Eurographics Symposium on Rendering 2025
B. Wang and A. Wilkie
(Guest Editors)

COMPUTER GRAPHICS *forum*
Volume 44 (2025), Number 4

Detail-Preserving Real-Time Hair Strand Linking and Filtering

T. Huang[*1,2], J. Yuan[*3], R. Hu[3], L. Wang[1], Y. Guo[3], B. Chen[4], J. Guo[†3], J. Zhu[‡5,1]

[1]Shandong University, China
[2]Mohamed bin Zayed University of Artificial Intelligence
[3]Nanjing University, China
[4]The University of Melbourne, Australia
[5]University of California Santa Barbara, US

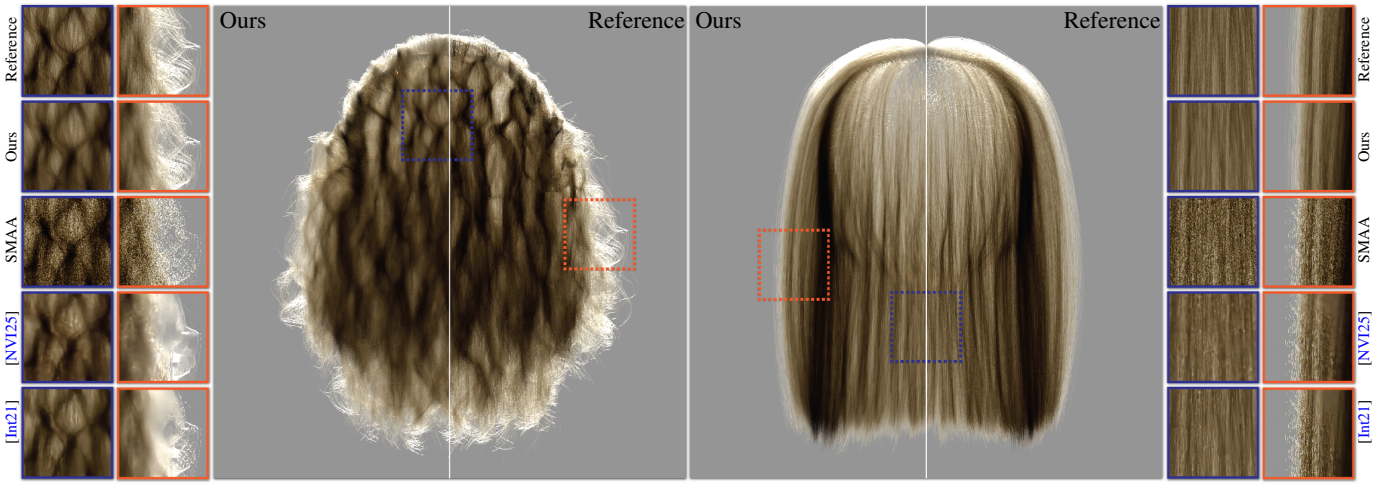

Figure 1: *We showcase the linking and filtering effects of our method in scenes with straight and curly hair. The input is a 1 spp noisy image. We compare our approach against SMAA, the optix denoiser [NVI25], and Oidn denoising methods [Int21]. Our technique not only effectively reconstructs broken flyaway strands but also preserves finer details in areas with dense hair.*

Abstract

Realistic hair rendering remains a significant challenge in computer graphics due to the intricate microstructure of hair fibers and their anisotropic scattering properties, which make them highly sensitive to noise. Although recent advancements in image-space and 3D-space denoising and antialiasing techniques have facilitated real-time rendering in simple scenes, existing methods still struggle with excessive blurring and artifacts, particularly in fine hair details such as flyaway strands. These issues arise because current techniques often fail to preserve sub-pixel continuity and lack directional sensitivity in the filtering process. To address these limitations, we introduce a novel real-time hair filtering technique that effectively reconstructs fine fiber details while suppressing noise. Our method improves visual quality by maintaining strand-level details and ensuring computational efficiency, making it well-suited for real-time applications in video games and virtual reality (VR) and augmented reality (AR) environments.

CCS Concepts

• ***Computing methodologies*** → *Collision detection;* • ***Hardware*** → *Sensors and actuators; PCB design and layout;*

† Corresponding author: guojie@nju.edu.cn.

‡ Corresponding author: zhujunqiu@mail.sdu.edu.cn.

* Dual First Authors.

1. Introduction

Hair plays a critical role in the perceptual fidelity of virtual characters in real-time rendering applications. However, real-time rendering of hair remains a fundamental challenge in computer graphics. Due to the complex interplay of high-frequency geometry and intricate light transport, hair fibers exhibit fine-scale structures and anisotropic scattering, which makes Monte Carlo path tracing highly susceptible to noise. Although recent advances in denoising techniques [V*18, Cor25a, MZV*20, Cor25b, Cor24] have enabled some level of real-time performance, existing approaches often suffer from excessive blurring and perceptual artifacts. In particular, fine strands, especially flyaway fibers, are prone to fragmentation or complete loss, severely degrading visual fidelity. Even state-of-the-art anti-aliasing [Cor25a, KB83] and reconstruction techniques [WLL*21, TN23] fall short in addressing these challenges, failing to preserve the fidelity of thin structures under dynamic lighting and viewpoint changes (Figure 1).

