

A Thin Shell Volume for Modeling Human Hair

Tae-Yong Kim and Ulrich Neumann

Computer Graphics and Immersive Technology Laboratory
Integrated Media Systems Center, University of Southern California
{taeyongluneumann}@graphics.usc.edu

Abstract

Hair-to-hair interaction is often ignored in human hair modeling, due to its computational and algorithmic complexity. In this paper, we present our experimental approach to simulate the complex behavior of long human hair, taking into account the hair-to-hair interactions. For long hair, we propose the thin shell volume (TSV) model for enhancing hair realism by simulating complex hair-hair interaction. The TSV is a thin bounding volume that encloses a given hair surface. The TSV method enables virtual hair combing for surface-model hairstyles. Combing produces the effects of hair-hair interaction that occur in real human hair. Any hair model based on a surface representation can benefit from this approach. The TSV is presented mainly as a tool to modeling human hair, but this model also gives rise to global hair animation control.

1. Introduction

Computer generated ‘virtual humans’ are widely used in many areas – for example, as virtual agents, avatars in virtual worlds, characters in games, or even in film production. Although methods for creating very convincing face images and animations exist, it remains difficult to obtain the realistic appearance and behavior of human hair. As a consequence, hair is usually simplified or hidden with hats. However, hair is a very important cue for recognizing people. To faithfully model the natural appearance of a specific person, hair has to be accurately modeled, rendered, and animated.

The simplest way to generate the appearance of hair is with textured polygons. This approach is widely used in real-time applications such as computer games and avatar agents. However, the complex shading and dynamics of human hair are not reproduced by this technique. We seek methods for modeling and synthesizing realistic hair appearance and behavior.

Volumetric textures extend a 3D volume of hair over the surface [4]. Volume textures can produce very realistic images of short hair, however the computational costs are high, and hair strand animation is difficult. The fake fur rendering technique approximates the probabilistic light reflectance patterns of shorthair furs at reasonably far viewing distances [6]. The lack of any underlying hair geometry model limits rendering and animation to fur or short hairs that do not move much. Long hairs are difficult to represent with *texture-based* models since they fail to

represent the appearance, interactions, and motions of individual hairs or groups of hair strands.

The majority of human hair modeling methods can be classified as *geometry-based*. These usually involve a large number of hair-strand primitives such as cylinders, surfaces, lines, and curves. With control over the appearance and motions of each primitive, these models are more suited for long hair and animation. The trigonal prism was proposed for representing each hair strand [1]. Anjo et al. applied cantilever beam physics to simulate an individual hair’s behavior [3]. Most geometry-based motion models use second-order motion equations (e.g., [5]). To create realistic images, pixel-blending buffers address the antialiased-rendering of small-scale hair strands [2].

Most geometry-based methods model and render each hair strand independently. In reality, hairs are not perceived independently nor do they behave independently. Neighboring hairs tend to attract or repel each other. The resulting hair pattern provides an important ‘signature’ of hair appearance. These behaviors are very hard to model, even for limited cases, thus *hair-hair interaction* (HHI) has been largely ignored in prior works.

This paper presents an efficient method for HHI modeling for a set of long hairstyles. Modeling HHI effects in all possible long hairstyles is difficult and beyond our scope. We limit our consideration to hair and hairstyles that are approximated by surfaces. Our goal is to add more realistic appearance and behavior to this common set of human hairstyles. Virtually, any hair model based on surfaces such as NURBS or polygons can benefit from our method.

The paper is organized as follows. In the next section, we provide an overview of the proposed thin shell volume method. Chapter 3 describes a virtual hair combing procedure based on the TSV model. Chapter 4 describes the rendering method used for our experiments. Chapter 5 shows the use of the TSV model for animation. Chapter 6 shows experimental results. In conclusion, we discuss the limitations and possible extensions of our current approach.

2. Thin Shell Volume (TSV) Model

At large scales, long hair tends to form smooth surfaces that can be approximated by polygons or parametric surfaces such as NURBS. At smaller scales, each hair strand is essentially a long, thin cylinder that can be represented by a

curve. To correctly model long hair, it is necessary to take both scales into account. The TSV model builds on the simplicity of the surface representation while adding the fine appearance details of long hair. A thin shell volume is constructed on each surface. Inside the volume, hairs are associated with particles that move under constraints that model the hair-hair interaction during combing. Individual hair strands are represented by a set of particle positions inside the volume.

