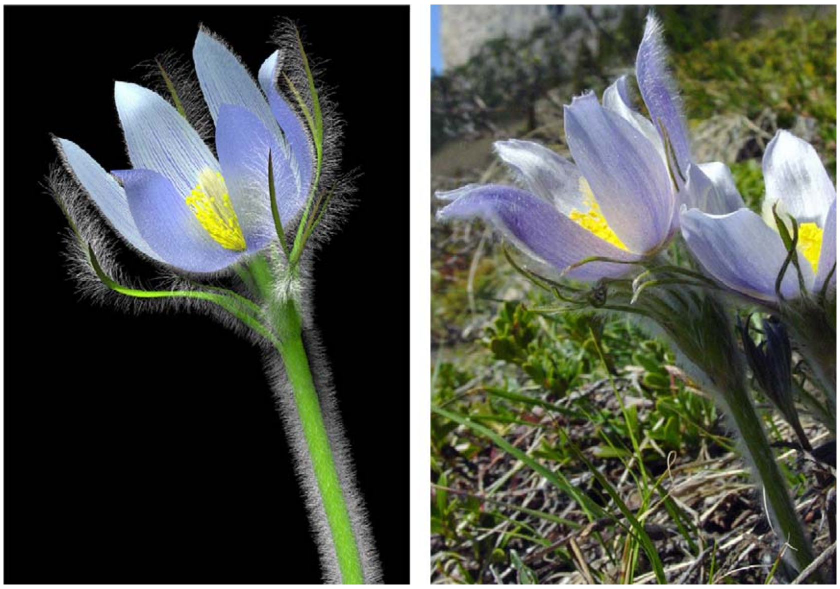
Fig. 9. Photograph (left) and rendered image (right) of wild crocus.

of the petals and finger-like leaves. Three different hair models have been used for the model. The fuzzy translucency of the petals has been obtained using Monte Carlo sampling. The vein patterns were generated procedurally during rendering by connecting sequences of jittered points with spline curves.

6. Conclusions and future work

Hairs are an important attribute of plant appearance, and cannot be neglected in the quest for increasingly realistic plant models and synthetic images. We have shown that they can easily be incorporated into plant models expressed using L-systems. Local hair parameters are controlled using modules generated by L-systems. The orientation information inherent to L-systems is used for setting hair parameters according to positional information, as well as for positioning and orienting hairs on the generalized cylinder segments.

The presented results lead to future work in three directions.

6.1. Improved efficiency of hair representation and rendering

The overall goal of the presented work has been to enhance realism in visualization of plants. To this end, we have chosen the flexible and conceptually straightforward approach of modeling the geometry of individual hairs. This approach makes it possible to model and preview plants with hairs at interactive rates, although high-quality rendering times are longer. It would be useful to improve this performance while maintaining the quality of the resulting images, and to offer trade-offs between computational efficiency (time and space), and the visual quality of the results. Previous work in the modeling of hair and fur suggests a number of techniques that may help achieve these goals. For example, texels may offer a good compromise for modeling clusters of very short and dense hairs, or fuzzy surfaces. Prefiltering rather than supersampling may further improve the computational cost of rendering.

6.2. Multiscale modeling based on biological simulations

Our approach to modeling plants with hair is purely phenomenological: we use a Poisson distribution to position hairs on their supporting surfaces, and reproduce the shape of individual hairs according to their observed appearance. It may be interesting, at least from a biological perspective, to simulate the underlying morphogenetic processes according to current biological theories and hypotheses.

6.3. Improved rendering techniques

Much work still remains to be done on the rendering of plants. For example, techniques such as subsurface scattering could further enhance the appearance of petals and leaves.

Acknowledgments

We thank Adam Runions for the procedurally generated textures of the Nankin cherry leaves, and Dave Fracchia and Brendan Lane for useful comments on this paper. The reported research has been supported in part by a Discovery Grant from the Natural Sciences and Engineering Research Council of Canada, and by postgraduate scholarships from NSERC and iCORE.

