Efficient and Realistic Hair Modeling and Animation Using Sparse Guide Hairs

Yizhou Yu Johnny T. Chang Jingyi Jin
Department of Computer Science, University of Illinois at Urbana-Champaign
1304 West Springfield Avenue, Urbana, IL 61801
E-mail: {yyz,jtchang,jingjin}@cs.uiuc.edu

Abstract

This paper presents a unified framework for modeling and animating curly hairs. Both modeling and animation start from a sparse set of guide hairs. An initial static sparse hair model is extracted from a superimposed vector field by tracing field lines. It can be effectively interpolated to produce a dense hair model. Random natural or artificial curliness can then be added to the dense model through a parametric hair offset function with a randomized distribution of parameters over the scalp.

In addition to challenges in modeling, hair also exhibits strong anisotropic dynamic properties which demand distinct animation techniques for single strands and hair-hair interactions. While a single strand can be modeled as a multibody open chain expressed in generalized coordinates, modeling hair-hair interactions is a more difficult problem. A dynamic model for this purpose is proposed based on a sparse set of guide strands. Long range connections among the strands are modeled as breakable static links formulated as nonreversible positional springs. Dynamic hair-to-hair collision is solved with the help of auxiliary triangle strips among nearby strands. Adaptive guide strands can be generated and removed on the fly to dynamically control the accuracy of a simulation.

Fine imagery of the final dense model is rendered by considering both primary scattering and self-shadowing inside the hair volume which is modeled as being partially translucent.

Index Terms: Hair Modeling, Guide Hairs, Hair Interpolation, Offset Functions, Hair Animation, Hair-Hair Interaction, Static Links, Collision Detection, Open Chain, Hair Rendering

1 Introduction

Hair is a crucial element of appearance. One of the many challenges in simulating believable virtual humans and animals has been to produce realistic looking hair. Creating realistic hair presents problems in all aspects of computer graphics technologies, i.e. shape modeling, dynamics and rendering [14, 29, 19, 36, 1, 6, 10, 38, 16, 11, 21, 12, 9, 20, 18, 28, 40, 4]. Hair rendering and shape modeling of fur like short hair is becoming increasingly available to animators. However, shape modeling and dynamics of long hair has been difficult. The difficulties arise from the number of hair strands, their geometric intricacies and associated complex physical interactions such as collisions, shadowing and static charges. These interactions contribute to both static and dynamic appearances of hair.

The most important thing in a hairstyle lies in the way hair strands curve and deform. The causes for hair deformation can be summarized as natural curliness, artificial hairstyling processes and deformation under external forces such as gravity, collision and static charges. According to [13], the degree of natural curliness is indicated by the curvature of a hair strand when it is not under any external forces. In an actual hairstyling process, many artificial procedures are provided by a hairdresser, such as perming, combing, shearing and the application of cosmetics. The effects of perming and cosmetics are most influential to cause long-term artificial hair deformation. They overcome natural curliness and make hairs hard to model. Lastly, hairs bend under gravity and collision. These factors should be considered in hair dynamics. Because of artificial hairstyling and external forces, hair strands do not grow and curve completely randomly in all directions, but follow certain global as well as local flow patterns. For example, long hair is usually draped down, and nearby strands usually form a cluster and deform in the same way. However, there is still small amount of randomness that distinguishes strands from each other in the way they curve.

In addition to challenges in modeling, hair also has highly anisotropic dynamic properties, i.e. hair strands are extremely hard to stretch but free to move laterally and interact with each other irregularly. Strands cannot penetrate each other when they intersect; yet, each strand does not have a fixed set of neighboring strands. These unique properties inform us that a custom designed dynamic model is necessary to achieve realistic results. The dynamics of long hair involve three aspects. First, an individual hair strand can deform and interact with the scalp, cloth and other objects. Second, an initial hairstyle can usually be recovered after subsequent head movement and the application of external force fields. This means a hairstyle can memorize its original configuration. Slight movement does not erase this memory. However, radical movement may permanently damage this memory and no complete recovery is possible. Third, there are dynamic collisions among different strands. A real person can have more than 50,000 hairs. Each hair can be modeled as dozens of hair segments. Directly detecting pairwise collisions among hair segments is neither necessary nor computationally practical. Therefore, we should model hair collisions at a higher abstraction level.

In this paper, we present a unified framework for hair modeling and animation. Both modeling and animation start from a sparse set of guide hairs. The proposed modeling approach can generate realistic wavy hairstyles by editing a basic sheared hair model with a generic offset function for curliness. The hair dynamics model has the following features: i) an initial hair connection model that allows hairstyle recovery after minor movement, ii) a hair mutual collision model that considers the hair volume as a collection of continuous strips, iii) an adaptive hair generation scheme to complement our sparse hair model. Since our framework is based on sparse guide hairs, designing hairstyles and solving physical interactions among hairs are computationally efficient without losing much of the quality from a method based on a dense model.

1.1 Related Work

The work we present in this paper has been made possible by previous work on hairs and other related topics. We limit the overview to the previous work on hair modeling and dynamics, focusing on explicit hair models. In these models, each hair strand is considered for shape and dynamics. They are more realistic and especially suitable for long hair.

There are a few methods for hair modeling. Due to effects of adhesive forces, hairs tend to form clumps. Watanabe introduced the wisp model in [36]. Yan *et al* [38] modeled the wisps as generalized cylinders. The wisp model is also used in [5]. Daldegan *et al* proposed to define a few characteristic hair strands in 3D and then populate the hairstyle based on them. Hadap and Magnenat-Thalmann [11] model hairs as streamlines of an ideal fluid flow. The user can set up a few flow elements around an object to design a hairstyle. Other researchers also tried to model and constrain hair using a single thin shell or multiple head hull layers [16, 20].

There has also been a number of proposed methods for hair animation. Rosenblum *et al* [29] and Daldegan *et al* [6] used a mass-spring-hinge model to control the position and orientation of hair strands. Anjyo *et al* [1] modeled hair with a simplified cantilever beam and used one-dimensional projective differential equation of angular momentum to animate hair strand. None of these previous attempts considered hair-hair interactions and hairstyle recovery after minor movement. Recently, Hadap and Magnenat-Thalmann [12] proposed a novel approach to model dense dynamic hair as continuum by using a fluid model for lateral hair movement. Hair-hair collision is approximated by the pressure term in fluid mechanics while friction is approximated by viscosity. This work presented an elegant and promising model for hair interactions using a dense set of strands. Plante *et al* proposed a wisps model for simulating interactions inside long hair [28]. Hair strands are clustered into wisps consisting of a skeleton and a deformable envelope. Collision forces among wisps follow an anisotropic formulation. This model is suitable for hairstyles with independently clustered wisps. Koh and Huang presented an approach by explicitly modeling hair as a set of 2D strips [18]. Collisions between hair strips are handled to create more realistic motion. Nonetheless, the volumetric aspect of the hair is not captured.

Recently, there has been a few feature films, such as Final Fantasy and Monsters Incorporated, with realistic hair simulations as well as some commercial software packages, such as Shave[33] and Shag[32]. Shave is considered as the best commercial hair modeling and simulation software in the industry. However, its single strand dynamics does not look realistic, and it does not have hair-hair collision. Final Fantasy is the film with the best simulations for long human hair. From the press releases, Aki's hair was modeled as a whole deformable exterior surface and some of the simulations were done using the Maya cloth plugin. That means hairs are constrained around the surface to enable very good hairstyle recovery, but much of the lateral freedom has been lost. In some situations, the hair flows like a piece of cloth instead of a set of individual stiff strands. On the other hand, Monsters Incorporated has long fur simulation [9]. Each hair is considered as particles linked in a chain by a set of stiff springs. A builder or a small snippet of code is used to generate the inbetween hairs. Hair-hair collision has not been considered.

1.2 Overview

Let us first clarify the types of hair models used in our system. We use both sparse and dense hair models. A sparse hair model has from dozens to hundreds of guide hairs. A dense hair model has around 50,000 strands which is close to the number of hairs human beings have. A hair model can also be static or dynamic. A static model does not change its geometric configuration over time. Hairs in a dynamic model change their positions and velocities over time when forces are present. Thus, we have four combinations of hair models as shown in Fig. 1(a).

Given a geometric model of a synthetic head, our system first generate a static sparse hair model H_{ss} using interactively defined vector fields and splines. A H_{ss} can be interpolated to generate a static dense hair model H_{sd} which can be further edited to have curliness. The dense curly hair model thus obtained can be rendered to produce final synthetic images (Fig. 1(b)).

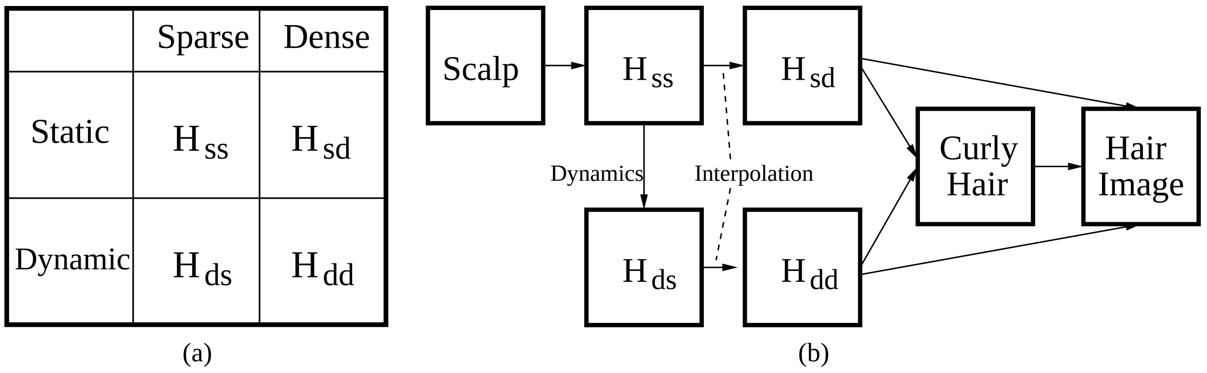


Figure 1: (a) Four types of hair models used in this paper; (b) a diagram of our hair modeling and animation system with various data flow paths.

A H_{ss} can also go through a different path in the system to produce hair animations. It can be the input to our simulation subsystem. Each guide hair from the H_{ss} is initially represented as a polyline with multiple vertices. The H_{ss} is then equipped with structural elements needed for dynamic simulation. For example, each vertex is considered as a rotational joint with a hinge. Connections and triangular meshes among guide hairs are then built for simulating hair-hair interactions. Such an enhanced model is then ready for dynamic simulation. Note that these enhanced structures are ”invisible”, which means they are never visualized during hair rendering although the effects they produce are incorporated into hair motion. Once an animation sequence of the sparse model is generated, we obtain a dynamic sparse hair model H_{ds} at each frame. A H_{ds} can then be interpolated to produce a dynamic dense model H_{dd} which can be rendered to produce a synthetic image corresponding to a specific frame in an animation sequence. Curliness can certainly be added before rendering as in the static case.