Below we list the procedures to perform a virtual hair combing using the thin shell volume model (Figure1).

- First, we model a hairstyle with a set of surfaces by using existing surface modeling methods. We require each surface to be parameterized. (a)
- We add *thickness* to each surface by offsetting the surface along its normal direction, generating a thin curved volume. (b)
- The curved volume is warped to a rectilinear 3D grid. Inside the grid, a number of hair strands are evenly distributed. Each cell of the grid is associated with a list of particles that reside inside it. (c)
- We perform a virtual hair combing function on the normalized volume. The hair combing function redistributes the hair particles, creating variations based on the HHI model, producing a set of *combed* hair particles. (d)
- Finally, we recover each hair strand by connecting the corresponding hair particles. These *stylized* hairs are a

group of curves where each curve represents one hair strand. The curves are passed to the rendering pipeline. (f)

2.1 Notation and Terminology

We define some notations and terminology for use throughout this paper. As shown in Figure 1(c), a TSV is parameterized by three variables (s, t, u). The \vec{t} vector is aligned with the major direction of the hair flow. The \vec{u} vector is aligned with the *depth* direction of the hair volume. The \vec{s} vector is orthogonal to both \vec{u} and \vec{t} . For simplicity, s, t, u are normalized to range from 0.0 to 1.0. Along the \vec{u} (depth) direction, we assign the innermost hairs the value of $u = 0$, and the outermost hairs (surface hairs) the value of $u = 1$.

Let (i, j, k) denote the integer grid index corresponding to a given (s, t, u) . ($0 < i \leq L, 0 < j \leq M, 0 < k \leq N$, where L, M, N represents the grid resolution of the volume.) $V_{i,j,k}$ identifies the $(i,j,k)^{\text{th}}$ volumetric cell position. Particles p are distributed among the cells (Figure 2) representing each strand passing through a cell. Each cell maintains a list of particles residing in it. The density of particles defines the hair opacity ρ later used for shadowing (Section 4.2). For a given particle, the cell containing it is determined by quantizing its s, t, u position values.

