

Real-time Level-of-detail Strand-based Rendering

T. Huang^{1,2} , Y. Zhou² , D. Lin³ , J. Zhu² , L. Yan² , and K. Wu¹ 

¹ LIGHTSPEED, USA

² University of California, Santa Barbara, USA

³ Nvidia, USA

Abstract

We present a real-time strand-based rendering framework that ensures seamless transitions between different level-of-detail (LoD) while maintaining a consistent appearance. We first introduce an aggregated BCSDf model to accurately capture both single and multiple scattering within the cluster for hairs and fibers. Building upon this, we further introduce a LoD framework for hair rendering that dynamically, adaptively, and independently replaces clusters of individual hairs with thick strands based on their projected screen widths. Through tests on diverse hairstyles with various hair colors and animation, as well as knit patches, our framework closely replicates the appearance of multiple-scattered full geometries at various viewing distances, achieving up to a $13\times$ speedup.

CCS Concepts

• **Computing methodologies** → **Rendering**;

1. Introduction

Strand-like structures contribute significantly to believable characters and immersive environments, particularly for the high fidelity of fur, hair, and knit cloth with fiber-level details. Unfortunately, most strand-like structures contain a significantly high number of primitives. Even an average human scalp, which is the type having the fewest strands, still consists of 100K - 150K hair follicles. A knitted full garment can have tens to hundreds of millions of fiber segments [WY17]. So, rendering and simulating realistic strand-like structures with complex geometries at such a scale poses significant challenges to computational efficiency for real-time applications. In the gaming industry, a common technique is to use flat, textured planes that simulate the visual appearance such as hair cards [Sch04, Jia16]. While this technique provides computational efficiency, the inherent coarse approximation limits the realism achievable in physics simulations and rendering.

Simulation and rendering directly using strands as primitives have gained prominence in recent production practices due to their ability to offer a more authentic portrayal of the behavior and appearance of strands [Taf19, Epi21, TS22, HYW*23, HWP*23]. Given the vast number of strands, a level-of-detail (LoD) strategy that simplifies the model at different viewing distances is essential to optimize computational resources for real-time applications. For hair, one approach is to use hair cards when the hair is at a distance. However, this introduces a notable discontinuity in dynamics and appearance during the transition from strands to cards due to their fundamentally different representations. An alternative strategy reduces the number of hair strands while increasing each

remaining strand's thickness. Regrettably, this method leads to an excessively bold and dry hair appearance compared to the realistic brightness and hue, as the thick hair meant to represent a cluster of hair strands should have distinct optical properties from a single hair strand. This distinction becomes particularly pronounced during multiple light bounces, exacerbating the visual disparity. Recent research proposes aggregated scattering models for a collection of furs [ZZW*22] and fibers [ZMA*23] that capture the aggregated appearance, incorporating the effect of multiple scattering. However, neither of their methods can be used in real-time strand-based rendering, like hairs and knits, due to the slow neural network inference [ZZW*22] and inaccurate handling of multiple scattering within an aggregated ply, exaggerated at model level [ZMA*23].

This work presents a novel real-time technique for strand-based rendering with seamless transitions between different LoD levels while ensuring a consistent appearance. At the non-zero LoD level, we use one thicker strand to represent a cluster of regular strands. To shade these thick strands, we consider the sparsity of strands in a cluster and reframe existing aggregation scattering models [ZZW*22, ZMA*23] to derive a Bidirectional Curve Scattering Distribution Functions (BCSDf) model, capturing both single and multiple scattering interactions between individual strands within the cluster. Our aggregated BCSDf is general and supports hairs and fibers across a wide range of roughness at different lobes. Then, our multiple scattering component modifies the dual scattering [ZYWK08] approximation by redistributing the lobe energies and introducing an extra light path series. Also, for hair that is without a clearly well-defined hierarchical structure, we proposed a hair hierarchy construction method, where we build hair clus-

tered at each LoD level and fit each cluster with thick hair to better conform to the overall shape. We present a real-time strand-based rendering framework that dynamically selects the appropriate LoD level, along with an on-the-fly width calculation method to cover represented fine strands tightly. We implement our real-time rendering pipeline in the modern GPU rasterization pipeline and conduct tests on various hairstyles with dynamics. In our experiments, our LoD framework, combined with our new BCSDf model, facilitates smooth transitions with a consistent appearance, resulting in neglectable computational overhead (<1%) for close-up views, a speedup of $2\times$ for middle views, and up to $13\times$ for far views.

We summarize our contributions as follows:

- A general strand-based aggregated BCSDf model to accurately capture single and multiple scattering within the cluster for both hairs and fibers;
- A real-time LoD hair rendering framework that dynamically and adaptively replaces hairs clusters with thick strands, achieving up to $13\times$ speedup.

2. RELATED WORK

Single strand BCSDf model. Kajiya and Kay [KK89] introduced the first physically-based analytic fiber shading model, considering the variation of reflectance in azimuthal directions and accounting for internal absorption and caustics. This model is later extended by Marschner et al. [MJC*03] into the well-known Bidirectional Curve Scattering Distribution Functions (BCSDf) model for hair, which remains actively used in the games and film industries. The analysis involved ray paths within dielectric cylinders, and the scattering is split into R , TT , and TRT modes. These modes are represented as separable products of azimuthal and longitudinal functions, where R and T signify one reflection and transmission inside the fiber, respectively. This model has undergone improvements by various researchers. Notable enhancements include adding a non-physical diffuse term for accurate fitting of measured data [ZRL*09], decoupling into artist-friendly lobes [SPJT10], adding a $TRRT$ lobe for energy conservation [dFH*11], and adding lobe for high-order internal reflections [CBTB16]. Besides, the diffuse term is also used to enhance the realism and flexibility for the fiber, ply, and yarn rendering, as introduced in [YTJR15, KSZ*15, ZMA*23]. Additionally, fiber reflectance model has been extended for elliptical hairs [KM17], animal hairs with interior medullas [YTJR15], and microfacet-based hair scattering model [HHH22]. In recent work, Xia et al. [XWM*20, XWH*23] propose a wave optics scattering model to capture the colorful glints of hair and fur.

Multiple scattering approximation. Multiple scattering is essential for realistic strand-based rendering but is computationally expensive. Moon et al. [MM06] use photon mapping to track indirect light bounces, storing incident radiance in a 3D grid of spherical harmonic (SH) coefficients [MWM08]. KT et al. [KJA*23] use MLP to learn the high-order scattered radiance between hairs. Zinke et al. [ZYWK08] introduce *dual scattering*, dividing multiple scattering into global and local multiple scattering, to achieve real-time strand-based hair rendering [YK08]. Our work builds upon dual scattering and introduces an additional local scattering component to efficiently approximate the complex scattering effects between aggregated strands in real-time scenarios.

Level-of-detail construction. The level-of-detail (LoD) strategy is commonly employed in real-time applications to optimize performance by using models with varying complexities based on viewer distance [KSW21]. Various simplification techniques have been developed to create LoD meshes at different complexity levels, including element removal [GH97, Hop96, LWC*02], re-triangulation [CSAD04, LN21], and primitives fitting [NW17, BL20]. However, traditional LoD methods do not suit filament-like shapes such as hair due to their unique geometry. Weier et al. [WZK*23] present an adaptive neural representation for correlation-aware LoD voxels designed for general purposes. In simulation [BKCNO3] and hair editing [WLJ*03], a bottom-up LoD strategy was employed to group hair strands into clusters by their scalp locations. Their approach, however, introduces noticeable appearance differences between LoD levels due to applying a single hair appearance model for a hair cluster, ignoring light interaction within the cluster.

Surface-based simplification – hair cards. Koh and Huang [KH01] pioneer the use of a polygon strip to represent a cluster of hair strands with alpha maps for real-time applications, a technique still prevalent in the game industry [Jia16]. However, creating hair cards for different LoD levels is time-consuming for artists and can cause discontinuities in dynamics and appearance during transitions. In contrast, the LoD solution proposed in this paper only requires a model with full hair strands, eliminating the need for additional manual authoring in LoD asset creation.

Strand-based simplification – aggregation model. Based on the hierarchical structure of the yarn geometries, the aggregation model has recently been used to depict the optical characteristics of a cluster of fibers in woven [MGZJ20] and knitted structures [MGJZ21]. Their model integrates normal mapping with a specialized BSDF that combines a specular component for specular reflection and transmission and a body component for multiple scattering within the fiber bundle. Cook et al. [CHPR07] introduced a stochastic technique by randomly selecting a subset of geometric elements and statistically altering those elements to preserve overall appearance. Zhu et al. [ZZW*22] proposed an aggregation BCSDf model for a bunch of fibers using neural networks. Later, to eliminate parameter fitting, Zhu et al. [ZMA*23] derived an analytical aggregated BCSDf model to account for single and multiple light scattering models for yarns based on dual-scattering theory.

This paper introduces a general aggregation strategy that builds on analytical aggregated scattering model [ZMA*23] with two significant improvements and supports hair BCSDf [MJC*03] with additional TRT lobes. Firstly, we redistribute lobe energies and introduce an extra light path series in the aggregation scattering model for more accurate light scattering approximation within the hair cluster. Secondly, our model supports an elliptical cross-section, creating a tighter fitting bound for strand clusters.

3. Background

This section overviews hair and fiber shading models, dual scattering for multiple scattering approximation, and previous aggregation model, which serve as the foundation for our method.