In this work, we introduce a real-time framework designed to reconstruct strand-level details while effectively suppressing noise robustly. First, for all pixels that are mistakenly identified as missed, we select the best candidate strands and extrapolate these strands to fill these pixels. Then we analyze screen-space information and employ an effective pixel coverage approximation scheme that ensures our results closely approximate the ground truth in an unbiased manner. Finally, guided by hair-specific geometric priors, we design a specialized filtering kernel that significantly reduces noise while successfully preserving fine-grained strand details.

We summarize our contributions as follows:

- **Strand Reconstruction via Geometric Extrapolation**: We propose a screen-space reconstruction technique that extrapolates strand parameters (e.g., tangent direction, ID, and parametric position) to address the fragmentation and complete loss of fine strands, so as to maintain structural coherence without relying on high sampling rate.
- **Pixel Coverage Estimation**: We present an efficient approximation technique that enables accurate and temporally stable estimation of pixel coverage for individual hair strands. Notably, our method relies solely on information extracted from a 1 sample per pixel (1 spp) rendering input, making it highly suitable for real-time applications with limited computational budgets.
- **Orientation-Aware Anisotropic Filtering**: We design an elliptical-shaped filtering kernel that jointly considers spatial proximity, directional alignment, and chromatic similarity, effectively reducing noise while preserving fine strand details.

2. Related Work

2.1. Hair Appearance Modeling

Hair appearance modeling is a complex task in computer graphics, typically divided into two primary areas: modeling individual hair fibers and simulating the multiple scattering interactions within hair. The foundation of modern hair rendering was established by Marschner et al. [MJC*03], who modeled hair fibers as rough dielectric cylinders and decomposed light interaction into multiple scattering lobes using angular parameterization. Their work introduced the separation of the bidirectional scattering distribution function (BSDF) into longitudinal and azimuthal scattering, laying the groundwork for realistic hair rendering.

Building on this framework, d'Eon et al. [dFH*11] proposed an energy-conserving model for hair, refining the physical accuracy of hair light transport. Yan et al. [YTJR15, YJR17] extended this to fur rendering, incorporating both near-field and far-field scattering effects to improve realism in hair and fur rendering under dynamic lighting conditions.

In recent years, research has focused on refining the geometric and optical properties of hair fibers. Huang et al. [HHH22] introduced a microfacet-based scattering model that more accurately captures the surface roughness and self-shadowing effects of individual hair strands. Benamira et al. [BP21] advanced hair modeling by combining scattering and diffraction models for elliptical hair fibers, enabling more precise rendering of anisotropic highlights and fine-scale details.

For high-density hair and fur rendering in production environments, Zhu et al. [ZZW*22] developed level-of-detail techniques for fur aggregation, making the scalable rendering of complex hair geometry more feasible. These approaches have been particularly valuable for applications in animation and visual effects.

In the context of real-time rendering, modern approaches aim to balance physical accuracy with computational constraints. Chiang et al. [CJL*16] demonstrated a GPU-friendly strand tracing technique using linear depth peeling, enhancing the efficiency of hair rendering. Karis [Kar13] introduced screen-space decomposition, which allows for the integration of multiple scattering in real-time rendering. Bhokare et al. [BMDY24] propose a real-time method for generating and representing individual hair strands with level-of-detail support, significantly facilitating efficient hair rendering. Our work extends these concepts with visibility buffer fitting, addressing sub-pixel discontinuities through parametric propagation, thereby improving the visual fidelity of real-time hair rendering.

Multiple scattering in hair rendering has been further advanced by Zinke and Yuksel's dual scattering framework [ZYWK08], which decouples local single scattering from global multiple scattering. Recent innovations have optimized these models using procedural and data-driven techniques. For instance, Zhu et al. [ZZW*22] proposed screen-space strand clustering with velocity-aware level-of-detail (LoD) transitions and mipmapped scattering kernels to preserve anisotropic appearance. Huang et al. [HZL*24] proposed a guide-hair-based LoD construction and a screen-space selection strategy, achieving up to 10× acceleration for strand-based hair rendering at far view distances while maintaining visual quality. KT et al. [KJA*23] employed neural networks to predict high-order scattering components from lower-order terms, effectively mapping scattering hierarchies to improve computational efficiency.

These works collectively provide the physical and algorithmic foundations for realistic hair rendering. In particular, accurate modeling of multiple scattering plays a crucial role in noise reduction. However, due to the sub-pixel scale and high-frequency nature of hair geometry, practical renderers often face insufficient sampling rates, leading to severe aliasing and noise. This challenge remains

a significant bottleneck for real-time hair rendering, and addressing it forms the core motivation for our approach.