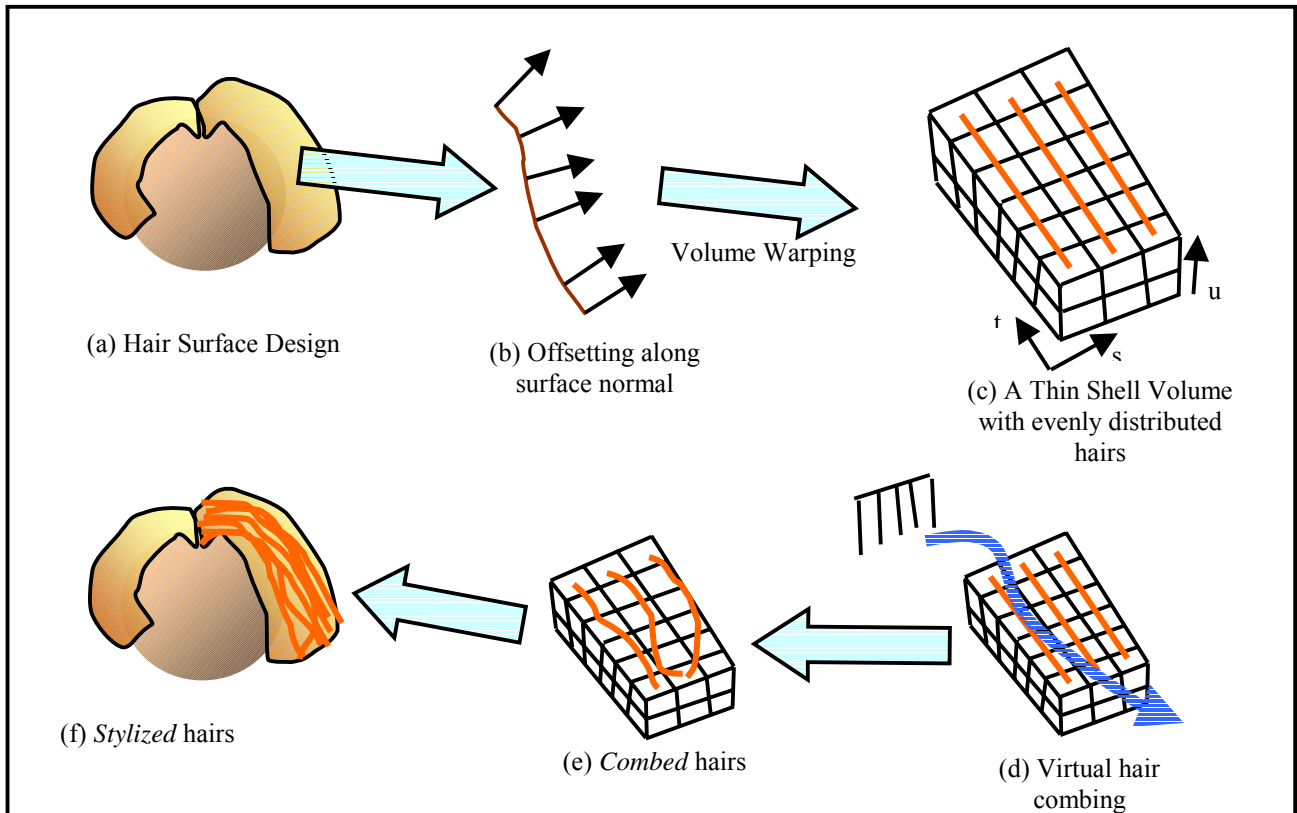


Figure 1. An overview of hair combing using a thin shell volume model.

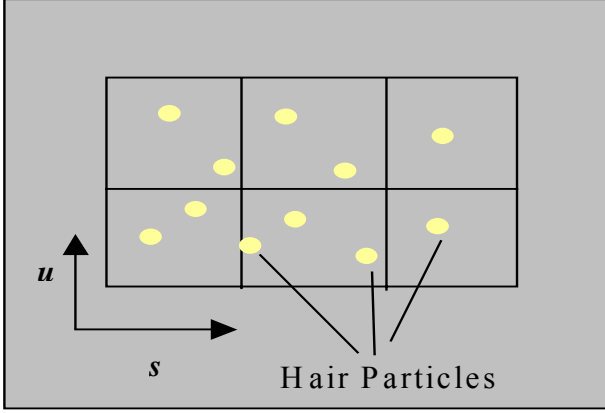


Figure 2. Cross-section of a TSV.

2.2 TSV Generation and Volume Warping

The initial TSV shape is determined from the hair surface (or reference surface). We start from a reference surface (e.g., NURBS) parameterized as $\mathbf{P} = \mathbf{S}(s, t)$.¹ The TSV is constructed by offsetting the surface along the normal (depth) direction, forming a thin curved volume (Figure 3).

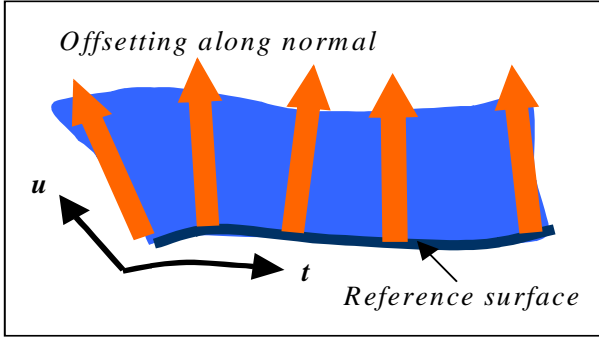


Figure 3. Generation of a thin shell volume

The reference surface is generally curved so the normal varies over the surface. The parameterization of a point \mathbf{p} inside the thin shell volume is (s_p, t_p, u_p) . The normal $\mathbf{n}(\mathbf{p})$ can be defined by taking the cross product of the partial derivatives of the reference surface \mathbf{S} along \vec{s} and \vec{t} .

