An Interactive Fur Modeling Technique

Allen Van Gelder and Jane Wilhelms
Computer Science Department
University of California, Santa Cruz, U.S.A. 95064
E-mail: avg@cse.ucsc.edu, wilhelms@cse.ucsc.edu

Abstract

A technique for modeling fur, but not long human hair, quickly, using the facilities of common graphics workstations is described. The user selects a variety of parameters to achieve the desired appearance for a particular animal, such as hair density, length, stiffness, and color properties. Undercoat and overcoat may have separate specifications, and degrees of randomness may be specified, for added realism. Standard GL facilities are used for modeling, lighting, and rendering. Hair density is automatically adjusted for viewing distance to compensate for the limit of single pixel resolution, thus avoiding the tendency to "melt" into a surface of uniform appearance. Gravitational effects are approximated. Four drawing methods are compared: single line, polyline, nurbs curve, and nurbs surface. The polyline method is judged to offer reasonable realism at substantially faster rendering rates than previously reported hair techniques.

KeyWords: Computer graphics, computer animation, modeling, natural phenomena, animals, fur, hair.

1 Introduction

Modeling and animating animals is a complex and difficult issue, and simulating their fur is a challenging aspect of this problem. Fur consists of many thousands of individual curved hairs that can move independently and interact in complex ways with light. Simulating fur exactly is impractical, if not impossible. However, fur is essential if the goal is to create acceptably realistic animals.

Fur and hair are really the same material, being made up of dead cells produced by follicles in the skin of mammals. Generally, we call it fur if it covers most of the body and is relatively short. We call it hair if it is limited to some body regions, such as the head or mane region. Fur is more interesting than hair in some ways, because it often consists of two or three different types on the same animal, and may vary widely in color on different parts of the body, or even on a single hair.

We have been exploring ways of modeling visually attractive fur that do not add too significantly to the cost of rendering and animating the animal. We are able to report that rather simple techniques, using the facilities of common graphics workstations, can produce reasonably realistic fur in interactive time frames.

2 Background

Two different techniques for modeling hair and fur have been described in the literature. The first technique models individual hair primitives [1, 2, 3, 6, 7, 10, 11, 12, 13, 15]. The second technique simulates the interaction of light with hair using three-dimensional textures [4, 5, 8, 9, 14].

Three-dimensional textures can generate many complex phenomena, including fur. Using this technique, the Kajiya teddy bear remains one of the most realistic images of simulated fur [5]. However, it is difficult to find the appropriate texture for the desired fur, and rendering is extremely slow. (Rendering times are usually given in hours.) Therefore, for our application, we use the first technique: individual hair primitives.

Previous techniques for individual hair are too slow for interactive animation, as shown in Figure 1, but were usually more concerned with long human hairs than with animal fur. Gavin Miller, in one of the earliest papers on modeling fur and other anisotropic materials, used straight line segments oriented according to the skin surface normal and tangent vectors to mimic fur [7]. Robert Skinner modeled hair and fur as a sequence of cylinders following a Bézier curve [13]. He implemented a specialized anti-aliasing hair renderer to make this more efficient. Though results looked quite good, it was too slow for interactive purposes. Rosenblum *et al.* used a similarly expensive physical model of individual hairs involving mass-spring chains [11].

Anjyo *et al.*. model human hair as *polylines* representing a chain of sticks, which is bent using a cantilever beam model [2]. Watanabe and Suenaga use a chain of trigonal prisms to approximate cylinders [15]. To avoid expensive calculations on each hair, they modeled a small proportion of the hairs carefully and used a *wisp* model to generated more hairs from these.

Daldegan *et al.* model human hair, including motion and collision detection, and render it using ray tracing [3]. Hairs are modeled as a curved chain of straight cylinders. Their human hair style typically has 100,000

Method	Hairs/sec.	Equipment
[13]	5	DEC Vax
[2]	400	SGI VGX
[15]	421	SGI 4D/210GTX
[12]	333	unspecified SGI
[1]	750	85 MIPS workstation
this paper	15,000	150MHz SGI RE-II

Figure 1: Reported rendering speeds for hair.

to 150,000 hairs. They also use a *wisp* model to reduce calculations. They use a shadow-buffer technique in ray-tracing the hair. They do not indicate how long their impressive ray-traced rendering of a synthetic actress' head takes.

