Eurographics Symposium on Rendering (2006)
Tomas Akenine-Möller and Wolfgang Heidrich (Editors)

A Novel Method for Fast and High-Quality Rendering of Hair

Songhua Xu[1,2,3†], Francis C.M. Lau[3], Hao Jiang[1,3] and Yunhe Pan[1]

[1] CAD & CG State Key Lab of China, Zhejiang University, P.R. China, 310027
[2] Department of Computer Science, Yale University, New Haven, CT, USA, 06520
[3] Department of Computer Science, The University of Hong Kong, Hong Kong, P.R. China

Abstract

This paper proposes a new rendering approach for hair. The model we use incorporates semantics-related information directly in the appearance modeling function which we call a Semantics-Aware Texture Function (SATF). This new appearance modeling function is well suited for constructing an off-line/on-line hybrid algorithm to achieve fast and high-quality rendering of hair. The off-line phase generates intermediate results in a database for sample geometries under different viewing and lighting conditions, which can be used to complete a large part of the overall computation and leaves only a few dynamic tasks to be performed on-line. We propose a model having four levels, from the whole hair volume to the very fine hair density level. We further employ an efficient disk-like structure to represent hair distributions inside a hair cluster. As the intermediate database carries opacity information, self-shadows can be easily generated. We present experiment results which clearly show that our methodology can indeed produce high quality rendering results efficiently. Supplementary materials and supporting demos can be found in our project website `http://www.cs.hku.hk/~songhua/hair-rendering/`.

Categories and Subject Descriptors (according to ACM CCS): I.3.3 [Picture/Image Generation]: Display algorithms,Viewing algorithms; I.3.5 [Computational Geometry and Object Modeling]: Curve, surface, solid, and object representations, Object hierarchies; I.3.7 [Three-Dimensional Graphics and Realism]: Color, shading, shadowing, and texture, Raytracing

1. Introduction

Hair modeling and rendering has been widely employed in various application areas of computer graphics, multimedia and computer vision. We are interested in a method that can achieve both efficiency and high-quality in hair rendering. Such a method is needed in applications including previewing during animation authoring and film design, computer games, interactive hair style design, and interactive digital painting using virtual hairy brushes [XTLP02, XTLP04]. The majority of the latest hair rendering algorithms are either designed for high quality rendering with non-interactive response or for fast/interactive rendering but with lowered quality. It seems that with currently available computing power, the goal of highly realistic hair rendering at interactive speed is still largely unrealized.

1.1. Main ideas and algorithm overview

We build appearance-related semantic information directly into the hair appearance modeling function. This appearance modeling function accepts as input hair density distributions and the viewing and lighting directions. We use both off-line and on-line steps to achieve the final high-quality rendering results with real-time performance. We introduce a four-level hair modeling hierarchy as well as a disk-like hair strand distribution model to represent the geometry and hair distribution inside a single hair cluster—they provide the underlying efficient data structure support.

With our off-line/on-line hybrid approach, most of the time-consuming rendering steps are completed in the off-line phase which generates re-usable intermediate computation results in a special purpose database. During the on-line

† Most of the work was done while Xu was at Zhejiang University. He is currently with Computer Graphics Lab, Department of Computer Science, Yale University. Contact him at songhua.xu@yale.edu.

phase, appearance maps are synthesized based on records in the database selected according to index terms provided to the appearance modeling function. After that, a mixture of Kajiya-Kay's model [KK89] and Marschner et al.'s model [MJC*03] is used to perform lightweight hair lighting calculations on the fly. A fast self-shadow generation algorithm is also designed which can further enhance the rendering realism.

The modeling of hair is via a four-level hierarchy in which a hair cluster (the primitive modeling unit) is represented internally as a generalized disk structure. This structure makes easy the generation of a density function to represent succinctly the hair strands in a cluster, which is used as the main "semantic" information for indexing the database. For the database, we compute a simplified reflectance representation of hair clusters as a function of hair density and the incoming and outgoing directions. This precomputation step only considers appearance changes due to lighting and viewing changes along the lateral directions of the hair. The database is indexed based on hair density distribution and these two azimuthal angles. During the on-line phase, this simplified precomputed representation containing reflectance and alpha values from the database is used to compute the radiance and opacity maps. The algorithm takes into account self-shadowing and shading of each layer of a hair cluster, using approximations to compute the final shading.

1.2. Contributions

Our main contributions are (1) a new appearance modeling methodology, which builds directly into the appearance modeling function appearance-related semantics of the rendering target; and (2) a hybrid off-line/on-line algorithm to achieve fast and high-quality rendering of hair and shadowing effects. Other contributions include (3) a four-level hair geometry hierarchy and a disk-like hair strand distribution modeling metaphor that capture the hair volume's highly complex geometry as well as the complicated hair strand distribution within a hair cluster; and (4) a fast shadow generating algorithm that is based on the database of hair rendering intermediate results.

1.3. Organization of the paper

The remainder of this paper is organized as follows. Sec. 2 surveys the most relevant work. Sec. 3 describes our hierarchical hair modeling approach and its associated data structure support. Sec. 4 discusses the representation of our semantics-aware texture function in a hair rendering intermediate result database. Sec. 5 discusses how to construct our rendering database in the off-line phase. Sec. 6 explains the on-line phase of our rendering algorithm. Sec. 7 presents our experiment results. Sec. 8 concludes this paper and points out directions for future work.

2. Related Work

2.1. Hair rendering

The major challenges for hair rendering are due to 1) the lighting and shadowing effects arising from the complex interactions between light rays and semi-translucent hair strands in massive number and with very tiny volumes; and 2) the aggregate lighting effect of the large number of individual hair strands, each of which having a complex geometry. Most of the past work tried to achieve the finest rendering effects but paid little attention to the response time issue [Ree83, PH89, KK89, Gol97, YXYW00, MJC*03]. There were some treatments focusing on faster algorithms [Len00, LPFH01, KHS04, Sch04]. Despite much wonderful work that was done so far, we are still short of a practical algorithm to render hair both fast (i.e., real-time response) and in high quality.

2.2. Appearance modeling

Surface reflectance modeling is critical to the accurate rendering of surface geometry. Surface appearance is subject to viewing and lighting conditions as well as the scale at which it is observed. At a coarse scale, where local surface variations are sub-pixel and local intensity is uniform, appearance is characterized by BRDFs (Bidirectional Reflectance Distribution Functions). At a fine scale, where surface variations give rise to local intensity variations, appearance can be characterized by BTFs (Bidirectional Texture Functions). A BTF can be interpreted as a mapping from a 4-D space of lighting and viewing directions to a space of 2-D texture images. A BTF dataset is a collection of images indexed by both the viewing direction and the lighting direction. In this paper, we propose yet another appearance modeling function which uses also an off-line database to render complex hair efficiently and in high quality.