2.2. Hair Filtering and Antialiasing

The refinement of hair image quality in rendering has often drawn upon techniques from image filtering and anti-aliasing (AA), which aim to restore fine details while mitigating aliasing artifacts. However, most filtering and AA algorithms are designed for general image processing and rarely consider the unique properties of hair. Image denoising and sparse sampling have been actively researched in rendering, particularly to reduce variance in Monte Carlo rendering while preserving fine details. A wide range of filtering methods has been proposed and systematically categorized into screen space domains [SZR*15], and path space [KDB16]. Among existing approaches, we place emphasis on those most relevant to our method, especially techniques that address sparse sampling and detail-preserving denoising in real-time rendering.

For real-time rendering, denoising typically employs edge-preserving filters [BEJM15, DSHL10], which are effective for thin, translucent surfaces. However, hair rendering poses a distinct challenge, as its geometry is inherently composed of edges, leading to sparse sampling of the scene.

To address the sparse sampling issue, some approaches leverage auxiliary features such as positions and normals [GO12], with machine learning algorithms used to optimize filter parameters [KBS15]. Bako et al. [BVM*17] adopted convolutional neural networks (CNNs) to learn the optimal filter kernels for offline rendering, while Chaitanya et al. [CKS*17] introduced recursive CNNs that achieved high-quality denoising for sparse samples in real-time rendering. Zeng et al. [ZWW*20] propose to use convolutional neural networks with different receptive fields for different scales of noises.

The only work that directly focuses on hair filtering presented by Roc R. Currius et al. [CAS22] explored the use of neural networks. Although promising, this approach introduced additional blurring and energy variation, and the time required to run a single network was longer than the rendering time itself.

Alias artifacts caused by insufficient sampling of fine structures, such as individual hair strands, can be considered a form of unresolved subpixel detail, thus falling under the broader challenges of AA. As demand for high-fidelity real-time rendering increases, screen-space AA techniques have become a practical alternative to traditional multisample anti-aliasing (MSAA) [FT20]. Morphological anti-aliasing (MLAA) [Res09, JME*18] and its derivatives, such as FXAA [Lot09], SRAA [CML11] employ post-processing filters to detect geometric edges and interpolate across them to suppress aliasing. These methods have become widely adopted in real-time graphics pipelines due to their balance between visual quality and performance. However, their effectiveness remains limited in scenes involving semi-translucent, high-frequency elements like hair, where aliasing artifacts persist despite edge detection. Although FXAA integrates real-time hardware optimization, and SRAA improves edge detection via multi-sample buffers, these methods often fail to preserve the intricate structure of hair, leading to blurring or residual aliasing. This limitation underscores the unique challenges faced in hair rendering, calling for specialized anti-aliasing techniques. SMAA [JESG12] introduced more precise pattern detection and temporal feedback, but the problem of faithfully reconstructing subpixel-level hair geometry remains unresolved, particularly under low sampling budgets or oblique viewing conditions. Additionally, temporal anti-aliasing (TAA) is frequently employed for both denoising and anti-aliasing [YLS20], yet it faces complications in hair rendering due to the severely fragmented artifacts by fine hair strands, complicating its correspondence with historical buffers.

3. Problem Formulation and Analysis

Rendering high-fidelity hair images requires faithfully capturing all physically-based shading effects for each individual strand. This is particularly challenging due to the extremely fine and complex geometry of hair. A core difficulty lies in accurately computing the radiance received by each pixel, especially when a strand only partially overlaps a pixel.

Let P denote a pixel's footprint on the image plane. The pixel intensity I_P is defined as the average outgoing radiance over this footprint along the camera viewing direction ω_p:

$$I_P = \frac{1}{|P|}\int_P L_o(x_p, \omega_p)\, dx_p, \tag{1}$$

where $L_o(x_p, \omega_p)$ is the outgoing radiance at point x_p in direction ω_p, and $|P|$ is the pixel's area.

To illustrate the challenge, we first analyze a simplified case where a single strand intersects the pixel alongside the background. Let $P^+ \subset P$ be the subregion occupied by the hair strand, and $P^- := P \setminus P^+$ be the remaining background region. The pixel intensity then becomes:

$$\begin{aligned} I_P &= \frac{1}{|P|}\left(\int_{P^-} L_b(-\omega_p)\, dx_p + \int_{P^+} L_o(x_p, \omega_p)\, dx_p\right) \\ &\approx \frac{|P^-|}{|P|} L_b(-\omega_p) + \frac{1}{|P|}\int_{P^+} L_o(x_p, \omega_p)\, dx_p, \end{aligned} \tag{2}$$

where $L_b(-\omega_p)$ denotes the background radiance, assumed to be approximately constant over P^-.

Accurately computing this expression introduces two major challenges. First, as demonstrated in [MJC*03], hair exhibits high-frequency shading variations due to glints and inter-reflections, making it difficult to estimate the second term in Euqation 2 using traditional Monte Carlo integration without introducing noise. Second, at typical viewing distances, the projected width of a single strand is often smaller than the pixel size, which complicates the estimation of pixel coverage $C = \frac{|P^+|}{|P|}$.

