A Simple and Fast Technique for Fur Rendering

Technical Report - Tutorial

Georgios Papaioannou

Department of Informatics, University of Athens

Contact information:

Georgios Papaioannou

Department of Informatics & Telecommunications,
Panepistimioupolis, Ilisia,
15784, Athens,
GREECE

e-mail : georgep@di.uoa.gr
web page : http://www.di.uoa.gr/~georgep/

Abstract. Recently an effective technique has been introduced in graphics bibliography for simulating and rendering properly shaded fur, grass or similar textures using multiple layers of concentric textures. This paper is a guide to simple step-by-step modeling of furry objects with shadows that can be directly implemented either in standard rendering software or in rendering APIs for real time applications, without requiring specialized graphics hardware.

1. Introduction

Realistic and efficient simulation of fur and similar textures is a problem that has captured the interest of computer graphics researchers since the late ’70s, and still remains an open issue. Techniques can be classified in three main categories: explicit geometric modeling, volume density modeling and approximation of the fur-coating lighting properties.

The first class simulates explicitly every hair [Miller 88, LeBlanc et al. 91], grass blade [Reeves, Blau 85] or strand of hair [Yang et al. 00] as an individual geometric entity. These techniques are capable of rendering accurate short fur-like texture or even long and stylized hair, at the expense of severe computational overhead. In [Van Gelder, Wilhelms 97] a faster approach was adopted, using simplified geometry (curves and polylines), but the results were of relatively poor visual quality.

The macroscopic effect of fur coating was simulated via the estimation of the BRDF in [Westin et al. 92] and a probabilistic lighting model that also implements shadow interaction and translucency in [Goldman 97]. The later is quite efficient and produces very convincing results when fur is viewed from a far distance.

Volume textures for fur rendering were introduced in [Perlin, Hoffert 89] and [Kajiya, Kay 89], while an interesting work in this direction has been conducted in [Neyret 98]. The above methods are mostly targeted towards ray tracing and suffer from increased rendering cost.

Two recent papers by J. Lengyel [Lengyel 00, Lengyel et al 01] present a multiple level-of-detail fur simulation approach that combines geometric modeling with a simplified volumetric method. When the finest detail is used for close-upo inspection, geometric hair representation is adopted, while when moving far from the object the fur coating is modeled as a set of concentric texture layers, with a displacement from the surface. These layers can be regarded as cut-sections of the volume texture.

In this paper, we describe a step-by-step guide for fur-like texture simulation, based on Lengyel’s ideas, from the implementation point of view. More specifically we concentrate on the layered texture approach, which can easily be both realized using commercial renderers and programmed in custom software. Furthermore we discuss the effect of standard shadow generation methods and propose a simple extension to the layered texture approach that incorporates hair-over-hair and hair-over-skin shadows.