2.3. Hair modeling

There are two kinds of hair models: the strand hair models and the cluster (or wisp) hair models. The strand hair models, e.g. [RCT91], attempt to model each hair strand individually. Manual modeling of every single hair strand is forbiddingly tedious. The cluster/wisp hair modeling methods, e.g. [CSDI99], group neighboring hair strands into an aggregate unit. Kim et al. [KN02] presented a multi-resolution hair model built upon a hierarchy of hair clusters. The user can edit every level of this hierarchy, including a single hair strand. In this paper we propose a four-level hierarchy and a disk-like structure for modeling highly complex hair geometries and the hair strand distributions within hair clusters.

3. Hair Modeling and Representation

Our work focuses on the rendering of hair, and less on the modeling. But the two cannot be easily separated, as it is

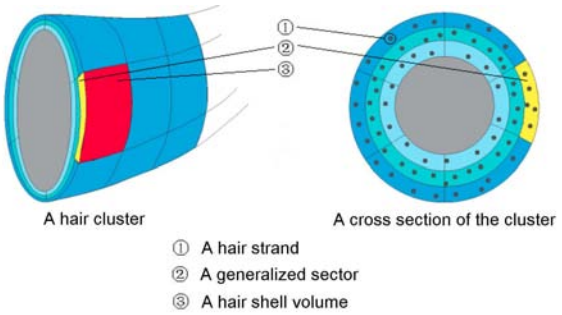

Figure 1: *A hair cluster modeled as a layered solid object with an interior core and multiple layers of volumetric shells.*

obvious that the hair model affects directly the quality and efficiency of the rendering results.

3.1. Four-level hierarchy of hair modeling

To model the hair and capture its appearance with high fidelity, we adopt a four-level hierarchy. The four levels are the entire hair volume, the hair macro-cluster level, the hair cluster level and the hair density level (hair strand level) respectively. In this hierarchy, a hair cluster (the third level) serves as our modeling primitive, which helps cut down the hair modeling cost as suffered by strand-based hair modeling approaches. Similar to the approach in [XLTP03], macro-clusters (the second level) are used to eliminate as much redundancy in hair modeling and simulation as possible; a macro-cluster groups together hair clusters whose geometries are similar but their physical positions may not be adjacent. In a hair macro-cluster, geometries of all the hair clusters can be trivially derived from each other via simple transformations. This is a reasonable strategy as there are usually only a limited number of sharply distinctive geometries among hair clusters in a hair volume. Hair macro-clusters thus make up the entire hair volume (the first level) at the top of the hierarchy. At the very bottom, we keep track of the distribution of hair strands inside a hair cluster in the form of a hair density field or an explicit representation of each hair strand's position as used in many strand-based approaches. The hair density field can be obtained from density based hair modeling tools, e.g. [XY01], or density based hair dynamics simulations, e.g. [BCN03].

3.2. Generalized disk structure for representing hair clusters

To support the four-level hierarchy, we propose a *generalized disk structure* to represent a hair cluster's envelop shape as well as the distribution of its constituent hair strands. It is called "generalized" because unlike its analogy, the real hard disk which is cylindric, the envelope of a hair cluster is a generalized cylinder.

We generate generalized cylinders through the general sweeping operation in CAD by sweeping a 2-D variable contour curve along a 3-D spline curve. Similarly to the concept of hair macro in Xu et al.'s work on realistic virtual brush [XLTP03], we sweep an ellipse along a 3-D curve to generate a generalized cylinder. During sweeping, we make sure that the ellipse always lies on the normal plane with respect to the 3-D curve. The shape of the moving ellipse can be varied during sweeping. We denote the sweeping trajectory as $T(t)(0 \le t \le 1)$, which is a B-spline in 3-D space, and the ellipse $E(t_0)$ lying on the normal plane of $T(t)|_{t=t_0}$ as:

$$\begin{aligned} E(t_0) \triangleq \{(x,y) | & x = A(t_0) v \cos\big(\theta(t_0)+u\big), \\ & y = B(t_0) v \sin\big(\theta(t_0)+u\big), (v \in [0,1], u \in [0,2\pi))\} \end{aligned} \quad (1)$$

where $A(t_0)$ and $B(t_0)$ are half of the lengths of the major and minor axes of the ellipse $E(t_0)$ respectively, and $\theta(t_0)$ is the corresponding self-twisting phase of the ellipse $E(t_0)$. Here $A(t)$, $B(t)$ and $\theta(t)$ $(0 \le t \le 1)$ are three one-dimensional B-splines. The geometry modeling parameters needed for constructing the hair cluster H can therefore be compactly stated as:

$$H \triangleq Modeling\Big(T(t), A(t), B(t), \theta(t)\Big), \ (0 \le t \le 1). \quad (2)$$

To represent the hair density field, we divide the hair cluster into two components: a heterogeneous volumetric shell part with mesostructures and a homogeneous transparent interior core (see Figure 1). The appearance of the cluster is principally determined by hair distributions in the volumetric shells. For a hair cluster, there are multiple layered shell surfaces, each of which is logically further divided uniformly into several *Hair Shell Volumes* (HSV). The division is done by connecting the neighboring points obtained by sampling the parametric equation of the generalized cylinder uniformly in the parametric domain $\{t, v, u\}$, where t $(0 \le t \le 1)$ is the relative arc length along the sweeping trajectory, with v $(0 \le v \le 1)$ and u $(0 \le u < 2\pi)$ being two parameters on the radius and phase of the sampling position as defined in (1). Thus, each of the samples is in terms of (t, v, u), and the set of all the sampling points $\mathbf{S} \triangleq \{(t_i, v_j, u_k)\}$ is collected by:

$$\begin{aligned} \mathbf{S} \triangleq & \{\tfrac{i}{s_t} | i = 0, 1, \cdots, s_t\} \times \{\tfrac{j}{s_v} | j = 0, 1, \cdots, s_v\} \\ & \times \{\tfrac{2\pi k}{s_u} | k = 0, 1, \cdots, s_u - 1\} \end{aligned} \quad (3)$$

where $\frac{1}{s_t}$, $\frac{1}{s_v}$ and $\frac{2\pi}{s_u}$ are the sampling step sizes for the three parameters respectively. In the rendering algorithm, we consider the density field within every HSV separately.

As illustrated in the right part of Figure 1, the structure of a cross section of our hair cluster model looks very much like that of a hard disk, hence the name "generalized disk structure". As shown in Figure 1, each plate (cross section) of the structure is made up of an interior core and some exterior

shells. Every cross section is divided into several "generalized tracks", each of which is further organized as multiple "generalized sectors". Notice that each HSV has two boundary generalized sectors.

3.3. Hair density field for sector

Hair strand distribution within a hair cluster is captured as a density field. The density field can be directly obtained from either a density based hair modeling tool, e.g., the V-HairStudio [XY01], or density based dynamic hair simulations, e.g., [BCN03]. In case the hair model to be rendered is not given in the form of hair density distribution, we need to first establish the density field for a generalized sector. This is usually the case when only hair strand positions are given. We assume the cross section of a hair strand is always circular and the hair's density due to the existence of a single hair strand follows a Gaussian distribution. The overall density distribution can thus be safely defined to be the sum of all these Gaussian distributions due to individual hair strands. We store values of this density field discretely at each grid point in the grid map for ease of processing later on. In the following, we use ρ to denote hair density.