Moreover, in practical scenarios, when multiple hair strands intersect within a single pixel (i.e., in dense regions), the pixel coverage can be effectively handled using standard rendering techniques. However, in sparse regions where only a few strands contribute to the pixel coverage, this approach becomes insufficient due to the lack of sufficient information from the G-buffer. To address this, we focus on enhancing the estimation for these low-coverage regions by supplementing the rendering with single-strand contributions,

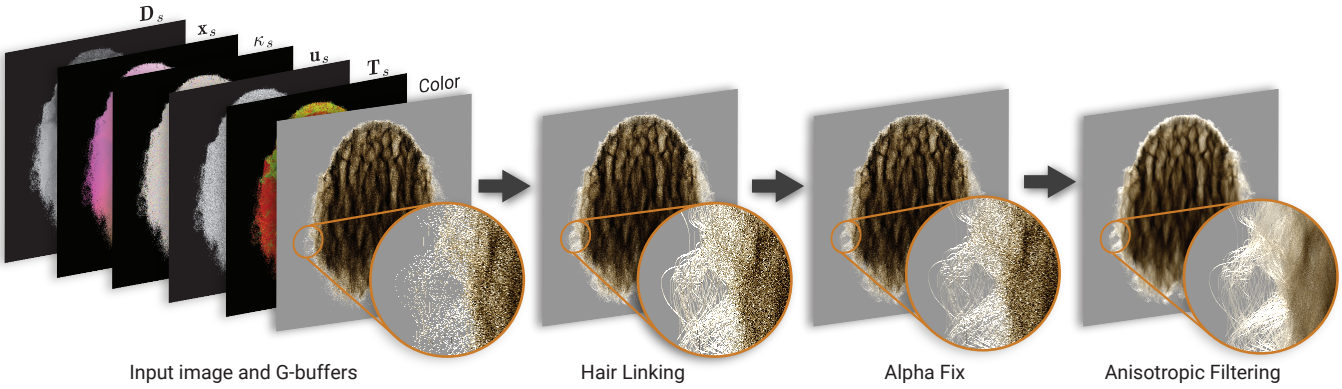

Figure 2: ***Overview of our hair linking and filtering pipeline.*** *Starting from G-buffer data and a 1-spp input image, we first link disconnected flyaway fibers via a hair linking procedure. Next, an alpha fix step enforces energy conservation, mitigating transparency-induced artifacts. Finally, we apply anisotropic filtering to reduce noise while preserving high-frequency hair details.*

thus ensuring more accurate shading and reducing artifacts. We begin by analyzing the rendering results under a 1spp constraint, denoted as $\mathbb{I}_c = \{I_0, I_1, ..., I_n\}$, and apply this approach specifically to the sparse regions.

For each pixel, we have an associated G-buffer:

$$G_P = \{\mathbf{D}_s, \mathbf{x}_s, \mathbf{T}_s, \kappa_s, \mathbf{u}_s\}, \tag{3}$$

where $\mathbf{D}_s$ is the sample depth, $\mathbf{x}_s$ is the world-space position, $\mathbf{T}_s$ is the local tangent, κ_s is the strand ID, and $\mathbf{u}_s$ is the curve parameter along the strand. The final pixel intensity can thus be modeled as a function of this data: $I_P = \Phi(G_P)$.

In dense hair regions where strands dominate the pixel footprint (e.g., near the scalp), TAA [YLS20] and similar techniques often yield acceptable results. However, in sparse regions—particularly at the hair contours where strand coverage is low—these methods struggle due to insufficient information in the G-buffer. This leads to noticeable aliasing and loss of fine details.

To address these limitations, we first introduce a **hair linking** step (Section 4.1) that recovers pixels mistakenly classified as background by associating them with the correct hair strands. This allows us to build an enhanced G-buffer $\hat{G}_P$ by propagating accurate per-strand attributes to these previously underrepresented pixels. Using this updated information, we can also estimate the strand coverage C_P more reliably.

In addition, we propose a fine elliptical-shaped filter kernel tailored to the anisotropic structure of hair (Section 4.3). This kernel accounts for the local tangent orientation and spatial extent of each strand, further reducing shading artifacts and aliasing in low-coverage regions.

4. Our Method

In this section, we detail each component of our proposed method, which consists of three main stages: Hair Linking, Alpha Fix and Orientation-Aware Anisotropic Filtering (Figure 2).

4.1. Hair Linking

As shown in Figure 3, to recover missing pixels P_{empty} in the initial G-buffer $\mathbb{I}_c$, we identify candidate strands in the 3x3 neighborhood around P_{empty} and select the one most likely to intersect it. This serves as the basis for initializing strand propagation.

Strand Selection. We assume that a single hair strand projected into screen space appears smoothly and continuously. Based on this observation, we make two assumptions:

- Tangent alignment: A valid strand's tangent direction should align with the vector connecting it to P_{empty}.
- Spatial proximity: A valid strand is likely to be spatially close to P_{empty} in screen space.

We compute an alignment score ϵ that combines these criteria:

$$\epsilon_i = \underbrace{\frac{(S_{empty} - S_i) \cdot \mathbf{T}_i^{proj}}{|S_{empty} - S_i|}}_{\text{tangent alignment}} \cdot \underbrace{\exp\left(-\frac{\|S_i - S_{empty}\|^2}{2\sigma^2}\right)}_{\text{spatial proximity}} \tag{4}$$

Here, S_i and S_{empty} are the screen-space positions of sample s_i and the center of P_{empty}, and $\mathbf{T}_i^{proj}$ is the screen-space projected tangent of strand κ_i.