$$\mathbf{n}(\mathbf{p}) = \mathbf{n}(s_p, t_p) = \frac{\delta \mathbf{S}(s_p, t_p)}{\delta s} \times \frac{\delta \mathbf{S}(s_p, t_p)}{\delta t}$$

¹ We chose the Coons Patch as our reference surface representation. (Refer to the Appendix A.) Other surface representations are equally applicable as long as they can be represented in the form of $\mathbf{P} = \mathbf{S}(s, t)$

² Note that high curvatures and folded surfaces can cause self-intersections, but many useful reference surfaces are relatively smooth and free of these problems.

The world coordinate of \mathbf{p} is then reconstructed by

$$\mathbf{p} = \mathbf{S}(s_p, t_p) + T \cdot \mathbf{u}_p \cdot \mathbf{n}(s_p, t_p)$$

, where T is a scalar defining the thickness of the volume.

It is more convenient to compute the hair-hair interaction inside a regular rectilinear volume rather than in a curved volume, so we use an implicit volume warp. Any point \mathbf{p} in the TSV with (s_p, t_p, u_p) can directly map to a corresponding point in a regular rectilinear volume we call a *normalized TSV*. The normalized volume has an orthonormal basis defined by the $\vec{s}, \vec{t}, \vec{u}$ vectors and greatly simplifies the description and calculation of our simulation for hair-hair interaction. In the remainder of the paper, we assume that TSV is normalized. (The distortions caused by this simplification are not visually significant since they only affect the hair-hair interaction behavior.)

3. Virtual Hair Combing

In this section we consider the thin shell volume as a modeling method for distributing long hairs. The distribution of long hairs is usually very simplified for mathematical simplicity. Often simple random variations such as translations and rotations are used. These random variations do not, however, model hair-hair interaction and result in unrealistic hair distributions and appearances. In our model, virtual hair combing provides a way of generating the hair patterns that often occur in real-world hair styling.

We create a number of hair strands by placing hair particles inside the volume. Initially, we generate n uniformly distributed straight hairs. For a TSV with a resolution of $L_x M \times N$, each hair is associated with M particles that lie along its path. There are roughly $n/(L_x N)$ particles in each cell. Each hair lies along the \vec{t} direction. Each hair strand starts where t is close to zero, and ends where t is close to 1.

We redistribute hair particles inside the TSV by simulating the effects of combing. Each hair that lies under the path of the comb moves under influence of the comb teeth. We generate various hair patterns by changing tooth shape, tooth distribution and the path of the comb. The cross-section of a comb tooth in the su plane is a trapezoidal radial influence function (Fig. 4).

Our implementation models each tooth *path* as a parametric curve on the st plane over the TSV. Each tooth of the comb has an influence function such that:

1. Only the particles hit by the tooth are affected
2. Each particle moves away from the center of the tooth's influence
3. Each affected particle tries to follow the path of the comb

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

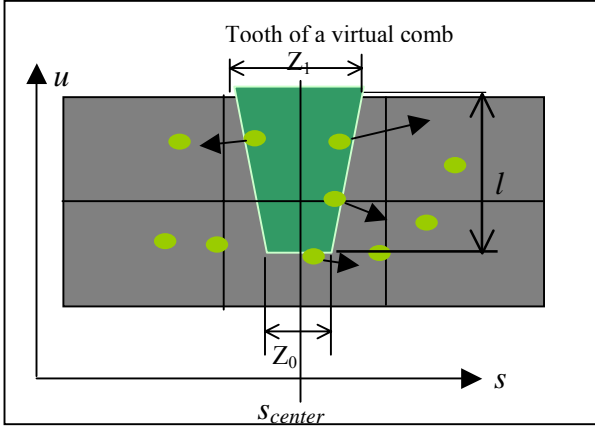


Figure 4. Virtual hair combing

3.1 Particle Selection (*Pickup*)

We test if each particle is affected by a comb tooth. We model the path of the tooth's center as a parametric curve C on the st plane, where s_{center} is the s -coordinate of the tooth trapezoid center.