4. HRIR-DB and Semantics-Aware Texture Function

Past good results are worth remembering and should be made available for future re-use. Our strategy is to execute as much of the time-consuming rendering tasks off-line as possible and store the intermediate results as *Hair Rendering Intermediate Result* (HRIR) records in a rendering database (HRIR-DB).

For the current HRIR-DB, we only attempt to model different rendering effects brought about by the changes of the illumination conditions along the lateral direction of hair. Illumination conditions along the lengthwise direction of hair would only affect the global lengthwise rendering result, which can be efficiently calculated on-line. To be more specific, an HRIR represents the appearance of a generalized sector, which is a boundary cross section of an HSV with a certain density field defined inside, under a certain viewing and lighting setting. The viewing and lighting directions are both 1-D angles confined in the plane that contains the generalized sector. Each HRIR is a 1-D signal sequence that carries two kinds of information: reflectance values and opacity (alpha) values.

Compared with storing a 2-D reflectance map and an alpha map for a small volume of the hair shell, our 1-D HRIR record format for a cross section of an HSV contributes significantly to miniaturization of the HRIR-DB. This downsizing is very important for it allows the possible loading of the entire HRIR-DB into the main memory for carrying out the fast on-line rendering process.

The HRIR-DB in fact serves as the range of our newly proposed appearance modeling function—*Semantics-Aware Texture Function* (SATF)—which accepts as parameters the key distinguishing features of the rendering target. In our present context, the rendering target is a hair cluster, and its semantics are represented compactly by a hair density distribution. Intuitively, our SATF is a database lookup based on the input parameters of the hair density distribution of the rendering target and the incoming and outgoing lighting directions. Thus, mathematically, SATF defines the following mapping relationship:

$$SATF(\beta, \mathbf{v}, \mathbf{l}) = HRIR\big(\rho_{feature}(\beta), \theta_v(\beta), \theta_l(\beta)\big) \qquad (4)$$

where $\theta_v(\beta)$ and $\theta_l(\beta)$ are the viewing and lighting directions, respectively, or namely, the azimuth angles defined in the normal plane of the HSV, and $\rho_{feature}(\beta)$ is a hair density distribution feature vector. The subscript "feature" of ρ indicates only selected features related to density distribution are used in the process, resulting in a much reduced number of dimensions for the database records.

The HRIR-DB is similar to a BTF database in that both of them contain image samples produced under different viewing and lighting settings. The important difference is, our method includes additionally an appearance-related parameter based on the density distribution of the generalized sector ($\rho_{feature}(\beta)$). Therefore, a record in the database is the salient appearance map of the rendering object, and not merely a texture map for a certain fixed geometry pattern.

5. Constructing the Database of Hair Rendering Intermediate Results

To construct the HRIR-DB, we first obtain a collection of samples of generalized sectors, $\{\beta_i\}$, by initializing control parameters of a generalized sector β using random values. For each of these generalized sector samples, two major tasks are performed for establishing the HRIR-DB: 1) to compute the HRIRs for β under a number of typical viewing and lighting directions (Sec. 5.1); these directions are all confined to the plane on which β lies; 2) to compute the hair density distribution feature vector $\rho_{feature}(\beta)$ for β, which is used as an index when storing these HRIRs arising from β in the database. The top part of Figure 2 summarizes the overall process of the HRIR-DB establishment.

5.1. Deriving an HRIR record

We introduce a "planar ray tracing" procedure to compute *HRIR* for the given input generalized sector β with the viewing direction $\mathbf{v}$ and lighting direction $\mathbf{l}$. The difference between "planar ray tracing" and those normal ray tracing procedures is that both the viewing and lighting directions are confined on the plane that the rendering target β lies on; and the rendering result *HRIR*, namely the hair appearance for β, is a 1-D signal rather than a 2-D image. We modify the ray tracing procedure for a volume as suggested in [YXYW00]

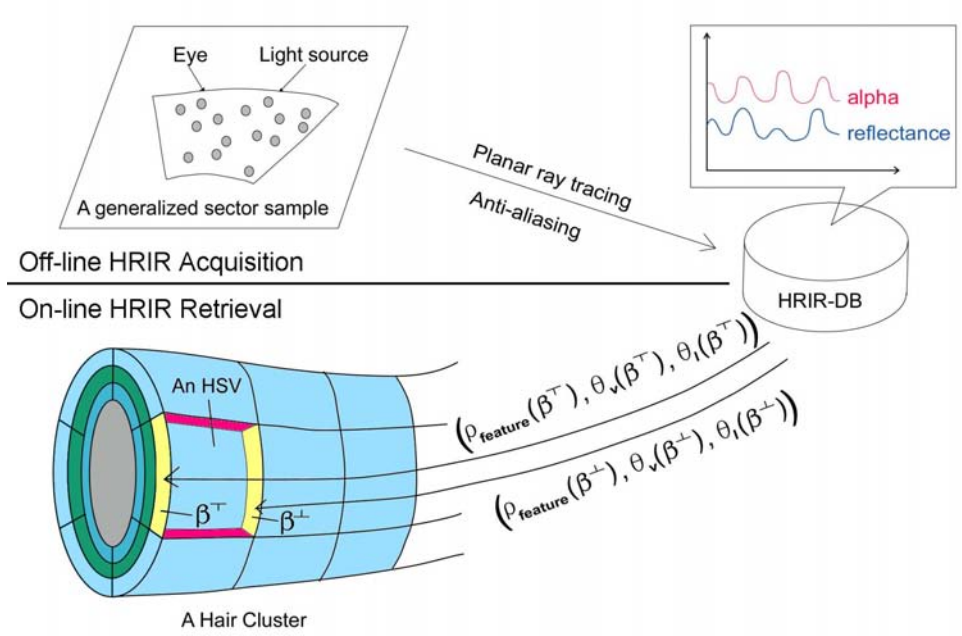

Figure 2: *HRIR-DB: In the off-line rendering phase, generalized sectors of different density distributions are ray-traced under various viewing and lighting directions to generate HRIRs for establishing the HRIR-DB; in the on-line rendering phase, HRIRs are retrieved according to the index vector.*

into a planar version and carry out the procedure on β. After tracing all the rays and recording all the accumulated reflectance and opacity in their corresponding locations, we obtain an array of reflectance and an array of alpha values. These two arrays constitute one single hair rendering intermediate result (HRIR) record.