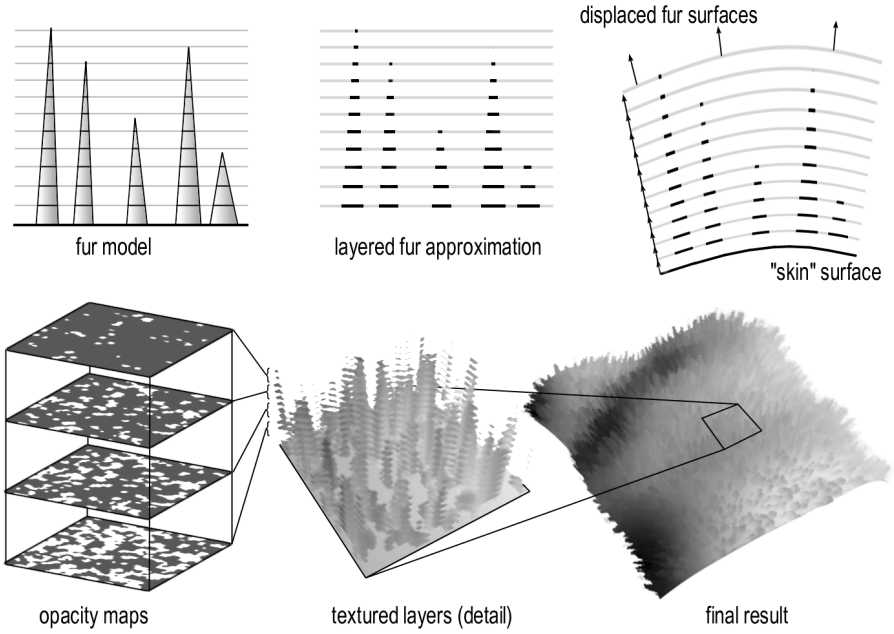

Figure 1. Procedure overview. The fur model is approximated by a set of surface layers. Opacity maps are used to control the transparency of the fur coating with respect to the distance from the skin.

2. Procedure Overview

The approach presented is similar to the volume texture techniques, but instead of tiling volume textures on the object surface, the modeling is simplified by using a small number of texture slices parallel to the surface. When these slices are tightly stacked one on top of the other, the visual effect of continuous volumetric fur is achieved. Each separate fur layer consists of a displaced version of the underlying object surface and appropriate texture mapping is used to modulate its opacity (Figure 1).

This technique requires no specific hair (or grass blade) representation model. As fur is modeled by two-dimensional opacity maps, the hair distribution and thickness may be generated in various ways, even by hand-painting them on the maps. In the next section we show a practical way to construct the texture maps for the individual texture layers. As can be seen in the close up detail of Figure 1, the layered fur can not stand a very close examination; however, it is a good tradeoff between an accurate but expensive geometric model or volume texture and a simple yet restricted to distant objects illumination model.

As will be described in section 4, this layered fur implementation facilitates the generation of accurate hair-over-hair and hair-over-surface shadow generation in a simple and unified manner.

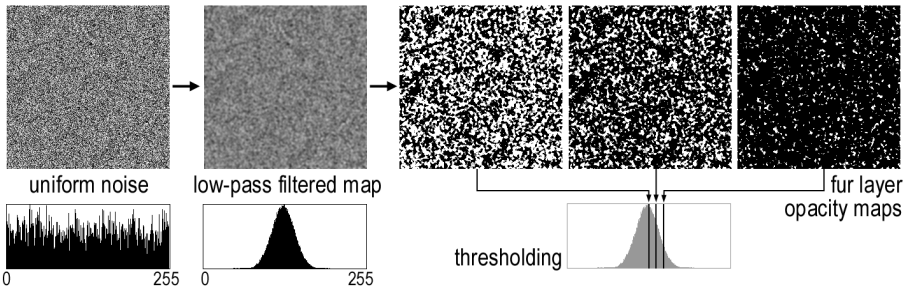

Figure 2. The fur opacity layers generation.

3. Generation of Fur Opacity Layers

In order to easily construct the separate fur layers, the object surface is displaced along the direction of the normal vector to create a set of surface shells. The number of shells depends on how close the fur is to be examined. The desired effect is achieved when the inter-layer distance corresponds to less than 2 pixels in the screen space.

The opacity maps for each layer can be derived from a single image as follows: At a first stage, a grayscale image is created using random pixel intensities (noise). Subsequently, this image is smoothed out using a blurring filter, such as a 3×3 or larger averaging mask, depending on the hair size. In the resulting image, high intensity pixels correspond to hair locations. The exact intensity of these pixels defines the corresponding hair length. Each separate opacity map is produced by thresholding the smoothed image at different threshold values.

In the common case, where the base of each hair is thicker than the tip, opacity maps near the skin surface are created using a low threshold that is increased for higher layers. As a result, fewer pixels pass the threshold near the tip of the hair creating the effect of a hair thinning with the distance from the skin.

The opacity map generation process is outlined in Figure 2. In this example, the original map is produced by assigning uniformly distributed random intensity values to each pixel. The image is then blurred with a 5×5 weighted averaging filter mask.