We select the source sample s_{src} and its strand κ with the highest score above a threshold ϵ_{align}:

$$s_{\text{src}} = \arg\max_{s_i} \epsilon_i \quad \text{subject to} \quad \epsilon_i > \epsilon_{\text{align}}. \tag{5}$$

If no candidate exceeds the threshold, P_{empty} is not filled. In this paper we set ϵ_i to 0.5 across all scenes.

Once the optimal source sample is identified, we proceed to propagate along strand κ to determine which other pixels it intersects, as described below.

Adaptive Propagation After selecting a valid candidate strand κ, we aim to fill in missing pixels it traverses. Direct intersection tests between a strand and pixel boundaries are expensive, so we opt

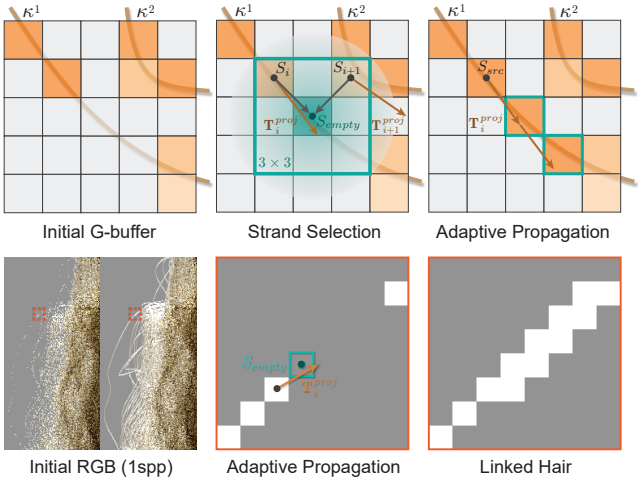

Figure 3: ***Illustration of Hair Linking****. When sampling rates are insufficient, thin flyaway hairs may be entirely missed, leading to visible discontinuities. We illustrate our hair linking strategy, which identifies missing pixels (Strand Selection) along fibers and adaptively extends the fiber forward (Adaptive Propagation) to restore geometric continuity. The last row shows the hair linking process and results on rendered image.*

for an efficient point-sampling approach along the strand's parametric curve. Our goal is not to verify whether a single pixel intersects with the curve, but rather to identify all fillable pixels that the curve passes through during propagation.

We initialize the parametric sampling step Δu based on the screen-space distance between P_{empty} and s_{src}:

$$\Delta u_0 = \frac{\|\mathbf{M}^{-1}(S_{empty}) - \mathbf{M}^{-1}(S_{src})\|}{\text{Length}(\kappa)} \tag{6}$$

where $\mathbf{M}^{-1}$ is the inverse camera projection matrix, Length(κ) is the length of this segment in world space.

We then iteratively adjust the step size based on screen-space distance between successive samples to prevent over- or under-sampling:

$$\Delta u_i \leftarrow \frac{\Delta u_{i-1}}{\|s_i - s_{i-1}\|} \tag{7}$$

Propagation Rules. Starting from s_{src}, we propagate forward (or backward) along the curve according to the strand's tangent direction. At each step, we determine if the sampled point lies within an empty pixel. If so, we assign the strand's attributes to that pixel.

The propagation terminates under one of the following conditions:

- The curve reaches its endpoint (i.e., the parametric u value reaches 0 or 1).
- The next projected point falls outside the image boundary.
- The sampled pixel is already filled.

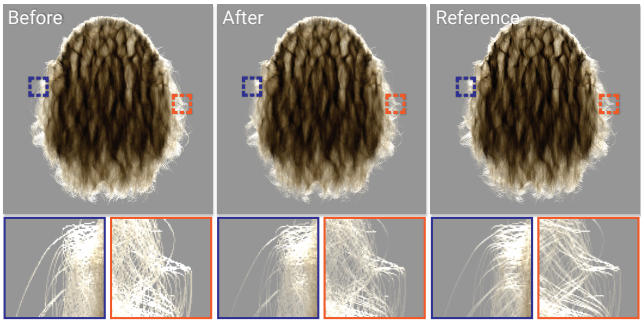

Figure 4: ***Effect of Alpha Fix.*** *Left: before alpha fix; middle: with alpha fix; right: reference. Due to the subpixel width of individual hair strands, transparency conveys the perceived thinness of hair. Without alpha correction, flyaway fibers appear overly thick, and regions of moderate density become unnaturally bright due to accumulated opacity. Our alpha fix enforces energy conservation, resulting in more visually plausible, softer hair with consistent radiance.*

This ensures efficiency while avoiding the introduction of spurious strands.