In the rendering stage, we consider both diffuse and specular reflection as well as partial translucency of each strand by integrating volume density rendering with a modified version of the opacity shadow buffer algorithm [17].

Fig. 1(b) shows the various paths in our system. Note that for curly hair, we have two levels of details. The sparse or interpolated hair model only has large-scale deformations without fine curly details. Each strand in these models serves as the spine of its corresponding curly strand. Curliness is added onto the interpolated dense hair model before rendering.

The organization of the rest of the paper is as follows. In the next section, we describe various aspects of hair modeling. Section 3 presents techniques for hair animation. Hair rendering is briefly discussed in Section 4. Section 5 presents our results and comparisons. And Section 6 provides conclusions.

2 Hair Modeling

Previous approaches either model individual hairs explicitly [1, 36, 29, 6, 5] or model all hairs collectively as a volume with a density at each point [14, 27]. We take the explicit approach here, considering the enormous variations of hairstyles. The input to our method is a polygonal model of a head including the scalp and face. Our overall virtual hairstyling process has the following multiple steps:

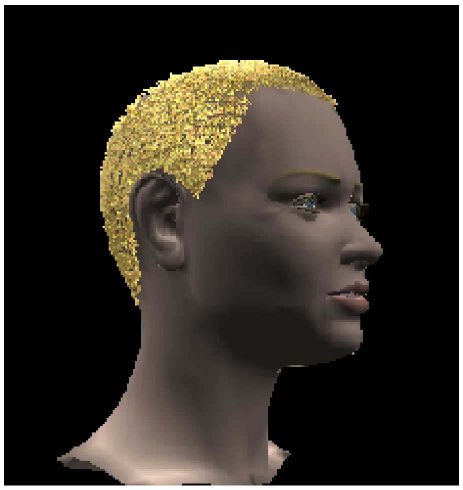
1. select a region on the scalp for hair growth, and specify a length distribution for hairs growing from the region (shearing);
2. model the polylines for a sparse set of guide hairs using a set of interactively defined vector fields, transform the polylines for the guide hairs into spline curves, and further adjust their deformations using their control points;
3. generate a dense hair model from the guide hairs through interpolation;
4. edit each strand in the dense model using an offset function to produce random curliness;
5. pull together nearby hairs to form clusters;

2.1 Shearing

We first need to define the region on the scalp where hair should be grown (Fig. 2(a)). The contour of this region is generated by linearly interpolating the 2D polar coordinates of a few points, which have been interactively selected in a flattened map of the scalp. The center of this map aligns with the top center of the scalp (Fig. 3(c)). Once specified, the same region can be repeatedly used for multiple hairstyles.

Shearing is the starting point of hairdressing. Every hairstyle needs a corresponding hair length distribution. The length of a particular strand is determined by the location of its root on the scalp. We use the vector between the root of the strand and the centroid of the scalp to define this location in a 3D polar coordinate system. Hairdressers sometimes cut hair with respect to some reference planes[39, 31]. For example, they would cut lower part of the hair to a horizontal plane at the bottom of the neck, and pull upper part of the hair straight up and cut it to a horizontal plane on top (Fig. 2(b)). At other times, the hair would be cut to have a smoothly varying length[39, 31].

According to these two shearing styles. We designed two representations for hair length distributions. The first one is based on a BSP tree (Fig. 2(b)). The plane defined for an intermediate node of the BSP divides a region on the scalp into two smaller ones. The plane at a leaf node of the BSP actually defines a reference plane for hair-cutting. All hairs rooted in the region corresponding to that leaf node should be cut to the reference plane given at the same leaf. This is done by setting the length of a strand to be the length of the shortest path outside the scalp between its root and that reference plane. So part of the shortest path may be curved and lying on the scalp. We actually use a sphere to approximate the shape of the scalp to accelerate this calculation. Based on the second shearing style, the second representation is a linear interpolation model based on the length specified at a few key locations on the scalp.


(a)

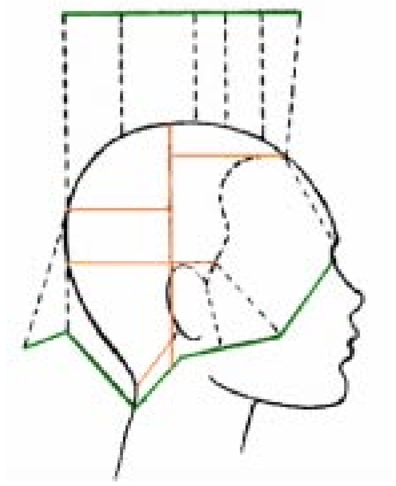
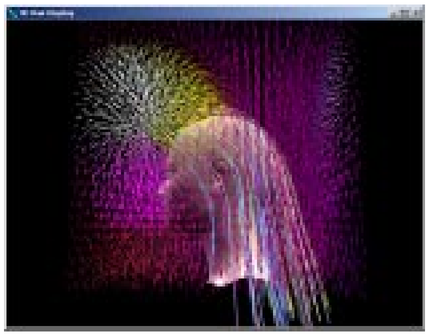
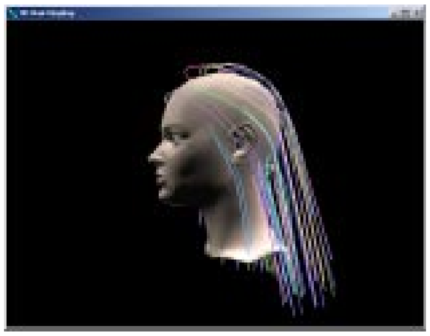

(b)

Figure 2: (a) A polygonal head model and the selected region on scalp for hair growth; (b) a length distribution represented with a BSP tree. Orange lines represent intermediate dividing planes; dark green lines represent reference planes at the leaf nodes.


(a)


(b)

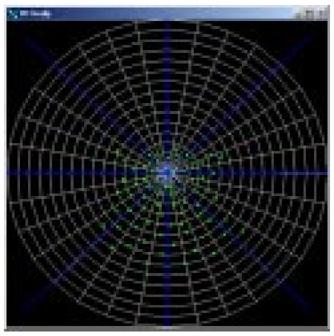

(c)

Figure 3: (a) A visualization of a vector field generated from two field primitives. One of the primitives is a vortex in front of the forehead. The orientation at each point is pseudo-colored with brightness indicating field strength; (b) the initial static sparse hair model extracted from the vector field in (a); (c) a user interface for the 2D flattened map of the scalp. The green dots represent the roots of the sparse guide hairs shown in (b).

2.2 Modeling Sparse Guide Hairs

To generate a detailed hair model with around 50,000 strands, we start with a sparse set of guide hairs uniformly distributed over the scalp. Each guide hair is initially represented as a polyline which is obtained by tracing a trajectory through an interactively defined global vector field [34]. To perform the tracing, we start from the root of the guide hair on the scalp. At every step, we

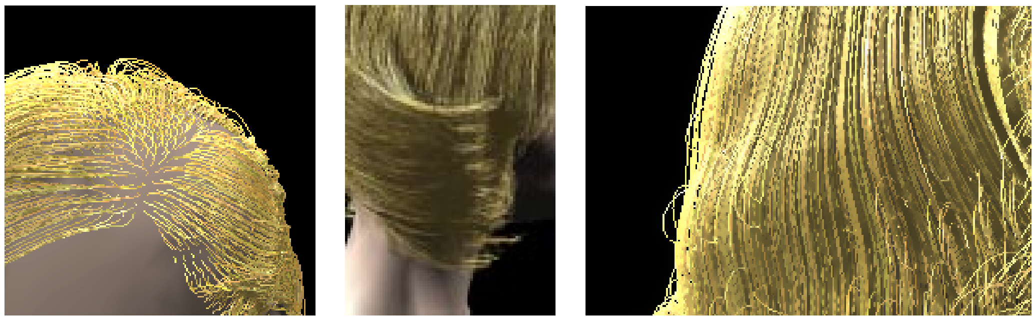


Figure 4: (a) As shown here, a thin ellipsoidal vector field primitive can be used for dividing hairs; (b) the magnetic field induced by a linear or arc current can be used as a perming roller to wrap hair around; (c) a field primitive that can generate a volumetric wave in the hair.

generate a new segment of the polyline along the current direction of the vector field. This is repeated until the guide hair has reached its predefined length.

The global vector field is actually a superposition of multiple vector field primitives with local influence. Since this approach is similar to the scheme in [11] which makes use of ideal fluid flow fields, we briefly introduce the concept and our adaptations below. A vector field primitive is a simple vector field with a rigid transformation between the world coordinate system and its local coordinate system. The rigid transformation can be interactively edited to change the position and orientation of the local coordinate system. Each field primitive is responsible for one large feature in the hair flows. For example, we can use one field primitive to bend hairs towards the back of the head; or make the hairs on the forehead curve differently from those on the back of the head.

The orientation and strength of a vector field primitive at a certain point are provided by two separate (procedurally defined) functions. To achieve local influence, the strength of a field is actually a product of the inherent field strength and a spatial term that diminishes with increasing distance from a reference point in the field. In practice, we have used three common functions for the spatial term, namely, uniform, inverse power and a smoothly decreasing function connecting two constants. It is straightforward to define the first two functions. The smoothly decreasing function is defined as

$$SD(r) = \begin{cases} s_0, & r \le r_0; \\ 0.5(s_0 + s_1) + 0.5(s_0 - s_1)\cos(\frac{r - r_0}{r_1 - r_0}\pi), & r_0 < r < r_1; \\ s_1, & r \ge r_1 \end{cases} \tag{1}$$

where a cosine function connects two constant parameters s_0 and s_1. Either s_0 or s_1 can be set to zero. Local influence can be achieved by setting s_1 to zero. Fig. 3 shows a sparse hair model extracted from a superposed vector field. Some of the field primitives we have used in practice are shown in Fig. 4.

The locality of a vector field primitive can also be specified on the scalp. We can associate a list of local regions on the scalp with a field primitive so that only hair strands growing from these regions can be affected by this field primitive. The union of these local regions is called the *domain* of the field primitive.

The guide hairs are represented as polylines so far. The smoothness of the polylines can be improved by Hermite spline interpolation. The tangent at each vertex required by a Hermite spline can be obtained by averaging the directions of the two line segments sharing the same vertex. We can also further edit the guide hairs by transforming Hermite splines into B-splines and interactively adjusting the positions of their control points.