This technique is not restricted to fur. In a similar way it may be applied to other patterns instead of noise, such as weaving to simulate fabric. The same image can be used as a bump map to enhance the embossed effect.

As layers are textured separately, different texture maps can be associated with each one. In this way we may simulate color gradient along the hair, although it is more common to use the same texture map for all layers. In uniform areas of the map, if hair-over-hair shadow is not calculated it is often hard to distinguish the shape of individual hairs. A simple trick to overcome this problem is to add a small amount of noise to the uniform areas of the texture map, so that each hair has a slightly different color.

4. Shadow Generation

When this technique is used in commercial rendering software, shadows are handled by the renderer. Shadow maps are not recommended for accurate shadows as shadows from objects outside the fur layers will be cast properly on the fur coat but hair-over-hair or hair-over skin dimming will not be produced. Z-buffer based shadow maps fail to penetrate the outermost fur layer. On the other hand, properly filtered shadow maps produce soft-edged shadows at the object boundaries, which are consistent with the lighting at the rim of furry patches. Ray-traced shadows work perfectly but are of course a lot slower, especially due to the fact that supersampling is necessary to avoid aliasing effects. In practice however, no accurate shadows are needed for the fur-coated regions of an object due to the fact that the slight hair color variations we introduced for better hair strand definition also simulate the effect of hair-to-hair light blocking (see examples a and b in figure 4).

When programming this technique in custom applications, a more accurate shadow generation algorithm can be derived for relatively flat surfaces. For a given fur layer *i*, all opaque texels of the layers above *i* cast a shadow on it. The areas of the affected layer (or the skin surface) that are in shadow can be easily estimated by modulating the color of the *i*-th layer by shifted versions of the *i*+1 and above opacity maps as demonstrated in figure 3. More specifically, if N_{layers} is the number

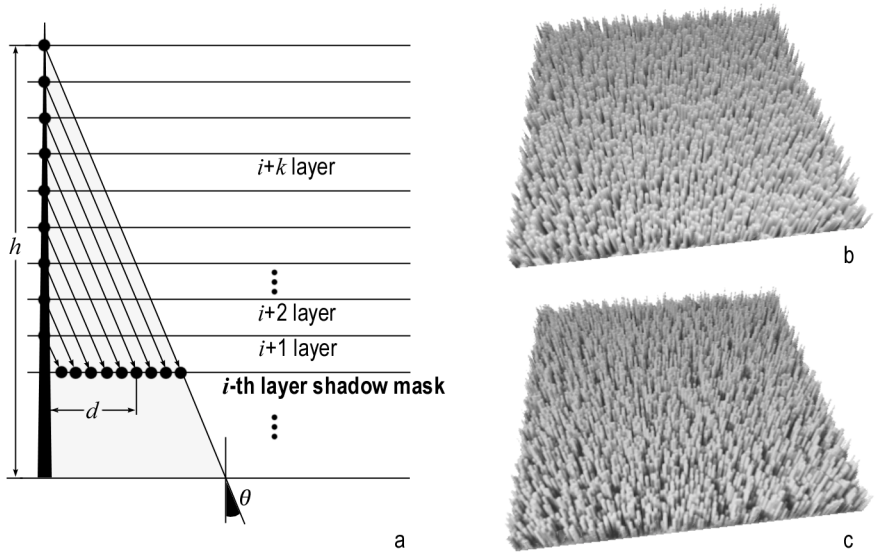

Figure 3. Real time shadow generation. (a) The shadow mask calculation for a given layer. (b) Shadows on a fur-coated patch that is illuminated by a light coming from the left. (c) The same patch as illuminated by a light source further on the left (longer shadows).

of fur layers, the *j*-th opacity map, $i < j \le N_{layers}$, is shifted by $d = h \cdot \tan(\theta) \cdot (j - i) / N_{layers}$ before modulating the color of layer *i*. *h* is the maximum hair height and θ is the angle of the incident light direction. In practice, this means that each fur layer is rendered in multiple passes. In the first pass the current layer color and opacity images are applied directly on the layer surface. All subsequent passes multiply the current surface color with the alpha channel of the overlaying layers, after shifting the texture coordinates according to *d*. As the shadow direction is controlled by the texture coordinates, using this shadow generation technique with highly curved surfaces may produce unrealistic results.