4.2. Alpha Fix

When filling empty pixels through strand propagation, our goal is to avoid introducing obvious energy increase in Eq. 2. Instead of accumulating albedo or radiance (which could lead to over-saturation), we adopt an alpha blending scheme. In this approach, newly filled pixels are treated as partially covered by the hair strand and partially by the background. By blending the hair and background colors according to the strand's original coverage, we ensure the added pixels do not make the hair appear brighter than intended. For each strand κ, let M denote the number of pixels originally rasterized from the strand, and N the number of additional pixels filled in via propagation. We define a strand-wise alpha as:

$$\alpha = \frac{M}{M+N} \tag{8}$$

This α represents the fraction of the strand's original coverage over the total post-propagation coverage. For each pixel i in $M \cup N$, we compute its final shaded color as:

$$C_{\text{final}}(i) = \alpha \cdot C_{\text{hair}} + (1-\alpha) \cdot C_{\text{bg}}(i) \tag{9}$$

Here, C_{hair} is the average hair color computed from the originally rasterized pixels (M), and $C_{\text{bg}}(i)$ is the average background color in the vicinity of pixel i. Even for pixels originally belonging to the strand ($i \in M$), we do not use the exact background value hidden behind the hair. Instead, we estimate $C_{\text{bg}}(i)$ by sampling background regions around the pixel. This strand-wise alpha blending strategy conserves the total energy contributed by each strand. The use of α for hair color guarantees that the total radiance across $M+N$ pixels matches the original radiance across M pixels:

$$\sum_{i \in M} C_{\text{hair}} = \alpha \sum_{i \in M \cup N} C_{\text{hair}} \tag{10}$$

By smoothly blending in background color based on α, we prevent over-saturation and ensure that no region becomes unnaturally bright or transparent. The result (Figure 4) is a visually consistent rendering of hair strands after propagation, maintaining both physical plausibility and perceptual quality, avoiding hard transitions between original and propagated areas. Figure 8 demonstrates that the alpha correction stage remains effective even in the presence of complex backgrounds.

4.3. Orientation-Aware Anisotropic Filtering

Finally, we apply a filtering stage to reduce residual noise and artifacts in the alpha matte, preserving fine hair structures. To achieve this, we perform anisotropic filtering aligned with each strand's screen-space tangent direction, denoted as $\mathbf{T}_p$, to avoid cross-strand blurring while maintaining the intricate details of hair (as shown in Figure 5). This anisotropic filtering method contrasts with traditional isotropic filters, which tend to smooth across both strand directions, leading to undesirable artifacts in the case of hair-like structures.

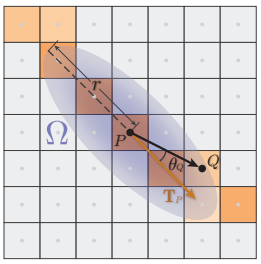

Figure 5: *Anisotropic filtering.*

In our approach, we design an elliptical kernel, oriented along the direction of $\mathbf{T}_p$, with the following parameters:

- r: the length of the kernel's major axis.
- ρ: the aspect ratio of the elliptical kernel.

Filtering is performed strictly within the bounds of the elliptical region, ensuring that smoothing occurs predominantly along the direction of hair growth and minimizes blurring of neighboring strands. The result is a visually coherent matte that enhances the fine details of hair while removing high-frequency noise. This step enhances visual coherence, eliminates high-frequency noise, and prepares the output matte for robust downstream applications such as compositing or neural rendering.

In our filtering kernel, the composite weight w_q for each neighboring pixel Q within the kernel domain $\Omega(P)$ is determined by three factors:

- **Spatial attenuation**: $\exp\left(-\frac{d_P^2}{\sigma_d^2}\right)$, where $d_P = \|P - Q\|$ is the screen-space Euclidean distance between pixels P and Q,
- **Directional alignment**: $\exp\left(-\frac{\theta_Q^2}{\sigma_\theta^2}\right)$, where $\theta_Q = \angle(\mathbf{T}_P, Q - P)$ is the angle between the tangent direction $\mathbf{T}_P$ and the vector from P to Q,
- **Color similarity**: $\exp\left(-\frac{\|C_P - C_Q\|^2}{\sigma_c^2}\right)$, where C_P and C_Q are the RGB color vectors of pixels P and Q.

The final weight w_q is computed as the product of these three factors:

$$w_q = \exp\left(-\frac{d_P^2}{\sigma_d^2}\right) \cdot \exp\left(-\frac{\theta_Q^2}{\sigma_\theta^2}\right) \cdot \exp\left(-\frac{\|C_P - C_Q\|^2}{\sigma_c^2}\right). \quad (11)$$

Here, σ_d, σ_θ, and σ_c are user-defined parameters that control the influence of spatial distance, directional alignment, and color similarity, respectively. In this paper, we set the parameters σ_d, σ_θ, and σ_c to 0.1, 0.1, and 0.9, across all scenes. These factors ensure that pixels closer in screen space, aligned with the hair tangent, and with similar colors have a greater contribution to the filtered result.