Note that during this guide hair modeling stage, we can interactively make changes to the vector fields and B-splines in real-time since the rendering time for a small number of splines in addition to the synthetic face model is minimal. This is the reason why we choose to model sparse guide hairs first.

2.3 Hair Interpolation

Since we eventually need a dense hair model for rendering, the remaining hair strands in the dense set are interpolated from the guide hairs. Intuitively, one could imagine a simple procedure by averaging the position of the neighboring strands. However, this approach tends to group strands together into unnatural clusters. We adopt a more sophisticated method that produces better interpolation results. It requires an approach for defining a local coordinate system at each potential hair root. A typical scheme for this uses a global UP vector, such as the vertical direction, and the local normal orientation. Our interpolation procedure works as follows:

- Find the nearest root of a guide hair and transform the segments of that guide hair from the world coordinates to its local coordinates. Name this transformation M_1.
- Take these segments in the local coordinates and transform them back to the world coordinate using the local-to-world coordinate transformation defined at the root of the interpolated strand. Name this transformation M_2^{-1}.

$$M_2^{-1} M_1 p \tag{2}$$

The procedure is summarized as equation (2), where p is the location of the guide hair in the world coordinate, M_1 and M_2 are the two transformations described previously. More than one nearby guide hairs can be used together to achieve smoother results by merging the multiple transformed guide hairs with some averaging scheme. Local clustering effects can be removed by interpolation from multiple guide hairs. However, when discontinuities caused by vector fields like the one in Fig. 4(a) are present, the guide hairs for interpolation should be chosen carefully so that they fall on the same side of any divider.

In summary, our procedure generates better results by taking into account the round shape of the scalp and considering both rotation and translation between local coordinate systems.

2.4 Generating Random Curliness with an Offset Function

Given the interpolated dense model from the previous section, we still need to add natural curliness of hair strands and even some local randomness in the sweep of each strand to create an overall natural appearance. To achieve this, we first define a specific offset function for hair strands in a canonical coordinate system, then reconstruct curly hairs by modulating the hair sweeps from vector fields with this function.

2.4.1 Offset Function

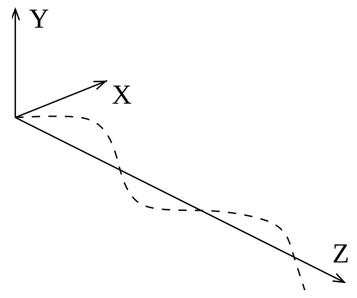


Figure 5: The canonical coordinate system for the hair offset function.

We assume that the underlying hair strand is a straight half line coincident with the positive z-axis of the canonical coordinate system for the offset function, and the origin is at the root of this strand. The offset function is a two-component function that returns the offsets along both $x-$ and $y-$ axes given a z value(Fig. 5).

According to [23], there are three primary types of natural hair waves, namely, a uniplanar wave, a dished wave and a helix. A uniplanar wave looks very much like a planar sinusoidal wave, while a dished wave looks like a sinusoidal wave which has been mapped onto half of a cylindrical surface with the axis of the wave lengthwise along the cylinder. [23] further presented the author's research results on the main reason for the formation of these waves, which is the sequence of periodical contraction and relaxation of small-scale muscles that are in control of the various follicle configurations. Please note that an artificial helix can also be shaped with a hair roller during perming[39] by first wrapping hairs around the roller and then removing the roller along its symmetric axis.

We can see that all three types of hair waves can be easily represented with our offset function although we do not have to be restricted to these types when synthesizing wavy hairs. The uniplanar wave can be represented with a sinusoidal offset for the x-axis and a zero offset for the y-axis. A helix can be represented with a sinusoidal offset for the x-axis and a cosinoidal offset for the y-axis. The dished wave can be represented with a sinusoidal offset for the x-axis, but the wave for the y-axis is a more complicated function which always returns a positive offset value and whose period is at most half of the period of the wave for the x-axis.

Although a variety of parametric or procedurally defined functions can be used, we designed a specific class of offset functions with a fixed number of parameters as follows.

$$\text{Wave}(t) = \text{Mag}(t)\sin(2\pi(Rt + P_0)t + \phi_0) + Bias \tag{3}$$

where t is the function variable; R, P_0, ϕ_0, and $Bias$ are constant parameters; and

$$\mathrm{Mag}(t) \quad = \quad A + Bt\exp(-\alpha t) + C(1 - \exp(-\beta t) + D\exp(\gamma(t - t_0)) \qquad (4)$$

where A, B, C, D, α, β, γ, and t_0 are all constant parameters. We can see this is basically a sinusoidal wave with variable period whose initial value is P_0, and with variable magnitude whose initial value is A. If $R > 0$, the period of the wave becomes smaller as we move closer to the tip of the hair, which is a common phenomenon exhibited by real wavy hairs. The last three terms in Eq.(4) allow us to vary offset magnitude at different parts of a hair strand. The second term reaches a maximum at $t = 1/\alpha$; the third term becomes close to C with sufficiently large t; and the last term increases exponentially especially when $t > t_0$ which can be used to model curly features at the end of a strand. With two sets of different parameters, this function can model offsets along both x- and y-axes. All the examples shown in this paper use this parameterization of the offset function.

2.4.2 Editing Hair Sweeps with the Offset Function

To modulate a hair sweep from vector fields with the above offset function, we need to consider the hair sweep as the parametric axis for the variable t in the offset function. For a certain point p on the sweep, its corresponding t value is the accumulated arc length between p and the root of the sweep. Thus, we can obtain a pair of offset values for each point on the sweep.

We still need to define a local coordinate system at each point on the sweep to impose the offsets. The z-axis of the local system at a point p is always the tangential direction of the sweep at p. We use the vector between the point p and the centroid of the scalp as an UP vector for the local system. Then both x- and y- axes can be derived from the UP vector and the z-axis. So p is originally at the origin of this local system. Its new location is always on the local xy-plane and is determined by the pair of offsets returned from the offset function. [30] presents a similar idea to edit details of a 2D curve.

The same editing operation can still be applied even when the hair sweep has dynamics which makes it deform from frame to frame. However, every local coordinate system, once initialized as above, should follow the same transforms that its corresponding point on the sweep undergoes.

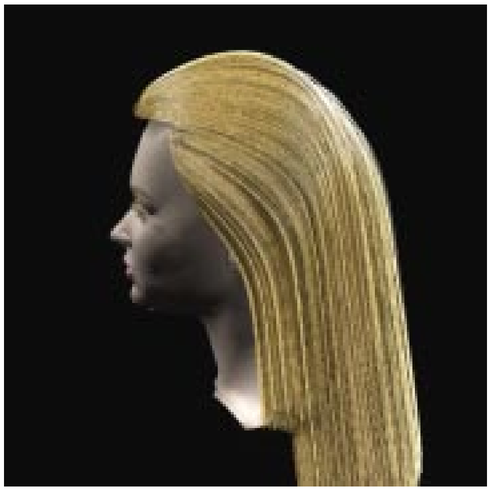
2.4.3 Random Curliness

To achieve natural appearance, obviously we should not edit all hair sweeps with the same set of offset parameters. On the other hand, random hair curliness is far from white noise. The author of [23] observed smoothly spatially varying initial phase, ϕ_0 in Eq.(3), in hair waves. During perming, usually a local cluster of hair is wrapped around a roller, which also suggests that hair waves from that cluster share similar shapes. In practice, we set up a multi-scale grid over the scalp in a polar coordinate system, and generate smoothly varying random parameters for the offset function using Perlin's noise[26] given the mean and standard deviation of each parameter for each scale.

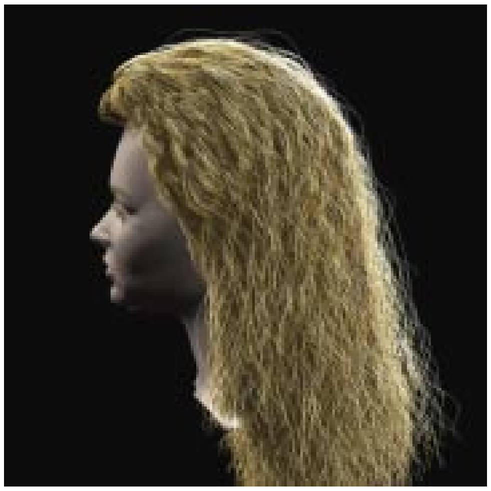
2.5 Hair Clustering

Because of cosmetics and static charges, nearby hairs tend to form clusters. This effect is partially dealt with in Section 2.4.3 when multiscale noise is synthesized. Hairs growing from the same cell in the highest resolution grid have the same parameters for the offset function, therefore tend to be close to each other. Here we introduce an additional offset to further reinforce this effect when the previous treatment is not sufficient. One hair is chosen to be the representative of the hairs growing from the same cell in the grid. Each vertex on other hairs from the same cell is forced to move by a certain distance towards its corresponding vertex on the representative hair. Hair clusters become obvious after this step.

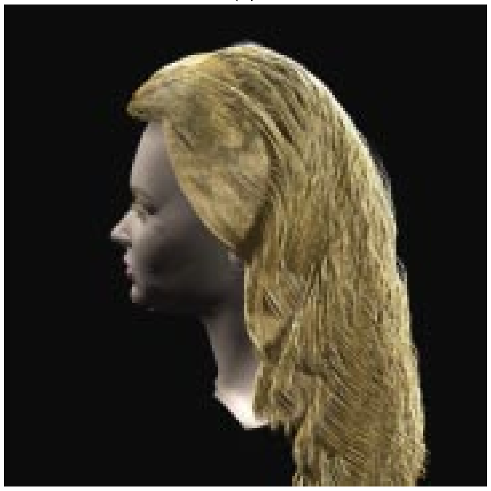
Fig. 6(a) shows the interpolation result. Fig. 6(b)-(c) shows the difference the offset function can make on hair appearance. Fig. 6(d) demonstrates the effectiveness of the hair clustering step.



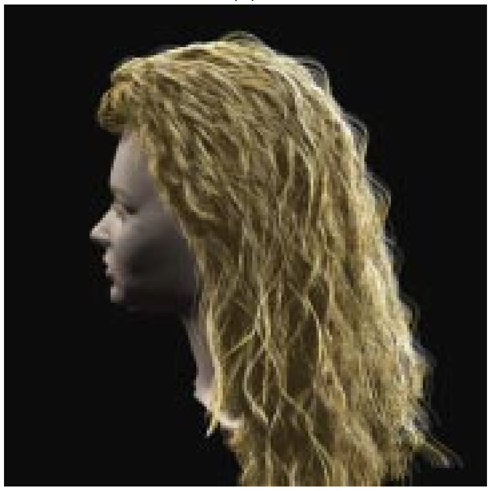
(a)



(b)



(c)



(d)

Figure 6: (a) A static dense hair model interpolated from the sparse model shown in Fig. 3(b); (b) an improved version after modulating the intial model in (a) with random offset functions at a fine scale; (c) another version by modulating the model in (a) with random offset functions at a coarse scale. (b) and (c) look quite different because of different offset functions; (d) the appearance of the model in (b) is improved by further clustering nearby hairs;

3 Hair Animation

The modeling approach in the previous section only generates static hair models. To elaborate hair animation techniques, let us begin with single hair dynamics. Then, a sparse model for hair-hair interactions will be introduced.