The general rendering equation for outgoing radiance L_o towards

a direction parameterized by longitudinal-azimuthal angles (θ, ϕ) is defined as:

$$L_o(\theta_o, \phi_o) = \int_{-\pi}^{\pi} \int_{-\frac{\pi}{2}}^{\frac{\pi}{2}} L_i(\theta_i, \phi_i) f(\theta_i, \phi_i, \theta_o, \phi_o) \cos^2(\theta_i) d\theta_i d\phi_i,$$

where L_i and L_o denote incoming radiance from direction (θ_i, ϕ_i) and outgoing radiance in direction (θ_o, ϕ_o) , respectively.

3.1. Single strand Shading Model

As suggested by Marschner et al. [MJC*03], the interaction between light and a single hair can be decomposed into three modes of reflection R , TT and TRT , where T and R stand for transmission and reflection across a cylinder interface. The single hair BCSDf, denoted as f , is defined as:

$$f^{\text{hair}}(\theta_i, \phi_i, \theta_o, \phi_o) = \sum_{p \in \{R, TT, TRT\}} M_p(\theta_h) N_p(\phi), \quad (1)$$

where p denotes different types of lobes, $\phi = \phi_o - \phi_i$ is the relative azimuthal angle, $\theta_h = (\theta_i + \theta_o)/2$ is the longitudinal half angle.

In terms of fibers, multiple scatterings inside the fiber, such as $TRT, TRRT, \dots$, are approximated by an additional diffuse lobe D , as suggested in [YTJR15]. Following [ZMA*23]. The attenuation A_p is extracted from N_p term and defined as a user-defined value.

$$f^{\text{fiber}}(\theta_i, \phi_i, \theta_o, \phi_o) = \sum_{p \in \{R, TT, D\}} A_p M_p(\theta_h) N_p(\phi), \quad (2)$$

3.2. Dual Scattering

Zinke et al. [ZYWK08] proposed dual scattering (DS) to approximate multiple scattering between millions of hair segments efficiently. The key idea is to split the multiple scattering into two additional lobes: global and local multiple scatterings. The global scattering approximates the amount of light remaining after forward scattering through a number of hairs, while local scattering estimates the total contribution through backward scattering after multiple bounces in the neighborhood. Thus, two scattering lobes are added to the local shading (Eq. 1) as an approximation:

$$f_F^{\text{ds}}(\theta_i, \theta_o, \phi) = \frac{I_F(\phi)}{\cos^2(\theta_i)} d_F A_F \sum_p G(\tilde{\sigma}_F^2, \theta_o + \theta_i) N_{F,p}, \quad (3)$$

$$f_B^{\text{ds}}(\theta_i, \theta_o, \phi) = \frac{I_B(\phi)}{\pi \cos^2(\theta_i)} d_B A_B G(\tilde{\sigma}_B^2, \theta_o + \theta_i), \quad (4)$$

where $A_{F|B}$, $d_{F|B}$, $\tilde{\sigma}_{F|B}$, and $I_{F|B}$ represent the forward/backward attenuations, scattering density constants, averaged spread variance, and binary backward hemisphere indicators, respectively. $N_{F,p}$ is the averaged azimuthal lobe p within the front hemisphere. Forward scattering attenuation A_F can be computed by the sequential product of averaged forward attenuations that the light passes through, while $A_B = A_1 + A_3$ includes all light paths with single backward scattering A_1 and with three backward scattering A_3 . $G(\beta, \theta) = \frac{1}{\sqrt{2\pi}\beta} e^{-\theta^2/2\beta^2}$ is the traditional Gaussian distribution. We refer readers to the previous work [ZYWK08] for more details. However, dual scattering only considers R and TT and assumes infinite hair strands behind the shading point.

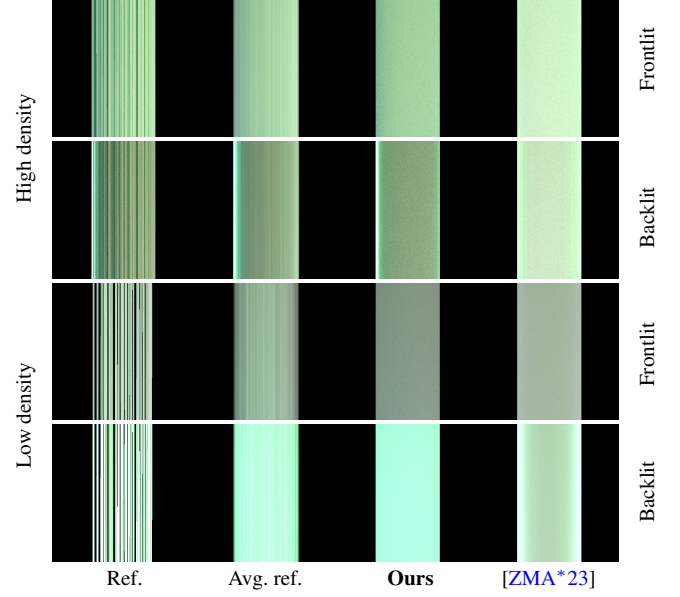


Figure 1: Compared to prior aggregated model [ZMA*23], ours is closer to avg. ref. that averages 100 different ply instances (ref.) for both frontlit and backlit with high and low hair densities. Both avg. ref. and ref. use single fiber BCSDf f^{fiber} .

3.3. Aggregated Shading Model

Leveraging the dual scattering, Zhu et al. [ZMA*23] proposed an aggregated BCSDf model to approximate light scattering within a ply consisting of n fibers as:

$$f^{\text{aggregated}}(\theta_i, \phi_i, \theta_o, \phi_o) = \sum_{p \in \{R, D, F, B\}} A_p M_p(\theta_h) N_p(\phi), \quad (5)$$

in which, in addition to reflection R and diffuse D lobes, they introduced forward scattering lobe F and backward scattering lobe B to capture multiple scattering components inside the aggregated ply (see Fig. 2b). The F lobe represents the distribution of remaining scattered energy that passes forward after multiple scattering inside the ply. The corresponding attenuation $A_F(\theta_d)$ is computed as:

$$A_F(\theta_d) = \prod_{i=1}^n a_F(\theta_d), \quad (6)$$

where a_F sums the corresponding outgoing radiance toward the front hemisphere, including contributions from R , TT , and D lobes of a single hair. However, as shown in Fig. 1 second column, their aggregated model does not match the reference well, especially under backlit conditions. Particularly, their F lobe represents the remaining energy after the ray passes through all the fibers in the light path. However, their R , B , and D lobes still exhibit a diffuse azimuthal distribution, which should be attenuated after multiple scattering. The double-counting leads to higher brightness. Additionally, in sparse fiber distribution, TT lobe exhibits a completely different azimuthal distribution compared to D and R lobes, and it cannot be represented by a single F lobe adequately.

4. Our Aggregated Shading Model

We propose an improved aggregated BCSDf model with three novel enhancements, including 1) modified lobe decomposition strategy for hair and fiber, 2) shadowing-masking term $S(\phi, \theta_d)$, and 3) elliptical cross-section handling, to match the reference accurately. Our aggregated BCSDf is defined as:

$$f^{\text{ours}}(\theta_i, \phi_i, \theta_o, \phi_o) = S(\phi, \theta_d) \sum_{p^{\text{single}} \cup \{\hat{B}\}} M_p(\theta_h) N_p(\phi), \quad (7)$$

We consider our aggregated model to characterize lobes of the last intersected strand inside an aggregated geometry. Our aggregated model consists of two parts: inherited lobes from the original single-strand BCSDf and an extra \hat{B} lobe that approximates the outgoing radiance toward the back hemisphere from the shading point. It is worth noting that our model does not depend on a specific strand type. The definition of p^{single} varies according to the underlying strand geometry: when aggregating human hair strands, it follows Eq. 1, whereas for cloth fibers, it adheres to Eq. 2.

Inherited lobes. The lobes originate from single-strand BCSDf are used to describe the scattering distribution of the counterpart lobe after being attenuated by $n - 1$ preceding strands. Here, n is the total number of strands intersected by a ray that carries incident radiance. When the ray leaves from the back hemisphere of the aggregated strand, n always equals 1, meaning that these inherited lobes should be exactly the same as single fiber BCSDf in this situation. The longitudinal and azimuthal scattering functions of inherited lobes are defined as:

$$M_{\hat{p}} = G(\tilde{\beta}_p^M, \theta_h), \quad N_{\hat{p}} = a_F^{n-1} N_p \quad (8)$$

where longitudinal scattering distributions are modeled by Gaussian functions with wider variance $\tilde{\beta}_p^M$ to account for the longitudinal spread due to front scattering. In particular, $\tilde{\beta}_p^M$ is calculated by accumulating all variances along the light path as:

$$\tilde{\beta}_p^M = (n - 1) \tilde{\sigma}_f^2(\theta_d) + \beta_p^M \quad (9)$$

The spread variance $\tilde{\sigma}_f^2(\theta_d)$ accumulates forward azimuthal variances of n strands. $\theta_d = \theta_i - \theta_o$ is the relative longitudinal angle.

And similar to global scattering, the azimuthal part is diminished by multiplying the forward scattering transmittance. This transmittance is calculated as the product of the attenuation per penetrated strand along the scattering path.

