Modeling and Rendering of Hair and Fur

Clemens Brandorff

Abstract

The very thin nature, the very high count and the illumination complexity of hair makes modeling and rendering hair a very complex task. Two main representations to model hair exist: volumetric and spline/polyline representations. Both representations have their advantages and disadvantages. This work is a State of the Art Report on how to model and render hair. It should give an overview over the techniques and methods available, including a small section about real hair aquisition.

CR Categories: I.3.7 [Computing Methodologies]: Computer Graphics-Three-Dimensional Graphics and Realism

Keywords: rendering hair/fur, illuminating hair/fur, modeling hair/fur

1 Introduction

In almost every part of computer graphics it is higly needed to render creatures (mostly either humans or animals). To gain visually plausible results of these synthetic creatures, hair or fur has to be rendered according to the physical reality. Speaking of humans, complex hairstyles are needed to truly represent a person. Until now, most of the characters in live action feature films, where it is needed to render a synthesized version of the actors for special effects,wear only simple hairstyles (like ponytails, where no individual hair strands are visible, short hair, or even they are just bald), to avoid the rendering and modeling complexity of the hairstyles (for example the Matrix series). Speaking of animals, where mostly short hair and fur are involved, it is nevertheless important to render this huge amount of hairs at an acceptable speed while keeping the results visually and physically plausible.

Modeling and rendering hair is a highly complex task. Not only the geometric complexity is hard to grasp and model with current computer graphic methods (splines, polylines, texels, polygons, etc.), the illumination properties of a single hair fiber is also highly complex since it is not just an opaque cylinder. Furthermore the small geometry of hair leads to very bad aliasing artifacts if no extensive subpixel sampling or antialiasing is used.

This work is a State of the Art report on how to render and model hair. It presents the most commonly used techniques to model and render hair. Since research in this sector has been done for quite some time (compared to other topics in computer graphics), and a final conclusion is far from been drawn, this work can not be complete. Nevertheless I want to show how difficult this topic can be, and want to present different techniques on how these problems could be solved.

In the following sections i will present some different representations and illumination techniques for hair: real hair aquisition is described in Section 2, where real hairstyles are captured using digital image processing techniques; the volumetric representation including some real-time techniques in Section 3. The polyline/spline approach (used for more complex hairstyles) in Section 4. In Section 5 some advanced illumination techniques are discussed, and in Section 6 two examples of rendering hair for movies are discussed.

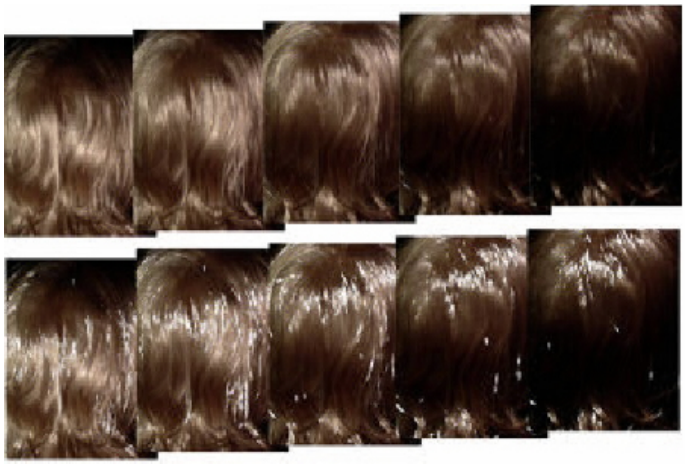


Figure 1: Upper row: pictures from a sequence. Lower row: corresponding masks.

2 Real Hairstyle Aquisition

Real Hair Aquisition tries to capture the geometry of real hairstyles by techniques originating in the field of digital image processing. This techniques generate a geometric model of the outermost hairs of a hairstyle. These geometric models cannot be used per se, since they are not good enough to represent a complex hairstyle, but nevertheless these models could be used as a starting point to interactively generate a pleasant looking hairstyle.

[Grabli et al. 2002] propose an aproach, where a sequence of images are taken from a viewport under controlled lighting conditions. From this sequence, geometrical 3D data of hair strands are generated. This technique works under the assumption that for a given pixel position all pictures in the sequence (from the same viewport) show projections of the same hair strand.

Since the properties of hair result in strong light scattering and high specularity, 3D-scanners are not able to capture the geometry of hairstyles. The authors use these properties to generate models of hairstyles, which could be used by an artist to faciliate the modeling process of hairstyles. The technique proposed here is based on shape-from-shading [Brooks and Horn 1989] and shape-from-specularities [Yang et al. 2003] approaches. First the images have to be taken under controlled conditions (that conditions contain intrinsic camera parameters, camera/object position and lighting conditions for each shot). One important property of the camera is a very high resolution, since hair strands are projected to pixel size. After image generation, a mask is computed for each image, that outlines the hair strands, which are most visible (see Figure 1). Therefore for a given pixel position a number of vectors are generated which represent the projection of the same hair in image space. Out of all these masks a single representative vector is chosen and stored within a so called sequence mask (visible in Figure 2). This mask contains the most visible hair strands of the whole sequence. The

Figure 2: A sequence mask, generated from the masks visible in figure 1.

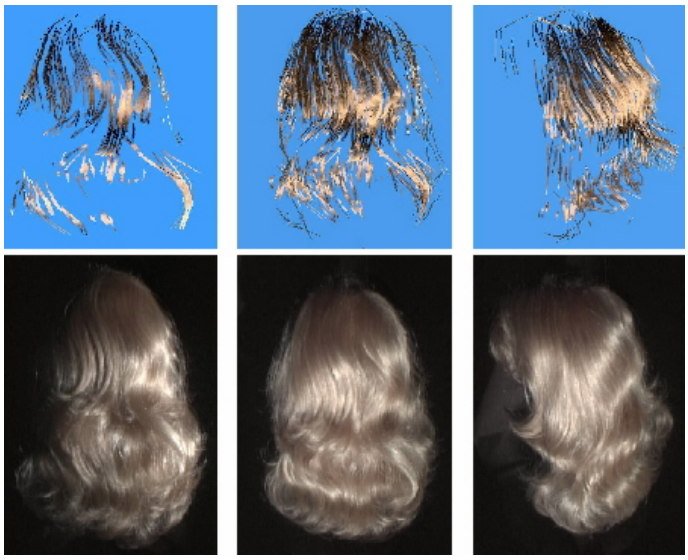


Figure 3: In the upper row are the rebuild hair strands (rendered as broad strands for visual convenience) from the differend viewports below.

2D vectors of this hair strands represent projections of 3D vectors to image space. These 3D vectors are generated using the sequence mask and the illumination changes within the image sequence (using a reflectance map). After the 3D vectors are computed, they are chained and result in the hairstrands.

Results of this technique are shown in figure 3. This technique was only designed as a proof of their theory and work only on a single viewport. Therefore the results look very sparse since only the most prominent hair strands are captured using only one viewport. The main drawbacks are, that the subject has to sit perfectly still while the images are taken, and that this technique is only able to capture the outermost geometry of the hairstyles (the hidden hairs can not be captured). Another drawback is, that a hairstyle is assumed to consist of thin hair strands and that their orientation is visible in the images, therefore hairstyles consisting of thick strands (like dreadlocks) or short hair pointing directly towards the camera cannot be captured

The next technique proposed by **[Paris et al. 2004]** is based on the work of [Grabli et al. 2002] and tries to improve the results of the former technique.

First, images are taken from a fixed viewport under a moving lightsource, using only one camera.The setup can be seen in Figure 4. To generate a 3D model of the whole hairstyle, these sequences have to be taken from various viewports. After the pictures are made, the properties of the images are analyzed using various filters (a large portion of this work considers the development and performance of these filters). In this 2D analysis the orientation of a segment (which is a small section, approximately 1mm) projected on a viewing plane is computed, which results in a plane where the segment is contained.The 2D analysis is followed by a 3D analysis of the illumination variations across the images,where a normal to this segment is computed, which results in a second plane. Intersecting both planes generate the 3D orientation of the segment. The linked segments generate the hair strands. The resulting 3D data of the different viewports has to be registered (this means that

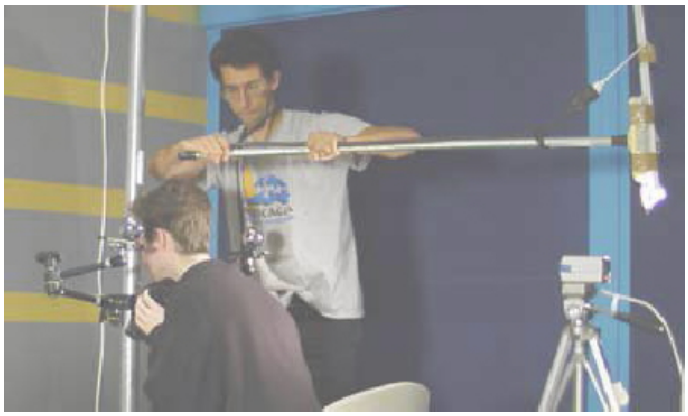


Figure 4: The setup to generate the images from a viewport.

Figure 5: Some models generated from the series of images (the upper right images are photographs as a reference).

the different 3D models are transformed into the same coordinate system). This registration process has to be done manually. There are several drawbacks to this technique. The main drawbacks of this technique are still the same as discussed at the technique of [Grabli et al. 2002]. The authors state that their setup to generate the images has some difficulties with long hair, since long hair tends to move, when the person to be scanned changes the position to get images from another viewport. Nevertheless this in an interesting technique which would be very interesting to be incorporated into other interacive modeling techniques as a starting point.

To render the the results displayed in Figure 5 they used the illumination model described in [Marschner et al. 2003] and rendered each hair strand as antialiased GL lines. Another problem is that the generation of one image sequence takes more than one minute in which the person should be perfectly still. These two problems could be eliminated using more fixed cameras and a generally faster setup to generate the images.

[Wei et al. 2005] propose an improvement over the technique described in [Paris et al. 2004] where they generate the 3D information from multiple viewports, which are registered automatically in advance, and the segments are computed from the registered viewports. This leads to slightly better results displayed in figures 6.

Figure 6: Results from the multiviewport technique (left: photographs of the original subject).

3 Volumetric Hair Representation

Volumetric hair representations try to capture the geometry of hair while representing this geometry in volumetric data structures, like texels,voxels, shell textures, etc. This representations save a lot of computation power since the real geometry is approximated using the volumetric representation, and there for it is suited for real-time rendering (which will be illustrated by three techniques in this section). But since single hair fibers are not individually represented this techniques are only usable for short hair or representations where single hairs are not individually visible (for example if viewed from far).

[Kajiya and Kay 1989] propose to render fur using texels,which are a generalisation of volume densities. Texels are three dimensional arrays of various parameters which approximate a collection of micro surfaces within the volume. These texels are used to approximate the real geometry at a higher level. The parameters stored in the three dimensional array are the density and a lighting model distributed within the volume. The density describes the distribution of the micro surfaces while the lighting model describes the light transport within the volume. The original definition according to [Kajiya and Kay 1989] is : 'A texel is a tripel ρ, B, ψ consisting of a scalar density $\psi(x,y,z)$, a frame bundle $B = [n(x,y,z), t(x,y,z), b(x,y,z)]$,and a bidirectional light reflectance function ρ.' The scalar density ρ approximates how much of the unit area of a volume cell is covered by microsurfaces. The frame bundle B describes the local orientation of the surfaces within the texel, it consists of a field of coordinate basis vectors n (normals), t (tangents) and b (binormal s). The bidirectional light reflectance function ψ describes the type of the surface contained within the texel. The equations to render texels are :

$$T = e^{-r} \sum_{s=tnear}^{tfar} \rho[x(s),y(s),z(s)] \tag{1}$$

$$\begin{aligned} B \quad = \quad & \sum_{t=tnear}^{tfar} e^{-r \sum_{u=tnear}^{tfar} \rho[x(s),y(s),z(s)]} \\ & \times [\sum_i I_i(x(t),y(t),z(t)) \psi(x(t),y(t),z(t),\theta,\phi,\rho)] \\ & \times \rho(x(t),y(t),z(t)) \end{aligned} \tag{2}$$

While Equation 1 computes the transmission (i.e. if the sum is infinite the transmission is zero and therefore the density is totally opaque).Equation 2 computes the brightness along the ray. The coefficient r converts the density into an attenuation coefficient. The I_i describes the incident intensities and are computed by shooting a ray towards each lightsource from point P (see Figure 7). Since rendering texels using these equations is impossibly expensive [Kajiya and Kay 1989] has developed an algorithm where these sums are approximated using a Monte Carlo treatment in the spirit of distributed ray casting. The algorithm works as follows:

1. Intersect a ray with all texel boundaries to find $tnear$ and $tfar$ for each texel. Sort all intersections from front to back and match with distance. Let $Tnear = \min(tnear)$ where the minimum is over all segments. Similary $Tfar = \max(tfar)$.