3.1 Single Hair Strand Dynamics

There are few techniques developed on modeling single hair strand dynamics [29, 1, 6, 12]. Some of the previous work [29, 6] models a single strand as particles connected with rigid springs. Each particle has 3 degrees of freedom, namely one translation and two angular rotations. This method is simple and easy to implement. However, individual hair strand has very large tensile strength, and hardly stretches by its own weight and body forces. This property leads to stiff equations which tend to cause numerical instability unless very small time steps are used. We model each hair strand as a serial rigid multibody chain. There is a rotational joint between two adjacent segments, and translational motion is prohibited. A single chain can be considered as a simple articulated body with joint constraints. Dynamic formulations of articulated bodies are addressed in robotics [7, 25] as well as graphics literature [37]. Both constrained dynamics with Lagrange Multipliers [2] and generalized(or reduced) coordinate formulation [7] can be used equally efficiently. The dynamics of a serial multibody chain and its generalized coordinate formulation have recently been applied to single hair simulation in [12]. Since our main contributions in hair animation concern hair-hair interactions, we describe the formulation of the serial multibody chain and our adaptations briefly in this section.

3.1.1 Kinematic Equations

In our model, we assume that the twisting of a hair strand along its axis is prohibited. This reduces each rotational joint in a strand to have two degrees of freedom. A rotational joint can be decomposed into two cascading one-dimensional revolute joints each of which has a fixed rotation axis [25]. The rotation angles at the 1D revolute joints represent the set of generalized coordinates in a multibody chain system. If a 1D revolute joint has a rotation axis ω along with a point q on the axis, the matrix transformation corresponding to a rotation around ω by an angle θ can be given by the exponential map $e^{\widehat{\xi}\theta}$ [25] where

$$\widehat{\xi} = \begin{bmatrix} \widehat{\omega} & v \\ 0 & 0 \end{bmatrix}, \widehat{\omega} = \begin{bmatrix} 0 & -\omega_3 & \omega_2 \\ \omega_3 & 0 & -\omega_1 \\ -\omega_2 & \omega_1 & 0 \end{bmatrix},$$

and $v = -\omega \times q$. Suppose a hair segment has n preceding 1D revolute joints in the chain and a local frame is defined at the segment. Assume the local-to-world transformation for this frame when all preceding joint angles are zero is $g_{st}(0)$. The updated local-to-world transformation after a series of rotations at the n joints becomes

$$g_{st}(\Theta) = e^{\widehat{\xi}_1\theta_1} e^{\widehat{\xi}_2\theta_2} \cdots e^{\widehat{\xi}_n\theta_n} g_{st}(0) \tag{5}$$

Thus, given an arbitrary series of joint angles, the position of every vertex in the chain can be obtained using this product of exponentials of its preceding joints. The exponential map actually is just another way of formulating a 4×4 homogeneous matrix. It can be calculated in constant time [25]. Therefore, the whole chain can be evaluated in linear time.

3.1.2 Dynamics of Hair Strand

Given the mapping in Eq. 5 which is from the set of generalized coordinates (joint angles) to real 3D world coordinates, hair strand simulation can be solved by integrating joint angular velocities and accelerations. Forward dynamics of a single strand in terms of joint angular velocities and accelerations can be solved using Featherstone's algorithm [7] or Lagrange's equations for generalized coordinates [25]. The former method is more efficient with a linear time complexity. Detailed formulations of these methods can be found in [7, 25]. In practice, we adopt an implementation of Featherstone's algorithm by Kuffner and Mirtich [24] for the stable simulation of cascading 1D revolute joints.

Both external and internal forces are indispensable for single hair dynamics. In this paper, hair-hair interactions are formulated as external forces in addition to gravity. The actual form of these external forces will be discussed in Section 3.2. At each joint of the hair chain, there is also an internal actuator force to account for the bending and torsional rigidity of the strand. We model the actuator force as a hinge with a damping term as in [29]. Since the hairs from our modeling stage may have deformations even when there are no external forces, we define a nonzero resting position for each hinge. Any deviation from the resting position results in a nonzero actuator force trying to reduce the amount of deviation. This setup helps a strand to recover its original shape after subsequent movement.

3.1.3 Strand-Body Collision

In order to simulate inelastic collision between the hair and human body, there is no repelling forces introduced by the human body. Once a hair vertex becomes sufficiently close to the scalp or torso, it is simply stopped by setting its own velocity to be the same as the velocity of the human body while all the following vertices in the multibody chain are still allowed to move freely.

Any acceleration towards the human body is also prohibited at the stopped vertices which, however, are allowed to move away from or slide over the human body. Frictional forces are added as well to those vertices touching the human body. Collision detection is handled explicitly by checking penetration of hair strand particles with the triangle mesh of the body parts.

This scheme cannot guarantee that the hair vertices do not penetrate other colliding surfaces in the middle of a time step. If penetration does occur, we need to move the part of the penetrating strand outside the surface in the same time step so that no penetration can be actually observed. It is desirable that the tip of the hair, if outside the surface, remains unchanged during this adjustment in order to introduce minimal visual artifacts. To achieve this goal, inverse kinematics [25] can be applied to adjust the positions of the intermediate vertices between the tip and the adjusted locations of the penetrating vertices. In our implementation we opt for a simpler method using iterative local displacements. Starting from the root, we move the first penetrated vertex p_1 to its nearest valid location p'_1, and then propagate this displacement by moving the subsequent vertices. More specifically, assume the following vertex of p_1 is p_2, we compute the vector $v' = ||p_2 - p_1|| \frac{v}{|v|}$, where $v = p_2 - p'_1$. The new location for p_2 after the adjustment is $p'_2 = p'_1 + v'$. We repeat this for all the vertices following p_1 until reaching the tip.

3.2 A Sparse Model for Hair-Hair Interaction

We devise a novel scheme to simulate only a sparse set of guide hairs for complex hair-hair interactions. A guide hair essentially represents a local hair wisp. Therefore, it should have the mass of a whole wisp instead of a single hair. We first introduce an elastic model to preserve the relative positions of the hair strands. The static links model the interaction of the hair due to interweaving, static charges and hairstyling. Second, the hair-hair collision and friction is simulated using the guide hairs and a collection of auxiliary triangle strips. Last, we provide an adaptive hair generation technique to complement our sparse hair model. The proposed method models the hair dynamics efficiently with good visual realism.

3.2.1 Static Links

It is evident that the hair strands tend to bond together with other strands in their vicinity because of cosmetics, static charges and the interweaving of curly hairs. As a result, the movement of each strand is on most part depended on the motion of other strands. These interactions can have relatively long range effects besides clustering in a small neighborhood. While hair local clustering is modeled by default using our sparse model, longer range interaction is not. Furthermore, slight head movements or external forces exerted on the hair do not change a hairstyle radically. This is partly because each hair strand has its internal joint forces and resting configuration. However, an individual hair's recovery capability is quite limited especially for long hairs. The bonding effect among hairs plays an important role. Dramatic movements can break the bonds created by hairstyling, static charges or interweaving.

To effectively model the bonding effect, we may view the hair as one elastically deformable volume. Traditional models for deformable bodies include 3D mass-spring lattice, finite difference, and finite element method [35, 41]. These models approximate the deviation of a continuum body from its resting shape in terms of displacements at a finite number of points called nodal points. Although the vertices of hair strands may serve as the nodal points inside this hair volume, directly applying traditional models is not appropriate for the following reasons. We are only interested in an elastic model for hair's lateral motion. Under strong external forces, the continuum hair volume may break into pieces which may have global transformations among them. Therefore, using one body coordinate system for the whole hair volume is inadequate.

We propose to build breakable connections, called static links, among hair strands to simulate their elastic lateral motion and enable hairstyle recovery. These connections are selected initially to represent bonds specific to a hairstyle since different hairstyles have different hair adjacency configuration. The static links enforce these adjacency constraints by exerting external forces onto the hair strands. Intuitively, one can use tensile, bending and torsional springs as bonds to preserve the relative positions of the hair strands. In practice, we opt for a simpler and more efficient method using local coordinates.

We introduce a local coordinate system to each segment of the hair strands. For each segment, we find a number of closest points on nearby strands as its reference points. To improve the performance, an octree can be used to store the hair segments for faster searching. We transform these points, which are in the world coordinates, to the segment's local coordinates (Fig. 7a). The initial local coordinates of these reference points are stored as part of the initialization process. Once strands have relative motion, the local coordinates of the reference points change and external forces are exerted onto these strands to recover their original relative positions (Fig. 7b). We model these external forces as spring forces with zero resting length. One advantage of using the local coordinates is that it eliminates the need for bending and torsional springs.

Let us consider a single hair segment h with m reference points. The initial local coordinates of these reference points are represented as $p^o_{h,i}, i = 1, ..., m$, while their new local coordinates are represented as $p^n_{h,i}, i = 1, ..., m$. The accumulated force this segment receives due to static links can be formulated as

$$\mathbf{f}_h = \sum_i \left[k^s_{h,i} |\mathbf{l}_i| - k^d \frac{\mathbf{v}_i \cdot \mathbf{l}_i}{|\mathbf{l}_i|} \right] \frac{\mathbf{l}_i}{|\mathbf{l}_i|} \tag{6}$$

We compute the spring force using the Hook's law in (6), where $k^s_{h,i}$ is the spring constant for the i-th reference point of segment h, and k^d is the universal damping constant. Since the resting length in our case is zero, $|\mathbf{l}_i = p^n_{h,i} - p^o_{h,i}|$ is multiplied by $k^s_{h,i}$ directly. $\mathbf{v}_i$ is the time derivative of $\mathbf{l}_i$.

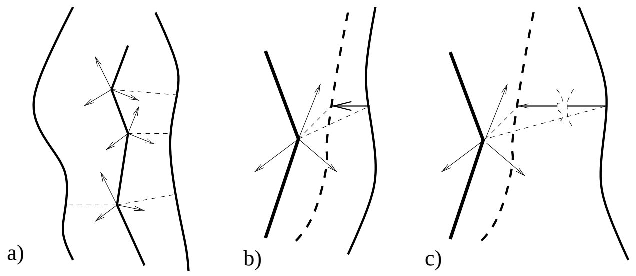


Figure 7: Each hair segment has its own local coordinate system where the forces from all static links (dashed lines) are calculated.