5. Results

We have used the fur rendering technique in both custom real-time applications and existing rendering software. Images (a)-(c) of figure 4 were rendered using 3DStudio MAX. The opacity maps for the hair coating of the doormat in image (c) were modulated by an additional map that represented the hair density. This way, the dark brown inscription on the mat is formed by shorter hair than the rest. A fur density map was also applied to the beard and the hair of the person in image (b) to define their extends.

Examples (a) and (b) use shadow maps while in image (c) shadows were generated via ray-tracing. As is demonstrated in Figure 4c, object interaction is seamless. The cardboard box sinks in the hair coating of the mat (the box penetrates the topmost layers of the fur) and the shadows from the window frame are diffused on the hair.

The image in plate (d) of Figure 4 is a screenshot from an OpenGL demo depicting an animated field blown by the wind, which runs at 10fps on a TNT2 accelerator. Notice that mip-mapping works well with the technique and, apart from eliminating aliasing, it helps obscure the distant strands in a natural way.

References

[Goldman 97] Dan Goldman. “Fake Fur Rendering.” In *Computer Graphics (Proc. SIGGRAPH ’97)*, pp. 127-134 (August 1997).

[Kajiya, Kay 89] James Kajiya and Timothy Kay. “Rendering Fur with Three Dimensional Textures.” In *Computer Graphics (Proc. SIGGRAPH ’89)*, 23(3):271-280 (July 1989).

[LeBlanc et. al. 91] André LeBlanc, Russel Turner and Daniel Thalmann. “Rendering Hair using Pixel Blending and Shadow Buffers.” *Journal of Visualization and Computer Animation*, 2(3): 92-97 (1991).

[Lengyel 00] Jerome Lengyel, “Real-Time Hair”, *11th Eurographics Workshop on Rendering*, 243-256 (June 2000).

[Lengyel et. al. 01] Jerome Lengyel, Emil Praun, Adam Finkelstein and Hugues Hoppe, “Real-Time Fur over Arbitrary Surfaces”, *ACM Symposium on Interactive 3D Graphics*, pp. 227-232 (March 2001).

[Miller 88] Gavin Miller. “From Wire-Frame to Furry Animals.” In *Proc. Graphics Interface ’88*, pp. 138-145 (June 1988).

[Neyret 98] Fabrice Neyret. “Modeling, Animating and Rendering Complex Scenes Using Volumetric Textures.” *IEEE Transactions on Visualization and Computer Graphics*, 4(1):55-70 (January-March 1998).

[Perlin, Hoffert 89] Ken perlin and Eric Hoffert. “Hypertexture.” In *Computer Graphics (Proc. SIGGRAPH ’89)*, 23(3):253-261 (July 1989).

[Reeves, Blau 85] William Reeves and Ricki Blau. “Approximate and Probabilistic Algorithm for Shading and Rendering Structured Particle Systems.” In *Computer Graphics (Proc. SIGGRAPH ’85)*, 19(3):313-322 (July 1985).

[Van Gelder, Wilhelms 97] Allen Van Gelder and Jane Wilhelms. “An Interactive Fur Modeling Technique.” In *Proc. Graphics Interface ’97*, pp. 181-188 (May 1997).

[Westin et al. 92] Stephen Westin, James Arvo and Kenneth Torrance. “Predicting Reflectance Functions from Complex Surfaces.” In *Computer Graphics (Proc. SIGGRAPH ’92)*, 26(2):255-264 (July 1992).