Shih and Guo [12] model human hair using the cantilever beam model suggested by Anjyo *et al.* [2]. Hairs are modeled as *polylines*, also. They point out that a human head has between 80,000 and 120,000 strands of hair, varying in width from 0.05 mm to 0.09 mm. In their model, collision detection can be performed, and hair blowing in the wind can be simulated.

Ando and Morishima discuss a relatively fast method of simulating human hair, including collision detection [1]. They also use a *wisp* technique and model a single hair which is duplicated hundreds of times in the neighborhood. Rankin and Hall modeled long pony-tail hairs and render in software [10]. They use a polyline technique, and do not give any rendering times.

As far as we can tell, given the published information, and the variety of machines used (see Figure 1), the method we describe here is significantly faster than previous methods, even allowing for equipment differences. For example, on a 150MHz SGI Reality Engine II using a single processor, we can render about 35,000 hairs per second (CPU time) of straight fur and 15,000 hairs per second of 8-segment polyline fur. In our judgment, our method produces fur that is competitive in image quality to the faster published techniques. However, our method is designed for modeling fur, and is not as suitable for high-quality modeling of long human hair, which may require more polyline segments and greater shape control.

3 Modeling Approach

A hair may be a shaded straight line, polyline, nurbs curve or nurbs surface ("nurbs" stands for "non-uniform rational B-spline"), all drawn using SGI GL graphics calls and the GL renderer. These methods are referred to as *hair styles*.

A number of user-controlled parameters define the hair properties. The number of hairs depends on *hair density*, and their roots are randomly dispersed about the skin surface, which is a triangle mesh. A user-controlled *weightless direction vector* is used to determine the hair direction at the root of the hair, but the final direction is affected by a gravitational influence, which is scaled by a *stiffness* factor. The hair *length* is also user-controlled; one length is specified for the overcoat and a second for the undercoat. Hairs are given material characteristics which, together with the light model, provide their color. Random factors can be added to hair orientation and shading to give more variation. Fur can be drawn as a combination of longer overcoat hairs and shorter undercoat hairs, which can have different materials.

The number of hairs is by default assumed to be about $100{,}000$ over the whole animal, or about 100 per square inch on our small monkey. This default allows rapid, acceptably realistic rendering. The user can interactively scale this number for more, or less, dense fur. We have used up to 500,000 hairs on the monkey to produce a full-looking coat for final images. The fur drawing dominates the rendering time at this density.

As there are $130{,}000$ triangles in the monkey's skin, we generate less than one hair per triangle on average, at the default setting, and about 4 on average at the density scale of 5.0. Due to variation in triangle area, many triangles still receive an allocation of zero or one hairs, as described in Section 3.2.

3.1 Overcoat and Undercoat

Many furred animals have two or three kinds of fur with different characteristics. We simulate this by allowing an undercoat and overcoat that can have different material characteristics, stiffness, randomness, and hair style, which are user-controlled. The user also specifies the percentage of hair that is overcoat and undercoat, and the relative length of the undercoat compared to the overcoat. Figure 4 shows a triangle with some overcoat and undercoat furs. Polyline fur is shown, with the undercoat more stiff than the overcoat. Figure 9 shows the use of an overcoat and undercoat with the monkey.

3.2 Hair Distribution and Barycentric Coordinates

The skin covering is a triangle mesh. Each triangle is allocated an appropriate number of hairs based on its area and the overall hair density. Specifically, let α be this triangle's fraction of the total skin area, let N be the desired total number of hairs for the animal, and let u be a uniform random number in $[0, 1)$. Then $\lfloor \alpha N + u \rfloor$ hairs are allocated to this triangle. This method of “random rounding” is important, because we cannot assume that the fractional part of αN is itself random. Regions of higher curvature have smaller triangles, and αN typically is less than 0.5 throughout such a region. This would leave a bald spot if the usual deterministic rounding were used.

Those hairs allocated to a triangle are distributed over the surface at random, according to a 2D uniform distribution. To generate points according to such a distribution, barycentric coordinates are used. Let u_1 and u_2 be independent uniform random numbers on $[0, 1]$. The first barycentric coordinate, b_1, is generated from a triangular distribution on $[0, 1.0]$ with the peak at 0. The second barycentric coordinate, b_2, is generated from a uniform distribution on $[0, 1.0 - b_1]$. The third makes the sum equal 1.0. Then the location of the hair root, **h**, is a weighted sum of the three vertex locations, $\mathbf{v}_1$, $\mathbf{v}_2$, $\mathbf{v}_3$, of the triangle.

$$\begin{aligned} b_1 &= 1.0 - \sqrt{u_1} \\ b_2 &= (1.0 - b_1)u_2 \\ b_3 &= 1.0 - b_1 - b_2 \\ \mathbf{h} &= b_1\mathbf{v}_1 + b_2\mathbf{v}_2 + b_3\mathbf{v}_3 \end{aligned}$$

As a precalculation, a table of 2048 random triplets of barycentric coordinates (b_1, b_2, b_3) is created, to avoid the need to store **h** for each hair. The position of the root of each hair is generated by cycling through the values in this table, and using them as barycentric coordinates for the triangle being drawn. Because we start at the beginning of the table each time the animal is redrawn, we generate the same hairs each time.