$$s_{center} = C(t)$$

We define an influence zone $Z(u)$ along the u (depth) direction. $Z(u)$ determines the vertical shape of the tooth. Our implementation models it as a trapezoid.

$$Z(u) = \begin{cases} \frac{1}{2} \cdot \{(1 - u') \cdot Z_0 + u' \cdot Z_1\}, & \text{if } u' \geq 0 \\ \varepsilon, & \text{otherwise} \end{cases}$$

$$u' = 1.0 + \frac{u - 1.0}{l}$$

ε denotes a very small value to avoid a division by zero in a later equation. l denotes the depth of the tooth. Using the distance from the center of the trapezoid, we define the zone influence function $Z(s,u)$ as follows.

$$Z(s,u) = \text{sign}(R(s)) \cdot \cos\left(\frac{\pi}{2} \cdot \text{clamp}\left(\frac{R(s)}{Z(u)}\right)\right)$$

$$R(s) = s - s_{center}$$

$$\text{clamp}(x) = \begin{cases} x, & \text{if } |x| \leq 1 \\ 0, & \text{otherwise} \end{cases}, \quad \text{sign}(x) = \begin{cases} 1, & \text{if } x \geq 0 \\ -1, & \text{if } x < 0 \end{cases}$$

$Z(s,u)$ is used as a probability function to check if a particle is inside the tooth's influence zone. Given a particle's position (s,t,u) , we first select the nearest trapezoid based on t , then we calculate $Z(s,u)$. For each particle, we make a probabilistic decision whether or not

the tooth picks up the particle. A random value is compared to the absolute value of $Z(s,u)$ to determine the binary decision outcome.

The sign of $Z(s,u)$ determines the direction of the particle motion. A positive value moves the particle in the increasing- s direction and a negative influence value moves it in the decreasing- s direction.

3.2 Particle Redistribution (s -direction)

Once a particle is picked up, we reposition the particle randomly (in s) within the range of the tooth's influence zone. Due to the pickup function $Z(s,u)$, the particles near the center of the influence zone are more likely to be repositioned. The particle distribution gets sparser at the center of the tooth's influence zone since hairs are pushed aside by the comb tooth. Figure 5 illustrates the particle density change due to combing.

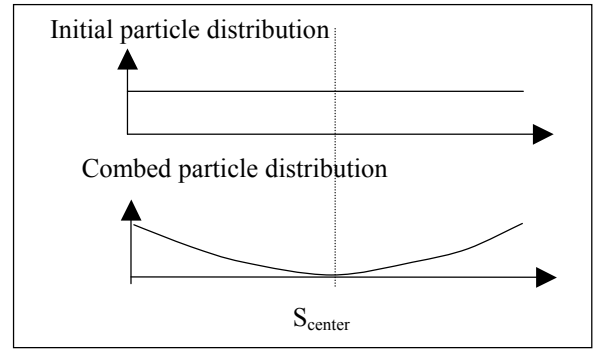


Figure 5. Particle redistribution in s direction

3.3 Sweeping Effect

Random redistribution of the particles representing a hair strand may generate unwanted curliness. A hair strand should be bent accordingly to the path of the comb. To generate the *sweeping effect*, we use a simple scheme (Figure 6). For each hair strand, we find the first particle p_j influenced by the comb. Let the particle's parameterization be (s_j, t_j, u_j) . We calculate the new s -coordinate s'_j for this particle. We calculate the distance $r = |s'_j - s_{center}|$. For the following particles p_l ($l > j$), we add the distance to the path of the comb. That is, for a particle p_l parameterized by (s_l, t_l, u_l) , the new s -coordinate $s'_l = C(t_l) + r$.

3.4 Vertical Motion (u -direction)

In a real combing, the combed hairs tend to be stacked on top of the remaining hairs. We simulate this behavior by moving combed particles upward (in u -direction). Simply moving all the particles upward may generate a crowding problem after repeated iterations. Some (not combed) particles should be pushed down too. To simulate these effects, we swap the combed particle with a particle from the upper cell. When there is no upper cell or there is no

particle in the upper cell, we move the particle upward with small amount (usually the size of cell in the u -direction). To ensure that the particles lie inside the volume, any out-of-bound particle is reflected back to the inside of the volume.