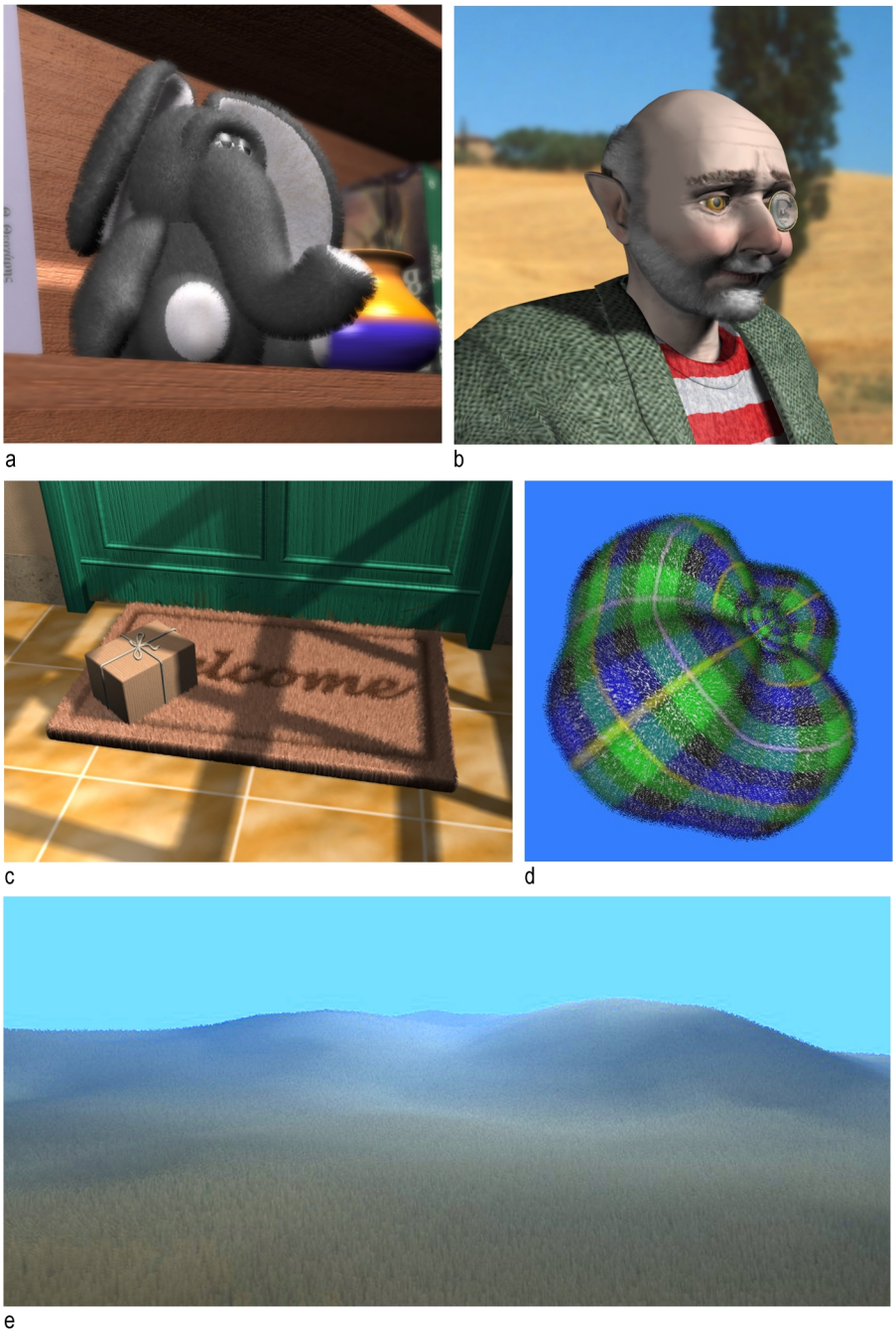

Figure 4. Examples of the method. Images a-c were produced with 3DStudio MAX. Plates d and e are screenshots from the companion OpenGL demos.