Modified \hat{B} lobe. In our approach, we employ the \hat{B} lobe to approximate the contribution of multiple scattering near the shading point after the ray has reached the corresponding strand. It is essential to note that, following the assumption of dual scattering that splits multiple scattering into forward and backward components, the forward scattering is distributed among the inherited lobes. The \hat{B} lobe is specifically designed to account for the outgoing path toward the back hemisphere at the shading point. Fig. 2 illustrates an example of our aggregated BCSDf for fibers, alongside the single-fiber BCSDf and the aggregated BCSDf from [ZMA*23].

To address the scenario that strand distribution is relatively sparse within the aggregate geometry, allowing light to easily traverse the “first” layer of strands and reach the strand behind (green path in Fig. 3a), we introduce an additional path series, denoted as

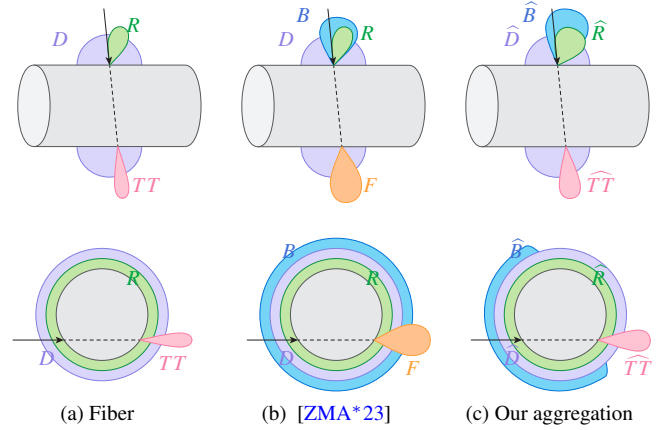


Figure 2: Illustration of single fiber, aggregated [ZMA*23], and our aggregated fiber BCSDf for longitudinal (top) and azimuthal (bottom) components. Our model replaces F lobe with \hat{TT} lobe and distributes part of its contribution to \hat{R} , \hat{TT} , and \hat{D} with larger variances. Additionally, our \hat{B} lobe is designed to only account for the outgoing paths toward the back hemisphere.

A_{1+} , for the backscattering light, to supplement existing paths with one A_1 and three A_3 backward scattering events (denoted by B), and the paths with even number of forward scattering events (denoted by F). This series represents the scenario where the light hits the strand behind the currently shaded strand first, undergoes an even number of forward scattering events and one backward scattering event, and then hits the back side of the current strand. This path series can be described as $F^{i+1} B F^i$, where $i \geq 0$ denotes the number of strands passed through behind the currently hit strand. The rationale behind considering the A_{1+} path series is that, when the strand density is relatively low, there is a higher probability that some strands behind the currently hit strand get illuminated through gaps, significantly contributing to the overall appearance of the outer strand. The corresponding attenuation is computed as follows:

$$A_{1+} = d_{1+} a_B(\theta_d) \sum_{i=1}^n a_F^{2i-1}(\theta_d), \quad (10)$$

where d_{1+} accounts for the possibility of the contribution of such paths coming from the side and rear directions. In practice, we approximate it using a constant value as $d_{1+} = 0.6$.

The attenuation $A_{\hat{B}}$, longitudinal $M_{\hat{B}}$ and azimuthal $N_{\hat{B}}$ distributions are as follows:

$$A_{\hat{B}} = A_{1+} + A_1 + A_3, \quad M_{\hat{B}} = G(\tilde{\beta}_{\hat{B}}^M, \theta_h), \quad N_{\hat{B}} = \frac{A_{\hat{B}} I_B(\phi)}{\pi}, \quad (11)$$

where the longitudinal variance $\tilde{\beta}_{\hat{B}}^M$ is the averaged accumulated variance along each possible path, weighted by its attenuation.

Estimation of strand number n . In order to represent a collection of strands using a thick cylinder with an elliptical cross-section, it is important to estimate the number of strands a ray intersects inside the aggregated cylinder for computing attenuation and scattering distribution. We define the unitless strand density ρ as the ratio of the area occupied by strands to the cross-sectional

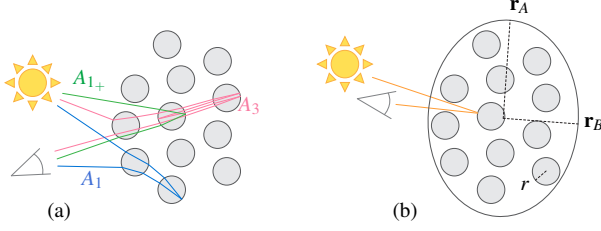


Figure 3: Illustrations of light pass in aggregated cross-section: (a) the blue A_1 and pink A_3 paths are considered in the traditional local scattering with one and three backward scattering events by B lobe. The green path illustrates our added A_{1+} paths, which encounter two deflections before returning to the shading strand. (b) when the incident and outgoing directions are close, the observed strand along the incident direction may not be occluded by others along the outgoing direction. All strands with radius r are contained by an elliptical cross-section, with major axis \mathbf{r}_A and minor axis \mathbf{r}_B .

area of the aggregated cylinder, given by $\rho = n_{\text{total}} r^2 / (|\mathbf{r}_A| |\mathbf{r}_B|)$, where n_{total} is the number of strands inside, r is the single strand radius, and \mathbf{r}_A and \mathbf{r}_B are the major and minor axes of the cross-section. Given the ray travel distance l , the number of intersecting strands n can be calculated as $n = \rho l \sqrt{\frac{n_{\text{total}}}{\pi |\mathbf{r}_A| |\mathbf{r}_B|}}$, where the last term estimates number of intersecting strands per unit distance.

Shadowing-masking $S(\phi, \theta_d)$. For a cluster of strands, even if ω_i and ω_o fall within the same hemisphere, it does not guarantee that the strand seen from the view direction can also be seen from the light direction. In other words, when the relative azimuthal angle $\phi \in (-\frac{\pi}{2}, \frac{\pi}{2})$, self-shadowing situations can occur within a thick aggregated geometry.

We propose a shadowing-masking term to compensate for self-shadowing for $\phi \in (-\frac{\pi}{2}, \frac{\pi}{2})$ by categorizing shadowing at the shading point into three categories: shadowing by a single strand, shadowing by multiple strands, and no shadow. We approximate the possibility of the shadow ray from a shading point needing to pass through a single strand to reach the light source as:

$$P_s = (1 - \rho)(1 - \cos(\phi)), \quad (12)$$

where $\cos(\phi)$ is used to approximate the difference between the incident and outgoing directions in the azimuthal plane. As shown in Fig. 3b, strands located further from the incident direction are more likely to be occluded by other strands along the light direction. The $1 - \rho$ term accounts for the fact that the shadow-masking effect becomes less prominent with denser strands, as a thick strand better approximates the cluster of strands.

The shading point can also be occluded by multiple strands along the shadow ray. We approximate the probability as:

$$P_m = (1 - \rho) c_m, \quad (13)$$

where c_m is a constant value empirically determined ($c_m = 0.2$ for all our experiments). The probability of the shading point reached directly by light with no-shadow can be computed as:

$$P_n = 1 - P_m - P_s. \quad (14)$$

Thus, the shadowing-masking $S(\phi, \theta_d)$ term is formulated as:

$$S(\phi, \theta_d) = P_s(\phi) a_F(\theta_d) + P_n \mathbb{1}, \quad (15)$$

where $\mathbb{1}$ represents a three-dimensional vector of ones. Based on the dual scattering assumption, the attenuation needs to be multiplied by a_F , as the light ray must undergo forward scattering once to reach the actual shading strand. Note that no lighting calculation is performed for P_m , given the assumption that its contribution to lighting is negligible.

5. Real-time Hair Rendering with Level-of-detail

This section outlines the construction process of the hair hierarchy and describes how we utilize it in rendering.

5.1. Initialization

Given a strand-based hair model with tens of thousands of *single hairs*, we build the hair hierarchy in four stages: 1) selecting a number of guide hairs for simulation, 2) clustering hair strands into n_L levels, 3) fitting the ellipses for thick hair cross-sections, and 4) computing interpolation weights. We denote the levels in the hierarchy by L_0 (coarsest) through L_{n_L-1} (finest).

1. Guide hair selection. Initially, a Catmull–Rom spline is fitted with n_c control points for each hair strand. Following [WYZG09], we use k-means clustering to create a user-defined number of hair clusters. The distance metric between two single hairs i and j is defined as the sum of distances between corresponding control points along the two single hairs, expressed as $D(i, j) = \sum_k^{n_c} |\mathbf{p}^{i,k} - \mathbf{p}^{j,k}|^2$, where $\mathbf{p}^{i,k}$ denotes the position of the k -th control point along the hair i . Within each cluster, the hair with the smallest average distance to all other hairs is selected as the guide hair, which serves as the basis for the physical-based simulation and is used to animate each single hair via linear skinning [Som15] in runtime. Note that the cluster used for guide hair selection in this stage would not be discarded afterward.

2. Hair strand clustering. After identifying guide hairs, each single hair locates three guide hairs that are closest to it. Single hairs sharing the same set of guide hairs are clustered together to form the coarsest level L_0 . The subsequent levels, from L_1 to L_{n_L-1} , are directly clustered from L_0 . While we do not maintain strict bounding relations between consecutive levels, we ensure that the cross-sectional area of the thick hairs at each level is approximately four times larger than that of the previous level.