More formally, when tracing a ray R_j, we denote the k-th grid point in the grid map (through which the hair density field is discretely recorded (Sec. 3.3)) met by R_j as $\eta_{j,k}$. We compute $\eta_{j,k}$'s opacity $\alpha(\eta_{j,k})$ as a value proportional to the local hair density $\rho(\eta_{j,k})$, and $\eta_{j,k}$'s reflectance $re(\eta_{j,k})$ using the Phong illumination model with the normal direction of hair strand surface approximated as the gradient of the local hair micro-density, i.e., $\mathbf{n}(\eta_{j,k}) \approx \nabla\rho(\eta_{j,k})$. During our planar ray tracing, after hitting R_j with $\eta_{j,k}$, we update R_j's accumulated reflectance $\widehat{re}(R_j)$ and its accumulated opacity $\widehat{\alpha}(R_j)$ by (5).

$$\begin{cases} \widehat{re}(R_j) = \widehat{re}(R_j) + \alpha(\eta_{j,k}) \times re(\eta_{j,k}) \times \left(1 - \widehat{\alpha}(R_j)\right) \\ \widehat{\alpha}(R_j) = \widehat{\alpha}(R_j) + \alpha(\eta_{j,k}) \times \left(1 - \widehat{\alpha}(R_j)\right) \end{cases} \tag{5}$$

We also apply a visibility test with the accumulated opacity of a ray such that our algorithm stops tracing R_j when either the ray's accumulated opacity $\widehat{\alpha}(R_j)$ becomes very close to 1, or it has penetrated the rendering target, i.e., β. After tracing all the rays, we record all the penetrating rays' accumulated reflectances $\widehat{re}(R_j)$'s and opacities $\widehat{\alpha}(R_j)$'s at their corresponding pixel locations to establish a 1-D array of reflectance values and another 1-D array of alpha values. They form the *HRIR*.

Since it is normal for a hair strand to cover only a fraction of a pixel, severe aliasing artifacts may arise. To overcome this, the above computed HRIR will be smoothed using a 1-D Gaussian kernel. The distribution feature parameter needed by the Gaussian kernel is set according to the distribution feature used when establishing the hair density field (Sec. 3.3). In our experiment, we empirically find that a ratio of 1.5 times between them would lead to visually most satisfying results.

5.2. Indexing an HRIR record

As outlined at (4), the complete index term for storing $SATF(\beta, \mathbf{v}, \mathbf{l})$ in the HRIR-DB is $\left(\rho_{feature}(\beta), \theta_v(\beta), \theta_l(\beta)\right)$. The schema is illustrated in the bottom part of Figure 2. The feature vector $\rho_{feature}(\beta)$ consists of three terms in our design, i.e., $\rho_{feature}(\beta) \triangleq \left(\overline{\rho}_{st}(\beta), \overline{\rho}_s(\beta), \overline{\rho}_t(\beta)\right)$ where $\overline{\rho}_{st}(\beta)$, $\overline{\rho}_s(\beta)$, $\overline{\rho}_t(\beta)$ are β's average hair density, s-dimensional and t-dimensional projected average hair density respectively. The average hair density of β, $\overline{\rho}_{st}(\beta)$, can be trivially computed since the hair density distribution within β is either initially given or derived in the preprocessing step (Sec. 3.3). To compute $\overline{\rho}_s(\beta)$ and $\overline{\rho}_t(\beta)$, assume $\{s,t\}$ is the 2-D parametric coordinate system defined over β. We first derive the den-

sity distribution histogram curves versus the s and t axes and then compute the average values of the derived histogram curves to be $\overline{\rho}_s(\beta)$ and $\overline{\rho}_t(\beta)$ respectively.

6. Fast and High Quality On-line Hair Rendering

6.1. Main steps of on-line hair rendering

As mentioned in Sec. 3, hair clusters are rendering primitives in our algorithm. Given a hair cluster with a certain hair density field, we can render its appearance under arbitrary viewing and illumination directions efficiently and realistically with the support of the off-line acquired HRIR-DB. Recall in Sec. 3.2, a hair cluster consists of a transparent interior core and an exterior ring volume, the latter of which is divided into multiple layers. By tessellating the outer surface of each of these layers, we obtain the layer's associated mesh. In the following, we use $P_{(u,v),k}$ to denote the point on the mesh for the k-th layer of the hair cluster, whose coordinate in the parameterized texture space of the mesh is (u,v) (see Figure 1). It is easy to see $P_{(u,v),k}$ and $P_{(u,v),k+1}$ have the same texture coordinates in their respective meshes.

The main steps of our on-line hair rendering are as follows, which are illustrated in Figure 3. For each layer of the mesh, we use the local hair density distribution and the current viewing and lighting directions as input of the SATF to construct a *re-map* and an α-*map* (Sec. 6.2). For each point on a layer we compute a lighting term for it, with the anisotropy of hair appearance taken into account (Sec. 6.3). And then we simulate the self-shadowing effect by utilizing the computed α-*map*s and organize the resultant shadow values as a shadow map (*s-map*) (Sec. 6.4). Finally we derive the shading for each layer by modulating all the lighting terms belonging to the layer using corresponding reflectance values and shadow values recorded at the layer's *re-map* and *s-map* respectively. The final overall shading for the hair cluster can be generated by blending together individual shadings of the layers according to their α-*map*s (Sec. 6.5).

6.2. SATF and *re-* and α-*map* construction

The first step of our on-line rendering is to construct the *re-map*s and α-*map*s for each layer in the hair cluster via the SATF. We realize this by first constructing local *re-map*s and α-*map*s for each HSV separately, and then patching up all these maps associated with the different HSVs in the same layer. There are two sub-tasks when constructing a local *re-map* and a local α-*map* for a given HSV: (1) to compute the parameters for the HSV's two boundary generalized sectors to retrieve the best matched HRIR(s) from the HRIR-DB; and (2) according to the extracted HRIR(s), to construct the *re-map* and the α-*map* for the HSV.

6.2.1. Calculating the SATF parameters for HSV

Each HSV is associated with two boundary generalized sectors, one at the top, $\beta^{\top}$, and one at the bottom, $\beta^{\perp}$ (see Figure 2). For each of them, say β, we calculate a hair density distribution feature vector $\rho_{feature}(\beta)$ (see Sec. 5.2). The SATF parameters for β are completed by putting together the current viewing and lighting directions w.r.t. β, namely $Params(\beta) = \big(\rho_{feature}(\beta), \theta_v(\beta), \theta_l(\beta)\big)$.

With both $Params(\beta^{\top})$ and $Params(\beta^{\perp})$ calculated as described in the above, we check whether the rendering context of the HSV—hair density distribution inside the HSV as well as the lighting and viewing directions across the HSV—is relatively uniform or not. It is considered uniform if (6) holds.

$$||Params(\beta^{\top}) - Params(\beta^{\perp})|| < \kappa^{threshold} \tag{6}$$

6.2.2. Constructing *re-map* and α-*map* for HSV

If the rendering context across the HSV is considered uniform, we simply use $Params_{ave} \triangleq \frac{Params(\beta^{\top})+Params(\beta^{\perp})}{2}$ to retrieve the best matched HRIR record, $HRIR(Params_{ave})$, from the HRIR-DB. Recall that an HRIR carries two 1-D arrays: a *re-array* and an α-*array* (Sec. 4). In this situation, we can construct the local *re-map* and α-*map* for the HSV by simply sweeping the *re-array* and α-*array* along the lengthwise direction of the HSV.