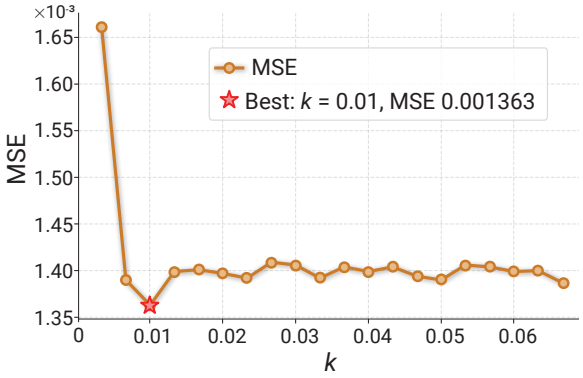

Figure 6: ***Selection of Filter Kernel Size*** *k. To determine an appropriate kernel size for filtering, we evaluate the denoised result under different kernel radii by computing the mean squared error (MSE) against a reference image. The kernel size yielding the lowest MSE is selected.*

The final radiance at pixel P is computed as the normalized weighted average of neighboring pixel colors:

$$\tilde{C}_P = \frac{\sum_{q \in \Omega(P)} w_q C_Q}{\sum_{q \in \Omega(P)} w_q}. \quad (12)$$

The length of the major axis of the filtering kernel r is dynamically chosen based on the screen-space projection of the hair, denoted as $\mathcal{B}_{\text{Hair}}$. This bounding box represents the extent of the hair in screen space and is computed as the projection of the hair geometry onto the 2D image plane. Then r is selected using a heuristic based on the screen distance to the object, with the following formula:

$$r = k \times \sqrt{\mathcal{B}_{\text{Hair},x} \times \mathcal{B}_{\text{Hair},y}}, \quad (13)$$

where $\mathcal{B}_{\text{Hair},x}$ and $\mathcal{B}_{\text{Hair},y}$ are the dimensions of the hair's bounding box along the x- and y-axes, respectively, and k is a scaling factor determined experimentally. In our implementation, k is typically set to approximately 0.01 which corresponds to about one-tenth of the bounding box's dimensions. The selection of k was derived by testing a range of values and optimizing the filtering performance based on mean squared error (MSE). (Figure 6 illustrates our selection process for the filter kernel size k). We test N parameter sets and select the one yielding the lowest MSE across a set of test cases. This adaptive approach allows the filter size to vary with the thickness and distance of the hair in the scene, ensuring that the filtering operation is neither too coarse (leading to oversmoothing) nor too fine (causing artifacts due to insufficient smoothing).

5. Implementation Details

This section outlines the details of our pipeline implementation.

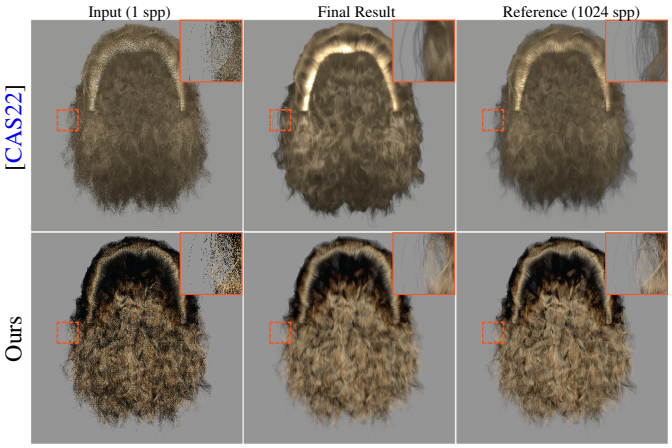

Figure 7: ***Comparison with State-of-the-Art Neural Hair Filtering Method****. We compare our method against a state-of-the-art neural network-based hair filtering approach [CAS22]. Note that each method is evaluated with its corresponding input and reference, as the neural model was specifically trained on hair materials without dual scattering and under their own rendering setup. The neural baseline exhibits excessive blurring and characteristic grid-like artifacts, with notable energy inconsistencies around the crown area. While it can reconstruct missing flyaway fibers, their intensity is inaccurate and visually diluted. In contrast, our method preserves the fine structure of flyaway fibers with correct energy and reconstructs sharper and more faithful primary hair regions.*

Figure 8: *Results of our method under complex lighting and background conditions. Despite the presence of noisy inputs caused by intricate environmental illumination, our approach robustly achieves strand linking and denoising for fly-away fibers, preserving fine hair structures. Additionally, under complex backgrounds, our alpha refinement module effectively corrects hair transparency, maintaining accurate matting.*

5.1. Hair Linking

For the hair linking stage, we employ pixel-level parallelism to ensure computational efficiency, making the approach highly compatible with GPU architectures and scalable to high-resolution renderings. Furthermore, since our reconstruction relies solely on screen-space attributes, we compress the G-buffer content using an RGBA32 texture. Through bitwise encoding, multiple values are compactly packed into the texture, allowing for significantly more efficient memory utilization.