3. Cross-section fitting. To construct a thick elliptical hair from a cluster of single hairs, we first create n_c sets of control points where each set \mathcal{P}^k contains all single hairs' k -th control points. For each set \mathcal{P}^k , we compute the average position of all control points $\bar{\mathbf{p}}_{cs}^k$ and normalized average tangent $\bar{\mathbf{t}}^k$ to define the center of the ellipse. Principal Component Analysis (PCA) is then employed to fit an ellipsoid for \mathcal{P}^k , after which we discard the axis closest to $\bar{\mathbf{t}}^k$. Finally, we identify four *corner control points* $\mathbf{p}_{0,1,2,3}^k$ from \mathcal{P}^k that are closest to the four endpoints of the remaining two axes.

4. Interpolation weight computing. In hair skinning, three guide hairs form a triangular prism for linear skinning. We compute and record interpolation weight for every single hair control

point, as well as for every thick hair cross-section center $\bar{\mathbf{p}}_{\text{cs}}^k$ and corner control points $\mathbf{p}_{0,1,2,3}^k$.

5. LoD level initialization. We compute the initial LoD levels for the starting view in rendering. Using a user-specified screen-space hair width threshold ϵ_w , we determine the LoD level for each cluster within L_0 as follows: we examine the LoD level from coarsest to the finest until we find the first level where the maximum thick hair width in the cluster is less than ϵ_w . At runtime, we adjust the LoD levels starting from the level used in the previous frame, making this process much more efficient. Note that the LoD selection is performed on a per-segment basis.

5.2. Runtime

With guide hairs dynamically simulated at runtime, our rendering framework uses the simulated control points of the guide hairs and the precomputed interpolation weights to determine the positions and shapes of the hairs at desired LoD levels (we update quantities like $\bar{\mathbf{p}}_{\text{cs}}^k$ and $\mathbf{p}_{0,1,2,3}^k$ by interpolation using these positions and weights). We dynamically compute the thick hair widths to determine the proper LoD levels and calculate density and lobes in our aggregated BCSDf.

Hair width calculation. To ensure that thick hairs selected by the width-based LoD metric adequately cover the corresponding single hairs, we dynamically compute the hair widths as follows. First, the four corner control points $\mathbf{p}_{0,1,2,3}^k$ are projected to the plane defined by $\bar{\mathbf{p}}_{\text{cs}}^k$'s position and its tangent. Then, we measure the largest screen-space distances (on the xy plane) and the largest depth differences (along the z axis) between these projected points as l_w and l_t as hair width and thickness, respectively. The corresponding density and estimated hair number are computed as $\rho = n_{\text{total}} r^2 / (l_w l_t)$ for shadow-masking in Eq. 15 and $n = \rho l \sqrt{\frac{n_{\text{total}}}{\pi l_w l_t}}$ for all lobes in Eq. 7.

Dynamic LoD selection. During runtime, we select the LoD level for each cluster in L^0 . The goal of our LoD selection is to ensure that all thick hairs rendered have widths that are just below the screen-space hair width threshold ϵ_w . Checking through all LoD levels can be expensive at runtime. Therefore, we propose an approximately correct, deferred LoD selection that reuses information between frames. For each cluster in every frame, we render its thick hairs at the current LoD level (the LoD level for the first frame is determined during initialization) and record the maximum width, denoted as w_{cur} . We also calculate the maximum width at one coarser level, denoted as $w_{\text{cur-}}$. Using these two widths, we determine the LoD level L for the next frame as:

$$L = \begin{cases} L + 1, & \text{if } w_{\text{cur}} > \epsilon_w, \\ L, & \text{if } w_{\text{cur}} \leq \epsilon_w < w_{\text{cur-}}, \\ L - 1, & \text{if } w_{\text{cur-}} \leq \epsilon_w. \end{cases} \quad (16)$$

Runtime pipeline. With the geometry generated for the given LoD level, we proceed with the conventional rendering pipeline, which includes three passes:

1. Shadow pass. Head and hair are rendered to generate a shadow map and deep opacity map [ZYWK08], respectively.

2. Shading pass. The width for each thick hair is calculated dynamically in the vertex shader. The maximum hair width for each cluster is recorded in a buffer using atomicMax. Then, tessellation shader subdivides hair strands based on their distance from camera, and geometry shader extends them into camera-facing strips. The aggregated BCSDf (Sec. 4) is used in fragment shader for shading.

3. Compute pass. The maximum hair width in each cluster at one coarser level is computed to determine the LoD for the next frame using the selection strategy (Eq. 16). Note that this compute pass only invokes the vertex shader.

Note that our runtime pipeline does not explicitly handle thick hairs overlapping, as hairs are rendered as camera-oriented strips for both the shading and deep opacity map passes.

6. RESULTS

We conduct all experiments on a system with an AMD Ryzen ThreadRipper 3970X 32-core CPU, 256 GB of memory, and an NVIDIA RTX 3090 GPU. We implement our method (aggregated BCSDf + LoD strategy) on both a CPU renderer and a GPU renderer (Fig. 4). In contrast to employing full hair geometry in path tracing (PT) and dual scattering (DS) [ZYWK08], we utilize reduced hair geometry achieved through our hair LoD solution. Specifically, we group individual hair strands into 256 clusters and fit each cluster with thick hairs across 5-6 LoD levels, depending on the total number of hair strands. Each thick hair is represented by a B-Spline with 16 control points. Each thick hair segment is assigned an individual LoD level for sufficient granularity.

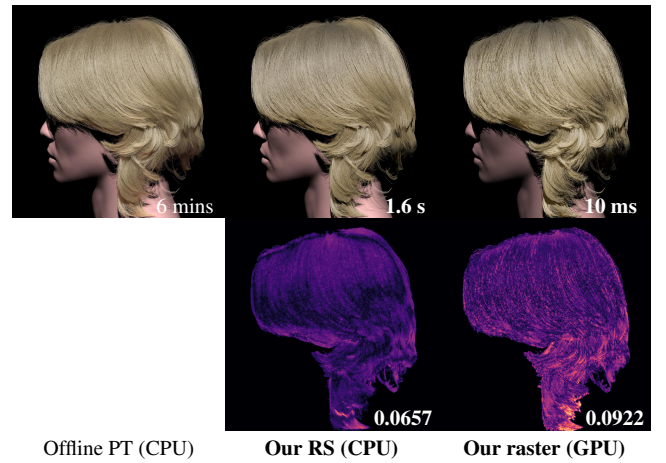


Figure 4: Our approach (offline ray shooting (RS) and real-time OpenGL rasterizer) achieves close results to path tracing (PT) reference, while our real-time implementation is 36000× faster. FLIP error maps are displayed at the bottom.

To validate the accuracy of our proposed method, we use a CPU render implemented in Embree [WWB*14] with 20 samples per pixel (spp). Similar to [ZYWK08], we use ray shooting to find all intersecting hairs along the shadow path to compute the transmittance for dual scattering. The CPU reference is generated by path tracing the full hair geometry with up to 70 ray bounces and 1024 spp. The hair BCSDf parameters are chosen generally consistent

Near view (60%)				Middle view (20%)				Far view (2%)	
Full w/ PT	LoD w/ ours	Full w/ DS	Full w/ PT	LoD w/ ours	Full w/ DS	Full w/ PT	LoD w/ ours	Full w/ DS	
970K	930K	970K	970K	790K	970K	970K	250K	970K	
	0.090	0.128		0.110	0.140		0.132	0.160	
1050K	1010K	1050K	1050K	857K	1050K	1050K	418K	1050K	
	0.066	0.086		0.062	0.082		0.065	0.086	
1900K	1856K	1900K	1900K	1535K	1900K	1900K	574K	1900K	
	0.053	0.070		0.125	0.141		0.141	0.137	
1600K	1510K	1600K	1600K	983K	1600K	1600K	330K	1600K	
	0.009	0.009		0.023	0.010		0.026	0.013	

Figure 5: Four hairstyles, ranging from straight to curly hair, under near, middle, and far views. Compared to path tracing with full hair geometry (full w/ PT), our approach (LoD w/ ours) utilizing reduced hair geometry through level-of-detail techniques can outperform existing dual scattering employing full geometry (full w/ DS) in both memory size and appearance. The FLIP error of ours increases for the far view due to the geometry simplification. *Pink* and *cyan* denote the number of segments and FLIP error of screen-space bounding box of hairs.

with the values provided by Marschner et al. [MJC*03]. To further evaluate our real-time performance and rendering quality, we implement our framework and counterpart full hair geometry and hair cards with DS using OpenGL 4, in which the shadow computation in all methods uses deep opacity maps.

We use a LoD threshold of $\epsilon_w = 2$ pixels for all our experiments. The output image resolution for all results is 1024×1024 . The initialization of our method takes from 5 to 10 minutes, depending on

the number of hair strands, with the majority of the time spent on hair clustering. To assess the impact of our LoD solution on rendering accuracy, we present results from near, middle, and far views with 60%, 20%, and 2% screen occupancy ratios, respectively. Note that our ray tracer produces a slightly different estimation of hair screen width than our rasterizer, which causes the hair count in the two systems to be different at the same view.