Skin motion during animation uses an anatomically based method described elsewhere [16, 17]. Each vertex of the skin’s triangle mesh is transformed to its present position. Although triangles shift and may change shape as a result, the hairs’ locations, being retained in barycentric coordinates, can be efficiently recalculated.

Associated with each skin *vertex* is a normal vector, which is used both in shading the skin and in orienting the fur. Again using the barycentric coordinates of each hair root, the normal vectors of the three vertices of the triangle are interpolated to provide a normal vector at the individual hair root, which is called the *skin normal*. Also associated with each skin vertex is a length for the fur at that vertex, and these lengths are interpolated with the barycentric coordinates to provide the lengths of individual hairs.

In principle, any method can be used to define hair lengths at all the skin vertices, but with over 100,000 such vertices, a hand-crafted assignment is not feasible. A default length is assigned for regions not subject to special treatment. A preprocessing procedure has been implemented to reduce the length of fur automatically over the face and hands, and other desired areas. The user can also interactively scale the length of all hairs at once or of all hairs in a selected 3D region.

Hair is given a user-defined material type consisting of ambient, diffuse, and specular reflectivity. To complete the rendering specification, it is necessary to define the hair’s shape, orientation, and normal vectors for lighting. These factors are influenced by segment direction and gravity, as discussed next. The other details are described separately for each hair style in following sections.

3.3 Distance-Based Culling

We find it desirable to scale automatically the number of hairs drawn, depending on how far the animal is from the viewer. If we do not do this, and use the full number of hairs at all distances, the animal’s surface looks like a single smooth material when viewed at greater distances. This is in part due to the fact that line, polylines, and nurbs curves drawn using the SGI hardware are minimally a pixel in width. Thus, when distance is multiplied by some factor c, the animal’s projected area is divided by c^2, but the hair remains one pixel wide, so its area is only divided by c.

To compensate automatically for limits of pixel resolution, we draw the full number of hairs at the distance such that approximately the area of the animal’s face, or less, is visible on the screen. We decrease the number of hairs drawn by the the zoom factor $1/c$ for greater viewing distances. That is, when the viewing distance doubles, the number of hairs drawn decreases by a factor of two.

For each triangle the number of hairs to be drawn is calculated according to the above factor, applied to the triangle’s allocated number of hairs. Then, sufficiently many hairs are culled to achieve the reduced allocation *on average*. Again, “random rounding”, as described in Section 3.2 for the triangle’s original allocation, is important to prevent small triangles from all getting rounded down to zero. To maintain the uniform spatial distribution at the reduced density, the hairs are accessed in the order they were created, which gives a spatially random sequence. As many as are needed are selected for drawing from the beginning of this sequence.

Figure 6 shows a monkey with polyline fur at two different distances. 190,000 hairs are drawn at a distance view at the top. The distance view at the bottom shows the effect of drawing 500,000 hairs at this distance. The close-up version of this fur (with 500,000 hairs) can be seen in Figure 9.

3.4 Influence of Segment Direction

The initial orientation of the hair, before applying gravity, is a weighted combination of the *skin normal* direction and a *body segment direction vector*. We call this direction vector the *weightless hair direction*.

Animal fur is often oriented along the longitudinal axis of a body segment, such as the leg or tail. We use a hierarchical body, in which skin vertices are associated with the body segment to which they are closest. Thus it is relatively easy to bend hairs on the skin according to their underlying segment. However, in a hierarchical model, the desired direction may not be obvious. I.e. if the body hierarchy is rooted in the lumbar region, the thoracic region and head have a longitudinal axis pointing up, while the legs and tail would have longitudinal axes pointing down. Animal hairs typically are oriented from head to tail.