2. Divide up the ray from $Tnear$ to $Tfar$ into ray segments S_i of length L_i where $\frac{1}{L}$ is a reference length parameter, the number of samples per unit distance in world coordiantes is set by the user.(The last segment may be shorter than L).

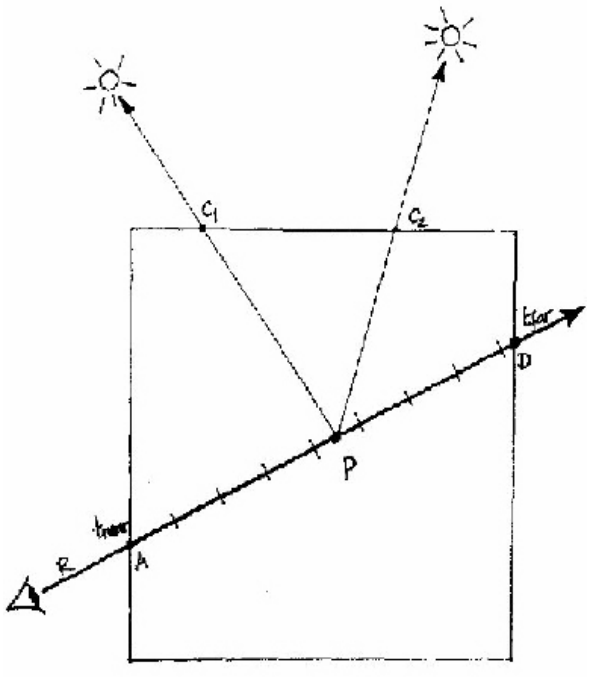


Figure 7: Rendering point P within a texel

3. Set transparency to unity.

4. For each segment:

   (a) Shoot shadow rays from the sample towards every light source to calculate the amount of light reaching the point.

   (b) Calculate brightness from lighting model and illumination intensity and multiply by transparency to give the overall contribution to the pixel ($pixel = pixel + trans * lightModel$).

   (c) Multiply transparency by $e^{r\rho}$ the transmission coefficient of the segments.

5. At the end segment, calculate brightness as above but normalize by fractional length of the segment.

After we have seen what texels are and how they could be rendered, I will present the method of [Kajiya and Kay 1989] on how to generate texels for fur. Since the bidirectional reflectance function ψ is the same for all texels of the same hair type it only has to be stored once for all texels. They treat a single hair fiber as an infinite thin, opaque cylinder and therefore the only necessary component in the frame bundle B is the tangent vector t, the normal and the binormal are omitted. For their reference model (teddy bear) they only used a single texel repeated over the whole surface. The bear texel was stored in a 40x40x10 array and the contents were designed based on the following criteria:

1. The hairs are distributed as a Poisson disk.

2. The Poisson disk is created with a torus topology, so the single texel can tile the entire bears surface without showing seams.

3. Animal fur comes often in two layers, an 'overcoat', and an 'undercoat'. The undercoat is a dense cover of short fur, while

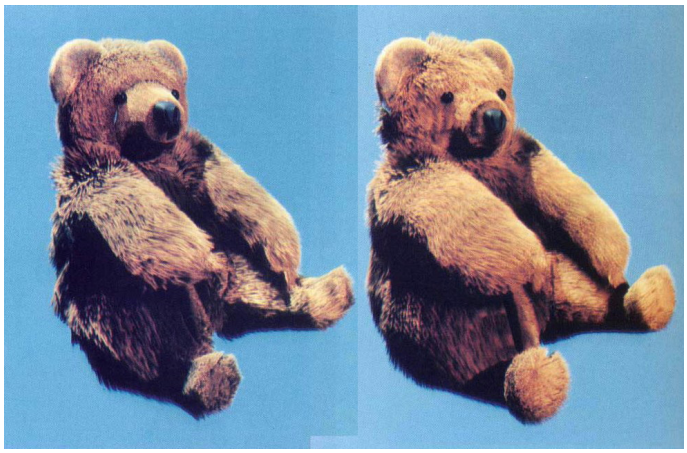


Figure 8: Two versions of the same geometrical teddy bear. The bear on the left used fewer larger texels than the one on the right.

the overcoat is a sparser distribution of long hair.(They found out that this is an important property to avoid a brushlike appearance).

The lighting model used here is seperated in a diffuse component , which is derived from the Lambertian shading model applied to a very small cylinder. The specular component is a model similar to the Phong light reflection model modified for cylindrical surfaces. The equation of the diffuse component is given by:

$$\psi_d = k_d \sin(t,l) \tag{3}$$

where l is the light vector, t is the tangent and k_d is the diffuse reflection component. The specular part is given by:

$$\psi_s = k_s((t \cdot l)(t \cdot e) + \sin(t,l)\sin(t,s))^p \tag{4}$$

where k_s is the specular coeffiicient, e is the vector pointing to the eye, and p is the phong exponent specifying the sharpness of the highlight.

The overall lighting model combines to:

$$\psi_{hair} = \psi_d + \psi_s \tag{5}$$

The render time for the images, the size of 1280x1024 (Figure 8) was about two hours on a network of large IBM mainframes, using twelve 3090 and two 3081 processors (30% to 40% processor usage of each of them). The results look like real Teddy Bears, the only flaw considering real hair is the appearance of the hair which definitively looks synthetic using their illumination model (nevertheless, since it is used to render a synthetic teddy, it is a very plausible illumination model for this task).

3.1 Real-Time Hair

In the field of real-time hair rendering, not much work has been done since hair geometry is not usable ad hoc in real time (either the computers were too slow to compute the real geometry in real-time, or a single hairy object consumes 95% of the computation power of the whole scene when rendered in real-time using splines or poly lines. Therefore some volumetric techniques had to be invented where texture shells are used to represent hair volumetric.

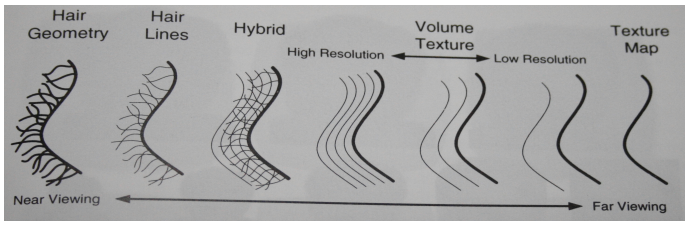


Figure 9: Different levels of detail to represent hair, using shell textures.

[Lengyel 2000] proposes a level of detail approach for real-time hair rendering. Four different view dependant levels of detail are proposed: a geometric representation, a line representation, a shell representation and a hybrid line-shell representation (see Figure 9).The author states that the geometric representation is not included in his system, but it should be a straightforward addidtion.

For the innermost level of his system, he uses alpha-blended, textured lines to represent the hair. As the viewport moves farther away from the object the line-shell hybrid is used. In this representation the alpha value is gradually scaled down to zero as the porgenerated texture shells take over the visual appearance. These shells are gradually removed until only one texture map is used to represent the hair.

The modeling of the different levels of detail is done by generating seeds directly on the surface of the mesh. The hair is then created by growing the seeds as particle system trajectories. This is used for the line representation. The texture shells are created by a simple normal offset of the vertices. The hair integration in the texture shells is done by sampling the particle system to the texture shells and filtering them afterwards by a Gaussian filter kernel. The layers representing lower levels of detail are either derived by filtering their predecessors, or by repeating the hair integration process with lower sampling rates.

To render hair, the author proposes two techniques, per-vertex and per-pixel. The per-vertex technique projects the light vector L and the specular half vector H onto the disk of normals given by the hair tangent, and use standard hardware lighting to actually compute the illumination (see Figure 10).The per-pixel approach only works for direct lighting and a non local viewer (a single view direction for all vertices). It uses the hardware texture matrix to encode the light and eye positions, and a 2D lookup to compute the per pixel lighting.

Furthermore he uses a soft shadow approach where a lightmap is computed every time the light position changes in respect to the object position. First the scene is rendered from the point of view of the light using a combined mesh UV texture map to generate the shadows of the meshes (hair geometry is omitted in this step). Second, samples are gathered from this rendereing, corresponding to each mesh (a small rendering window is used here, so that the number of samples does not fully cover the illuminated region of the object. The last step is to splat a small Gaussian for each light-view sample to fill in between the missing samples and to generate soft edged shadows.

The bear example (visible in Figure 11) discussed in this paper consists of 9406 triangles, 129466 triangles when expanded to 16 shells, and renders at 15 fps (5fps with a moving light source) on an Intel P3 733 MHz and a Nvidia Geforce 256 graphics board. the results of this technique looks pleasant and far better framerates should be archieved using up-to-date graphics hardware.

[Lengyel et al. 2001] propose a technique which archieves real-time framerates while being capable of displaying fur over arbitrary

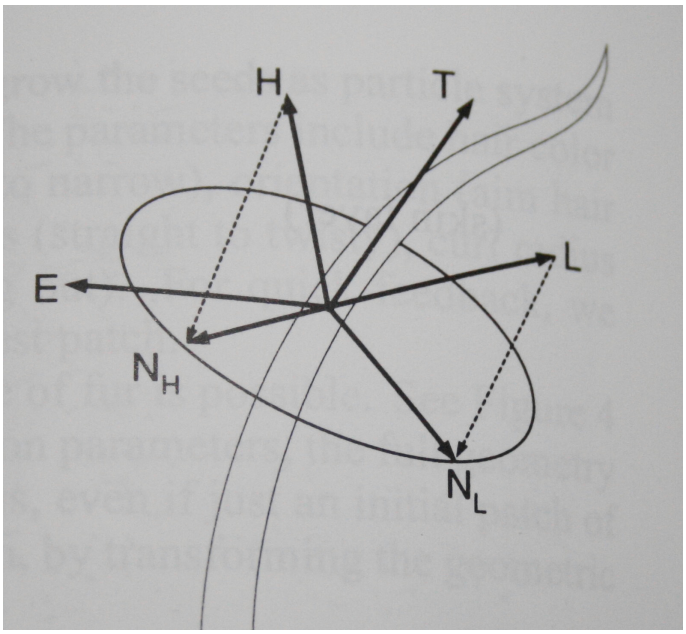


Figure 10: Per-vertex lighting projects the lighting vector L and the specular half vector H onto the disk given by the hair tangent T.

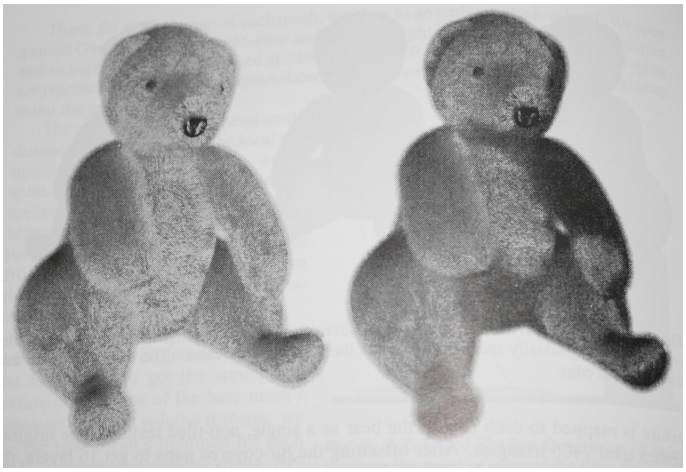


Figure 11: The bear example, left: volume texture only, right: modulated by soft shadows.

surfaces. To archieve this goal they use shell textures proposed in [Lengyel 2000]. This system takes as an input a triangle mesh and a parametric model for hair. Before runtime two operations have to be performed: a geometry preprocessing, and a texture preprocessing. In the geometry preprocessing step the lapped patches parametizing the surface are generated like in [Praun et al. 2000]. Therefore the method grows patches over random uncovered locations on the surface and locally parametrize them onto the texture domain using a fast optimization process. In the texture preprocessing step the components of the texture are created (the shell textures and the fin textures).