References

[1] H. Abelson, A. DiSessa, Turtle Geometry: The Computer as a Medium for Exploring Mathematics, MIT Press, Cambridge, MA, 1981.
[2] K. Anjyo, Y. Usami, T. Kurihara, A simple method for extracting the natural beauty of hair, In Proceedings of SIGGRAPH, pp. 111–120, 1992.
[3] A. Daldegan, N. Thalmann, T. Kurihara, D. Thalmann, An integrated system for modeling, animating and rendering hair, In Computer Graphics Forum (1993) 211–221.
[4] K.W. Fleischer, D.H. Laidlaw, B.L. Currin, A.H. Barr, Cellular texture generation, In Proceedings of SIGGRAPH, pp. 239–248, 1995.
[5] D.R. Fowler, P. Prusinkiewicz, J. Battjes, A collision-based model of spiral phyllotaxis, In Proceedings of SIGGRAPH, pp. 361–368, 1992.
[6] A.V. Gelder, J. Wilhelms, An interactive fur modeling technique, In Proceedings of Graphics Interface, pp. 181–188, 1997.
[7] D. Goldman, Fake fur rendering, In Proceedings of SIGGRAPH, pp. 127–134, 1997.
[8] J.T. Kajiya, T.L. Kay, Rendering fur with three dimensional textures, In Proceedings of SIGGRAPH, pp. 271–280, 1989.
[9] T. Kim, U. Neumann, Interactive multiresolution hair modeling and editing, In Proceedings of SIGGRAPH, pp. 620–629, 2002.
[10] J. Koenderink, S. Pont, The secret of velvety skin, Machine Vision and Applications 14 (2003) 260–268.
[11] J. Lengyel, E. Praun, A. Finkelstein, H. Hoppe, Real-time fur over arbitrary surfaces, In Proceedings of the ACM Symposium on Interactive 3D Graphics, pp. 227–232, 2001.
[12] O. Leyser, S. Day, Mechanisms in Plant Development, Blackwell, Oxford, 2003.
[13] G. Miller, From wire-frame to furry animals, In Proceedings of Graphics Interface, pp. 138–146, 1988.
[14] D.P. Mitchell, Generating antialiased images at low sampling densities, In Proceedings of SIGGRAPH, pp. 65–72, 1987.
[15] M. Nakajima, S. Saruta, H. Takahashi, Hair image generating algorithm using fractional hair model, In Signal Processing: Image Communication 9 (1997) 267–273.
[16] P. Prusinkiewicz, R. Karwowski, R. Měch, J. Hanan, L-studio/cpfg: a software system for modeling plants, in: M. Nagl, A. Schürr, M. Münch (Eds.), Applications of Graph Transformation with Industrial Relevance, Springer-Verlag, Berlin, 2000, pp. 457–464.
[17] P. Prusinkiewicz, A. Lindenmayer, The Algorithmic Beauty of Plants, Springer-Verlag, New York, 1990.
[18] P. Prusinkiewicz, L. Mündermann, R. Karwowski, B. Lane, The use of positional information in the modeling of plants, In Proceedings of SIGGRAPH, pp. 289–300, 2001.
[19] Y. Rodkaew, S. Siripant, C. Lursinsap, P. Chongstitvatana, An algorithm for generating vein images for realistic modeling of a leaf, In Proceedings of the International Conference on Computational Mathematics and Modeling, pp. 73–78, 2002.
[20] A. Schnittger, U. Folkers, B. Schwab, G. Jürgens, M. Hülskamp, Generation of a spacing pattern: the role of TRIPTYCHON in trichome patterning in *Arabidopsis*, Plant Cell 11 (1999) 1105–1116.
[21] J.C. Uphof, Plant Hairs, Gebrüder Borntraeger, Berlin, 1962.
[22] X.D. Yang, Z. Xu, T. Wang, J. Yang, The cluster hair model, Graphical Models 62 (2) (2000) 85–103.