Therefore, the user can specify a segment orientation vector for each body segment, if a segment direction influence is desired. The user can also specify a weighting factor, which determines the relative influence on the weightless hair direction of the skin normal and the segment direction vector. Figure 5 shows the monkey model with a weightless hair direction based on 0% segment direction and 100% skin-normal direction on left, and 50% skin-normal direction and 50% segment direction on right. There is no gravitational influence on the fur in these images.

3.5 Influence of Gravity

The location of the hair root, the hair's *length*, *skin normal*, and *weightless hair direction*, are determined as described above. These values, together with the *floppiness* factor ($floppiness = 1.0 - stiffness$), determine the hair's final orientation and shape.

First, the *weightless ending point* is determined simply by scaling the weightless hair direction by hair *length* and adding it to the hair root location. The resulting point may be perturbed randomly by a user-specified amount, to provide some irregularity in hair directions.

The *ending point* of the hair is at a distance given by *length* (overcoat and undercoat hairs have different lengths), in a direction determined by the weightless hair direction and floppiness, as shown in Figure 3. Let θ be the angle between the weightless hair direction and the skin normal, and let f be the floppiness factor, between 0.02 and 1.0. Then $f \cos\theta$ is subtracted from the Z coordinate of the hair's weightless hair direction vector. (Z is up in the world coordinate system and all vectors are in world coordinates.) This results in a new direction vector, which is then renormalized to length 1.0, and scaled by *length*. Adding this to the hair root location gives the actual *ending point*.

Figure:	8	9
Stiffness	0.2	0.2
Normal Randomize	0.0	0.5
Hair End Randomize	0.2	0.1
Segment Direction Wt	0.2	0.3
Ambient (RGB)	0.3 0.14 0.0	0.15 0.07 0.0
Diffuse (RGB)	0.3 0.14 0.0	0.3 0.14 0.0
Specular (RGB)	0.6 0.6 0.6	0.6 0.6 0.6
Shininess (RGB)	16	50

Figure 2: Properties of fur in various Figures.

A floppiness near 0 causes the *ending point* to be very near the *weightless ending point*, while a floppiness near 1 causes the *ending point* to be very near the plane of the skin triangle, and in the direction that the hair would fall from its weightless ending point. Figure 7 shows the effect of a floppiness of 0.8 on the three hair styles line, polyline, and nurbs curve.

3.6 Straight Line Fur

The simplest hair is a straight line emanating from the hair root, proceeding to the ending point described in Section 3.5. This line is drawn using the skin normal for shading purposes. The far left image in Figure 8 shows the monkey with line fur. This fur was rendered at about 35,000 hairs/second on an SGI Reality Engine II. The defining characteristics used are shown in Figure 2.

3.7 Polyline Fur

A polyline hair is a sequence of straight line segments simulating a curve. We found that seven lines give a good visual approximation for fur. This is probably too few for long human hairs. Using individual line segments makes it possible to change the shading normal for each segment.

As described in Section 3.5, the *weightless ending point* and the *ending point* are found from the hair root direction vector, the hair root location, and the hair length, taking into account the floppiness. The plane of the hair is defined by the hair root, the weightless ending point and the ending point. To avoid degeneracy, floppiness is required to be at least .02. Now eight "control points" are transformed into this plane such that the sequence begins at the hair root, proceeds first toward the weightless ending point, then bends around toward the ending point (Figure 3). A linear B-spline is defined, in effect, by these control points, but the GL line drawing routines, not the spline routines, are used because they