If by (6) the rendering condition across the HSV is non-uniform, we would fetch two HRIR records, $HRIR_1 = SATF\big(Index(\beta^{\top})\big)$ and $HRIR_2 = SATF\big(Index(\beta^{\perp})\big)$. The local *re-map* and α-*map* for the HSV can then be generated by interpolating between $HRIR_1$ and $HRIR_2$. Assume $HRIR_1 \triangleq \{c_{1,1}, c_{1,2}, \cdots, c_{1,n}\}$ and $HRIR_2 \triangleq \{c_{2,1}, c_{2,2}, \cdots, c_{2,n}\}$, where $c_{i,j}$ is either the j-th reflectance value or the j-th alpha value carried in $HRIR_i$ $(i = 1,2; 0 < j \leq n)$. Then the intermediate value can be derived through a linear interpolation:

$$\lambda(i,\gamma) = c_{1,i} \times (1-\gamma) + c_{2,i} \times \gamma \tag{7}$$

where γ is the interpolation parameter, which indicates the relative position of the intermediate value between $HRIR_1$ and $HRIR_2$. This simple linear interpolation might not appear to be good enough to generate a smooth looking texture transition. Our experiment results, however, show that this simple method can indeed yield visually satisfying results. This could be due to the following two reasons: 1) local hair lighting conditions and density distributions change continuously, and so the difference in reflectance and alpha values between two adjacent HRIRs is limited; 2) as long as the surface of a hair cluster is not too severely undersampled, the sufficient number of sample points would reduce the texture difference between corresponding sample points in the two HRIRs.

We construct *re-map*s and α-*map*s for all the layers in the generalized disk, which are to be used for deriving shading for these meshes on the fly (Sec. 6.5). In the following, we use $re_{(u,v),k}$ and $\alpha_{(u,v),k}$ to denote the reflectance and opacity

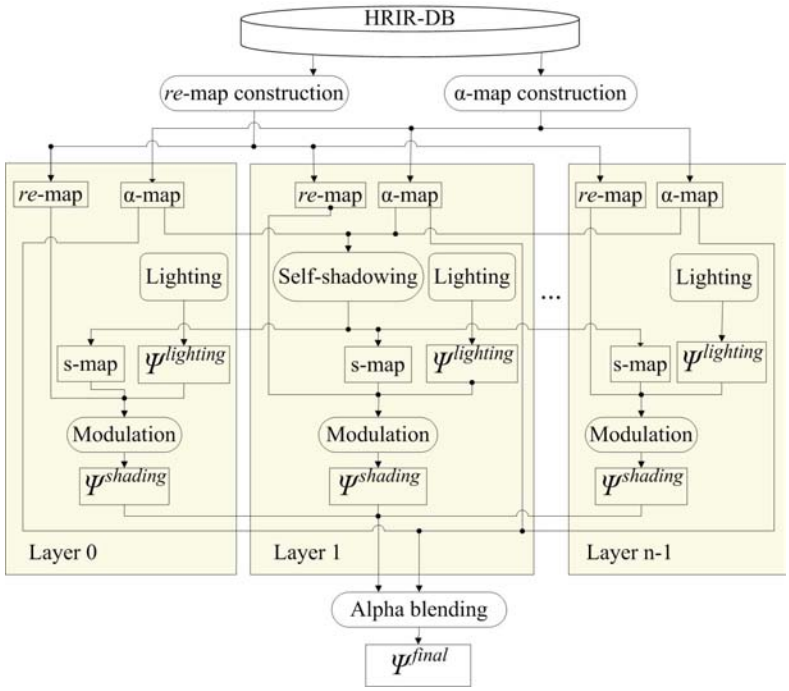

Figure 3: *Our on-line phase rendering algorithm: For each layer of the cluster shell, first a reflectance map and an alpha map are constructed procedurally from the HRIR-DB according to the hair density field within the layer and the current viewing and illumination directions. And then a shadow map is computed by our fast self-shadowing algorithm. Having prepared all the rendering maps, our algorithm performs shading for each layer of the mesh. For each pixel in a certain layer of the mesh, lighting is simulated. The resultant lighting term is modulated by the reflectance values sampled from the reflectance map and the light transmittance sampled from the shadow map to calculate the RGBA value of the pixel. Finally, all the layers of meshes are blended together according to the alpha maps to generate the final rendering image.*

of $P_{(u,v),k}$ recorded in the reflectance map and alpha map of the k-th layer, namely *re-map*$_k$ and α-*map*$_k$, respectively.

6.3. On-line hair lighting

Similar to the practice in [Sch04], our hair rendering pays special attention to the anisotropic characteristics of hair appearance.

For each $P_{(u,v),k}$, we evaluate a diffuse term $\Psi^{dif}_{(u,v),k}$, a primary specular term $\Psi^{speI}_{(u,v),k}$, and a secondary specular term $\Psi^{speII}_{(u,v),k}$. That is, in our system, the on-line hair lighting term $\Psi^{lighting}_{(u,v),k}$ for $P_{(u,v),k}$ is the overall effect of the above three terms:

$$\Psi^{lighting}_{(u,v),k} \triangleq \Psi^{dif}_{(u,v),k} + \Psi^{speI}_{(u,v),k} + \Psi^{speII}_{(u,v),k}. \tag{8}$$

Like [KK89], we compute the diffuse term of the lighting model as:

$$\Psi^{dif}_{(u,v),k} \triangleq \kappa^{dif} \sin(\mathbf{t}_{(u,v),k}, \mathbf{l}_{(u,v),k}) \phi^{dif} \tag{9}$$

where κ^{dif} is the diffuse coefficient; $\mathbf{t}_{(u,v),k}$ and $\mathbf{l}_{(u,v),k}$ are the tangent direction and the lighting direction at the point $P_{(u,v),k}$ respectively; $\phi^{dif} = (\phi^{dif}.r \;\; \phi^{dif}.g \;\; \phi^{dif}.b)^T$ is the diffuse color of the hair in the form of RGB values. Notice: the operator $\sin(\mathbf{X}, \mathbf{Y})$ here calculates the sine value of the angle spanned by the two vectors $\mathbf{X}$ and $\mathbf{Y}$.

Following Marschner et al.'s work [MJC*03], we simulate two highlight regions at our on-line hair lighting stage: one is the primary region which is shifted towards the tip part of the hair and the other is a secondary region which is shifted towards the root part of the hair. To evaluate the specular term for the primary highlight, we employ (10), which is a modified version of [KK89]'s specular equation.

$$\begin{cases} \Psi^{speI}_{(u,v),k} \triangleq \kappa^{speI} \left((\tilde{\mathbf{t}}_{(u,v),k} \cdot \mathbf{l}_{(u,v),k})(\tilde{\mathbf{t}}_{(u,v),k} \cdot \mathbf{v}_{(u,v),k}) + \right. \\ \qquad\qquad \left. \sin(\tilde{\mathbf{t}}_{(u,v),k}, \mathbf{l}_{(u,v),k}) \sin(\tilde{\mathbf{t}}_{(u,v),k}, \mathbf{v}_{(u,v),k}) \right)^{\kappa^h} \phi^{speI} \\ \tilde{\mathbf{t}}_{(u,v),k} \triangleq \frac{\mathbf{t}_{(u,v),k} + s_{(u,v),k} \times \mathbf{n}_{(u,v),k}}{||\mathbf{t}_{(u,v),k} + s_{(u,v),k} \times \mathbf{n}_{(u,v),k}||} \end{cases} \tag{10}$$