First the geometric hair is created by generating hair strands within a rectangular bounding box using a particle system. The shell textures are created by a RGBA-Image for each layer of the shell model by overlaying a grid on the bounding box containing the hair strands. At each gridpoint for every RGBA-Image, colour and opacity are computed using a tall gaussian filter with the geometric hair. the innermost shell is assigned full opacity since it represents the skin. In their implementation they usually used 16 layers, and the texture patches have a resolution of 128x128. The fin textures are used for silhouettes of the model. Only one instance of this texture is generated for the whole model. To generate this instance a slab of the geometric hair along an arbitrary surface tangent direction is chosen where the width of the slab corresponds with the density of hair associated with the fin. This width is chosen manually according on how much density is visually appropriate for a given model and texture. To render the model in real time, the first step is to render an opaque version of the whole mesh setting the z-buffer. The next step is to render the textured fins. The fins are quadrilaterals which are rendered on the silhouette-edges and extended out along the surface normals. To maintain temporal coherence the fins are gradually faded in as the fins approach the silhouette. The last step is to render the offset shells from the innermost to the outermost. For each layer the patches of the lapped texture are composed over what has already been rendered. For lighting a 32x32 texture map table according to [Lengyel 2000] is used. For the shadow, Banks selfshadowing approximation is used to darken the shells near the skin, and therefore to fake selfshadowing.

In addition to lighting and camera interaction, they implemented three more interactive controls: Hair colour,hair length and hair direction.

The authors generated the images in figure 12 on an AMD K7-700Mhz with a Nvidia GeForce DDR32 graphics card using DirectX 7. In Figure 13 are the framerates they archieved rendering the models shown in Figure 12, where the upper row shows the flat-shaded models which were used to generate the furry models below.

[Yang et al. 2006] propose an improvement over the method proposed by [Lengyel et al. 2001]. While the former method uses uniformly distributed texture layers the new method takes into account the eye position and the fur states at various regions on how many texture layers are needed to render the furry model faster while gaining the same results (see Figure 14). Therefore different regions are rendered with a different number of layers (see Figure 15). It is clear that every time the viewpoint changes in respect to the model the suitable number of layers has to be estimated again. While this process takes some time, it greatly reduces the number of polygons rendered. Figure 16 gives an impression on how the framerate improves using the heirarchical model over the uniform model while visually gaining similar results. The second part of this paper considers the modeling of fur. Since this method is represented by multilayer meshes, where these meshes are generated by shifting the vertices of the original mesh outwards from the model surface, the authors propose two fields (a scalar field and a vector field) to interactively control the appearance of fur.The user is able

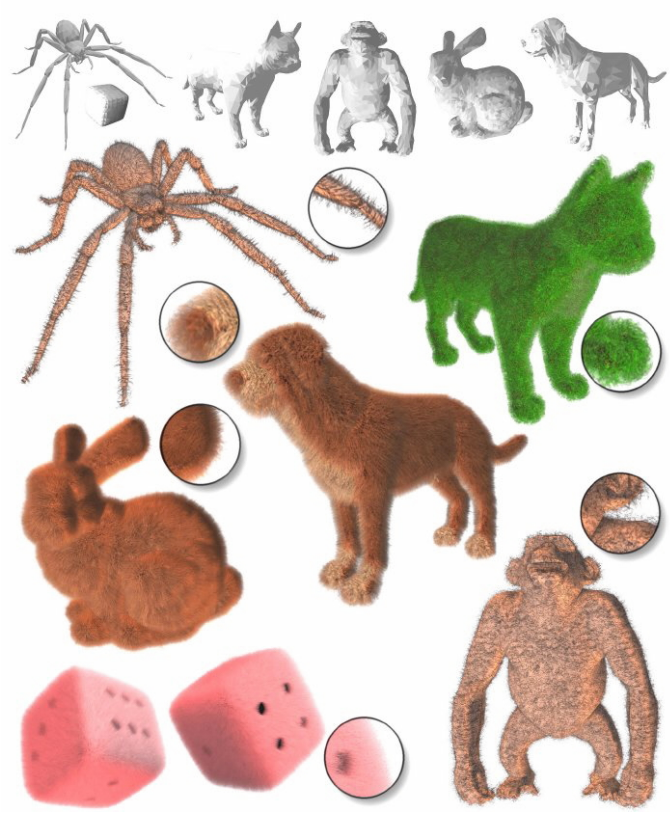


Figure 12: The upper row shows the flat shaded polygon meshes used to generate the furry models below. the models use 16 layers (only the dice uses 32).

Model	bunny	cat	spider	dog	chimp	1die
# faces	5,000	5,000	5,000	5,000	4,910	1,450
# edges	7,547	7,500	7,500	7,500	7,365	2,400
# patches	306	252	338	192	228	246
½ layers (fps)	23	26	31	20	21	25
All layers (fps)	12	14	15	10	11	13

Figure 13: Model complexities and render times.

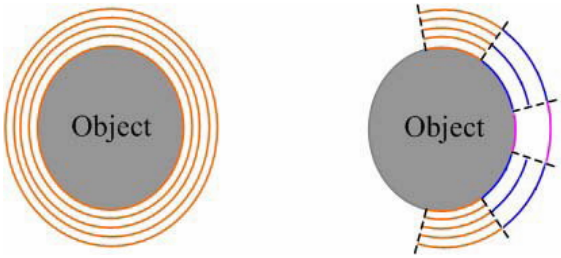
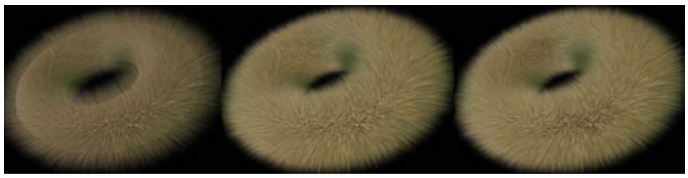


Figure 14: While the left object uses an uniform number of layers, the right object uses only that layers that are needed according to the viewpoint and the geometry.

Figure 15: From left to right: the regions represented with 16, 31 and 61 layers. On the far right: all three hierarchies combined.



a. uniform 16 layers fps: 169 b. uniform 61 layers fps: 49 c. hierarchical layers fps: 86

Figure 16: A comparison between the framerates archieved using uniform layers and hierarchical layers.

to control the fur slant by modyfying the vector field which sets the direction of each vertex when generating multi-layer meshes. The scalar field controls the magnitude of each vertex which in turn controls the fur length. With these two tools to interactively modify the fur, the authors realized five effects including combing of fur, blowing fur (where interesting dynamic effects could be generated), interpolating fur, smoothing fur and disturbing/unifying fur.

The authors implemented their method in Visual C++ 6.0 and measured the framerates on a 2.8Ghz Pentium CPU with 2 GB RAM and a GeForce FX5950 Ultra with 256 MB VRAM.In Figure 17 three models are shown, while the torus is designed mainly using the disturbance operation, the other two incorporate more of the effects.

4 Spline and Polyline Hair Representation

The most intuitive representation for hair fibers or strands is using polylines or splines. This representation consumes a lot of computation, since the amount of hair rendered is usually huge. But

Figure 17: Three models generated using different reshaping techniques.

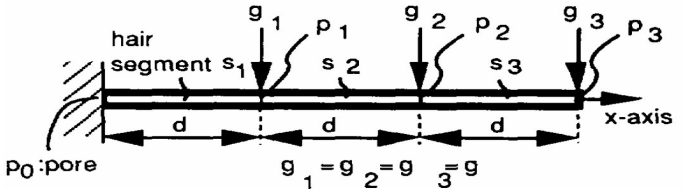


Figure 18: Schematics of a hair as a cantilever beam.

nevertheless this representation is the only usable one when trying to render complex hairstyles where individual hair fibers or strands are visible. Usually this approach does not render in real time, if more complex scenes are rendered. Only one paper uses this representation to archieve interactive framerates (greater 5 fps), and they only render one human head including a complexs hairstyle for modelling purposes, but to generate the final image they dont bother with real-time framerates.

[ichi Anjyo et al. 1992] proposed a model for human hair consisting of a technique for modeling hairstyles and another technique to simulate the dynamical behaviour of hair.

The modeling of hairstyles consists of three steps:

1. Definition of an ellipsoidal hull of the head model which is a rough approximation of the head. The region of the pores (where the hair could grow) is defined on this hull afterwards.

2. Hair bending is calculated according to a simplified simulation of the cantilever beam. Hair-head collisions are calculated in this step too.

3. In the last step the hair is cut and some minor adjustments are applied in order to get the desired hairstyle.

A cantilever beam originally was defined as a straight beam with a one-sided and fixed support (see Figure 18).To calculate hair bending a numerical simulation of the cantilever beam deformation is used. As force is uniformly applied on the beam, two deformations may happen: a shearing force (which is neglected since the authors state that it is not needed for hair bending), and the bending momentum which is crucial for the technique described here. Since only the bending momentum is used here, the cantilever beam simulation is reduced to a two-dimensional case, where the x-axis is the initial direction of the beam away from the head, and the y-axis represents the deflection of the beam. This equation governs the elastic deformation process:

$$\frac{d^2y}{dx^2} = -\frac{M}{E*I} \tag{6}$$

Where E is Youngs modulus and I denotes the second momentum of the area. The term $E*I$ usually is referred to as the flexural rigidity.

Another important part in this cantilever beam simulation is the head-hair collision detection. Since collision detection is a computational very expensive task, only the rather rough approximation is used here.The modeling process is depicted in Figure 19, where 19a shows the initial state where the head and the hair pores are defined. In this state the hair-beams stand radially away from the head. After applying gravity the result depicted in Figure 19b is archieved. Note that the hairs near the top of the head are almost not deformed because the bending momentum is relatively small there. After the cutting operations and after applying an external force field (to bend the hair from Figure 19b) to one side of the hairstyle Figure 19c

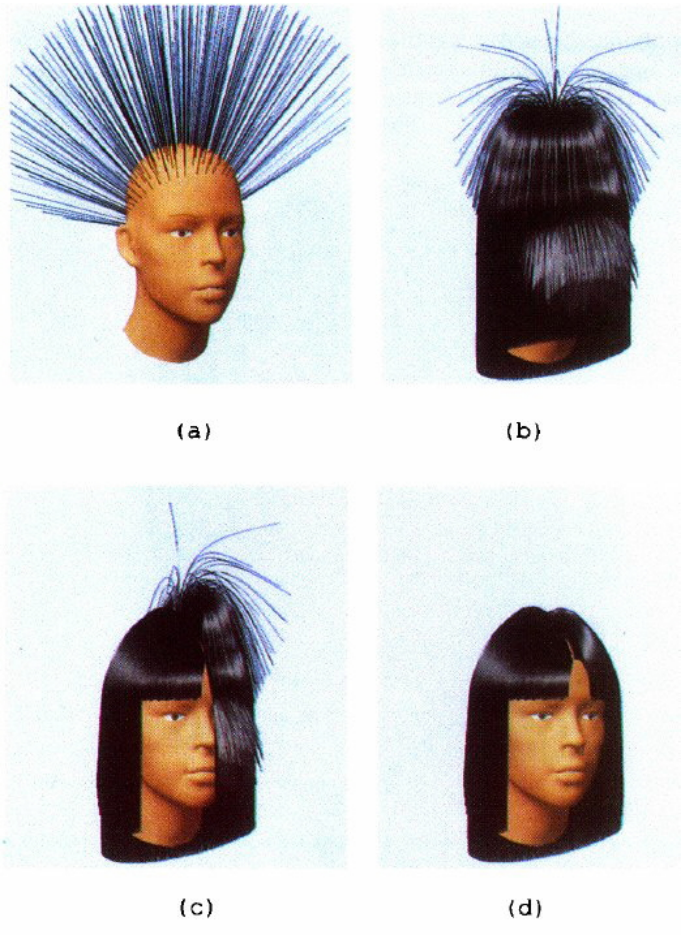


Figure 19: The modelling process using the cantilever beam simulation.