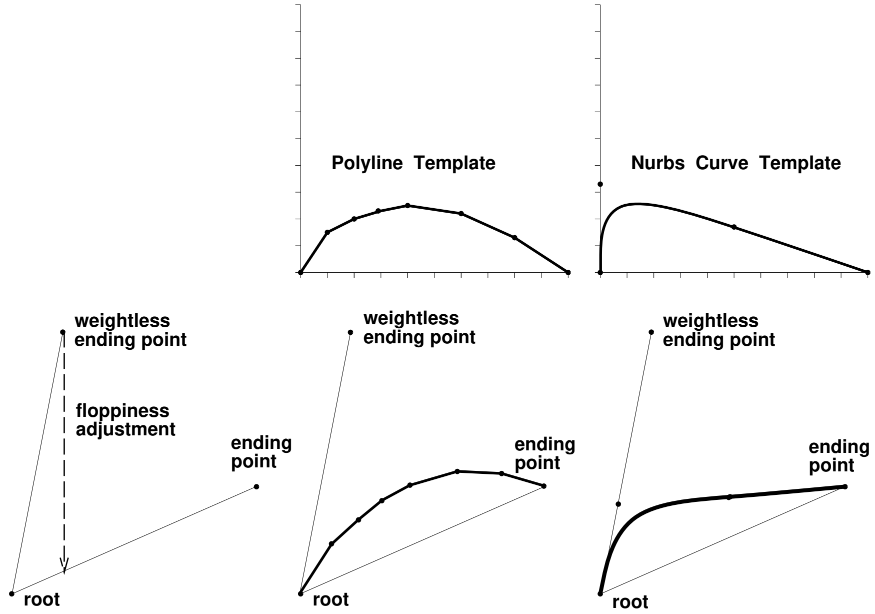

Figure 3: *Weightless ending point* is translated by *floppiness adjustment* ($floppiness = 1.0 - stiffness$) in the direction of gravity to determine the direction for *ending point*. Root, weightless ending point and ending point determine a non-orthogonal coordinate system, into which the template control points are transformed. Controls points, which may be for polyline, nurbs curve, or nurbs surface, define the hair.

permit shading normals and texture coordinates to be specified at each control point.

The essential innovation of this technique is the definition of shading normals. First, let us assume that the user has not requested any randomness in the shading normals. Then all shading normals will be in the "plane of the hair", defined above. At each control point the shading normal is perpendicular to the line segment that ends at that point. The zero-th control point is the hair root, and its shading normal is perpendicular to the direction of the first segment. The reason behind this method is that the line segment of hair is being thought of as a very thin cylinder. All normals to this cylinder surface are perpendicular to the cylinder axis, which is the line segment.

Now, let us assume the user has requested randomness in the shading normals. Then each shading normal is rotated through a random angle, with the line segment as axis. Thus it remains perpendicular to the line segment, and is consistent with a very thin cylinder's normal. This randomness of shading normals is intended to capture the effect of hairs "catching the light" differently.

The second from left image in Figure 8 shows the monkey with polyline fur defined as in Figure 2. This fur was rendered at about 15,000 hairs per second.

3.8 Nurbs Curve and Nurbs Surface Fur

For nurbs curve or nurbs surface fur, we calculate the appropriate control points and use the GL library to draw the curve or surface. In the case of a nurbs curve, this only allows one normal vector to be associated with the hair for shading, and does not provide for a texture map to be specified, so the entire hair receives a uniform color. Our implementation currently uses the skin normal as the shading normal. Nurbs surfaces have the advantage that a separate surface normal is computed at each control point, and texture coordinates may be defined at each control point, as well.

The nurbs-curve control points are defined in the plane determined by the root, weightless ending point, and ending point. This plane is found by the same method as for polylines (Section 3.7).

For a nurbs surface, each nurbs-curve control point is replaced by a circular sequence of control points in a plane orthogonal to that of the original control points. The result is that the nurbs surface is a very narrow tube around what would be the nurbs curve.

Although control points may specify any nurbs curve, our implementation defines a Bézier cubic that begins at the hair root, in the direction of the skin normal, and curves to end at the ending point, remaining in one plane (Figure 3). This requires just four control points. However, the nurbs technique permits hairs that trace a genuine 3D shape, such as a helical curve, at the expensive of more control points. Such spiral shapes are appropriate for some animals.

We found it significantly slower to use the nurbs-curve method, compared to polylines, and the use of a single normal limits the lighting effects. Nurbs surfaces are orders of magnitude slower than line-based methods.

The second from right image in Figure 8 shows the monkey with nurbs curve fur and the right image shows the monkey with nurbs surface fur, both defined as in Figure 2. The nurbs curve fur was rendered at about 12,500 hairs per second, and the nurbs surface fur was rendered at about 3,200 hairs per second. More fur images can be seen on our web site: `www.cse.ucsc.edu/~wilhelms/fauna`.

4 Discussion and Conclusions

We do not normally explicitly store information about individual hairs. We experimented with storing this information for straight line fur, and found it no faster than recalculating the fur from scratch. Recalculation is more desirable, because whenever the animal moves, the effect of gravity causes hair to change orientation, dependent on body position.

We were most satisfied with the combination of an overcoat of relatively curved fur, and an undercoat of relatively straight fur about 0.3 times as long. We found a small amount of randomization of the hair direction and normals creates a more realistic effect. We generally preferred polyline fur to the other choices, as it gives a curved shape, gives control over shading normal vectors, and is acceptably fast. Figure 9 shows the monkey with a fur defined by both an overcoat and undercoat. Figure 2 shows the characteristics for this fur. Figure 10 shows a selection of monkey images.