Here κ^{speI} is the primary specular coefficient. κ^h is the specular Phong exponent specifying the sharpness of the highlight. $\mathbf{l}_{(u,v),k}$ and $\mathbf{v}_{(u,v),k}$ are the lighting and viewing vectors at $P_{(u,v),k}$ respectively. $\phi^{speI} = (\phi^{speI}.r \;\; \phi^{speI}.g \;\; \phi^{speI}.b)^T$ is

the primary specular color of the hair, which is same as the color of the light source because primary highlight is mainly due to the light reflected off the hair surface. $\tilde{\mathbf{t}}_{(u,v),k}$ is the perturbed version of $\mathbf{t}_{(u,v),k}$, which is introduced here to shift the primary highlight region towards the tip of the hair by adding a small portion of the normal vector of the cluster surface, $\mathbf{n}_{(u,v),k}$, onto the true hair tangent direction, $\mathbf{t}_{(u,v),k}$. Such a primary highlight region shift process is controlled by a positive parameter $s_{(u,v),k}$.

The secondary specular term is computed through a very similar equation, as follows.

$$\begin{cases} \Psi^{speII}_{(u,v),k} \triangleq \kappa^{speII}\big((\tilde{\mathbf{t}}_{(u,v),k}\cdot\mathbf{l}_{(u,v),k})(\tilde{\mathbf{t}}_{(u,v),k}\cdot\mathbf{v}_{(u,v),k})+ \\ \qquad \sin(\tilde{\mathbf{t}}_{(u,v),k},\mathbf{l}_{(u,v),k})\sin(\tilde{\mathbf{t}}_{(u,v),k},\mathbf{v}_{(u,v),k})\big)^{\kappa^h}\phi^{speII} \\ \tilde{\mathbf{t}}_{(u,v),k} \triangleq \frac{\mathbf{t}_{(u,v),k}-s_{(u,v),k}\times\mathbf{n}_{(u,v),k}}{||\mathbf{t}_{(u,v),k}-s_{(u,v),k}\times\mathbf{n}_{(u,v),k}||} \end{cases} \quad (11)$$

Here $\phi^{speII}=(\phi^{speII}.r\ \ \phi^{speII}.g\ \ \phi^{speII}.b)^T$ is the secondary specular color of the hair, which is the same as the color of the hair because the secondary highlight is mainly caused by the light transmitted into the hair strands and its reflection. κ^{speII} is the secondary specular coefficient. This time, $\tilde{\mathbf{t}}_{(u,v),k}$ serves as a perturbed version of the hair's tangent direction to shift the secondary highlight region towards the root of the hair by deducting a small portion of the normal vector of cluster surface, $\mathbf{n}_{(u,v),k}$, from the true hair tangent direction, $\mathbf{t}_{(u,v),k}$. Such a highlight region shift process is also controlled by a positive parameter $s_{(u,v),k}$.

6.4. On-line hair self-shadowing

During on-line hair rendering, we compute two kinds of hair self-shadows: local shadows within a hair cluster and global shadows among multiple hair clusters. To simulate global shadows among hair clusters, we employ the shadow volume technique [Cro77]. In the following, we explain how we simulate local shadows within a hair cluster.

For local shadows within a hair cluster, complicated interactions between light rays and hair volume need to be considered. Kajiya and Hersen [KK89] first applied ray tracing to compute self-shadows during hair rendering. Later researchers proposed methods based on shadow maps [LV00, KN01, AL04]; the same idea was recently exploited by [MKBR04, KHS04] to achieve fast rendering with hair self-shadowing utilizing the power of the GPU.

Inspired by shadow map techniques, we use shadow maps (*s-map*s) to record results of simulated self-shadowing effects, one for each layer. In *s-map*$_k$, the *s-map* for the k-th layer, we store transmittance of light $\tau_{(u,v),k}$ at point $P_{(u,v),k}$, which approximates the portion of external light penetrating the hair volume to reach $P_{(u,v),k}$. To calculate $\tau_{(u,v),k}$ precisely, all the points that are passed through by the same light ray as $P_{(u,v),k}$ need to be identified, which is computationally expensive. Current shadow map based approaches, e.g. "opacity shadow map" [KN01], avoid this expense through a separate pass to render the scene from the light's point of view to prepare the shadow maps. This separate pass however is in any case a non-zero computation load, which is still undesirable for a fast rendering algorithm. Our method completely does away with this additional separate pass by taking advantage of the property of our layered generalized disk structure for hair clusters: self-shadows are always cast from the outer layers onto the inner layers. Since human eyes are not highly sensitive to the accuracy of shadowing, we introduce a simple and fast method to simulate the shadowing effect which gets darker as we move from the outer layers to the inner layers. Therefore, in our method, $\tau_{(u,v),k}$ is estimated as:

$$\begin{aligned} \tau_{(u,v),k} &\triangleq e^{-\sum_{m\in\chi(k)}\kappa^{shadow}\times\alpha_{(u,v),m}} \\ &\approx 1-\kappa^{shadow}\times\textstyle\sum_{m\in\chi(k)}\alpha_{(u,v),m}. \end{aligned} \quad (12)$$

Here $\chi(k) \triangleq \{m|\text{layer } L_m \text{ covers layer } L_k, \text{namely } L_m \text{ is an outer layer w.r.t. } L_k\}$, κ^{shadow} is a self-shadow intensity parameter controlling the darkness of self-shadowing, and $\alpha_{(u,v),m}$ is the light attenuation of $P_{(u,v),m}$ as recorded at α-*map*$_m$.

Compared with recent shadow map approaches, our shadow map generation method is very fast because it has a computationally trivial shadow map preparation process in which we only need to sample α-*map*s to read out alpha values with the same texture coordinates. Although our method is not theoretically exact, practically it can achieve visually satisfying results, as can be seen in the example in Figure 4; this is due to the fact that by design the HSVs in our generalized disk structure are not very large, and hence the hair layers are not very thick.