Hairstyles. In Fig. 5, we evaluate our approach on Four

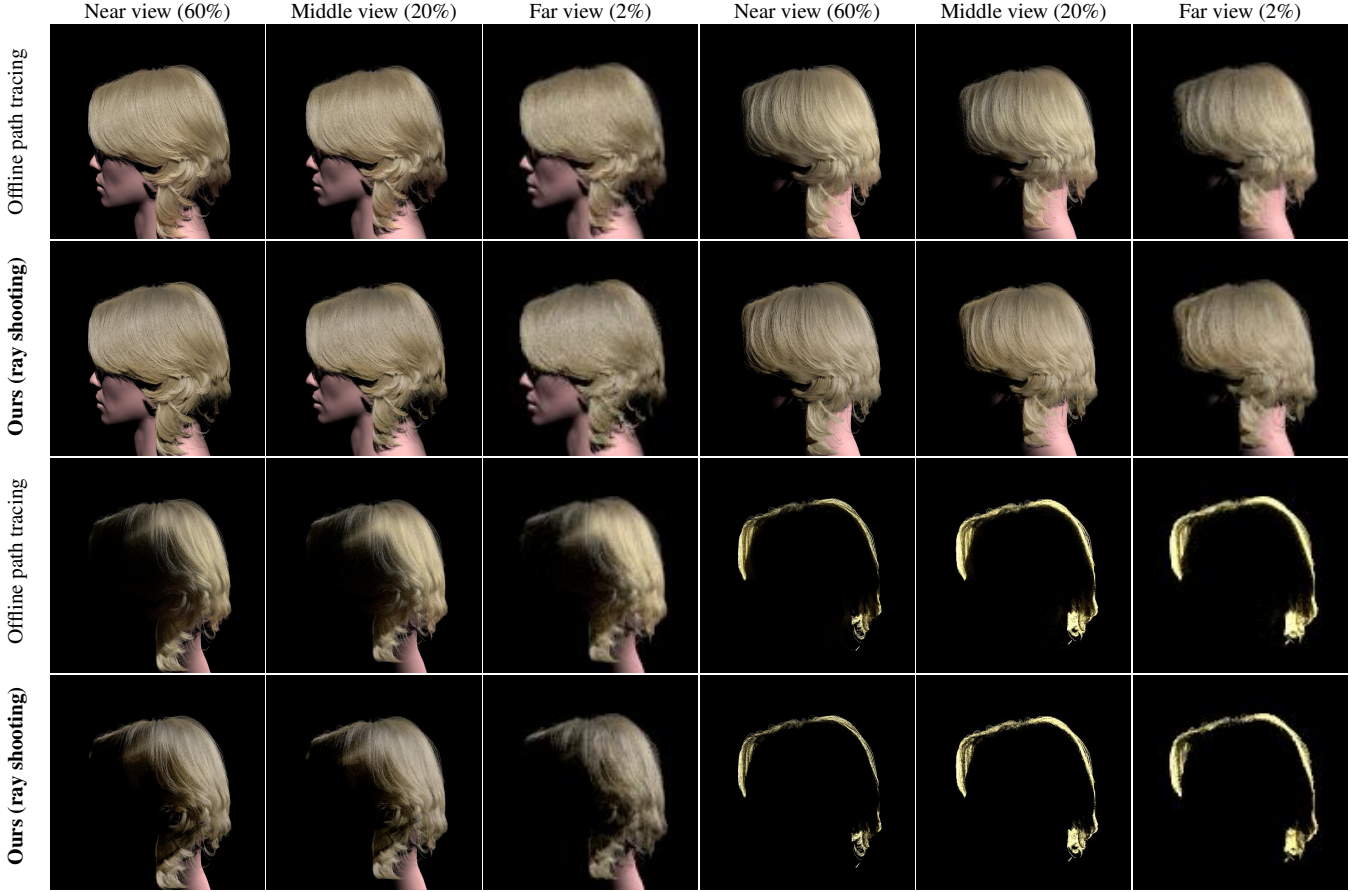


Figure 6: Our approach can yield results closely resembling those obtained from offline path tracing when the angles between the view direction and light direction are 0° , 45° , 90° , and 180° . It's worth noting that offline path tracing employs full hair geometry with 1050K segments, while ours utilizes 1010K, 857K, and 418K segments for near, middle, and far views, respectively.

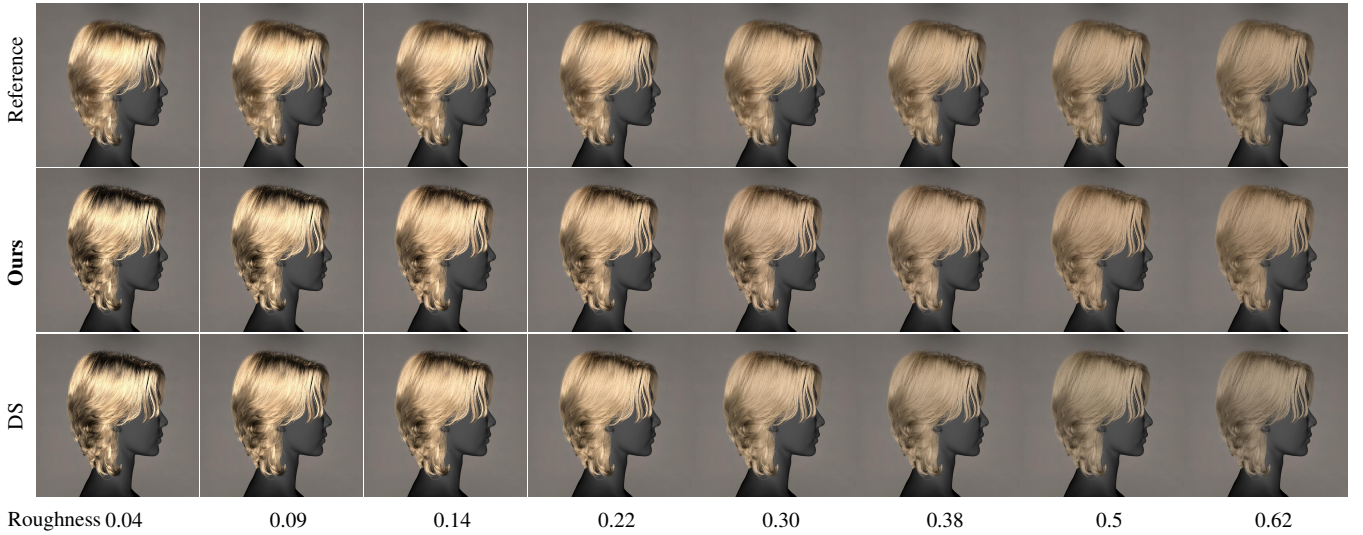


Figure 7: Our A_{1+} term effectively compensates for excessive darkening and color distortion caused by traditional DS as roughness increases.

hairstyles, ranging from straight to curly. Two of these hairstyles are sourced from Unreal Metahumans [Epi21], and two are cap-

tured from the real-world examples [DJBDT13, SSW*23]. For the four hairstyles, we use different hair colors: brown, blonde,

red, and black. Compared to using full hair geometry with DS, our method consistently delivers higher accuracy with lower \mathcal{H} LIP errors for all close views and middle and far views across the first three hairstyles. The additional light path series A_{1+} introduced by our method significantly enhances shading accuracy. However, as the viewing distance increases, the error of our method increases slightly due to geometric simplification.

Various light directions. Fig. 6 demonstrates that our method can produce results closely resembling those obtained from offline path tracing across various light directions, including 0° (frontlit), 45° , 90° , and 180° (backlit), and from different viewing distances.

Marschner model with on various roughnesses. In Fig. 7, we have covered almost the entire range of commonly observed roughness values, from extremely smooth roughness (0.04) to nearly diffuse roughness (0.62). Our method consistently maintains a high level of similarity to the ground truth across all these cases. At the same time, this also demonstrates that under high roughness conditions, our A_{1+} term effectively compensates for the contributions that DS fails to capture. Fig. 8 shows an evaluation of our method using Marschner hair BCSDf [MJC*03] with the full geometry in offline path tracing. Our method consistently and effectively reduces the number of hairs while maintaining a high level of appearance accuracy.

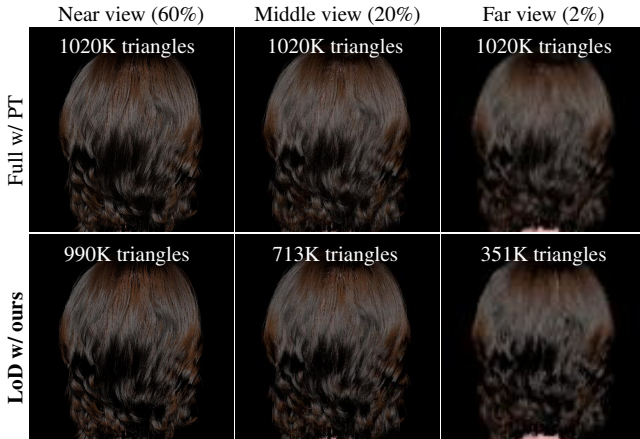


Figure 8: Marschner hair BCSDf [MJC*03] for dark-colored hair. Our strand-based LoD model (LoD w/ ours) effectively reduces the number of strands and preserves appearance

Animated hairs. Fig. 9 demonstrates that our method maintains visual accuracy with hair animation. We deployed a strand-based hair simulator [HWP*23] to simulate two hairstyles with 256 guide hairs and derived all hairs via linear skinning. Our method allows a smooth transition between different LoD levels in real-time. In extreme cases of intense hair movement, the finest LoD level may be required to maintain the correct screen space width. However, in our experiments, even during the most intense motion, our method achieves a significant level of simplification while maintaining nearly identical visual quality to full geometry.