In conclusion, we were pleasantly surprised to find that it is possible to make reasonably realistic fur comfortably quickly using line and polyline methods, in combination with graphics hardware acceleration.

Acknowledgments

Research supported by a gift from Research and Development Laboratories, and by NSF Grant CDA-9115268.

References

[1] M. Ando and S. Morishima. Expression and motion control of hair using fast collision detection. In R. Chin *et al.*, editor, *Image Analysis Applications and Computer Graphics*, pages 463–70, Hong Kong, December 1995. Springer-Verlag.

[2] Ken-ichi Anjyo, Yoshiakai Usami, and Tsuneya Kurihara. A simple method for extracting the natural beauty of hair. *Computer Graphics (ACM SIGGRAPH Proceedings)*, 26(2):111–120, July 1992.

[3] A. Daldegan, N. M. Thalmann, T. Kurihara, and D. Thalmann. An interated system for modeling, animating, and rendering hair. In *Eurographics '93*, pages 211–21, Barcelona, Spain, 1993.

[4] J.-M. Dischler and D. Ghazanfarpour. A geometrical based method for highly structured texture generation. *Computer Graphics Forum*, 14(4):203–15, October 1995.

[5] James T. Kajiya. Rendering fur with three dimensional textures. *Computer Graphics (ACM SIGGRAPH Proceedings)*, 23(3):271–280, July 1989.

[6] N. Magnenat-Thalmann, S. Carion, M. Courchesne, P. Volino, et al. Virtual clothes, hair and skin for beautiful top models. In *Proceedings of Computer Graphics International*, pages 132–141, South Korea, 1996. IEEE Computer Society Press.

[7] Gavin Miller. From wire-frames to furry animals. In *Graphics Interface '88*, pages 138–145, Edmonton, Alberta, June 1988.

[8] F. Neyret. A general and multiscale model for volumetric textures. In W. Davis and P. Prusinkiewicz, editors, *Proceedings of Graphics Interface '95*, pages 83–91, Toronto, Ontario, Canada, May 1995.

[9] Ken Perlin. Hypertextures. *Computer Graphics (ACM Siggraph Proceedings)*, 23(3):253–262, July 1989.

[10] J. Rankin and R. Hall. A simple naturalistic hair model. *Computer Graphics*, 30(1):5–9, Jan. 1996.

[11] R. E. Rosenblum, W. E. Carlson, and E. T. Tripp. Simulating the structure and dynamics of human hair: Modeling, rendering, and animation. *The Journal of Visualization and Computer Animation*, 2:141–148, 1991.

[12] Zen-Chung Shih and Hurng-Dar Guo. The modeling and animation of human hair. In *Pacific Graphics '94*, pages 215–228, Beijing, China, 1994.

[13] Robert Skinner. Modeling hair with structured particle systems. Master's thesis, University of California, Santa Cruz, Santa Cruz, CA, June 1989.

[14] John M. Snyder. *Generative Modeling for Computer Graphics*. Academic Press, Boston, Mass., 1992.

[15] Y. Watanabe and Y. Suenaga. A trigonal prism-based method for hair image generation. *IEEE Computer Graphics and Applications*, 12(1):47–53, January 1992.

[16] Jane Wilhelms. Animals with anatomy. *IEEE Computer Graphics and Applications*, 17(3), May 1997.

[17] Jane Wilhelms and Allen Van Gelder. Anatomically based modeling. In *Computer Graphics (ACM SIGGRAPH Proceedings)*, Aug. 1997.

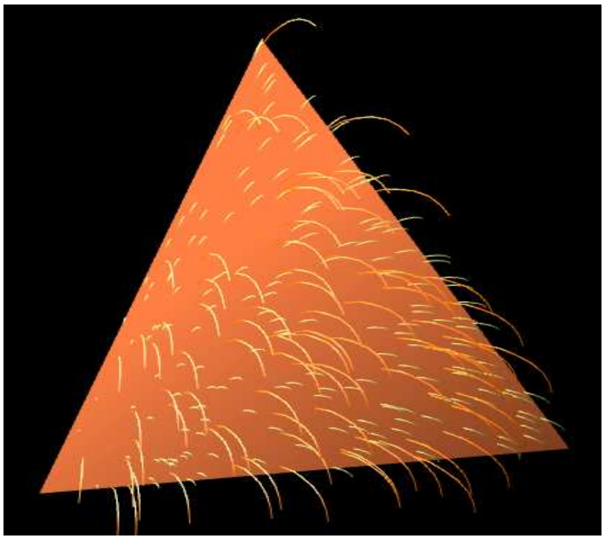

Figure 4: Fur Overcoat and Undercoat combined, as discussed in Section 3.1.