6.5. Deriving shading through integrating all the rendering effects together

By integrating all the rendering effects obtained in the previous steps, we derive the shading for each layer. For $P_{(u,v),k}$, its overall shading result, $\Psi^{shading}_{(u,v),k}$, in terms of RGB value is the product of its lighting term, $\Psi^{lighting}_{(u,v),k}$, its reflectance term, $re_{(u,v),k}$, as recorded in *re-map*$_k$, and the light transmittance term, $\tau_{(u,v),k}$, as recorded in *s-map*$_k$:

$$\Psi^{shading}_{(u,v),k} \triangleq \Psi^{lighting}_{(u,v),k}\times re_{(u,v),k}\times\tau_{(u,v),k}. \quad (13)$$

With $\Psi^{shading}_{(u,v),k}$, the RGBA value for $P_{(u,v),k}$ is extracted as $(\Psi^{shading}_{(u,v),k}.r, \Psi^{shading}_{(u,v),k}.g, \Psi^{shading}_{(u,v),k}.b, \alpha_{(u,v),k})$. Once the shading results for all the layers are calculated, we can derive the final hair rendering image Ψ^{final} by blending the shading results of all the layers through a standard alpha blending process:

$$\begin{aligned} \Psi^{final}_{(u,v)} \triangleq\ & \Psi^{bk}_{(u,v)}\times\textstyle\prod_{k=0}^{n-1}\left(1-\alpha_{(u,v),k}\right)+ \\ & \textstyle\sum_{k=0}^{n-1}\left(\Psi^{shading}_{(u,v),k}\times\alpha_{(u,v),k}\times\prod_{j=k+1}^{n-1}\left(1-\alpha_{(u,v),j}\right)\right). \end{aligned} \quad (14)$$

(a) (b) (c) (d)

Figure 4: *Comparison of rendering results: (a) rendering result produced by Scheuermann's algorithm; (b) result produced by our algorithm with secondary highlight effect disabled; (c) result produced by our algorithm with self-shadowing effect disabled; (d) result produced by our algorithm with all the effects enabled. A comparison between rendering results of a human hair model is shown at Fig. 5.*

Here $\Psi^{final}_{(u,v)} = (\Psi^{final}_{(u,v)}.r \quad \Psi^{final}_{(u,v)}.g \quad \Psi^{final}_{(u,v)}.b)^T$ is the RGB value in the final hair rendering image that is in correspondence to the eye ray $R_{(u,v)}$. $\Psi^{bk}_{(u,v)} = (\Psi^{bk}_{(u,v)}.r \ \Psi^{bk}_{(u,v)}.g \ \Psi^{bk}_{(u,v)}.b)^T$ is the RGB value of the penetrating light from the interior core on $R_{(u,v)}$, which is taken from the background color.

7. Experiment Results

We implemented our algorithm using Microsoft Visual C++ 6.0 and with the support of Microsoft Direct3D V9.0 on a PC with a Pentium 4 3.0GHz processor, 1G main memory and an NVIDIA GeForce 6600 GT graphics card. In our implementation, we take advantage of the current programmable graphics cards' computing power by executing most of the on-line rendering tasks through pixel shaders and vertex shaders.

We show the rendering results of four hair clusters with different geometries and density distributions in Figure 6, and four hair models on a human head in Figure 7. A partial sequence of hair animation results are shown in Figure 8. Table 1 reports some statistics of all these examples and the time needed to fulfill various stages of our on-line phase of the rendering algorithm, which include estimating the hair density distribution parameter for the SATF (Index calculation); computing the SATF, i.e., HRIR-DB lookup, and the *re*/α-*map* construction (map Construction); texture mapping the constructed *re*/α-*map*s onto the generalized disk structure (texture Mapping); on-line hair lighting (Lighting); and on-line hair shadowing (Shadowing). For the hair animation results (Figure 8), the statistics reported are the average performance. The performance is clearly of real-time quality when rendering a single hair cluster and of interactive quality when rendering a human hair model.

Our approach to achieve the high quality hair rendering effects can be seen as similar to that of Kajiya-Kay for hair modeling and rendering via volume densities [KK89]. We added three features: (I) varying visual effects to reflect changes in hair distributions as well as the viewing and lighting directions; (II) shifted secondary highlight effect as proposed by Marschner et al. [MJC*03]; and (III) self-shadowing effect. We compare the rendering results produced by our algorithm with those by Scheuermann's algorithm [Sch04]. We choose Scheuermann's for comparison because among all the existent work, his algorithm is the closest to ours in that his algorithm is also based on Kajiya-Kay's; also both our and Scheuermann's algorithms are designed for the purpose of efficient hair rendering. But Scheuermann's algorithm has incorporated only the effect (II) above. Figure 4 shows the comparison results, where (a) is Scheuermann's result and (d) is our result. In comparison, (a) looks flat and lacks the sense of stereo because his algorithm does not simulate the effects (I) and (III) in the above. The figure ((b) and (c)) also shows how the effects (II) and (III) would enhance the realism of the rendered result. Figure 5 compares the two algorithms with a complete hair model.

8. Conclusion and Discussion

Our proposed rendering algorithm is based on a database of precomputed intermediate results, which can achieve interactive rates in hair rendering without lowering the image quality. The mapping from input parameters to the precomputed hair rendering intermediate results forms our semantics-aware texture function (SATF), which is a form of parameterized BTF. The dimensionality involved in the mapping is reduced by making some simplification assumptions concerning visual aspects. These intermediate results are then used to compute for each hair strand a *re-map* and an α-*map*. These maps are further used to compute images including different lighting effects, which are then combined to form a single final rendering result. For a volumet-

(a) (b)

Figure 5: *Comparison of rendering results of a human hair model: (a) rendering result produced by Scheuermann's algorithm; (b) result of human hair by our algorithm.*

Table 1: *Statistics of hair rendering shown in Figures 6, 7 and 8 including the total number of hair clusters (C#), layers (L#), hair strands (S#) in K and HSVs (H#) in K in the hair model. After the H# row, each table entry presents the computation time in milliseconds consumed by each step of the on-line phase of our rendering algorithm, i.e., index calculation (I), map construction (C), texture mapping (M), online lighting (L), shadowing (S) and the overall frames-per-second rate (FPS) achieved by our rendering algorithm.*

Fig.	6(a)	6(b)	6(c)	6(d)	7(a)	7(b)	7(c)	7(d)	8
C#	1	1	1	1	17	12	4	6	40
L#	15	15	10	15	51	60	60	60	80
S#	3	3	2	3	14	20	20	9	21
H#	16	16	11	16	54	61	127	63	42
I	5	5	4	5	28	36	35	29	36
C	29	28	14	31	44	39	61	36	43
M	2.6	1.7	1.9	2.0	4.5	6.9	3.5	5.2	4.3
L	0.3	0.2	0.1	0.2	0.2	0.5	0.3	0.7	0.6
S	0.2	0.1	0.1	0.2	0.2	0.5	0.3	0.3	0.4
FPS	27	28	42	26	13	12	10	14	12

ric model of a generalized cylinder, the rendering treats hair strand meshes like shells, projecting them from the innermost ring to the outer-most one.

Comparing with conventional appearance modeling approaches such as BRDF and BTF, which only model the light transmission under certain viewing and lighting conditions for fixed material or a fixed geometry pattern, our new approach can be interpreted as a way to model the aggregate appearance of the target as a special kind of virtual "material appearance property". By such an aggregate virtual material appearance property, the challenge of rendering the highly sophisticated geometry of the target object can be reduced to applying an image based rendering over a much smoother and simplified surface geometry followed by some computationally inexpensive on-line lighting and shadowing procedures.

Our off-line/on-line rendering algorithm is also related to image based rendering (IBR) [MB95, LH96, GGSC96]. But unlike IBR, whose rendering capability derives mainly from the information captured in the image database, by including appearance-related semantics features in the modeling function and the extension of a lightweight and effective on-line rendering phase, our rendering algorithm can perform much more comprehensively in terms of its support for dynamically changing rendering scenes. In fact, our work can be viewed as a special image-based rendering algorithm capable of responding to dynamic changes in the target.