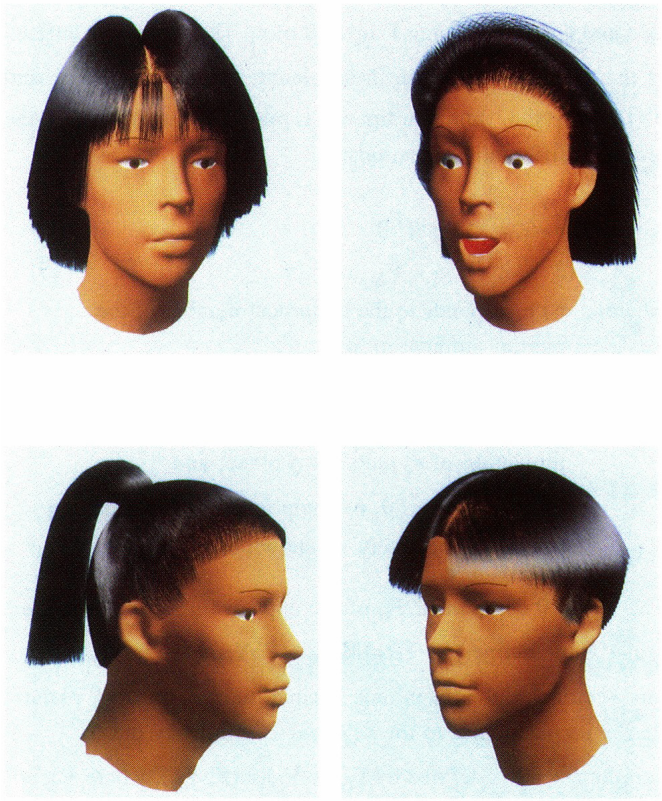


Figure 20: Different hairstyles using the cantilever beam simulation.

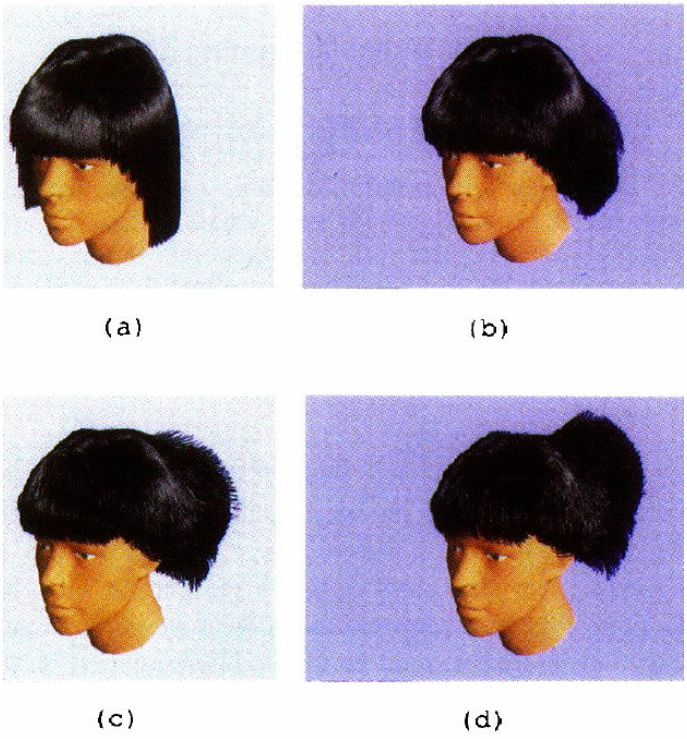


Figure 21: A wind gust scene.

is archieved. Figure 19d is archieved after applying the same cutting operation and forcefield to the other side.Figure 20 shows some more hairstyles generated with this technique.

The second important part of this work is the dynamical behaviour of hair. The authors state that some aesthetic features in hair animation are obtained as a result of inertia versus applied force. Since hair is modeled as a collection of linked linear segments, this segments are treated as rigid sticks for animation. Using this approach simple one-dimensional differential equations are needed to compute the animation. To solve this equations a rough approximation is made using an external pseudo-force field.

Figure 21 shows a wind-gust scene while Figure 22 shows the collision avoidance using a pseudo force field. Since the main aspect of this paper lies in the modeling and animating of hair, i will omit the rendering technique used to gain the resulting images.

The technique described here was implemented on a Silicon Graphics Iris Power Station workstation with a VGX graphics board. Typically the model consists of 20.000 hairs, each of which consists at most of 20 linear segments. It took about 50 seconds. to obtain the bending, while approximately 40 seconds per frame for the animation (bending time not included).

[Kim and Neumann 2000] propose a technique for long hairstyles modeled by surfaces, which includes hair-hair interactions.

The sketches in Figure 23 illustrates the steps of this technique:

1. First the hairstyle is modeled as a gently curved surface. Note that this technique only covers hairstyles which could be modeled as surfaces or polygons (Figure 23a).

2. Next a curved volume is generated by adding a thickness along the normal vectors to the former generated surface (Figure 23b).

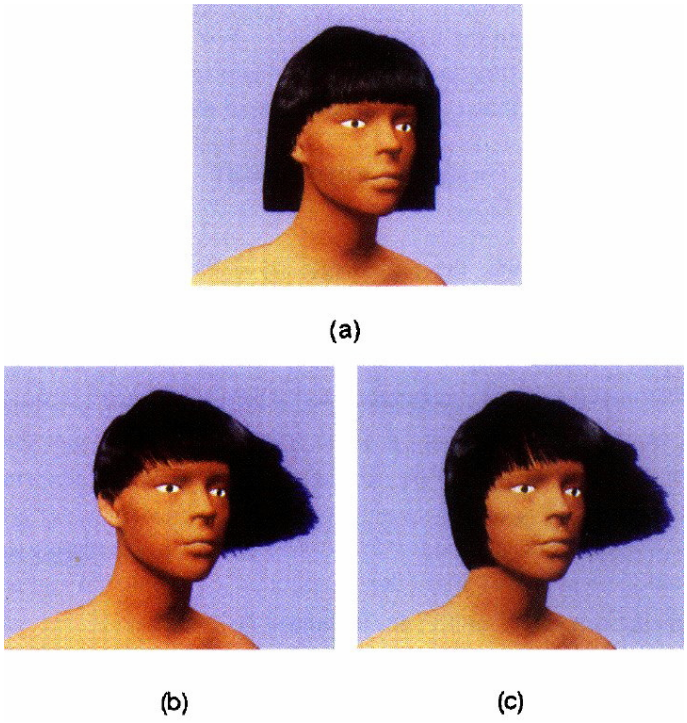


Figure 22: Example of a dynamic collision avoidance.

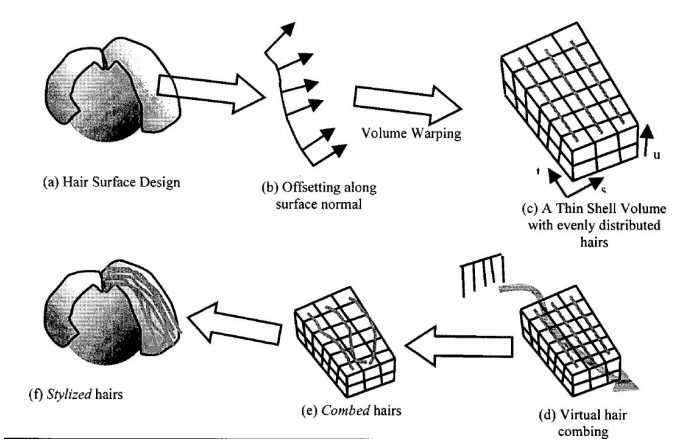


Figure 23: The modeling process using thin shell volumes.

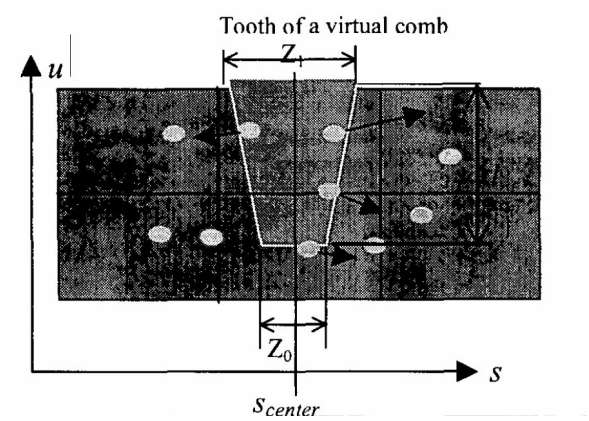


Figure 24: The tooth of the virtual comb.

3. To simplify hair-hair interaction it is easier to use a regular rectilinear volume. That is why the curved volume is implicit warped to a regular rectilinear volume called a normalized thin shell volume. The authors state that the distortions made in this step are visually neglectible, since it only influences hair-hair interaction behaviour. Long hair strands are uniformly distributed within the normalized thin shell volume. The individual hair strands are generated by particles on a path along the length of the hair (Figure 23c).

4. The next step consists of the virtual combing of the hair strands within the thin shell volume. The combing is represented as a path for each tooth of the comb (Figure 23c, Figure 24 shows a tooth of the comb). Each tooth of the comb affects the hair strands as follows:

   (a) Only particles hit by the tooth are affected.

   (b) Each particle moves away from the center of the tooth's influence.

   (c) Each affected particle tries to follow the path of the comb.

   (d) The vertical motions of particles are generated in a way that combed hair moves upward (outward) to lie on top of unaffected hair.

5. After combing each hair strand is recovered by connecting the corresponding hair particles. These stylized hairs are a group of curves, where each curve represents one hair strand (Figure 23e,f).

This technique was implemented on a SGI Onyx R10000 (only a single CPU and the graphics pipeline were used).The images shown in Figure 25 have a size of 512x512 the thin shell volumes used throughout the test have a size of 20x20x10, and 4000 hair strands where computed. It took about 20-30 seconds to comb and generate the model, while the rendering times were quite fast (1-2 seconds per frame). The authors state that most of the computing time was needed to reconstruct the hair strands from the particles in the thin shell volume. To animate the hairstyle, the reference surface could be deformed. Furthermore it is possible to add dynamics to the reference surface. Since the hairs within the thin shell volume are constrained due to the hair-hair interaction technique, it is possible to add free hairs which could be animated using standard dynamics techniques. The results of this technique is very well suited to

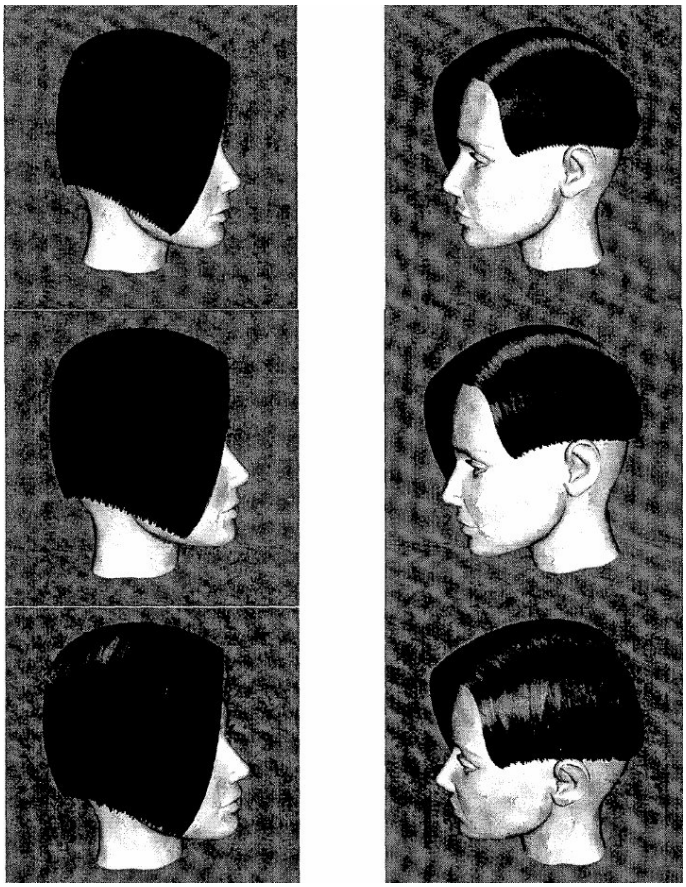


Figure 25: Results of the thin shell volume technique.