Real-time performance. Fig. 10 illustrates the real-time performance of our GPU rasterizer pipeline. The most demanding scenario occurs with a close view (50% screen occupation ratio),

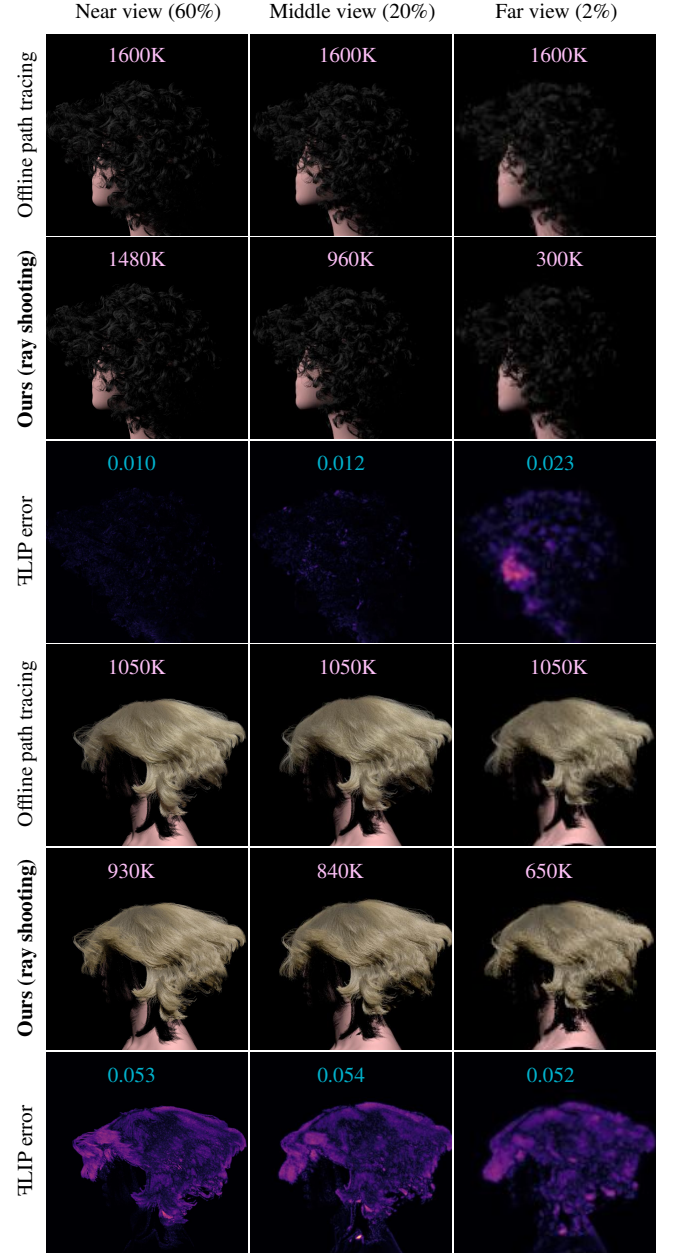


Figure 9: Animated hairs from two simulation sequences exhibit varying shapes. Pink and cyan denote the number of segments and \mathcal{H} LIP error of screen-space bounding box of hairs.

where only a limited number of single hairs can be represented by thick hairs. In this case, our approach needs to render a similar number of hair segments as the full hair geometry, along with additional compute passes for thick hair width computation. Nonetheless, our approach exhibits only a slight overhead ($<1\%$). As the camera distance increases, hair strands gradually transition to thicker hairs, reducing the number of segments required for rendering. This leads to a noticeable speedup of $2\times$ for the middle view (10% of screen) and up to $13\times$ for the far view (0.3% of screen). The breakdown of

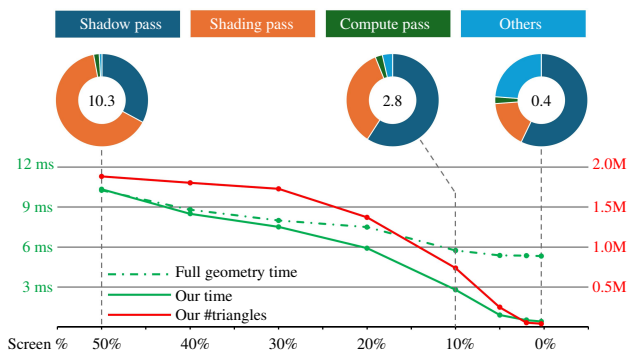


Figure 10: A performance visualization for the example in Fig. 11. As screen occupancy decreases from 50% to 0.3%, our GPU rasterizer progressively outperforms rasterizing full hair geometry, achieving a slight overhead of $0.99\times$ (10.3ms vs. 10.2ms) to a significant improvement of $13\times$ (0.4ms vs. 5.3ms). Pie charts provide a breakdown of the timings of three screen percentages of hairs.

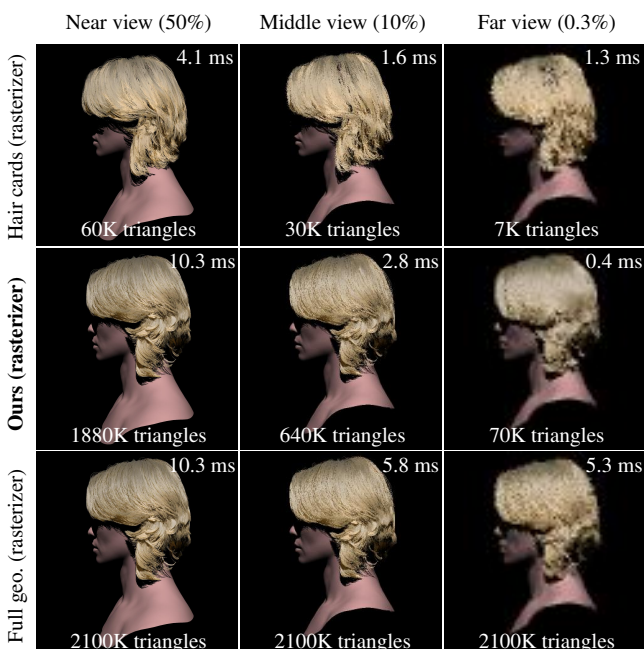


Figure 11: Full hair strand geometry, our strand-based LoD model, and hair card model at near, middle, and far views.

single frame time indicates that the percentage of time taken by the shading pass decreases with the camera distance due to the reduction in hair geometry and fewer pixels required to shade. Additionally, our pipeline requires a constant time of about 0.1 ms for hair data organization based on new LoD level for next frame rendering.

Comparison to hair cards. Fig. 11 compares our strand-based model with a hair cards model. The hair cards model has 3 LoD levels, featuring 600K, 300K, and 70K triangles, respectively. We manually configure two distances for the hair cards to do the LoD switching and render them using our real-time GPU rasterizer. Despite requiring more triangles in our LoD structure, our approach facilitates a seamless transition between LoD levels and supports

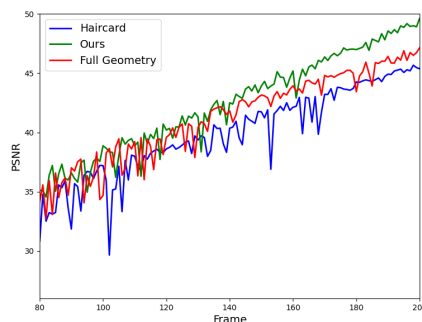


Figure 12: Evaluation on transition smoothness.

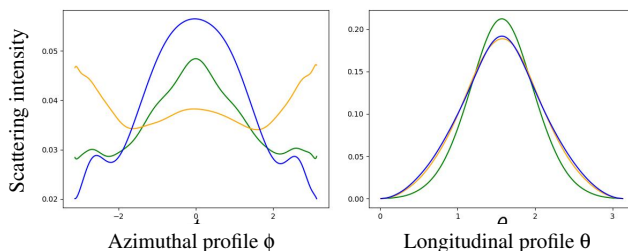


Figure 13: The average scattering intensity (green) across 100 ply instances with distinct fiber geometries. Our method (blue) offers a more accurate approximation than that of [ZMA*23] (orange), particularly in the azimuthal profile.

strand-based hair simulation. In contrast, the hair card textures at different LoD levels often exhibit inconsistencies, leading to noticeable popping artifacts during the LoD transition.

Note that we render haircards using an order-independent rasterization, meaning shading only the closest fragment without using alpha blending, for performance purposes. A threshold is set for the coverage texture, and fragments are discarded when their coverage falls below this threshold. Notably, in far-view scenarios, although hair cards have fewer triangles, they necessitate reading the coverage texture to compute accurate shadows. This requirement leads to higher computational costs compared to our method.

Additionally, we measured the smoothness of our transitions by comparing the similarity of adjacent frames using PSNR. As shown in Fig. 12, our method does not experience the drastic changes that occur with hair cards during LoD transitions. Furthermore, due to the appropriate LOD level, the flickering in our real-time rendering is less pronounced than with full geometry. Our PSNR remains consistently at the highest level among the three methods.

Memory cost and initialization time. In addition to the original full-size hair strand data, we store the positions of the centerline control points and the indices of four guiding points for thickness calculation. This results in extra memory consumption of approximately three-quarters of the original model size (78% for the four models used in the paper). Our initialization time ranges from 6 to 12 minutes for hair models with strands ranging from 60,000 to 120,000. This process is performed only once, and the results can be stored as configuration.