The virtual combing on a TSV can be performed repeatedly, varying the tooth distribution, tooth shape (in our case, Z_0 , Z_1 and l), and the path of the comb (refer to the Chapter 6 for the results).

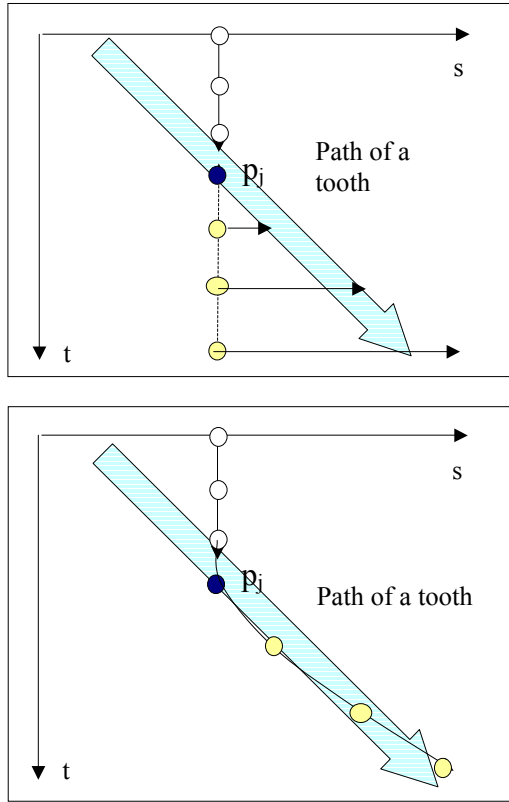


Figure 6. Bending of a hair strand

4. Rendering TSV

4.1 Hair reconstruction

We reconstruct each hair strand by connecting its corresponding particles. A straight line can be used to connect particles, but lines can generate sharp corners that seldom occur in real hair. We use a Lagrange Interpolation Curve [11] for the particle connections.

4.2 Probabilistic Shadow Generation

While there are known methods for hair shadow generation [2,7,8], we developed a simple shadow generation algorithm based on the TSV model and similar to the one used in [8]. For shadow generation, we assume that lighting is diffuse and omnidirectional. We model the inner hairs as more likely to be in shadow than the outer hairs. We propagate the probability of shadow starting from the

outer cells. Let $\Phi_{i,j,k}$ represent the amount of shadow in cell $V_{i,j,k}$,

$$\Phi_{i,j,k} = \begin{cases} 0.0 & , \text{if } k = L \text{ (outermost cell)} \\ \Phi_{i,j,k+1} + Th \cdot \rho(V_{i,j,k+1}) & , \text{otherwise} \end{cases}$$

where Th is a constant to control the optical thickness of each hair.³ The opacity function $\rho(V_{i,j,k})$ is proportional to the number of the particles in the cell.

We vary the shadow values for each particle inside $V_{i,j,k}$ by adding a small noise to $\Phi_{i,j,k}$. Note that these shadow effects only change when combing or animation changes the particle distributions. A change in lighting or viewpoint does not affect these values.

4.3 Shading and Antialiasing

For each hair strand's local shading, we use the widely used local illumination model developed by Kajiyama et al. [4]. Each hair strand is broken into a series of connected fine lines usually with an image length of 5-10 pixels. We calculate the result of the local shading for each node in hair strand. The shading value is multiplied by the shadow value. Each line segment is drawn using the standard Z-buffer algorithm.

Since hair is a very thin object when viewed under normal viewing conditions, one should be careful to reduce the artifacts from sampled line drawing. We employ supersampling by the accumulation buffer method [12]. We draw the hair image N times, each time shifting the image plane slightly. Each image is scaled by $1/N$ and added to the accumulation buffer. To avoid repeated evaluation of curves, we first prepare a set of lines with associated colors based on the shadow and shading model. At the accumulation step, we only draw these lines repeatedly. Line drawing is usually very fast if hardware acceleration is used.

5. Animation

Now we consider the TSV model as an animation tool for global control of hair motion. A global animation for each thin shell volume can be generated with simply deforming the reference surface. Furthermore, it is possible to add a simple dynamics to the reference surface. Collision detection and dynamics of smooth patches have been well studied [10, 14] and may be applied to control reference surface motions.