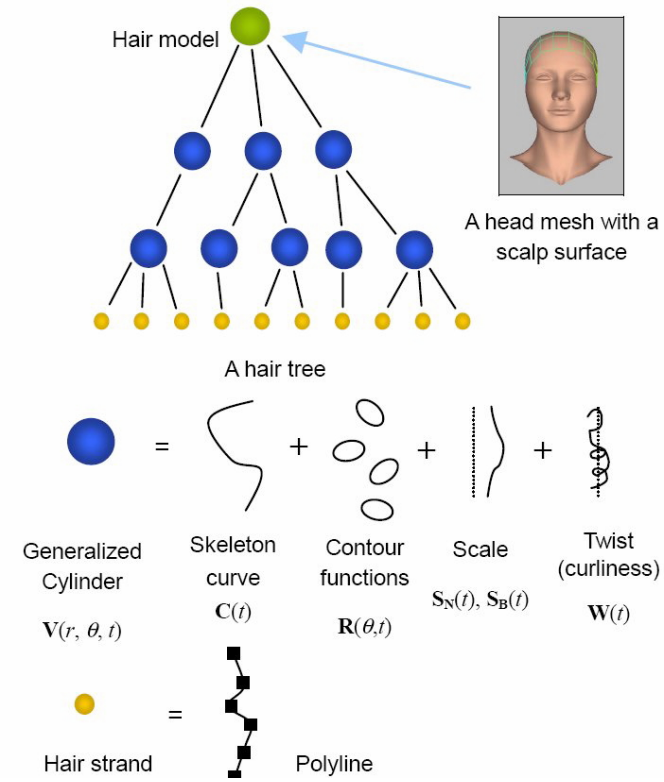


Figure 26: A graphical overview it the multiresolution model.

model hairstyles which could be approximated as curved surfaces. The combed hairs within the thin shell volume add to the realistic appearance of the hairstyle.

[Kim and Neumann 2002] describe a very powerful tool to interactively model and edit hairstyles. Figure 26 shows a graphical view of this technique, starting at a scalp surface (the patch where hair could originate). The actual hair is modeled as a set of generalized cylinders. Each generalized cylinder defines a boundary of each hair cluster, controlling hair strands or a whole group of clusters. The user interactively subdivides these hair clusters to gain more detail until the desired appearance is archieved.

A generalized cylinder is described by a skeleton curve and a set of contours placed along the curve.The scalp surface (see Figure 27 left) is a parametrized patch which represents the region where the user is able to place hairs. The scalp space (see Figure 27 middle) is spanned by u and v, and the generalized cylinders are placed on the scalp using arbitrary scalp space shapes (see Figure 27 right), which are allowed to overlap (see Figure 28).Within these contours the hair strands are generated (represented as polylines).

After the initial generalized cylinders are created, the subdivision stage begins to add detail. If a parent cluster is divided, the child clusters inherit the contours of the parent cluster. To place the child clusters on the scalp, random sampled positions are generated first (Figure 29 left) and the position relaxation algorithm by [Turk 1991] is used to improve the distribution (Figure 29 right).Now that the child clusters are positioned, the ownership of the individual hair strands need to be reassigned.

The next part describes the interactive techniques to manipulate this hair structure. The user may control the position, contours, scale and twist of any generalized cylinder, which results in different levels of detail. Since the hair is represented as a tree, if a non-

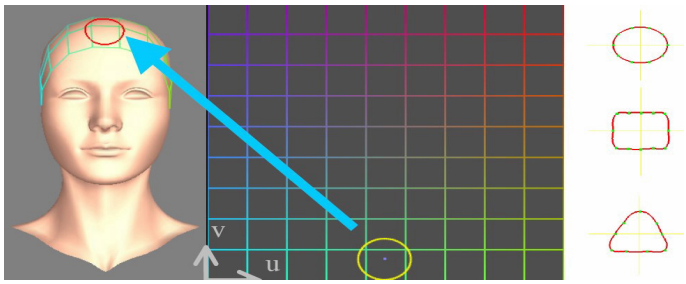


Figure 27: The placement of a shape (right) on the scalp (left) using the 2D scalp space (middle).

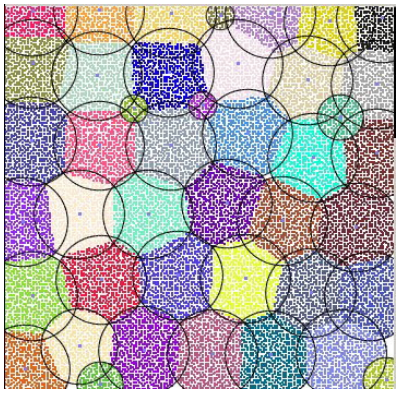


Figure 28: Overlapping shapes in the scalp space (the colours indicate which hair strands belont to which shape).

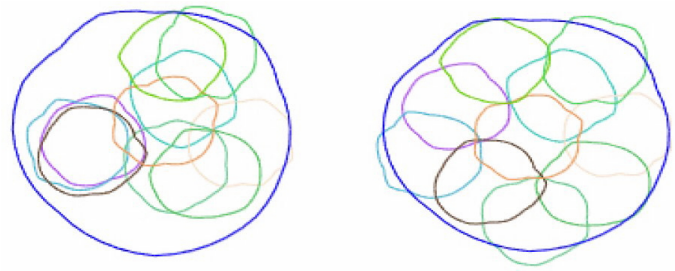


Figure 29: Subdivision of contours using random samples (left) and after the relaxation (right).

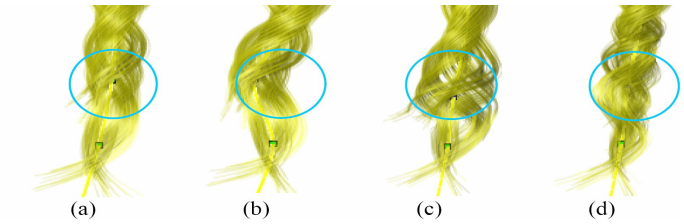


Figure 30: The user picks a control point (the black dot in the circle in picture image a), and is able to move (b), scale (c) and twist (d) the region.

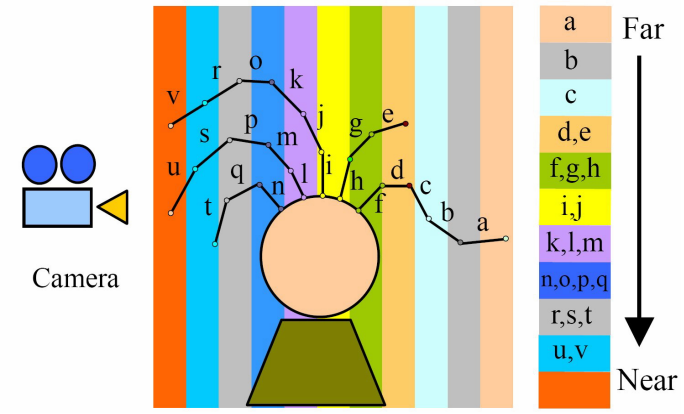


Figure 31: The sorting of the segments of the polylines for smooth drawing.

leave cluster is modified all of its decendant clusters must follow the shape-changes to preserve their relative position and orientation (Figure 30). Another operation, which the user is able to perform, is copy & paste, where any node in the hair tree could be copied to any other node regardless of the depth. Every copy operation generates a potential style template which can be used as a default in the subdivision stage.

To facilitate the edit operations there are four different ways to select parts of the hair. 1) The user is able to pick a cluster or region with a sphere. 2) Standard 2D-selection techniques are used to select clusters within the scalp space. 3) A cluster is selected and the hair-tree is used to move to other clusters. 4) Clusters regarding their depth within the hair tree could be selected.

To avoid hair-head-penetration the closest-point-transformation (CPT) algorithm described in [SEAN 2000] is used.

For the interactive rendering process the hair is rendered as polygons in OpenGL and the shading model used here is similar to the technique proposed by [Kajiya and Kay 1989]. For the self-shadowing of the hair, they used their opacity shadow maps algorithm [Kim and Neumann 2001] which is an approximation of deep shadow maps by [Lokovic and Veach 2000]. Since antialiasing is very important, due to the very thin hair strands, and the OpenGL antialiased line drawing option is not sufficient since the correct result depends on the drawing order, the authors use a visibility ordering algorithm inspired by [Levoy and Whitted 1985]. With this algorithm the bounding box of all segments is sliced by a plane parallel to the camera.This resulting bins are displayed in Figure 31. After the head mesh is drawn, the z-buffer is disabled and the segments are drawn as antialiased lines so that the ones indexed by the farthest bin are drawn first. To speed up the rendering, the authors

Figure 32: Some results generated using the interactive multiresolution modeling technique.

implemented a level of detail method which allows the user to define the number of hair strands and the number of segments per strand.

Figure 32 shows some results of this technique. The authors state that it took about 10-20 minutes to generate such a hairstyle and most of the time for modelling the hairstyle is consumed by manually placing the initial 10 -30 root clusters. They implemented their model on a Pentium3 700Mhz with 512MB RAM and a nVidia Quadro 2 Pro.

Typical time measurements for a hairstyle with 10.000 hair strands and 40 segments per strand, calculating 40 opacity maps of 200x200 pixels are: shading 1.2 seconds, visibility ordering 0.43 seconds, 5000 strands are updated per second, and shadow calculation 5.88 seconds per light source. For the interactive session usually 150.000 segments (5.000 strands with 30 segments per strand) are used,which renders with about 5 frames per second at a window size of 512x512. This tool seems to be a very powerful one, considering the manual modeling of complex hairstyles. The variety and complexity of the modeled hairstyles is very great, and the modeling time is acceptable.

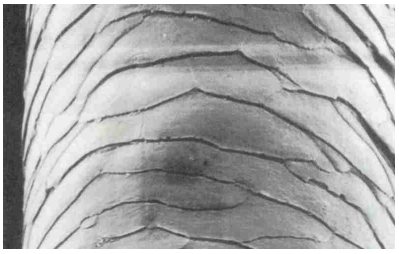


Figure 33: An electron micrograph of a hair fiber [??Robbins 1994]

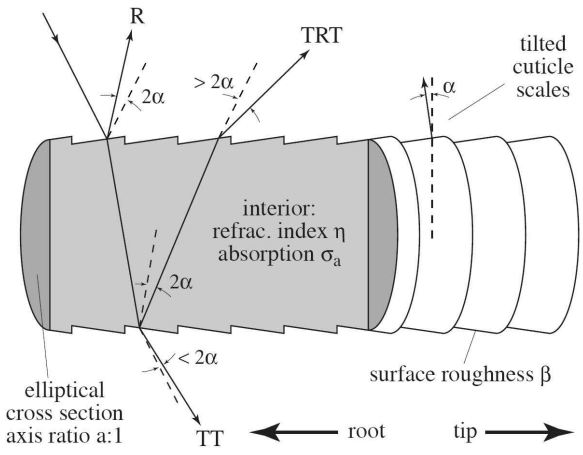


Figure 34: A model of a hair fiber.

5 Advanced Illumination of Hair

In this section, i will present some more advanced illumination techniques for hair and hair fibers. These range from very realistic lighting of hair fibers, over some global illumination techniques, image based lighting, to an illumination model where very short hair is represented without geometry (only the initial mesh is rendered) and the appearance of hair is only faked by the lighting computation.

[Marschner et al. 2003] developed a simple practical shading model based on experimental and theoretical study of the scattering of light from individual hair fibers.

Figure 33 shows a picture of a hair fiber taken from an electron microscope. This original hair fiber is approximated by an cylinder with tilted surface scales depicted in Figure 34. Figure 34 shows the light interaction with a hair fiber as well.

1. *R* denotes the shift of the primary specular peak towards the root. This effect is hypothesized to be due to the tilted scales on a hair fibers.

2. *TT* denotes the strong forward scattering property of light from coloured hair. This property is strongly observed on light coloured hair which looks much brighter when lit from behind.