Similar to the bonds of stylized hair, static links can be broken upon excessive forces. We set a threshold for each static link. If the length change of a static link is greater than the threshold, the static link breaks (Fig. 7c). Once a link is broken, the damage is permanent; the link will remain broken until the end of the simulation. To be more precise, we model the spring constant $k^s_{h,i}$ as shown in Fig. 8. As $|\mathbf{l}_i|$ increases beyond δ_1, the spring constant begins to decrease gradually and eventually becomes zero at δ_2 as the spring snaps. The spring constant will not recover even when $|\mathbf{l}_i|$ shrinks below δ_1 again. This nonreversible spring model would make the motion of the hair look less like a collection of rigid springs.

When external forces recede, the original hairstyle may not be completely recovered if some of the static links have been broken. New static links may form for the new hairstyle with updated neighborhood structures.

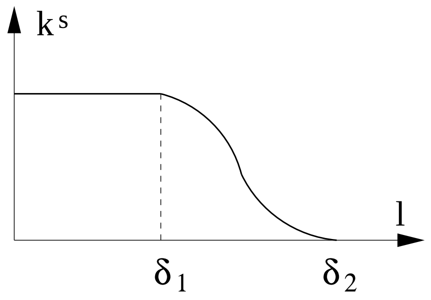


Figure 8: Spring constant $k^s_{h,i}$ vs displacement graph.

3.2.2 Dynamic Interactions

Elastic deformation only introduces one type of hair-hair interactions. Hairs also interact with each other in the form of collision. To effectively simulate hair-hair collision and friction using a sparse hair model, we need to have a dynamic model that imagines the space in between the set of sparse hairs as being filled with dense hairs. Collision detection among the guide hairs only is much less accurate. Let us consider a pair of nearby guide hairs. The space between them may be filled with some hairs in a dense model so another strand cannot pass through there without receiving any resistance. To model this effect, we can either consider the guide hairs as two generalized cylinders with sufficiently large radii to fill up the gap between them, or build an auxiliary triangle strip as a layer of dense hair between them by connecting corresponding vertices. The triangular mesh can automatically resize as the guide hairs move, but it is trickier to resize the generalized cylinders. Therefore, we propose to construct auxiliary triangle strips between pairs of guide hairs to approximate a dense hair distribution. If we consider the set of dense hairs collectively as a volume, a triangle strip represents a narrow cross section of the volume. A number of such cross sections can reasonably approximate the density distribution of the original hair volume.

Since the distance between a pair of vertices from two hairs may change all the time during simulation, we decide to use the distance among hair roots. A triangle strip is allowed as long as two guide hairs have nearby hair roots. Each triangle only connects vertices from two guide hairs, therefore is almost parallel to them. Note that the triangle strips may intersect with each other. This does not complicate things because each triangle is treated as an independent patch of hair during collision detection. The triangles are only used for helping collision detection, not considered as part of the real hair geometry during final rendering. They do not have any other dynamic elements to influence hair movement. However, some triangles may have nearby static links which can help them resist deformation. The triangle edges are not directly constructed as static links because static links only connect nearby hair segments while not all the segments connected by triangles are close to each other.

As in standard surface collision detection, two different kinds of collision are considered, namely, the collision between two hair segments and the collision between a hair vertex and a triangular face. Since each guide hair represents a local hair cluster with a certain thickness, a collision is detected as long as the distance between two hair elements falls below a nonzero threshold. Once a collision is detected, a strongly damped spring force is dynamically generated to push the pair of elements away from each other [3]. Meanwhile, a frictional force is also generated to resist tangential motion. A triangle redistributes the forces it receives to its vertices as their additional external forces. Both the spring and frictional forces disappear when the distance between the two colliding elements becomes larger than the threshold. The spring force in effect keeps other hairs from penetrating a layer corresponding to a triangle strip. An octree is used for fast collision detection. All the moving hair segments and triangles are dynamically deposited into the octree at each time step. An octree node has a list of segments and triangles it intersects with.

Hair also exhibits strong anisotropic dynamical properties. Depending on the orientation of the penetrating hair vertex and the triangular face, the repelling spring force might vary. For example, hair segments of similar orientation with the triangle strip should experience weaker forces. We scale the repelling spring force according to the following formula.

$$\mathbf{f}_r = \lambda(1 - |\mathbf{a} \cdot \mathbf{b}|)\mathbf{f}_s \tag{7}$$

The original spring force $\mathbf{f}_s$ is scaled in Eq. (7), where $\mathbf{a}$ is the normalized tangential vector of the hair at the penetrating vertex, $\mathbf{b}$ is the interpolated hair orientation on the triangular face from its guide hair segments, and λ is a scale factor. When $\mathbf{a}$ and $\mathbf{b}$ are perfectly aligned, the scaled force $\mathbf{f}_s$ becomes zero. On the contrary, when they are perpendicular, the spring force is maximized. The collision force between two hair segments can be defined likewise.

The hair density on each hair strip is also modeled as a continuum. It can be dynamically adjusted during a simulation. If there is insufficient hair on a strip, the strip can be broken. This would allow other hair strands to go through broken pieces of a hair strip more easily. This is reasonable because sometimes there is no hair between two hair clusters while at the other times, there may be a dense hair distribution. In our current implementation, the length of the triangle edges serves as the indicator for when the hair density on a strip should be adjusted. If a triangle becomes too elongated, it is labeled as broken. If a triangle is not broken, the magnitude of the collision force in Eq. (7) is made adaptive by adjusting the scale factor λ according to the local width of the triangle strip to account for the change of hair density on the triangle. Unlike the static links, this process is reversible. Once the two guide hairs of a strip move closer to each other again, indicating the hair density between them is increasing, the generated collision force should also be increased, and the triangle strip should be recovered if it has been broken. If every triangle strip in our method is modeled as broken from the beginning, our collision model becomes similar to the wisp model in [28] since every guide hair in our method actually represents a wisp.

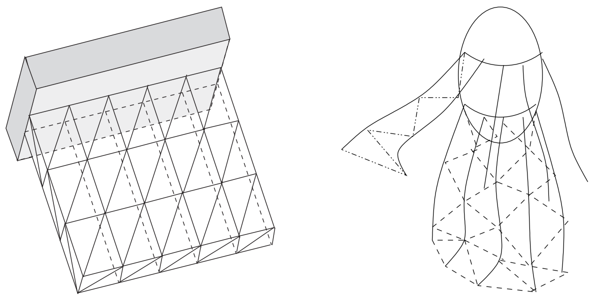


Figure 9: (a) For a brush, triangle strips can be inserted between horizontally and vertically adjacent guide hairs. (b) For a human scalp, triangle strips are inserted only between horizontally adjacent guide hairs

It may not be necessary to build triangle strips among all pairs of nearby strands. For a simple brush in Fig. 9a, we can only insert triangle strips between horizontally and vertically adjacent guide hairs. For human hair, we sometimes find it practically good enough to build triangle strips between guide hairs with horizontally adjacent hair roots (Fig. 9b). This is because hairs drape down due to gravity, and the thickness of the hair volume is usually much smaller than the dimensions of the exterior surface of the hair volume. In such a situation, using triangles to fill the horizontal gaps among guide hairs becomes more important.

3.2.3 Adaptive Guide Hair Generation

Initially, we select the guide hairs uniformly on the scalp. However, it is not always ideal to pick the guide hairs uniformly. During a run of the simulation, some part of the hair may be more active than the other parts. For example, when the wind is blowing on one side of the hair, the other side of the hair appears to be less active. As a result, some computation is wasted for not so active regions. For not so active regions, fewer guide hairs combined with interpolation is sufficient. However, for more active regions, it is desirable to use more guide hairs and less interpolation for better results. We design an adaptive hair generation method to complement our sparse hair model.

We generate additional guide strands adaptively during the simulation to cover the over interpolated regions. The distribution and the initial number of guide strands are determined before the simulation. However, as the simulation proceeds, more hair strands can be added. The hair model may become more and more computationally intensive if hairs can only be inserted. We notice that the inserted hairs may become inactive again later in the same simulation. Therefore, we also allow them to be deleted if necessary. To keep our hair strands relatively sparse, we may also set a limit on how many adaptive hair strands can exist at the same time. Picking the right place to generate adaptive guide hair is important.

We use a simple technique to detect where to add and remove adaptive guide hairs. For each pair of guide strands, we compute the distance between all pairs of corresponding vertices of the strands. If any pair of vertices become farther away than a threshold, it indicates that the hair in between these two guide strands is relying too much on the interpolation. We then add an adaptive guide hair half-way between these two strands. At the same time, we exam the adaptive guide hairs from the last step of the simulation. If some of the guide strands are no longer needed (when the two neighboring strands are sufficiently close), we remove those strands and save them for future hair generation. When an adaptive hair is generated, its initial vertex positions and velocities are obtained by interpolating from those of the two initiating guide hairs. If there was a triangle strip between these two initiating hairs, it should be updated to two strips with the new hair in the middle. The new adaptive hair then follows its own dynamics from the next time step, colliding with nearby strands and triangle strips. To avoid discontinuous motion on the rest of the hairs, a new adaptive hair does not spawn static links with other strands.

3.2.4 Adaptations for Curly Hair

In our approach (Fig. 1(b)), curliness is modeled in the last step before rendering. Guide hairs of a curly hair model are actually quite straight except for large-scale deformations. They represent the spines of curly hairs. However, the mutual interactions among the guide hairs should reflect the features in the final dense hair model since the motion part of the guide hairs is still going to be used for rendering even after hair curliness has been changed. In the case of a dense curly hair model, there should be stronger static links because curly hairs have more interweaving effects, and weaker hair-hair collisions because curliness generates a soft cushion effect among hairs. In practice, we follow these intuitions to set up the strength of mutual interactions.

3.3 Dynamic Dense Hair Modeling

Hair animations are generated in two passes in our system. In the first pass, the set of guide hairs are simulated from frame to frame with all the external forces and hair-hair interactions described in Section 3.2. During the simulation of each individual guide hair as described in Section 3.1, forces due to hair-hair interactions are treated as additional external forces. Hair-object collisions are also detected during this simulation. By the end of the first pass, we have obtained the positions and velocities of each guide hair at each frame.

In the second pass, we generate a dense hair model for each frame according to the guide hair positions. Curliness modeled using our offset function in Section 2.4 can also be added if necessary. To maintain the consistency from frame to frame, each hair in the dense model is assigned a fixed offset function throughout the whole sequence. Therefore, at each frame, we merely modulate the same offset function onto the underlying deforming spine to produce the desired curly strand. However, the offset function may still vary from strand to strand.