One can think of the hairs inside the TSV as constrained hairs due to our hair-hair interaction. One can also add free hairs that are unconstrained. The animation of these additional hairs can be done using the standard dynamics techniques for each strand's motion. The free hairs can be rendered separately on local illumination model and blended. The TSV model does not prevent the creation of

³ For example, thicker hairs will generate more shadow (occlude more light) than thinner hairs.

additional hair motions with current existing methods such as [2, 3, 5, 8].

6. Implementation and Results

In our system, the user first roughly designs the desired hairstyle with a set of surfaces. We use the Coons Patch as the representation for the reference surface (see appendix A). Plate 1 shows a hairstyle designed with 4 bilinear Coons Patches.

Plate 2 shows the actual combing function applied to a patch. (a) shows the reference surface. (b) shows a virtual comb with 20 teeth. The path of each tooth is modeled as a Lagrange Interpolation Curve. The paths are drawn as yellow curves. (c) shows 4000 initial hair strands for the TSV model. (d) shows 4000 hair strands after combing.

As an animation test, we prepared two sets of hairstyles and performed linear interpolation (morphing) between them. Plate 3 shows a set of frames taken from the animation. The complete animations in Plate 3 can be seen in <http://cerignola.usc.edu/tsv.htm>

Plate 4 shows the effect of changing the comb parameters (7000 hairs were used). In our implementation, each path of the tooth is specified as a curve in the *st* plane. Currently, the combing functions are created and processed as batch operations. In plate 4, the number of teeth increases from bottom to top. In the top row, 40 teeth were used. In the middle row, 20 teeth were used. In the bottom row, 12 teeth were used. The width of each tooth was scaled accordingly so that neighboring teeth do not interfere. The depth of each tooth was set to roughly half the thickness of the volume. Small random variations were added to the path of each tooth.

Our implementation runs on an SGI Onyx R10000 (only a single CPU and graphics pipe is utilized). It takes about 20~30 seconds to comb and generate the model shown in Plate2 (4000 hairs). Rendering is fast (1 – 2 seconds per frame). All the images shown are rendered at 512 x 512 pixel resolution. We used the resolution of 20x20x10 for all the TSVs in our tests. A large fraction of the computing time is consumed in the reconstruction of the hair strand from the TSV particles due to the numerous evaluations of the Coons Patch. For local lighting of the test scenes, two lights were used for the diffuse and specular shading.

7. Conclusion

In this paper, we presented a new method to simulate the complex behaviors of human hairs. Virtual hair combing is used to generate realistic looking hair distributions. Although the general case of hair-hair interaction is still unsolved, our method creates some complex effects efficiently for a set of target hairstyles with long, surface-like hair. Although our method maintains the control advantages of a surface representation while providing some fine hair details, it remains to be seen if the TSV model can be generalized to include more complex cases such as curly hairs and short spiky hairs.