3. *TRT* denotes a secondary coloured specular peak shifted towards the tip of the hair fiber from the first specular peak.

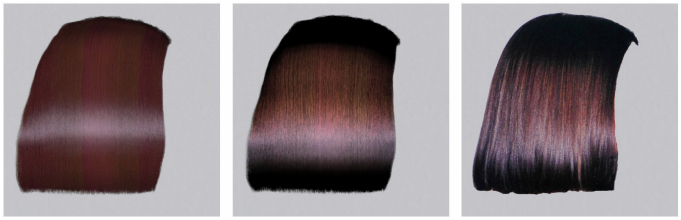


Figure 35: The result of the lighting model form [Kajiya and Kay 1989] under a single point light (left), the result of the model proposed by [Marschner et al. 2003] under the same lighting conditions (center), and a photograph (right)

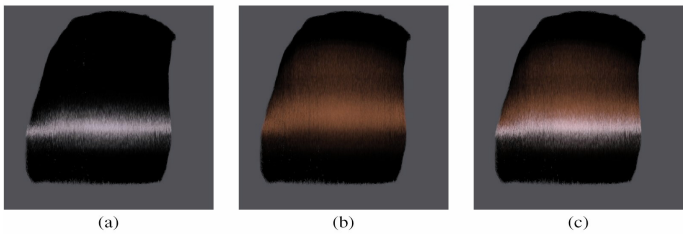


Figure 36: The components of the lighting model: (a) the R component (primary highlight), (b) the TRT component (secondary highlight), and (c) the complete model.

The experimental and theoretical study resulted in a scattering model which predicts that light from a directional source will be scattered into a perfect cone and the distribution around the cone is a sum of three distinct terms (M_R, M_{TT}, M_{TRT}) the exact discussion of the three scattering terms is found in [Marschner et al. 2003]. Figure 35 shows a comparison between a photograph, the model developed here and the model of [Kajiya and Kay 1989], where the hair fibers are treated as opaque cylinders. Switching from the simpler model of.[Kajiya and Kay 1989] to the model proposed here increased the rendering time from 6 minutes to 8 minutes on a two processor Pentium 3 1GHz system, and resulted in far better visual quality.Figure 36 shows the result of the R component, the TRT component and the results of the whole model. Figure 37 shows the result of this model under varying illumination directions, where the movement of the two specular peaks can be observed..

These results were generated using a commercial rendering software called Sasquatch (Worley Laboratories, http://www.worley.com). The hair models consisted of 50.000 to 100.000 spline curves. This illumination model captures the complexity of hair very well and is almost not to distinguish from real hair, as the results show.

[Moon and Marschner 2006] propose a technique to compute the scattering within light coloured hair using a photon map approach. Figure 38 again shows some results of the work of [Marschner et al. 2003].

Two effects can be seen which are very important for the apperance of hair.The first effect is visible in Figure 39 between the direct illuminated image and the image, which was computed using photon mapping, and therefore includes indirect illumination between hair fibers. It can be observed that if only direct illumination is considered the overall reflectance is much less than using indirect illumination, which leads to different colours.The second effect, which could be observed in Figure 40, where hair is illuminated using a sharp spot, is that there is a light glow around the edges of the spot.

Figure 37: A hair model under various lighting angles. It can be observed how the two highlights move.

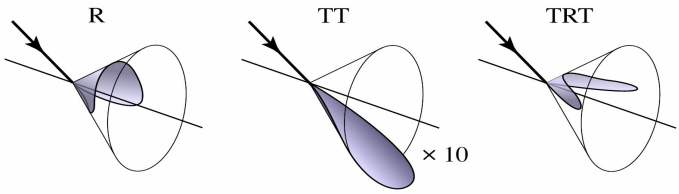


Figure 38: The three different scattering modes of hair.

Figure 39: A hair strand with direct illumination (left) and the same strand with direct and indirect illumination

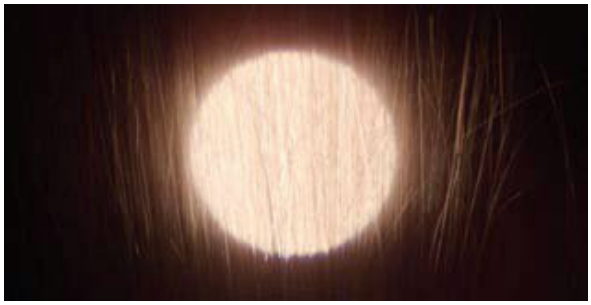


Figure 40: A photograph of hairs under a single sharp spotlight.

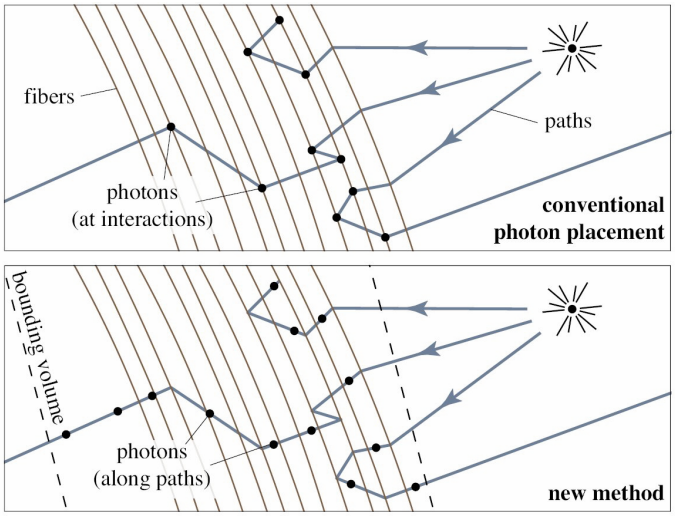


Figure 41: conventional photon placement method (top), placing method proposed by [Moon and Marschner 2006] (bottom)

Their technique, which includes both of that scattering effects, treat the hair fibers as one-dimensional fiber, neglecting the variations of thickness within a hair fibers, and only computing an average radiance per hair fiber. Since their scattering function is a five dimensional function (three dimensional point and two dimensional angular direction), this technique uses a 5D photon map to store the photons after they have been traced through the hair and followed multiple scattering events. The method to trace the photons here is similar to the method described in [Jensen and Christensen 1998] but with some differences accounting for the geometrical and optical properties of hair used with this techniques. All rays interact only with the geometry, since no continuous material is used here, and the photons are deposited with uniform probability along a ray, instead of only mapping them at interactions (a comparison between the old and the new technique can be seen in Figure 41).The last difference is that a 5D density estimate is used rather than a 3D, which handles the strong directional variations in reflectance better.

To speed up computation a radiance cache is proposed, which is efficient since viewing rays that lie close together generally lead in spatially close-by positions, and if similar scattered positions are choosen, it results in gathering almost the same photons.Figure 42 shows the results compared between direct illumination, the old photon mapping and the technique described here.

The technique described here was implemented as a single-

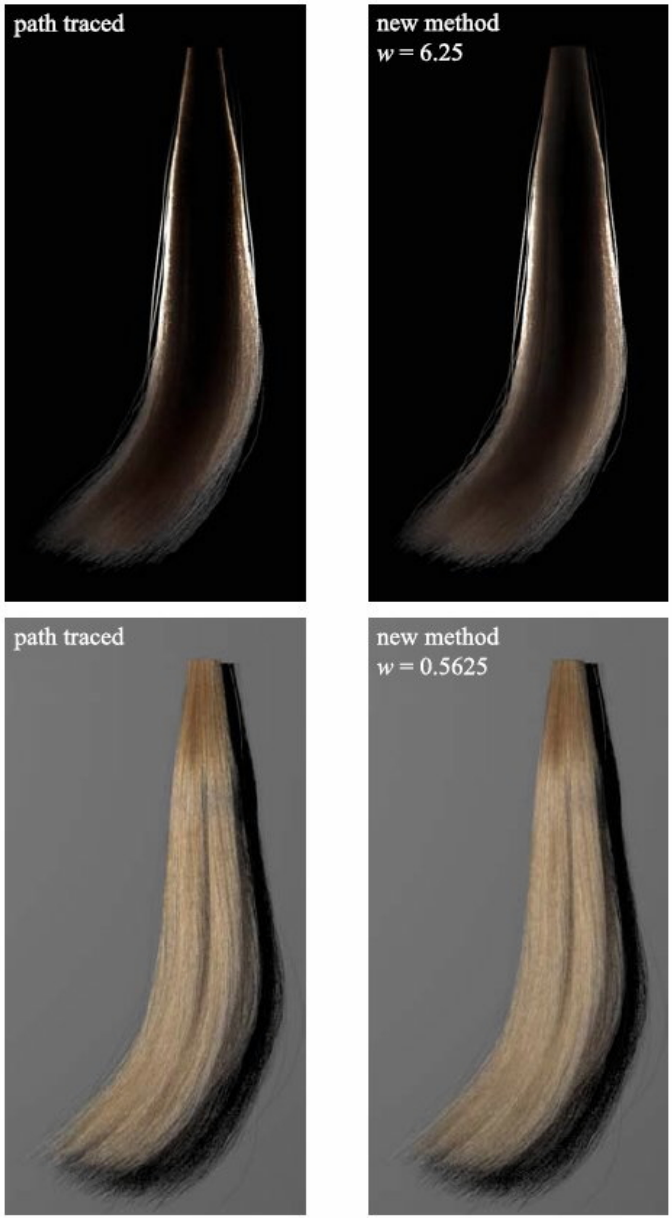


Figure 42: A comparison between conventional path traced hairs and hairs rendered with the method proposed by [Moon and Marschner 2006]

threaded java application and was used on a dual Intel Xeon 3.8GHz workstation. As seen in the results, this technique yields the visually same results as a conventional pathtracer while saving a lot of computation time (This technique reduces the conputation time from 60 to 100 hours of a conventional pathtracer down to 2.2 to 2.7 hours.)

In their sketch **[Yuksel and Akleman 2006]** propose a global illumination model for hair. Most of the hair rendering techniques only consider direct illumination since hair geometry is highly complex and that complexity results in scenes where a single hairy object consumes much more computation time than the rest of the scene, using radiosity or raytracing techniques. If only direct lighting is used, inconsistencies in the overall appearance could be observed. The authors propose some simplification techniques to reduce computation time while still generating good results.

$$L_r(\phi) = \int_S F(\theta,\phi)\gamma(x)L_i(x,\theta)d\theta \tag{7}$$

Equation 7 describes the reflected radiance L_r from a point in direction ϕ. Where F is the geometry term, γ is the visibility of x which are the points of the incoming radiance L_i and θ is the radiance form x. Since hair inhibits strong forward scattering the authors state, that they can safely ignore inter-hair-reflection of hair strands and therefore rewrite the equation as follows:

$$L_r(\phi) = \int_S F(\theta,\phi)\Gamma(\theta)\Lambda(\theta)d\theta \tag{8}$$

Where Γ is the visibility, and Λ is the incoming radiance in the direction θ. In this equation, hair strands do not contribute to Λ and only affect Γ, and therefore Λ is computed using standard global illumination techniques, and Γ is calculated by projecting the hairs onto the unit sphere. The authors suggest two more simplification techniques based on the level of detail. The first technique takes only one random subset of the hair geometry to represent all hairs. The second simplification eliminates subsets of selected hairs according to their distance from the computation point. Therefore hairs that contribute less (which are farther away) are computed with less accuracy while hairs with strong contribution are computed without any accuracy loss. Figure 43 shows the results with direct lighting (top) compared to the results with global illumination (bottom).It can be seen that the appearance of the directly illuminated hair is much darker and that computing global illumination for hair adds more realism. The rendering technique was based on a photon map with final gathering approach proposed by [Jensen 1996] and the shading model from [Kajiya and Kay 1989] was used .