Since small objects may miss all the guide hairs, but still hit some of the strands in the dense model, we decide to run hair-object collision detection for each (curly) hair in the dense model. Although this involves a certain amount of computation, the computing power available nowadays on a single processor workstation has already become sufficient to perform this task in a very short amount of time. If a hair penetrates an object, a scheme similar to the one described in Section 3.1.3 is used to adjust the hair.

4 Hair Rendering

Although the primary focus of this paper is on hair modeling and animation, we will discuss briefly our approach to rendering realistic hair. The kind of physical interaction considered here includes self-shadowing and light scattering. Hair strands are not completely opaque. Therefore, the interaction between light and hair leads to both reflection and transmission. When a dense set of hairs is present, light gets bounced off or transmitted through strands multiple times to create the final exquisite appearance. Basically, we can view a dense hair as a volume density function with distinct density and structures everywhere. The hair density is related to the local light attenuation coefficient while the structures, including the local hair orientation, are related to the phase function during scattering. In this section, we discuss how to efficiently render animated sequences of hair with high visual quality by considering the above factors.

While secondary scattering can improve the rendering quality, primary scattering and self-shadowing are considered much more important. Since the rendering performance is our serious concern when generating hair animations, we decide to simulate the latter two effects only. This is equivalent to solving the following volume rendering equation [15],

$$L(x,\vec{\omega}) = \int_{x_0}^{x} \tau(x',x)\sigma(x') \sum_{l} f(x',\vec{\omega}_l,\vec{\omega}) I_l(x') dx' \qquad (8)$$

where $L(x,\vec{\omega})$ represents the final radiance at x along direction $\vec{\omega}$, $f(x',\vec{\omega}_l,\vec{\omega})$ is the normalized phase function for scattering, $I_l(x')$ is the attenuated light intensity from the l-th light source, and $\tau(x',x) = \exp(-\int_{x'}^{x}(\alpha(\xi)+\sigma(\xi))d\xi$ where $\alpha(x)$ is the absorption coefficient and $\sigma(x)$ is the scattering coefficient. As in volume rendering, the final color of a pixel can be approximated as the alpha-blending of the colors at a few sample points along the eye ray going through that pixel. To perform alpha-blending correctly, the sample points need to be depth-sorted. In terms of hair, the sample points can be the set of intersections between the eye ray and the hair segments. Note that the input to the rendering stage is a large number of hair segments resulted from the discretization of the spline interpolated dense hairs. In order to obtain the set of intersections at each pixel efficiently, scan conversion is applied to the segments and a segment is added into the depth-sorted list of intersections at a pixel once it passes that pixel. Antialiasing by supersampling each pixel can help produce smoother results.

To finish rendering, we still need a color for alpha-blending at each of the intersections. It should be the reflected color at the intersection. The reflectance model we use is from [10]. It is a modified version of the hair shading model in [14] by considering partial translucency of hair strands. Since other hairs between the light source and the considered hair segment can block part of the incident light, the amount of attenuation is calculated using the opacity shadow maps [17] which can be obtained more efficiently than the deep shadow maps [22]. Basically, the algorithm in [17] selects a discrete set of planar (opacity) maps perpendicular to the lighting direction. These maps are distributed uniformly across the volume being rendered. Each map contains an approximate transmittance function of the partial volume in front of the map. Thus, the approximate transmittance of the volume at any point can be obtained by interpolating the transmittance at corresponding points on the two nearest opacity maps. In our implementation, exponential interpolation has been used since the attenuation of light through a volume is exponential. The exponential interpolation can be written as

$$\exp(-\alpha_1 \frac{d_2}{d_1+d_2} - \alpha_2 \frac{d_1}{d_1+d_2})$$

where $\exp(-\alpha_1)$ and $\exp(-\alpha_2)$ are the attenuation at the two nearest maps, and d_1, d_2 are the distance from the point to the two maps, respectively.

When the hair is rendered together with other solid objects, such as the head and cloth, which we assume to be completely opaque, the color of the solid objects needs to be blended together with the hair's during volume rendering. The solid objects also have their separate shadow buffer for each light source. Anything in the shadow of the solids receives no light while those solids in the shadow of the hair may still receive attenuated light.

5 Results

5.1 Hair Modeling

We have applied our modeling technique to a variety of hairstyles with different degrees of curliness, and obtained satisfactory results. Some of the hairstyles are shown in Fig. 10. They look natural and realistic. Vector field primitives can either be interactively adjusted by the user or be randomly generated. Typically, we need around ten vector field primitives to interactively model a hairstyle. The time for user interaction is usually between one and two hours. On the other hand, Fig. 10(d) uses 273 field primitives randomly generated and distributed over the scalp.

For each local cluster of hairs, the parameters in Eq. (3) are randomly generated. The mean and standard deviation of each parameter at each scale are fixed for all hairs, and are specified by the user. For the results shown here, we use two levels of scales to generate the random parameters. The final hair waves in Fig. 10(a) and in Fig. 6(d) actually have the same average magnitude, 5mm. However, hair waves in Fig. 6(d) look more random because they have a much shorter period and a much larger standard deviation for the initial phases of the waves. The hairs in Fig. 10(c) have an average magnitude of 1mm to give a smoother look.

5.2 Hair Animation

We have also successfully tested our hair dynamic model in a few animations. In our experiments, we used around 200 initial guide hairs during the animation of the sparse model with 15 segments for each strand. During each time step, a strand is interpolated with a Hermite spline and discretized into around 50 smaller segments. Based on this set of resampled sparse hairs, a dense hair model with 50,000 strands is generated on the fly at each frame for the final rendering. The guide hair animation stage takes about one second per frame on a Pentium III 800MHz processor. Hair interpolation, hair-object collision detection and antialiased rendering takes another 20 seconds per frame on a Pentium 4 2GHz processor. Fig. 11(a)-(b) show a sparse hair model with static links and the dense interpolated model. Static links can prevent excessive changes to a hairstyle during motion (Fig. 11(c)-(d)) as well as provide hairstyle recovery after motion. Fig. 12 shows two synthetic renderings of animated hair.

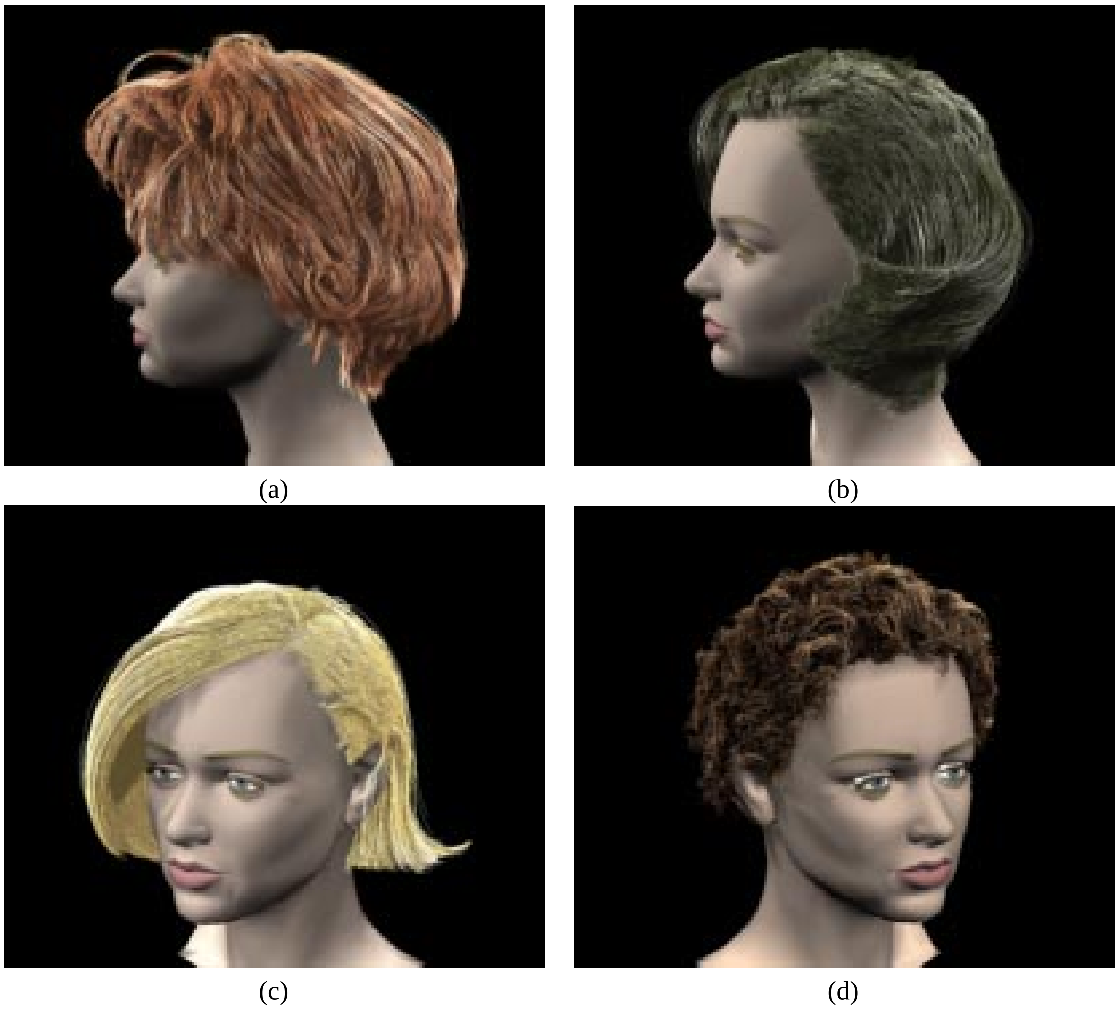


Figure 10: Four hairstyles produced from our techniques. These hairstyles have different length distributions and different levels of curliness.

5.2.1 Comparison with Ground Truth

A synthetic head shaking sequence is compared with a real reference sequence in Fig. 13. The hair strands in the real sequence obviously have mutual connections since they move together. We use relatively strong static links to simulate this effect. The head motion in the synthetic sequence was manually produced to approximate the real motion. Nonetheless, the synthetic hair motion reasonably matches the real one.

5.2.2 Dynamic Collision

To demonstrate the effectiveness of our hair collision strategy, we built a simple braided hair model and let it unfold under gravity. There are basically two sets of guide hairs in the model, and static links and triangle strips are only built among hairs from the same set. Therefore, the two sets can move away from each other. A comparison is given between images from two synthetic sequences in Fig. 14, one with collision detection and the other without. In the simulation without collision detection, hairs go through each other. But in the sequence with collision detection, hairs unfold correctly in a spiral motion.