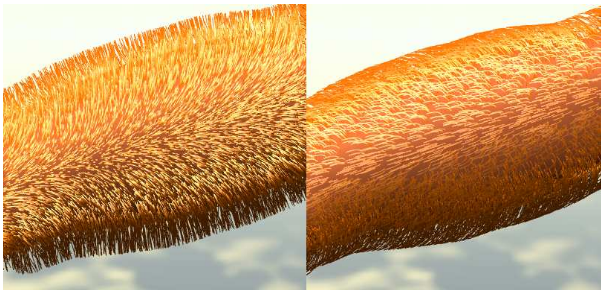

Figure 5: Weightless hair direction based on 0% segment direction and 100% skin-normal direction on left, and 50% skin-normal direction and 50% segment direction on right (Section 3.4).

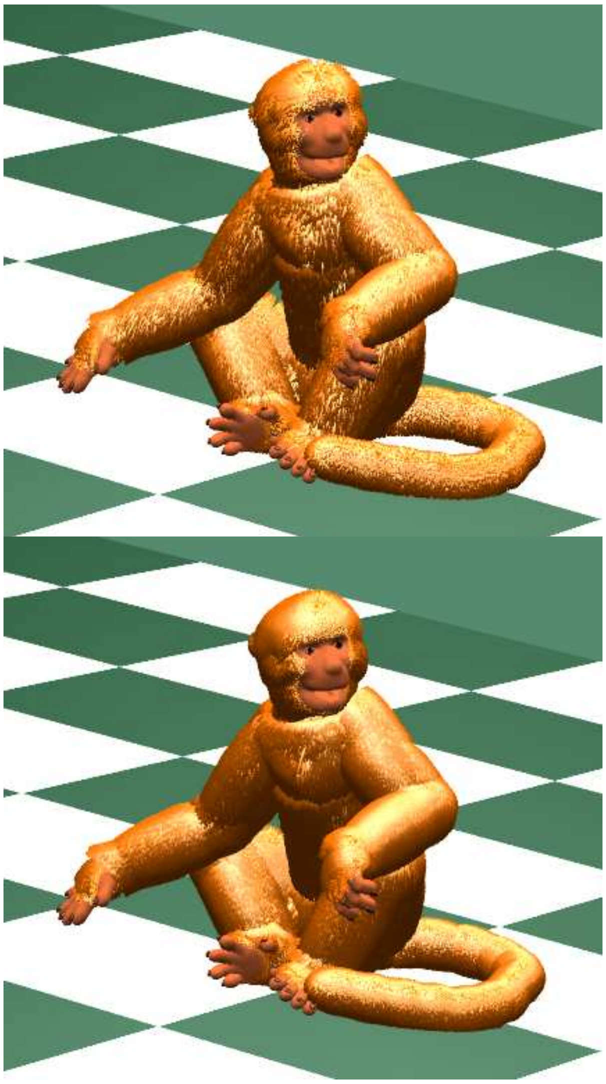

Figure 6: Effect of distance culling (top) and no distance culling (bottom), relative to full hair in the Figure 9. The hair is drawn using an over and undercoat of polylines.

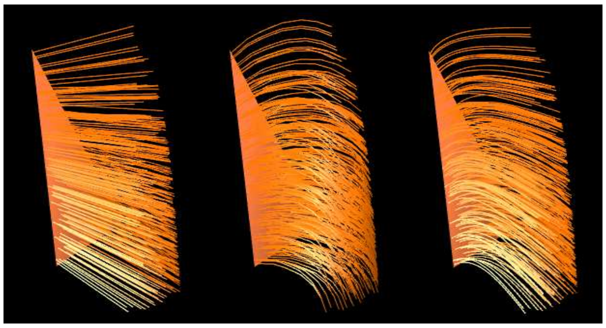

Figure 7: Effect of 0.2 stiffness on line, polyline, and nurbs curve fur, as discussed in Section 3.5.