Comparison to [ZMA*23]. We utilize CPU path tracing to ren-

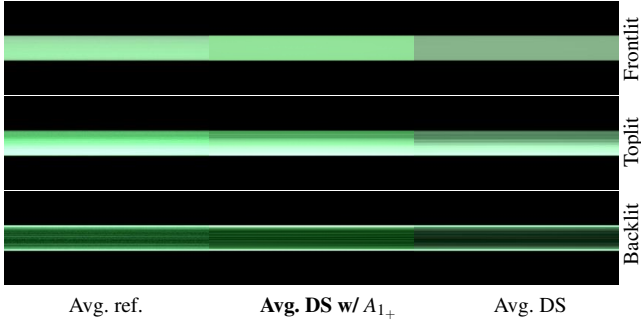


Figure 14: Given 100 fibers forming an elliptical cross-section with a major axis to minor axis ratio of 4 : 1, A_{1+} term can compensate for dual scattering to closely match averaged reference (avg. ref.).

der 100 ply instances with diverse fiber geometries and average the rendering outcomes. Additionally, we illustrate the scattering intensity at the azimuthal and longitudinal planes in Fig. 13. The aggregated model proposed by Zhu et al. [ZMA*23] does not align well with the reference, particularly under backlit conditions, whereas our model provides a more accurate approximation.

Ablation study on A_{1+} . To assess the influence of A_{1+} path series, we conduct an ablation study involving the averaging of rendering results from 100 different fiber instances. Each instance is rendered with single BCSDf with full single fiber geometries using CPU path tracing. The fiber cluster forms an elliptical cross-section with a major axis to minor axis ratio of 4 : 1. Fig. 14 demonstrates that the A_{1+} term effectively compensates for dual scattering, resulting in close alignment with the average reference (avg. ref.).

Validation on strand number estimation. To validate the accuracy of strand number estimation under highly eccentric cross-sectional shapes, we reused the model from Fig. 14 and conducted a comparison between two approaches: full geometry model with Dual Scattering implemented via ray shooting, and a single ply directly shaded by our aggregated model. The results in Fig. 15 show that our method yields consistently accurate results under different lighting conditions, regardless of whether the camera is aligned with the major or minor axis of the ellipse.

Knit patches. Our aggregated BCSDf can also be used for fabric rendering. Our method handles twisted fibers by rotating the local frame at each shading point. We evaluate the efficacy of our aggregated BCSDf using two knit patches. Fig. 16 showcases the comparison. The reference images, denoted as ref., are generated using fiber-level geometries (9.7M and 8.9M segments for each patch, respectively) with single fiber BCSDf, while other methods render ply-level geometries with significantly fewer segments (130K and 49K segments for each patch, respectively). Our aggregate BCSDf closely resembles the reference appearance. In comparison, the absence of the shadowing-masking term contributes to the observable increase in error. Using single fiber BCSDf results in significantly brighter results. The prior aggregated BCSDf [ZMA*23] also results in increased brightness by double-counting R and D lobes.

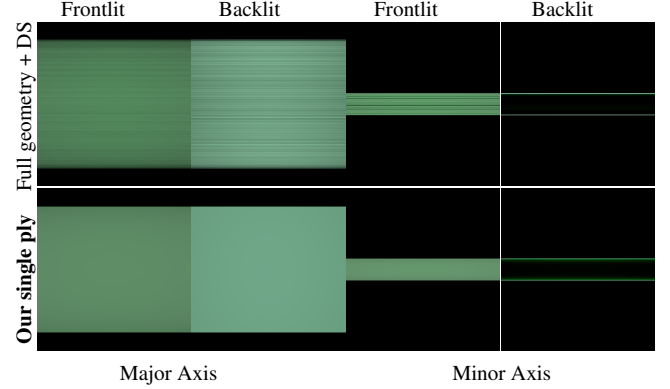


Figure 15: Top and bottom show the results of fiber-level geometry rendered using Dual Scattering and a single highly elliptical ply shaded using our aggregated model, respectively, under various light and view directions.

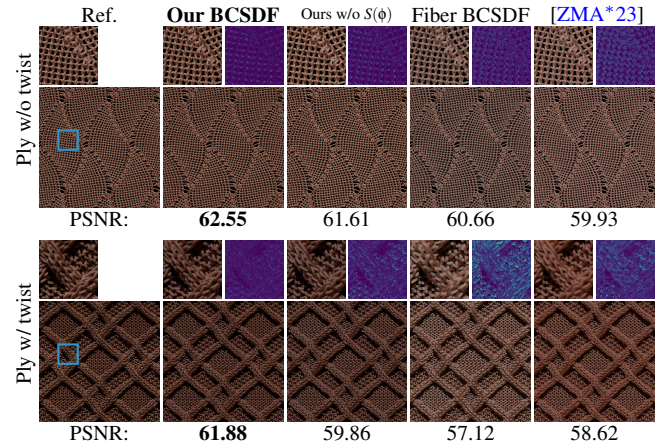


Figure 16: Knit patches: Ref. is obtained using one fiber-level geometry instance with path tracing and single BCSDf, while others render a collection of fibers as a ply with single BCSDf and the aggregated BCSDf [ZMA*23]. All results are generated using CPU path tracing. Our approach produces the closest appearance to GT (highest PSNR). Notably, the absence of the shadowing-masking term leads to a noticeable brightness mismatch. Magnified insets (highlighted in blue) and corresponding error maps are provided.

7. CONCLUSION

We have presented a Level-of-detail framework for real-time strand-based hair rendering. Our approach incorporates a comprehensive BCSDf that models both single and multiple scattering interactions among individual hairs within a hair cluster. Additionally, we have outlined a method for constructing a hierarchical representation of hair and a strategy for selecting appropriate levels of detail and determining the thick hair width on the fly, minimizing geometry data for efficient real-time rendering without perceptible loss in appearance. Furthermore, we have demonstrated that our proposed system can be implemented efficiently on the GPU to render various hairstyles in real time.

Limitations. Our method, aligning with the concept of dual scat-

tering, shares its limitations. In particular, the local scattering in dual scattering theory assumes that the geometric information and reflection model of the fibers surrounding the shading point is essentially consistent, which is not well-suited for hairs with various textured albedo and certain hairstyles, especially curly styles with light hair color, which would introduce biases in the results, as shown in Fig. 17. Also, our pipeline cannot preserve glints perfectly, as our LoD selection criteria are based on the assumption that the hair width is around 2 pixels on the screen to reduce hair geometries, while glints that appear usually need a sub-pixel (<0.5 pixels) details. Additionally, our current dynamic LoD selection ensures that rendered thick hairs remain within a certain width threshold (2 pixels), thereby minimizing the likelihood of one "thick hair" being seen through another. However, if the width threshold is increased, order-independent transparency (OIT) would be necessary to achieve correct rendering results, albeit at the cost of reduced performance. Last, we recognize the disparity between our research prototype and established commercial engines like Unreal Engine [Epi21].

Future Work. For curly or coily hairstyles, our current LoD structure exhibits relatively limited capability in preserving the characteristic features of such hair types. This limitation could potentially be addressed by incorporating more advanced clustering strategies, such as [ZCP*23], during LoD construction. Also, in our current GPU implementation, we adopt a three-layer DOM structure with splitting points determined in the initial frame to support Dual Scattering. Ideally, these splitting points should be updated on a per-frame basis to account for variations in the underlying hair geometry caused by hair motion and LoD transitions. We believe that dynamic adjustment of the splitting points also represents a promising direction for future work.



Figure 17: Our method inherits limitations from dual scattering, particularly in accurately approximating the appearance of curly, light-colored hairs when illuminated from a side light direction.