5.2.3 Hair-Air Interaction

Hair-air interaction is traditionally modeled as air drag which only considers the force exerted on the hair from the air. However, the velocity field of the air is also influenced by the hair. The method in [12] can be adapted to our model for hair-air interaction. That is, the air is simulated as a fluid and it generates a velocity field. Each hair vertex receives an additional external force from the air. This force can be modeled as a damping force using the difference between the velocity of the air at the vertex and the velocity of the hair vertex itself. The force exerted from the hair back to the air can be modeled similarly. If the air is simulated using a voxel grid [8], the velocity of the hair at each grid point can be approximated using the velocities of the nearby hair vertices and auxiliary triangles.

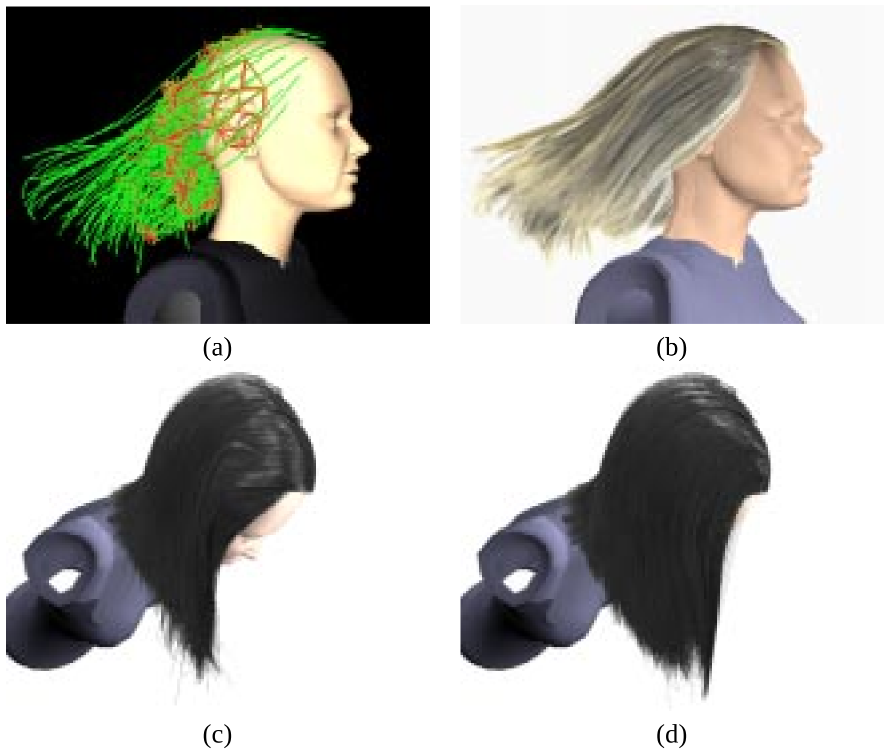


Figure 11: (a) A sparse hair model displayed with static links, (b) a rendered image of the interpolated dense model, (c)-(d) a comparison between hairs with and without static links. The image in (c) has static links, while the one in (d) does not.

Fig. 15(top) shows images from a hair animation with a wind. The wind velocity field is driven by an artificial force field with a changing magnitude and direction. The head and torso are considered as hard boundaries in the wind field while the wind can go through hairs with a certain amount of attenuation.

5.2.4 Brush Simulation

In addition to human hair interactions, we simulate the dynamics of brushes. Fig. 16 shows images from a sequence with a sphere colliding with a synthetic brush. The mutual interactions are weak when only a small number of hairs drape down behind the sphere. However, when more and more hairs drape down, they stabilize much faster because of the collisions.

5.3 Hair Rendering

An artistic flavor can also be added to the images by rendering the hair with increased translucency and specularity. Fig. 15(bottom) shows some re-rendered images from one of the wind blowing sequences with higher specularity. Fig. 17 shows a comparison between normally rendered hair and hair with high translucency.

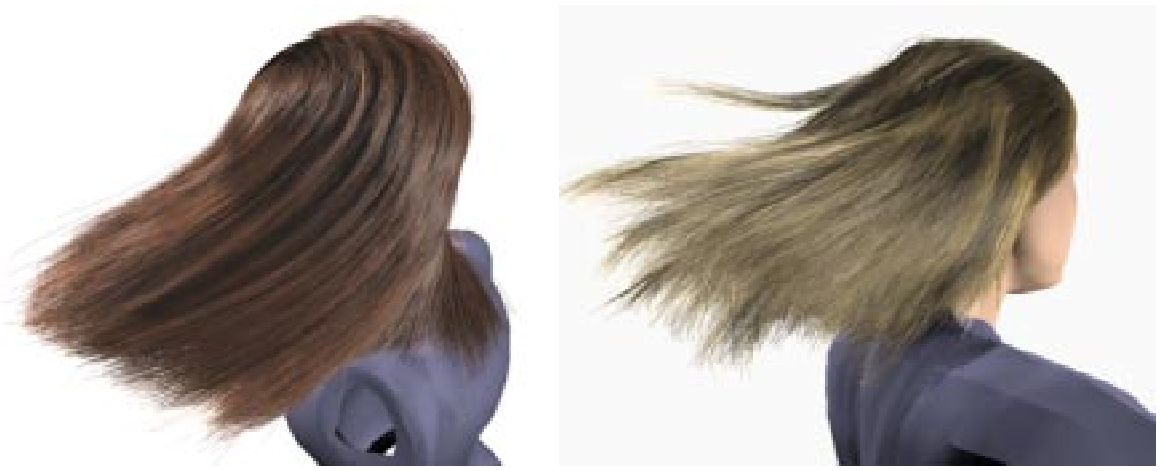


Figure 12: Two synthetic renderings of animated hair.

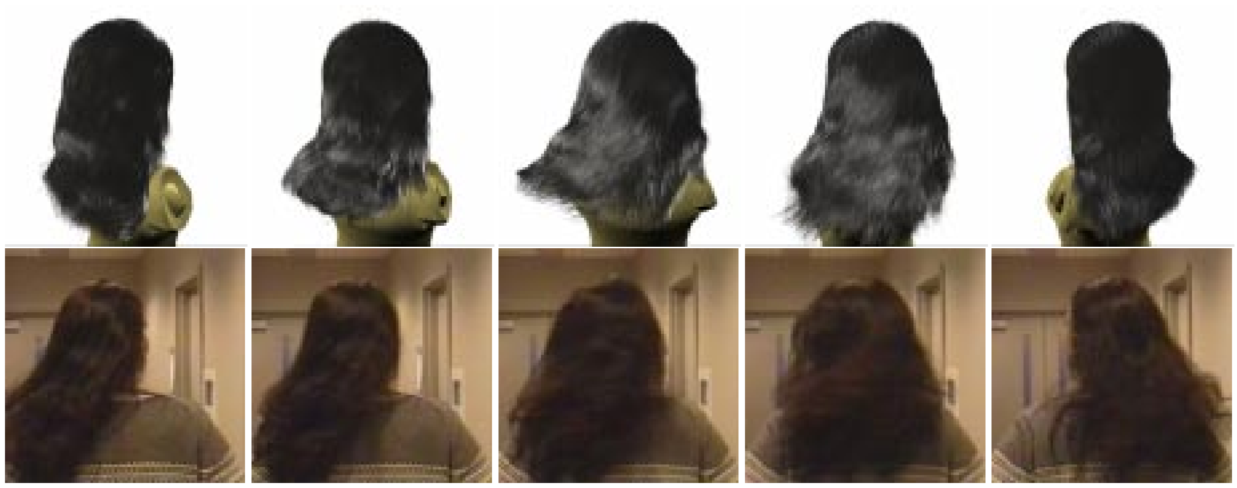


Figure 13: A comparison between a simulated hair animation and a real video. The hair motion is caused by the underlying head motion. Top row: images from the simulated hair motion sequence. Bottom row: images from the real video. The simulated hair motion approximately matches the real hair motion in the video.

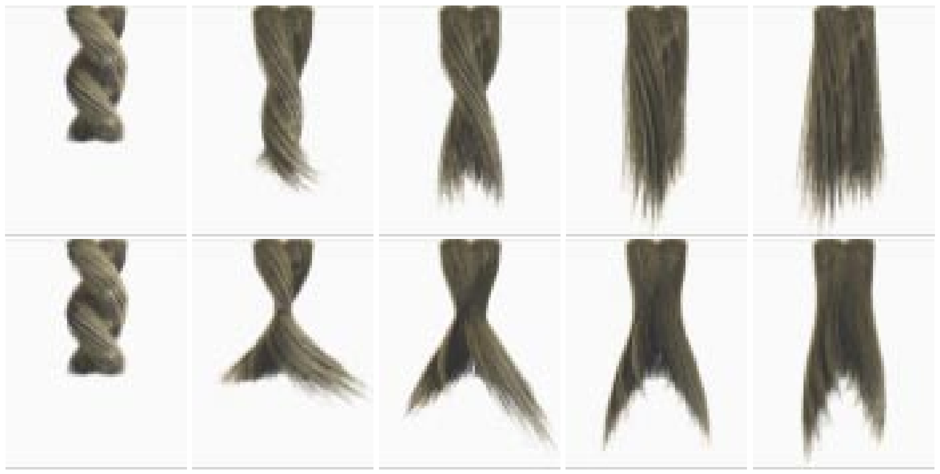


Figure 14: A comparison between two hair animations with and without collision detection. Top row: braided hair unfolds correctly in a spiral motion because of the collision detection. Bottom row: hairs penetrate each other when there is no collision detection.

6 Discussions and Conclusions

In this paper, we presented a unified framework for hair modeling and animation. Both modeling and animation start from a sparse set of guide hairs. The proposed modeling approach can generate realistic wavy hairstyles by editing a basic sheared hair model with a generic offset function for curliness. we also presented an integrated model for hair animation. Specifically, the dynamic model can perform the following functions: the static links and the joint actuator forces enable hairstyle recovery; hair-hair collision becomes more accurate by inserting auxiliary triangle strips and performing collision detection among strands as well as between strands and triangle strips; stable simulation of individual strands is provided by the formulation for multibody open chains. Although our model is not originally designed for hairs without obvious clustering effects, with our multiple hair interpolation scheme, visual results for this kind of hair turned out quite reasonable.