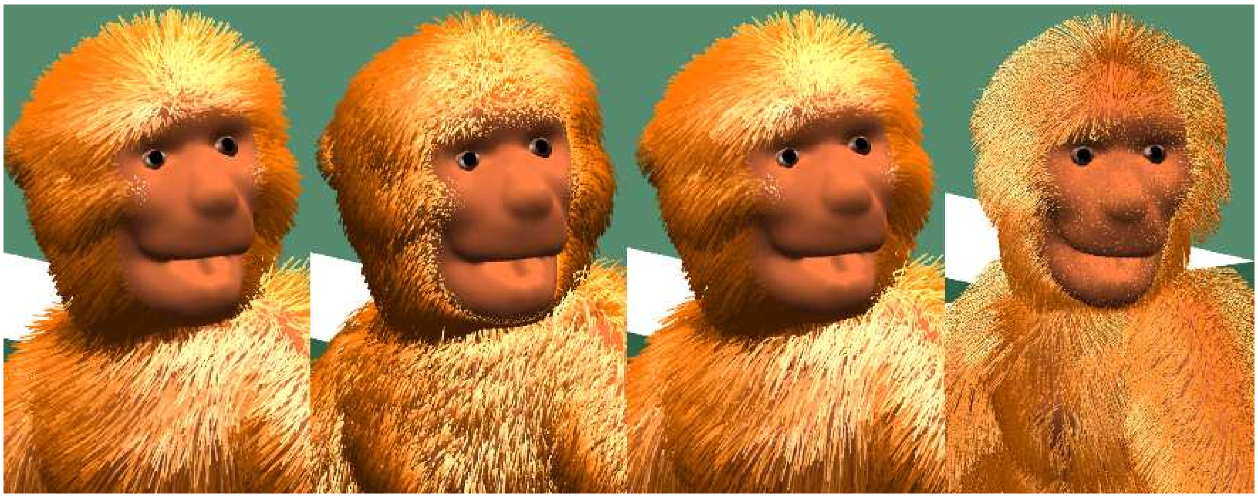

Figure 8: The four fur types (left to right): *line* (2.6 sec.), *polyline* (6.1 sec.), *nurbs curve* (70.2 sec.), *nurbs surface* (277.5 sec.), clipped from full monkey images of about 90,000 hairs and drawn in a 900x900 pixel image.

Figure 9: Part of image of monkey with an overcoat and undercoat of polyline fur shown close up (500,000 hairs on whole animal). Rendering of whole image at 900x900 pixels took 39.4 CPU seconds on an SGI R.E. II with a 150 MHz processor.

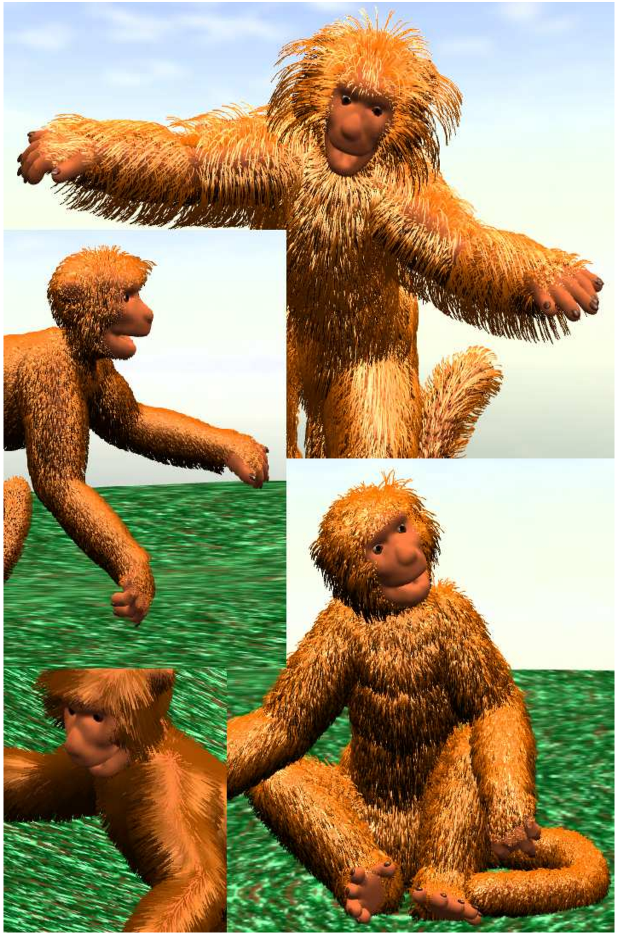

Figure 10: A selection of monkey images with varied furs: The upper is extra long polyline fur. The middle left image is short dense polyline fur. The lower right image is polyline fur with considerable randomness. The lower left image is line fur with little randomness.