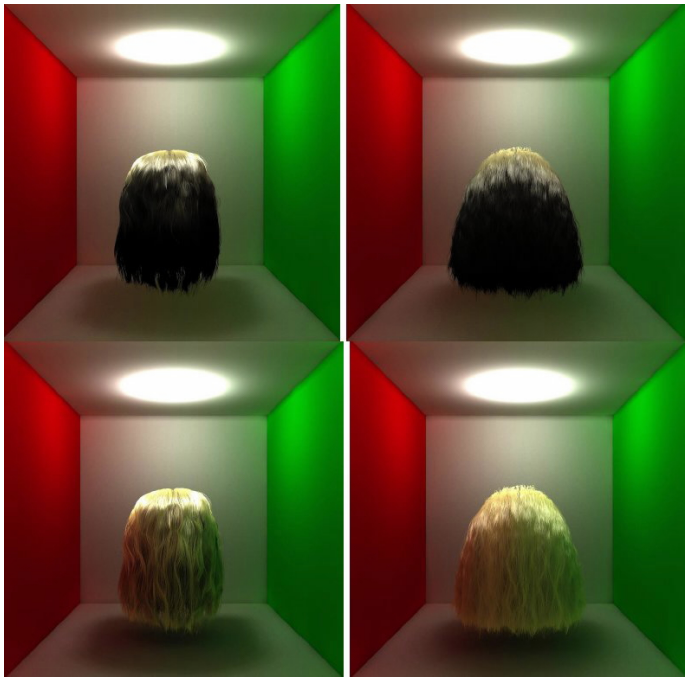


Figure 43: hair rendered with direct illumination only (top), and rendered with indirect illumination (bottom)

[Neulander 2004] proposes a fast approximative solution for image-based lighting of curved hair. The hair is modeled by a curve using a set of control points, where each control point includes world position and another four real values used in their occlusion model. The author states that other attributes like colour, hair thickness, etc. are useful, but not relevant for this model.

The lighting model used here is based on [Kajiya and Kay 1989]. Since image-based lighting has to consider numerous light directions, the diffuse and specular models used here are summations of the Phong diffuse and specular model over multiple shading normals, distributed orthogonally to the curve tangent. For each diffuse and specular shading sample, multiple look-ups into the image based lighting texture, with an occlusion appoximation per look-up, are performed. The image-based lighting texture is prefiltered by a pair of cosine-filter kernels to reduce the diffusion samples to a plane and the specular samples to a cone (see Figure 44). This

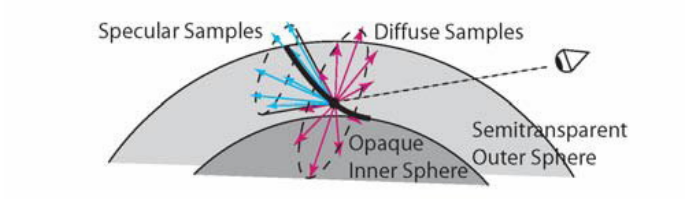


Figure 44: A schematical view of the specular and diffuse samples.

Figure 45: Two results where the furry torus is lit by a HDR image.

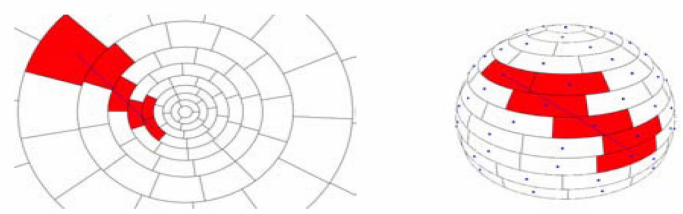


Figure 46: Line and face intersections on an igloo.

vastly reduces the rendering complexity with an acceptable loss in accuracy.

For the occlusion model used here, an extension of the model proposed by [Neulander and van de Panne 1998], two additional parameters are stored at each control point. The unit occlusion vector N_0 and an occlusion height scalar h_0. For an incident light vector L the fractional visibility is given by the following expression, where ρ is some density constant:

$$e^{-\rho(-h_0 N_0 L + \sqrt{1-h_0^2(1-N_0 L^2)})} \tag{9}$$

The extension of the occlusion model described here uses an opaque inner sphere centered at the same point as the outer sphere (see Figure 44). This is useful for modeling shorter hair which can be abruptly shadowed by skin.

The author states that this technique is well suited for GPU implementation, and that its main limitation is the assumption of a locally spherical, homogeneous medium, which precludes accurate shadowing. Some result of this technique is shown in Figure 45, where the usability of this technique for furry objects can be observed. Furthermore it can be observed that for furry objects, this technique yields nice image-lit results.

[Yuksel and Akleman 2005] propose a projection based framework for indirect illumination of hair-like structures including self-shadowing. This framework is based on a visibility calculation suitable for line segments. This visibility calculationis based on a hemisphere projection method. This projection method uses a structure that resembles an 'igloo' (see Figure 46). Using this igloo, the visibility problem reduces to line-circle intersection on a infinite plane (Fee figure 46). The framework also can be implemented using other approximations of hemispheres like hemicubes, etc.

The authors state that the images in Figure 47 are rendered in ten minutes on a Pentium 4 3Ghz CPU and no tricks, like different colours for each hair strand (to fake selfshadowing), where used. Furthermore they mention that no artifacts where produced in animated sequences.

[Goldman 1997] proposes a probabilistic lighting model for thin coats of fur. This method is much faster than hair-by-hair or volumetric methods. This 'fakefur' algorithm is used to render mam-

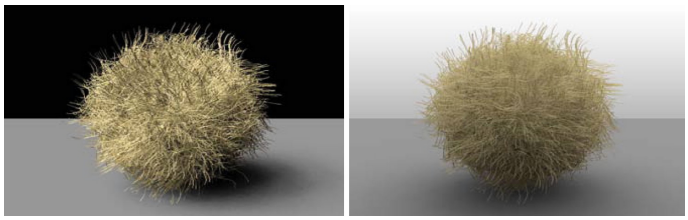


Figure 47: Left, rendered with an aera light, right rendered with an ambient occlusion.

malian fur creatures to be composed into live-action feature films (like '101 Dalmatians' by Disney). The outline of this illumination algorithm is given as follows:

1. Compute the mean hair geometry within the sample region. This is the "reference hair".
2. For each light:
   (a) Using the fakefur opacity function (described later), compute the hair over hair shadow attenuation.
   (b) Compute the reflected luminance of the average hair in the sample region.
   (c) Using the fakefur opacity function to compute the hair over skin shadow factor.
   (d) Compute the reflected luminance of the underlying skin.
   (e) Using the fakefur opacity function to compute the hair/skin visibility factor.
   (f) Blend the reflected luminance of the skin and hair using the visibility ratio to optain the final reflected luminance of the sample region.
3. Sum the reflected luminance for each light to obtain the total reflected luminance for the sample region.

The fur geometry on a surface is parametrized by hair length, hair radius, hair density, hair tangents at root and tip. The reflectivity of individual hairs is described by diffuse reflectivity (wavelength dependant), specular reflectivity, specular exponent, several directionality factors for transmissivity or reflectivity control and Lambertian macro behaviour. This parameters may vary over the creatures surface, either described procedurally or by texture maps.

For the reflected luminance of a single hair fiber a modified approach from the algorithm of [Kajiya and Kay 1989] is used. Since this model lacks the directionality it had to be modified. This modification uses two new attenuation factors, the relative transmissivity (ρ_{trans}) and the reflectivity factor ($\rho_{reflect}$). These two factors are used for forward (ρ_{trans}) and backward scattering ($\rho_{reflect}$), and are combined in their directional attenuation factor (f_{dir}) which is computed as follows:

$$f_{dir} = \frac{1+\kappa}{2}\rho_{reflect} + \frac{1-\kappa}{2}\rho_{trans} \tag{10}$$

κ characterizes the relative directionality of a given incident light ray L, eye ray E, and hair tangent T, and is defined as follows :

$$\kappa = \cos(\bar{T}\times\bar{L}, \bar{T}\times\bar{E}) = \frac{(\bar{T}\times\bar{L})(\bar{T}\times\bar{E})}{|\bar{T}\times\bar{L}||\bar{T}\times\bar{E}|} \tag{11}$$

Therefore κ is greater than zero for frontlighting and less than zero for backlighting.

Another factor has to be added to compute smooth shadows within the surface, the surface normal factor:

$$f_{surface} = 1 + \rho_{surface}(smoothstep(\bar{N} \cdot \bar{L}, \omega_{\min}, \omega_{\max}) - 1) \quad (12)$$

$\bar{N}$ is the normalized surface normal and *smoothstep* is the smooth Hermite interpolation between $\omega_{\min}$ and $\omega_{\max}$ defined as:

$$smoothstep(x,a,b) = \left\{ \begin{array}{c} 0, \forall x < a \\ 1, \forall x > b \\ -2(\frac{x-a}{b-a})^3 + 3(\frac{x-a}{b-a})^2, \forall a < x < b \end{array} \right\} \quad (13)$$

f_{dir} and $f_{surface}$ are multiplied into the equation $\psi_{hair}$of [Kajiya and Kay 1989] (see Equation 5):

$$\psi_{hair} = f_{dir} f_{surface}(\psi_d + \psi_s) \quad (14)$$

Like in [Kajiya and Kay 1989] this model is a first order approximation, which is most accurate when the hair albedo is low, and no secondary scattering of light off the hairs onto other hairs or skin is considered.

The fakefur opacity function computes the mean opacity for a fur patch and is described by the hair geometry, the hair distribution and the viewing angle. Considering the hair geometry, hairs are viewed as truncated cones of radius r_b at the base, radius r_t at the tip and length l where the values are considered as follows:

$$\begin{array}{rcl} l & \gg & r_b \\ r_b & \geq & r_t \end{array} \quad (15)$$

For the distribution of hairs two assumptions are made:

1. All hairs in the sample region share the identical direction and geometry.
2. The distribution of hairs in a small region has Poisson characteristics: Within a zone of uniform density, a sample of half the size will contain half the hairs and the hairs are placed independantly of each other.

Therefore the fakefur opacity function α_f is given by:

$$\alpha_f = 1 - \left(1 - \frac{DA_h g(\bar{E}, \bar{T}, \bar{N})}{n_i}\right)^{n_i} \quad (16)$$

Where n_i is a constant denoting the number of hairs within a sample region and D is the local density of hairs. A_h is the computed aera of projection onto the viewing plane as:

$$A_h = \frac{l(r_b + r_t)}{2} \quad (17)$$

The probability of a random ray striking a single hair α_h from E is given by :

$$\alpha_h = \frac{A_h}{A_s} g(\bar{E}, \bar{T}, \bar{N}) \quad (18)$$

where the projection dependent part of α_h is g defined as:

$$g(\bar{E}, \bar{T}, \bar{N}) = \frac{\sin(\bar{E}, \bar{T})}{\bar{E} \cdot \bar{N}} \quad (19)$$

This opacity function is used in the illumination process for three different computations: hair-over-skin shadows, hair-over-hair shadows, and hair-over-skin visibility. The other large scale geometry steps (skin-on-hair shadows, skin-on-skin shadows, skin illumination, etc.) are not included in this paper since the authors state that these are well handled by existing techniques.

Figure 48: A image of the film 101 Dalmatians by Disney.

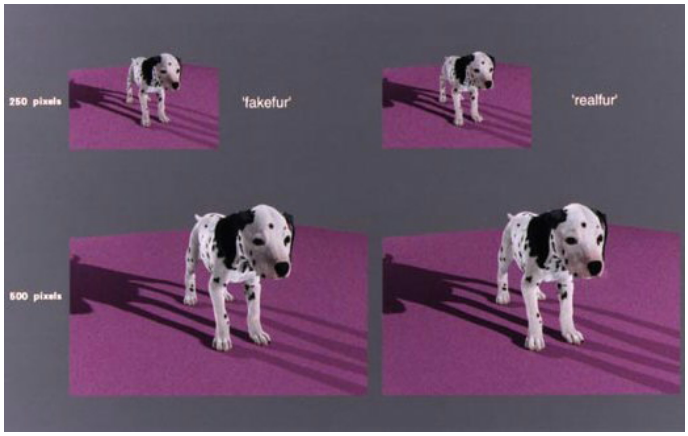


Figure 49: A comparison between the fakefur algorithm and the realfur algorithm.

There are two cases in which the result of the fakefur algorithm is not accurate: case one is the 'hot spot' where the viewing vector $\bar{E}$ is close to the illumination vector $\bar{L}$. Case two is the 'halo' where the viewing and illumination vector are nearly opposite. The author states that these illumination enviroments are generally avoided by cinematographers and since this model was designed for live-action feature films, it has no impact on the visual results of the fakefur algorithm.