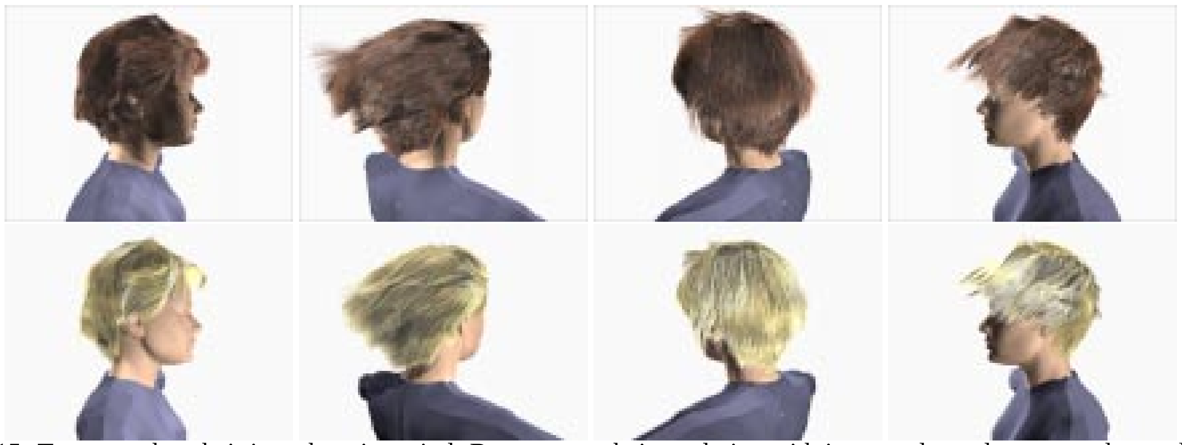


Figure 15: Top row: short hair in a changing wind. Bottom row: hair rendering with increased translucency and specularity to convey an artistic flavor.

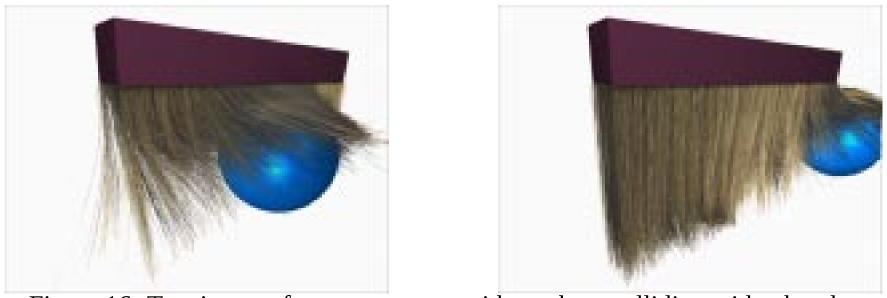


Figure 16: Two images from a sequence with a sphere colliding with a brush.

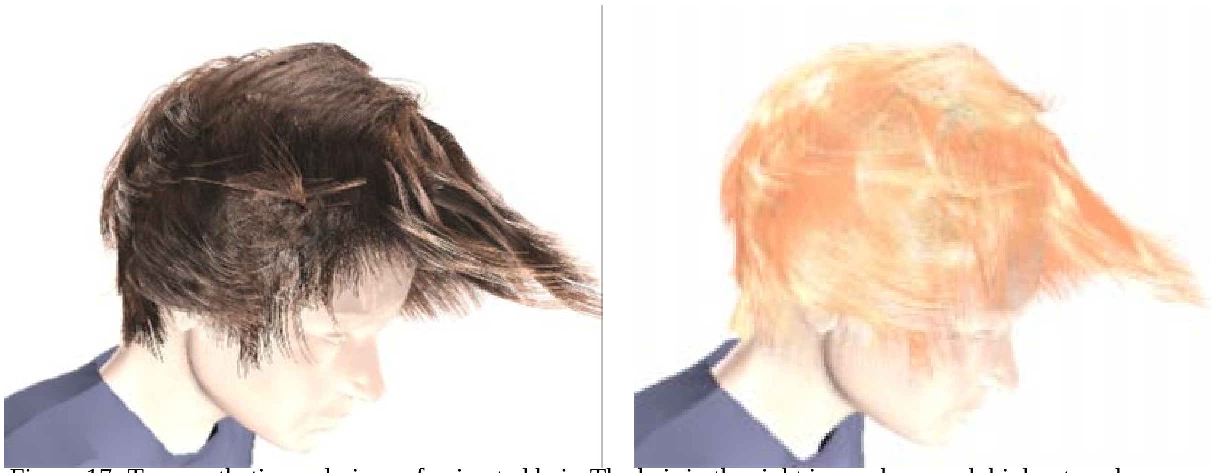


Figure 17: Two synthetic renderings of animated hair. The hair in the right image has much higher translucency.

Acknowledgments

This work was supported by National Science Foundation CAREER Award CCR-0132970 and start-up funds from the University of Illinois at Urbana Champaign.

References

[1] K. Anjyo, Y. Usami, and T. Kurihara. A simple method for extracting the natural beauty of hair. In *Proc. of SIGGRAPH'92*, pages 111–120, 1992.

[2] D. Baraff. Linear-time dynamics using larange multipliers. In *Proc. of SIGGRAPH'96*, pages 137–146, 1996.

[3] D. Baraff and A. Witkin. Large steps in cloth simulation. In *Proc. of SIGGRAPH'98*, pages 43–54, 1998.

[4] J.T. Chang, J. Jin, and Y. Yu. A practical model for hair mutual interactions. In *Proceedings of ACM SIGGRAPH Symposium on Computer Animation*, pages 73–80, 2002.

[5] L.-H. Chen, S. Saeyor, H. Dohi, and M. Ishizuka. A system of 3d hair sytle synthesis based on the wisp model. *The Visual Computer*, 15(4):159–170, 1999.

[6] A. Daldegan, N.M. Thalmann, T. Kurihara, and D. Thalmann. An integrated system for modeling, animating and rendering hair. *Computer Graphics Forum(Eurographics'93)*, 12(3):211–221, 1993.

[7] R. Featherstone. *Robot Dynamics Algorithms*. Kluwer Academic Publishers, 1987.

[8] R. Fedkiw, J. Stam, and H.W. Jensen. Visual simulation of smoke. In *SIGGRAPH 01 Conference Proceedings*, pages 15–22, 2001.

[9] M. Fong. Animating monster fur. In *SIGGRAPH course 36 notes*, 2001.

[10] D. Goldman. Fake fur rendering. In *Proc. of SIGGRAPH'97*, pages 127–134, 1997.

[11] S. Hadap and N. Magnenat-Thalmann. Interactive hair styler based on fluid flow. In *Computer Animation and Simulation 2000. Proceedings of the Eleventh Eurographics Workshop*, 2000.

[12] S. Hadap and N. Magnenat-Thalmann. Modeling dynamic hair as continuum. In *Eurographics Proceedings. Computer Graphics Forum, Vol.20,No.3*, 2001.

[13] R.R. Ogle Jr. and M. J. Fox. *Atlas of Human Hair(Microscopic Characteristics)*. CRC Press, 1999.

[14] J. Kajiya and T. Kay. Rendering fur with three dimensional textures. In *Proc. of SIGGRAPH'89*, pages 271–280, 1989.

[15] J.T. Kajiya and B.P. von Herzen. Ray tracing volume densities. *Computer Graphics (SIGGRAPH 84 Proceedings)*, 18(3), 1984.

[16] T.-Y. Kim and U. Neumann. A thin shell volume for modeling human hair. In *Proc. of IEEE Computer Animation*, 2000.

[17] T.-Y. Kim and U. Neumann. Opacity shadow maps. In *Proc. of Eurographics Workshop on Rendering*, pages 177–182, 2001.

[18] C. K. Koh and Z. Huang. A simple physics model to animate human hair modeled in 2d strips in real time. In *Proceedings of Eurographics Computer Animation and Simulation*, 2001.

[19] A.M. LeBlanc, R. Turner, and D. Thalmann. Rendering hair using pixel blending and shadow buffers. *Journal of Visualization and Computer Animation*, pages 92–97, 1991.

[20] D.-W. Lee and H.-S. Ko. Natural hairstyle modeling and animation. *Graphics Models and Image Processing*, 63:67–85, 2001.

[21] J. Lengyel. Real-time hair. In *Proc. of Eurographics Workshop on Rendering*, pages 243–256, 2000.

[22] T. Lokovic and E. Veach. Deep shadow maps. In *Proc. of SIGGRAPH'00*, pages 385–392, 2000.

[23] A.G. Lyne and B.F. Short. Biology of the skin and hair growth. In *Proc. of a Symposium held at Canberra, Australia*, 1964.

[24] Multibody dynamics package. http://robotics.stanford.edu/ kuffner/software. Developed by J. Kuffner and B. Mirtich.

[25] R.M. Murray, Z. Li, and S.S. Sastry. *A Mathematical Introduction to Robotic Manipulation*. CRC Press, 1994.

[26] K. Perlin. An image synthesizer. In *Proc. of SIGGRAPH'85*, pages 287–296, 1985.

[27] K. Perlin and E.M. Hoffert. Hypertexture. In *Proc. of SIGGRAPH'89*, pages 253–262, 1989.

[28] E. Plante, M.-P. Cani, and P. Poulin. A layered wisps model for simulating interactions inside long hair. In *Proceedings of Eurographics Computer Animation and Simulation*, 2001.

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

[30] M.P. Salisbury, S.E. Anderson, R. Barzel, and D.H. Salesin. Interactive pen and ink illustration. In *Proc. of SIGGRAPH'94*, pages 101–108, 1994.

[31] M.T. Scali-Sheahan. *18 Men's Styles*. Milady Publishing Company, Albany, NY, 1994.

[32] Shag (plugin for 3d studio max). home.abac.com/ddag/hair.html.

[33] Shave (plugin for lightwave). www.joealter.com/software.html.

[34] J. Stam. *Multi-Scale Stochastic Modeling of Complex Natural Phenomena*. PhD thesis, University of Toronto, 1995.

[35] D. Terzopoulos, J.C. Platt, and A.H. Barr. Elastically deformable models. In *Proceedings of SIGGRAPH'87*, pages 205–214, 1987.

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

[37] J. Wilhelms. Using dynamic analysis for realistic animation of articulated bodies. *IEEE CG&A*, 7(2):12–27, 1987.

[38] X.D. Yan, Z. Xu, J. Yang, and T. Wang. The cluster hair model. *Graphics Models and Image Processing*, 1999.

[39] K. Young. *28 Styles for Student Practice with Basic Cutting and Styling Guides*. Milady Publishing Company, Albany, NY, 1992.

[40] Y. Yu. Modeling realistic virtual hairstyles. In *Proceedings of Pacific Graphics*, pages 295–304, 2001.

[41] O.C. Zienkiewicz and R.L. Taylor. *The Finite Element Method: Solid and Fluid Mechanics Dynamics and Non-Linearity*. McGraw-Hill Book Company, 1989.