8. References

- [1] Y. Watanabe and Y. Suenega, *A trigonal prism-based method for hair image generation*, IEEE Computer Graphics and Application, Vol. 12, (1), pp. 47-53, January 1992.
- [2] A. LeBlanc, R. Turner and D. Thalmann, *Rendering hair using pixel blending and shadow buffers*, The Journal of Visualization and Computer Animation, Vol. 2, pp. 92-97, 1991.
- [3] K. Anjyo, Y. Usami and T. Kurihara, *A simple method for extracting the natural beauty of hair*, SIGGRAPH Proceedings, pp. 111-120, 1992.
- [4] J. Kajiya and T. Kay, *Rendering fur with three dimensional textures*, SIGGRAPH Proceedings, pp. 271-280, 1989.
- [5] R. Rosenblum, W. Carlson and E. Tripp, *Simulating the structure and dynamics of human hair: Modeling, rendering and animation*, The Journal of Visualization and Computer Animation, Vol. 2, (4), pp. 141-148, October-December 1991.
- [6] D. Goldman, *Fake fur rendering*, SIGGRAPH Proceedings, pp. 127-134, 1997
- [7] W. Reeves, D. Salesin and R. Cook, *Rendering antialiased shadows with depth maps*, SIGGRAPH Proceedings, pp. 283-291, 1987.
- [8] W. Kong and M. Nakajima, *Visible volume buffer for efficient hair expression and shadow generation*, IEEE Computer Animation, pp. 58 – 65, 1999
- [9] S. A. Coons, *Surfaces for computer aided design of space forms*, MIT Project TR-41, MIT, Cambridge, MA, 1967
- [10] P. Volino, M. Courchesne, and N. Thalmann, *Versatile and efficient techniques for simulating cloth and other deformable object*, SIGGRAPH Proceedings, pp. 137-144, 1995
- [11] A. Rockwood, P. Chambers, *Interactive Curves and Surface*, Morgan Kaufmann Publishers Inc., San Francisco, 1996
- [12] *Programming with OpenGL: Advanced Techniques*, SIGGRAPH Course Note, pp. 60-61, 1997
- [13] N. Thalmann, S. Carion, M. Courchesne, P. Volino, and Y. Wu, *Virtual Clothes, Hair and Skin for Beautiful Top Models*, Computer Graphics International, pp. 132-141, 1996
- [14] D. Baraff and A. Witkin, *Large Steps in Cloth Simulation*, SIGGRAPH Proceedings, pp. 43-54, 1998

Appendix

A Coons Patch

A Coons Patch is defined by 4 boundary curves. These curves can be any parametric curve as long as their endpoints are connected to form a closed boundary.

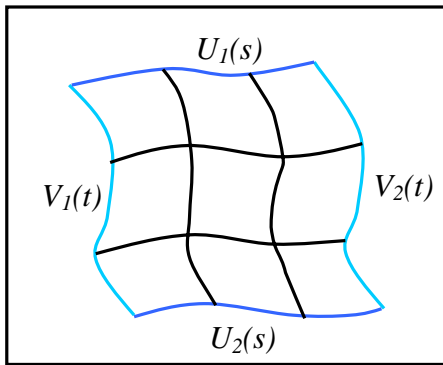
Let us say the 4 boundary curves are: $U_1(s)$, $U_2(s)$, $V_1(t)$, $V_2(t)$. Any point $P = S(s,t)$ is calculated as follows

$$P = S(s,t) = (1-t)U_1(s) + tU_2(s) + (1-s)L(t) + sR(t)$$

, where

$$L(t) = V_1(t) - (1-t)V_1(0) - tV_1(1)$$

$$R(t) = V_2(t) - (1-t)V_2(0) - tV_2(1)$$



A Coons Patch

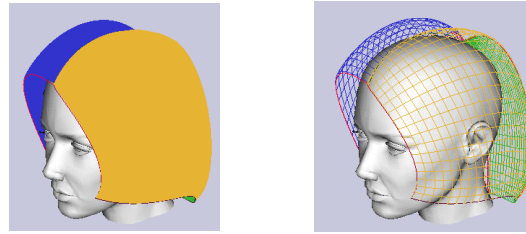
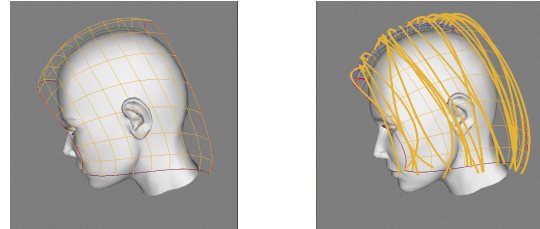
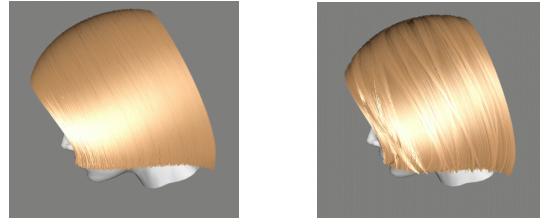


Plate 1. A hair design with 4 patches.
(Each patch has a different color)



(a) Hair surface design

(b) Combing function



(c) Hair model before combing

(d) Combed Hair

Plate 2. Result of hair combing