In our design of the on-line/off-line two-phase rendering algorithm, to capture the reflectance, we split the computation task into two parts: in the off-line process we only consider appearance changes arising due to lighting and viewing changes along the lateral directions of the hair; in the online phase we focus on reproducing appearance changes due to lighting conditions along the lengthwise direction of the hair. The split is based on the following regarding the effects caused by lighting conditions. To capture the effects caused by different lighting conditions in the lateral direction, we need to take into account the delicate hair geometry and its tiny volume, distribution, and positions, which is the most time consuming part of the rendering work. In comparison, the effects caused by different lengthwise direction lighting conditions are much easier to compute, which only needs to be operated at the level of hair cluster geometry. The split leads to much time saving in the online phase. A secondary benefit is that this can narrow the spectrum of HRIR samples that the HRIR-DB is supposed to capture, giving rise to a database that is more compact in size.

Currently, most of the hair we rendered are of the straight style. To support curly hair rendering, we will need to introduce additional semantics in our current hair appearance modeling function. Developing a better user interface for creating and customizing curly hair styles is a highly practical goal.

The research work presented in this paper is an initial attempt to build semantics features directly into the appearance modeling function. These features are density related for the case of hair, and because of that, we believe the same methodology can be applied to the modeling and rendering of other objects such as smoke, cloud, grass, fire, and the like. Our success in designing the fast off-line/on-line hybrid algorithm has to do with the capturing of appearance-related semantics. For hair as well as grass, cloud, smoke, etc., their appearance under certain viewing and lighting conditions is mostly determined by their density distribution. Thus we can categorize density distribution features for these targets as semantics-related information and build them into the ap-

pearance model. An off-line/on-line algorithm can then be designed, just like what has been done in this paper.

Acknowledgements

We appreciate the many constructive suggestions of the anonymous reviewers which helped improve the presentation of this paper. Chao Tian participated at an early stage of this research. Jiefeng Jiang provided a hair animation prototype system for our rendering experiment. This research is supported in part by the grant "Intelligent systems for painting and calligraphy" from RGC, Hong Kong. Songhua Xu is supported in part by a fellowship from Yale University.

References

[AL04] AILA T., LAINE S.: Alias-free shadow maps. *Proceedings of Eurographics Symposium on Rendering* (2004), 161–166.

[BCN03] BANDO Y., CHEN B.-Y., NISHITA T.: Animating hair with loosely connected particles. *Computer Graphics Forum (Proceedings of Eurographics) 22*, 3 (2003), 411–418.

[Cro77] CROW F. C.: Shadow algorithms for computer graphics. *Proceedings of ACM SIGGRAPH Conference* (1977), 242–248.

[CSDI99] CHEN L., SAEYOR S., DOHI H., ISHIZUKA M.: A system of 3d hairstyle synthesis based on the wisp model. *The Visual Computer 15*, 4 (1999), 159–170.

[GGSC96] GORTLER S. J., GRZESZCZUK R., SZELISKI R., COHEN M. F.: The lumigraph. *Proceedings of ACM SIGGRAPH Conference* (1996), 43–54.

[Gol97] GOLDMAN D. B.: Fake fur rendering. *Proceedings of ACM SIGGRAPH Conference* (1997), 127–134.

[KHS04] KOSTER M., HABER J., SEIDEL H.-P.: Real-time rendering of human hair using programmable graphics hardware. *Proceedings of Computer Graphics International* (2004), 248–256.

[KK89] KAJIYA J. T., KAY T. L.: Rendering fur with three dimensional texures. *Proceedings of ACM SIGGRAPH Conference* (1989), 271–280.

[KN01] KIM T.-Y., NEUMANN U.: Opacity shadow maps. *Proceedings of Eurographics Workshop on Rendering Techniques* (2001), 177–182.

[KN02] KIM T.-Y., NEUMANN U.: Interactive multiresolution hair modeling and editing. *ACM Transactions on Graphics 21*, 3 (2002), 620–629.

[Len00] LENGYEL J.: Real-time fur. *Proceedings of Eurographics Rendering Workshop* (2000), 243–256.

[LH96] LEVOY M., HANRAHAN P.: Light field rendering. *Proceedings of ACM SIGGRAPH Conference* (1996), 31–42.

[LPFH01] LENGYEL J., PRAUN E., FINKELSTEIN A., HOPPE H.: Real-time fur over arbitrary surfaces. *Proceedings of ACM Symposium on Interactive 3D Graphics* (2001), 227–232.

[LV00] LOKOVIC T., VEACH E.: Deep shadow maps. *Proceedings of ACM SIGGRAPH Conference* (2000), 385–392.

[MB95] MCMILLAN L., BISHOP G.: Plenoptic modeling: an image-based rendering system. *Proceedings of ACM SIGGRAPH Conference* (1995), 39–46.

[MJC*03] MARSCHNER S. R., JENSEN H. W., CAMMARANO M., WORLEY S., HANRAHAN P.: Light scattering from human hair fibers. *Proceedings of ACM SIGGRAPH Conference* (2003), 780–791.

[MKBR04] MERTENS T., KAUTZ J., BEKAERT P., REETH F. V.: A self-shadow algorithm for dynamic hair using density clustering. *Proceedings of Eurographics Symposium on Rendering* (2004).

[PH89] PERLIN K., HOFFERT E.: Hypertexture. *Proceedings of ACM SIGGRAPH Conference* (1989), 253–262.

[RCT91] ROSENBLUM R., CARLSON W., TRIPP E.: Simulating the structure and dynamics of human hair: Modeling, rendering and animation. *The Journal of Visualization and Computer Animation 2*, 4 (1991), 141–148.

[Ree83] REEVES W. T.: Particle systems – A technique for modeling a class of fuzzy objects. *ACM Transactions on Graphics 2* (1983), 91–108.

[Sch04] SCHEUERMANN T.: Practical real-time hair rendering and shading. *ACM SIGGRAPH 2004 Sketch* (2004).

[XLTP03] XU S., LAU F. C. M., TANG F., PAN Y.: Advanced design for a realistic virtual brush. *Computer Graphics Forum (Proceedings of Eurographics) 22*, 3 (2003), 533–542.

[XTLP02] XU S., TANG M., LAU F. C., PAN Y.: A solid model based virtual hairy brush. *Computer Graphics Forum (Proceedings of Eurographics) 21*, 3 (2002), 299–308 & 625.

[XTLP04] XU S., TANG M., LAU F. C., PAN Y.: Virtual hairy brush for painterly rendering. *Graphical Models 66*, 5 (2004), 263–302.

[XY01] XU Z., YANG X.: V-hairstudio: An interactive tool for hair design. *IEEE Computer Graphics and Applications 21*, 3 (2001), 36–43.

[YXYW00] YANG X., XU Z., YANG J., WANG T.: The cluster hair model. *Graphical Models 62*, 2 (2000), 85–103.