References

- [BKCN03] BERTAILS F., KIM T.-Y., CANI M.-P., NEUMANN U.: Adaptive wisp tree: A multiresolution control structure for simulating dynamic clustering in hair motion. In *Proceedings of the 2003 ACM SIGGRAPH/Eurographics Symposium on Computer Animation* (Goslar, DEU, 2003), SCA '03, Eurographics Association, p. 207–213. 2
- [BL20] BAUCHET J.-P., LAFARGE F.: Kinetic shape reconstruction. *ACM Trans. Graph.* 39, 5 (jun 2020). 2
- [CBTB16] CHIANG M. J., BITTERLI B., TAPPAN C., BURLEY B.: A practical and controllable hair and fur model for production path tracing. *Comput. Graph. Forum* 35, 2 (2016), 275–283. 2
- [CHPR07] COOK R. L., HALSTEAD J., PLANCK M., RYU D.: Stochastic simplification of aggregate detail. *ACM Trans. Graph.* 26, 3 (jul 2007), 79–es. 2
- [CSAD04] COHEN-STEINER D., ALLIEZ P., DESBRUN M.: Variational shape approximation. *ACM Trans. Graph.* 23, 3 (aug 2004), 905–914. 2
- [dFH*11] D'EON E., FRANÇOIS G., HILL M., LETTERI J., AUBRY J.: An energy-conserving hair reflectance model. *Comput. Graph. Forum* 30, 4 (2011), 1181–1187. 2
- [DJBDDT13] DEROUET-JOURDAN A., BERTAILS-DESCOUBES F., DAVIET G., THOLLOT J.: Inverse dynamic hair modeling with frictional contact. *ACM Trans. Graph.* 32, 6 (Nov. 2013), 159:1–159:10. 8
- [Epi21] EPIC GAMES: Unreal engine, 2021. URL: <https://www.unrealengine.com>. 1, 8, 12
- [GH97] GARLAND M., HECKBERT P. S.: Surface simplification using quadric error metrics. In *Proceedings of the 24th Annual Conference on Computer Graphics and Interactive Techniques (USA, 1997)*, SIGGRAPH '97, ACM Press/Addison-Wesley Publishing Co., p. 209–216. 2
- [HHH22] HUANG W., HULLIN M. B., HANIKA J.: A microfacet-based hair scattering model. *Computer Graphics Forum* 41, 4 (2022), 79–91. 2
- [Hop96] HOPPE H.: Progressive meshes. In *Proceedings of the 23rd Annual Conference on Computer Graphics and Interactive Techniques (New York, NY, USA, 1996)*, SIGGRAPH '96, Association for Computing Machinery, p. 99–108. 2
- [HWP*23] HSU J., WANG T., PAN Z., GAO X., YUKSEL C., WU K.: Sag-free initialization for strand-based hybrid hair simulation. *ACM Trans. Graph.* 42, 4 (jul 2023). 1, 9
- [HYW*23] HUANG L., YANG F., WEI C., CHEN Y. J. E., YUAN C., GAO M.: Towards realtime: A hybrid physics-based method for hair animation on gpu. *Proc. ACM Comput. Graph. Interact. Tech.* 6, 3 (aug 2023). 1
- [Jia16] JIANG Y.: The process of creating volumetric-based materials in uncharted 4. *Siggraph Courses: Advances in Real-Time Rendering in Games, Anaheim, CA 0, 0* (2016), 0. 1, 2
- [KH01] KOH C. K., HUANG Z.: A simple physics model to animate human hair modeled in 2d strips in real time. In *Proceedings of the Eurographic Workshop on Computer Animation and Simulation* (Berlin, Heidelberg, 2001), Springer-Verlag, p. 127–138. 2
- [KJA*23] KT A., JARABO A., ALIAGA C., CHIANG M. J.-Y., MAURY O., HERY C., NARAYANAN P. J., NAM G.: Accelerating hair rendering by learning high-order scattered radiance. *Computer Graphics Forum* 42, 4 (2023), e14895. 2
- [KK89] KAJIYA J. T., KAY T. L.: Rendering fur with three dimensional textures. *SIGGRAPH Comput. Graph.* 23, 3 (jul 1989), 271–280. 2
- [KM17] KHUNGURN P., MARSCHNER S.: Azimuthal scattering from elliptical hair fibers. *ACM Trans. Graph.* 36, 2 (apr 2017). 2
- [KSW21] KARIS B., STUBBE R., WIHLIDAL G.: A deep dive into nanite virtualized geometry. *ACM SIGGRAPH 2021 Talks* 0, 0 (2021), 0. 2
- [KSZ*15] KHUNGURN P., SCHROEDER D., ZHAO S., BALA K., MARSCHNER S.: Matching real fabrics with micro-appearance models. *ACM Trans. Graph.* 35, 1 (2015), 1:1–1:26. 2
- [LN21] LI M., NAN L.: Feature-preserving 3D mesh simplification for urban buildings. *ISPRS Journal of Photogrammetry and Remote Sensing* 173 (Mar. 2021), 135–150. 2
- [LWC*02] LUEBKE D., WATSON B., COHEN J. D., REDDY M., VARSHNEY A.: *Level of Detail for 3D Graphics*. Elsevier Science Inc., USA, 2002. 2
- [MGJZ21] MONTAZERI Z., GAMMELMARK S., JENSEN H. W., ZHAO S.: Practical ply-based appearance modeling for knitted fabrics. *Computer Graphics Forum* 42, 4 (July 2021), 2–11. 2

- [MGZJ20] MONTAZERI Z., GAMMELMARK S. B., ZHAO S., JENSEN H. W.: A practical ply-based appearance model of woven fabrics. *ACM Trans. Graph.* 39, 6 (nov 2020). 2
- [MJC*03] MARSCHNER S. R., JENSEN H. W., CAMMARANO M., WORLEY S., HANRAHAN P.: Light scattering from human hair fibers. *ACM Trans. Graph.* 22, 3 (2003), 780–791. 2, 3, 7, 9
- [MM06] MOON J. T., MARSCHNER S. R.: Simulating multiple scattering in hair using a photon mapping approach. *ACM Trans. Graph.* 25, 3 (2006), 1067–1074. 2
- [MWM08] MOON J. T., WALTER B., MARSCHNER S.: Efficient multiple scattering in hair using spherical harmonics. In *ACM SIGGRAPH 2008 Papers* (New York, NY, USA, 2008), SIGGRAPH '08, Association for Computing Machinery. 2
- [NW17] NAN L., WONKA P.: Polyfit: Polygonal surface reconstruction from point clouds. In *2017 IEEE International Conference on Computer Vision (ICCV)* (Venice, Italy, 2017), IEEE, pp. 2372–2380. 2
- [Sch04] SCHEUERMANN T.: Practical real-time hair rendering and shading. In *ACM SIGGRAPH 2004 Sketches* (New York, NY, USA, 2004), SIGGRAPH '04, Association for Computing Machinery, p. 147. 1
- [Som15] SOMASUNDARAM A.: Dynamically controlling hair interpolation. In *ACM SIGGRAPH 2015 Talks* (New York, NY, USA, 2015), SIGGRAPH '15, Association for Computing Machinery. 5
- [SPJT10] SADEGHI I., PRITCHETT H., JENSEN H. W., TAMSTORF R.: An artist friendly hair shading system. In *ACM SIGGRAPH 2010 Papers* (New York, NY, USA, 2010), SIGGRAPH '10, Association for Computing Machinery. 2
- [SSW*23] SHEN Y., SAITO S., WANG Z., MAURY O., WU C., HODGINS J., ZHENG Y., NAM G.: Ct2hair: High-fidelity 3d hair modeling using computed tomography. *ACM Trans. Graph.* 42, 4 (jul 2023). 8
- [Taf19] TAFURI S.: Strand-based hair rendering in frostbite. *ACM SIGGRAPH Courses: Advances in Real-Time Rendering in Games Course* 0, 0 (2019), 0. 1
- [TS22] TATARCHUK N., SCHNEIDER A.: Advances in real-time rendering in games: part ii. In *ACM SIGGRAPH 2022 Courses* (New York, NY, USA, 2022), SIGGRAPH '22, Association for Computing Machinery. 1
- [WLJ*03] WARD K., LIN M. C., JOOHI L., FISHER S., MACRI D.: Modeling hair using level-of-detail representations. In *Proceedings 11th IEEE International Workshop on Program Comprehension* (New Brunswick, NJ, USA, 2003), IEEE, pp. 41–47. 2
- [WWB*14] WALD I., WOOP S., BENTHIN C., JOHNSON G. S., ERNST M.: Embree: a kernel framework for efficient cpu ray tracing. *ACM Trans. Graph.* 33, 4 (jul 2014). 6
- [WY17] WU K., YUKSEL C.: Real-time fiber-level cloth rendering. In *Proceedings of the 21st ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games* (New York, NY, USA, 2017), I3D '17, Association for Computing Machinery. 1
- [WYZG09] WANG L., YU Y., ZHOU K., GUO B.: Example-based hair geometry synthesis. In *ACM SIGGRAPH 2009 Papers* (New York, NY, USA, 2009), SIGGRAPH '09, Association for Computing Machinery. 5
- [WZK*23] WEIER P., ZIRR T., KAPLAYAN A., YAN L.-Q., SLUSALLEK P.: Neural prefiltering for correlation-aware levels of detail. *ACM Trans. Graph.* 42, 4 (jul 2023). 2
- [XWH*23] XIA M., WALTER B., HERY C., MAURY O., MICHIELSEN E., MARSCHNER S.: A practical wave optics reflection model for hair and fur. *ACM Trans. Graph.* 42, 4 (jul 2023). 2
- [XWM*20] XIA M. M., WALTER B., MICHIELSEN E., BINDEL D., MARSCHNER S.: A wave optics based fiber scattering model. *ACM Trans. Graph.* 39, 6 (nov 2020). 2
- [YK08] YUKSEL C., KEYSER J.: Deep opacity maps. *Computer Graphics Forum (Proceedings of EUROGRAPHICS 2008)* 27, 2 (2008), 675–680. 2
- [YTJR15] YAN L.-Q., TSENG C.-W., JENSEN H. W., RAMAMOORTHY R.: Physically-accurate fur reflectance: Modeling, measurement and rendering. *ACM Trans. Graph.* 34, 6 (nov 2015). 2, 3
- [ZCP*23] ZHOU Y., CHAI M., PEPE A., GROSS M., BEELER T.: Groomgen: A high-quality generative hair model using hierarchical latent representations. *ACM Trans. Graph.* 42, 6 (Dec. 2023). 12
- [ZMA*23] ZHU J., MONTAZERI Z., AUBRY J., YAN L.-Q., WEIDLICH A.: A practical and hierarchical yarn-based shading model for cloth. *Computer Graphics Forum* 42, 4 (July 2023), 2–11. 1, 2, 3, 4, 10, 11
- [ZRL*09] ZINKE A., RUMP M., LAY T., WEBER A., ANDRIYENKO A., KLEIN R.: A practical approach for photometric acquisition of hair color. *ACM Trans. Graph.* 28, 5 (2009), 165. 2
- [ZYWK08] ZINKE A., YUKSEL C., WEBER A., KEYSER J.: Dual scattering approximation for fast multiple scattering in hair. *ACM Trans. Graph.* 27, 3 (2008), 32. 1, 2, 3, 6
- [ZZW*22] ZHU J., ZHAO S., WANG L., XU Y., YAN L.: Practical level-of-detail aggregation of fur appearance. *ACM Trans. Graph.* 41, 4 (2022), 47:1–47:17. 1, 2