Figure 48 is taken from the film '101 Dalmatians' by Disney, while Figure 49 is a comparison between the fakefur and a 'realfur' (implemented for evaluation, where fur is geometrical represented) algorithm. As the figures show, the visual performance of the fakefur algorithm is very well in comparison to the realfur algorithm. Considering the geometrical complexity of a realfur algorithm, the fakefur algorithm is a very good alternative for very short hair where lots of computation power can be saved, as long as the two situations where this technique does not work are avoided..

6 Hair Rendering for Movies

In this section i want to present two sketches considering hair rendering for movie productions. The first sketch illustrates the problems when choosing the representation of hair for the rendering,

Figure 50: Scrat, the hero of Ice Age: the Melt Down.

where the benefits of the decision could lead to other not considered problems later in the production process. The second sketch describes a very powerful method to groom realistic looking hair, used in Peter Jacksons King Kong.

[van Swaaij 2006] illustrates some problems and work-arounds when rendering fur for a film production like 'Ice Age: The Melt Down'.The tragic-comic hero, Scrat (see Figure 50) is discussed as an example here. When his hair is modeled as B-spline curves (1.3 million) with six control points where each control point has five values (three coordinates,colour and opacity), and adding support for deformation, blur, bounding boxes and subdivision of the B-splines into Bezier segments, Scrats fur would require more than 600 MB memory. Because CGI Studio is a ray-tracer, the geometry has to be in memory while rendering. While this huge memory requirements are a big problem there are more drawbacks with this approach. Because the hairs have a tiny radius, ray-tracing this geometry, which is much smaller than a pixel, would result in severe aliasing if no tremendous amount of subsampling is done. The worst of all is that CGI Studio is based on axis-aligned bounding box hierarchies so that the intertwined nature of hair would lead to very long rendering times.

To overcome these problems the Blue Sky Studios decided to use voxels, which are similar to the texels used in [Kajiya and Kay 1989], to represent the geometry. The use of voxel proved to be a big advantage considering memory usage since the actual geometry is contained within the voxels and therefore the number of hairs does not determine the memory allocation. Another advantage is that the rasterization is antialised by a filter function allowing the voxels to be rendered without additional anti-aliasing. The third advantage is that it proved to be much more efficient marching through a regular voxel grid than ray-tracing highly overlapping bounding boxes.

This improvement resulted in much faster rendering, if preprocessing is neglected. Since one frame is often rendered multiple times during lighting design this preprocessing is amortized over several renderings by caching the voxels on disk. Nevertheless, new problems arose using voxels,which are the actual generation of the voxels, so when they are rendered they look the same as if rendered the actual geometry, and motion blur was very problematic for the voxel system, because it used alot of memory and several memory reduction strategies had to be employed.

They looked for an alternative technique to reduce memory usage, but in the end they found out, that using the voxel system turned out to be a real memory hog because of the motion blur.

Figure 51: Using more deformers from left to right.

[Preston and Hill 2006] explain how their proprietary fur grooming and rendering program, called Bonobo by Weta Digital, for Peter Jacksons 'King Kong' works. The program allows both, styling and animating, using a form of visual programing.

The grooming part consists of a dependancy graph of deformers, which will modify each hair on the subdivision surface, which is initially grown standing straight out from the geometry. A lot of different deformers where used to model the appearance of fur (strands, clumping, collision, etc.), and even the dynamics were realized using deformers, generated by a soft-body simulation or a curve-based dynamic simulation, which are applied at the end of the grooming program. After the deformers are set, the grooming program reaches the renderer (PRman) for fur growth. About four million curve strands were created on King Kongs body. Besides this fur, the same groom program was applied to additional geometry within the fur to create mud, leaves, dirt, etc. The Renderman shader used here was based on the work of [Marschner et al. 2003], adopted to the characteristics of Yak-hair. They incorporated nonperfect cuticles in the individual hairs, light absorption through the volume and the effect of inter-hair light transport. Some results of the layered deformers are shown in Figure 51.

7 Conclusion

As we have seen the high complexity of hair, considering the geometry of the very thin hair fibers, and the illumination properties of hair, most of the techniques described here do not work for all sorts of hairs and hairstyles. Volumetric representations have the advantage that the real geometry is approximated and therefore a lot of memory could be saved. This may be true for the representation, but as we have seen in the sketch about Ice-Age this memory reduction does not work for the motion blur they used, and therefore this advantage had also its pitfall included. Nevertheless, volumetric representations are very well suited to generate short hair and fur where individual hairs do not affect the overall appearacne of

that hairstyle. The other representation method as splines or polylines has much more memory consumption, and due to the very thin hair fibers, extensive subpixel sampling or other antialiasing methods have to be used to avoid artifacts. Although this representation has many drawbacks, it is the only representation to capture complex hairstyles, where individual hair fibers or strands have a huge impact on the visual appearance. The method to approximate hairstyles with thin shell volumes is only suitable for a few hairstyles, but this hairstyles could be captured very well and compared to the polyline/spline representation a lot of computation time could be saved. The real hair aquisition techniques, have, as we have seen, many drawbacks, but research in this direction is quite young, and maybe more usable techniques will be developed in the future. Nevertheless, this techniques could be incorporated into other interactive modeling tools as a starting point to faciliate the modeling process.

If someone has the task to render realistic looking hair, a lot of considerations have to be made (what hairstyles should be rendered, how much memory and computation time is available,etc.). Taking this considerations as a starting point, some techniques can be chosen to achieve this goal. Perhaps more than one technique have to be put together to achieve this goal, until today there is no general technique to generate every kind of hairstyle.

References

BROOKS, M. J., AND HORN, B. K. P. 1989. Shape and source from shading. 53–68.

GOLDMAN, D. B. 1997. Fake fur rendering. In *SIGGRAPH '97: Proceedings of the 24th annual conference on Computer graphics and interactive techniques*, ACM Press/Addison-Wesley Publishing Co., New York, NY, USA, 127–134.

GRABLI, S., SILLION, F., MARSCHNER, S. R., AND LENGYEL, J. E. 2002. Image-based hair capture by inverse lighting. In *Proc. Graphics Interface*, 51–58.

ICHI ANJYO, K., USAMI, Y., AND KURIHARA, T. 1992. A simple method for extracting the natural beauty of hair. In *SIGGRAPH '92: Proceedings of the 19th annual conference on Computer graphics and interactive techniques*, ACM Press, New York, NY, USA, 111–120.

JENSEN, H. W., AND CHRISTENSEN, P. H. 1998. Efficient simulation of light transport in scences with participating media using photon maps. In *SIGGRAPH '98: Proceedings of the 25th annual conference on Computer graphics and interactive techniques*, ACM Press, New York, NY, USA, 311–320.

JENSEN, H. W. 1996. Global illumination using photon maps. In *Proceedings of the eurographics workshop on Rendering techniques '96*, Springer-Verlag, London, UK, 21–30.

KAJIYA, J. T., AND KAY, T. L. 1989. Rendering fur with three dimensional textures. In *SIGGRAPH '89: Proceedings of the 16th annual conference on Computer graphics and interactive techniques*, ACM Press, New York, NY, USA, 271–280.

KIM, T.-Y., AND NEUMANN, U. 2000. A thin shell volume for modeling human hair. *ca 00*, 104.

KIM, T.-Y., AND NEUMANN, U. 2001. Opacity shadow maps. In *Proceedings of the 12th Eurographics Workshop on Rendering Techniques*, Springer-Verlag, London, UK, 177–182.

KIM, T.-Y., AND NEUMANN, U. 2002. Interactive multiresolution hair modeling and editing. In *SIGGRAPH '02: Proceedings of the 29th annual conference on Computer graphics and interactive techniques*, ACM Press, New York, NY, USA, 620–629.

LENGYEL, J., PRAUN, E., FINKELSTEIN, A., AND HOPPE, H. 2001. Real-time fur over arbitrary surfaces. In *SI3D '01: Proceedings of the 2001 symposium on Interactive 3D graphics*, ACM Press, New York, NY, USA, 227–232.

LENGYEL, J. E. 2000. Real-time hair. In *Proceedings of the Eurographics Workshop on Rendering Techniques 2000*, Springer-Verlag, London, UK, 243–256.

LEVOY, M., AND WHITTED, T. 1985. The use of points as a display primitive. Tech. rep., Computer Science Department, University of North Carolina at Chapel Hill.

LOKOVIC, T., AND VEACH, E. 2000. Deep shadow maps. In *SIGGRAPH '00: Proceedings of the 27th annual conference on Computer graphics and interactive techniques*, ACM Press/Addison-Wesley Publishing Co., New York, NY, USA, 385–392.

MARSCHNER, S. R., JENSEN, H. W., CAMMARANO, M., WORLEY, S., AND HANRAHAN, P. 2003. Light scattering from human hair fibers. In *SIGGRAPH '03: ACM SIGGRAPH 2003 Papers*, ACM Press, New York, NY, USA, 780–791.

MOON, J. T., AND MARSCHNER, S. R. 2006. Simulating multiple scattering in hair using a photon mapping approach. In *SIGGRAPH '06: ACM SIGGRAPH 2006 Papers*, ACM Press, New York, NY, USA, 1067–1074.

NEULANDER, I., AND VAN DE PANNE, M. 1998. Rendering generalized cylinders with paintstrokes. In *Graphics Interface*, 233–242.

NEULANDER, I. 2004. Quick image-based lighting of hair. In *SIGGRAPH '04: ACM SIGGRAPH 2004 Sketches*, ACM Press, New York, NY, USA, 43.

PARIS, S., BRICEñO, H. M., AND SILLION, F. X. 2004. Capture of hair geometry from multiple images. In *SIGGRAPH '04: ACM SIGGRAPH 2004 Papers*, ACM Press, New York, NY, USA, 712–719.

PRAUN, E., FINKELSTEIN, A., AND HOPPE, H. 2000. Lapped textures. In *SIGGRAPH '00: Proceedings of the 27th annual conference on Computer graphics and interactive techniques*, ACM Press/Addison-Wesley Publishing Co., New York, NY, USA, 465–470.

PRESTON, M., AND HILL, M. 2006. Grooming, animating & rendering fur for "king kong". In *SIGGRAPH '06: ACM SIGGRAPH 2006 Sketches*, ACM Press, New York, NY, USA, 43.

SEAN, M., 2000. A fast algorithm for computing the closest point and distance function.

TURK, G. 1991. Generating textures on arbitrary surfaces using reaction-diffusion. In *SIGGRAPH '91: Proceedings of the 18th annual conference on Computer graphics and interactive techniques*, ACM Press, New York, NY, USA, 289–298.

VAN SWAAIJ, M. 2006. Ray-tracing fur for ice age: the melt down. In *SIGGRAPH '06: ACM SIGGRAPH 2006 Sketches*, ACM Press, New York, NY, USA, 44.

WEI, Y., OFEK, E., QUAN, L., AND SHUM, H.-Y. 2005. Modeling hair from multiple views. In *SIGGRAPH '05: ACM SIGGRAPH 2005 Papers*, ACM Press, New York, NY, USA, 816–820.

YANG, R., POLLEFEYS, M., AND WELCH, G. 2003. Dealing with textureless regions and specular highlights-a progressive space carving scheme using a novel photo-consistency measure. In *ICCV '03: Proceedings of the Ninth IEEE International Conference on Computer Vision*, IEEE Computer Society, Washington, DC, USA, 576.

YANG, G., SUN, H., WANG, W., AND WU, E. 2006. Interactive fur modeling based on hierarchical texture layers. In *VRCIA '06: Proceedings of the 2006 ACM international conference on Virtual reality continuum and its applications*, ACM Press, New York, NY, USA, 343–346.

YUKSEL, C., AND AKLEMAN, E. 2005. Rendering hair-like objects with indirect illumination. In *SIGGRAPH '05: ACM SIGGRAPH 2005 Sketches*, ACM Press, New York, NY, USA, 21.

YUKSEL, C., AND AKLEMAN, E. 2006. Rendering hair with global illumination. In *SIGGRAPH '06: ACM SIGGRAPH 2006 Research posters*, ACM Press, New York, NY, USA, 124.