- **Channels RG**: Encode screen-space curve normals (2x32-bit floating point)
- **Channel B**: Stores the curve's parametric coordinate $t \in [0,1]$ (32-bit floating point)
- **Channel A**: Bitmask field containing:
  - *Bit 0*: Hair presence flag (1 = hair pixel, 0 = non-hair)
  - *Bit 1*: Fitting source indicator (1 = filled by algorithm, 0 = original)
  - *Bits 2-31*: Primitive ID storage (30-bit integer for hair strand identification)

In our implementation, we adopt a visibility buffer strategy, originally developed for hair rendering [SDJ19], to effectively support multiple spp. Specifically, each pixel on the visibility buffer stores at most one fragment, ensuring unambiguous reconstruction even in multi-sample settings. Multiple visibility buffer pixels are then aggregated onto a single screen pixel, enabling multi-sample reconstruction without conflict. This design not only facilitates correct accumulation of samples but also preserves the efficiency necessary for real-time rendering. Although real-time applications typically operate under low spp, our framework remains compatible with high spp configurations when required (Figure 9).

5.2. Alpha Fix

In this stage, we record two quantities for each curve segment: M, the number of pixels originally hit from the strand, and N, the number of additional pixels filled in via propagation for each curve segment. Given that a typical human scalp comprises tens of millions of segments, we employ a hash map to balance concurrency efficiency and memory usage. The size of the hash table is dynamically adjusted based on the image-space bounding boxes of the hair regions:

$$Size_{hash} = \lceil W \times H \times 1.2 \rceil, \tag{14}$$

where W and H denote the width and height of the axis-aligned bounding box (AABB) containing all the strands in image space.

6. Results

We validate our method through a series of experiments across diverse hair geometries and rendering scenarios. All results are rendered using a Marschner hair model [MJC*03] with dual scattering enabled [ZYWK08]. The input is a 1 spp noisy image generated via OptiX path tracing, simulating a challenging yet realistic rendering setup. Unless otherwise stated, our reconstructions operate at the segment level, where each hair strand is represented by multiple connected segments (1 strand 60–70 segments). The reconstruction and filtering are therefore aligned with this physically grounded segmentation granularity. **Comparison with Anti-Aliasing Techniques.** As illustrated in Figure 1, our method substantially outperforms SMAA [JESG12], which fails to recover thin and broken flyaway strands. While SMAA helps alleviate aliasing artifacts around coarse edges, it lacks the semantic and geometric understanding necessary for recovering disconnected segments. Our approach reconstructs such fibers consistently and preserves

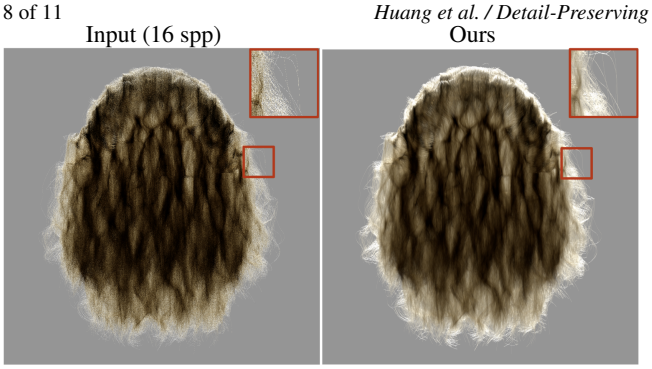

Figure 9: *Result under 16 spp input condition. Our method can work roubstly under different input spp.*

the high-frequency geometry in both straight and curly hair regions. Figure 10 presents a comparison between our method and MSAA [FT20], SSAA [FvDFH90], and TAA [YLS20], illustrating that existing popular anti-aliasing techniques fail to handle hair rendering effectively.

Comparison with General-Purpose Denoisers. We further compare our method against OptiX's built-in denoiser [NVI25] and Intel's OIDN [Int21]. As shown in Figure 1, both denoisers can reduce pixel-level noise but tend to over-smooth high-frequency structures, especially in regions with dense overlapping strands or fine flyaway fibers. OIDN, in particular, introduces visible blurring near the roots and fails to recover strand-level continuity. In contrast, our method accurately reconstructs individual segments with sub-pixel consistency, preserving fine hair structure even in complex occlusion scenarios. **Comparison with Neural Hair Filtering.** In Figure 7, we benchmark our method against a recent learning-based hair filter [CAS22]. Although the neural approach is specifically trained on hair materials, it suffers from spatial blurring and introduces characteristic grid-like artifacts due to its screen-space processing pipeline. Additionally, energy inconsistencies are observed near critical regions such as the crown and hairline. Our method, in comparison, produces sharper results with physically consistent brightness and improved fidelity for both primary and flyaway hair structures. Notably, we reconstruct sharper glints and highlight transitions without sacrificing fine detail.

Ablation Study. Figure 11 presents a step-by-step evaluation of our pipeline. Beginning from the raw 1 spp input, we show progressive reconstruction quality after candidate strand extrapolation, pixel coverage correction, and final hair-aware filtering. The qualitative improvements are consistent across various hair colors, including blonde, black and red; various hair types include curly, straight and ponytail. This validates the generality and robustness of our method across heterogeneous inputs.

Performance. Our framework operates in real-time on modern GPUs. The entire pipeline, including candidate selection, extrapolation, and filtering, takes under 8 ms per frame for scenes with 50,000 strands (3.4 million segments in curly hair scenes, 1.2 million in straight, 1.6 million in ponytail). We observe stable runtime performance regardless of hair complexity, as our segment-level

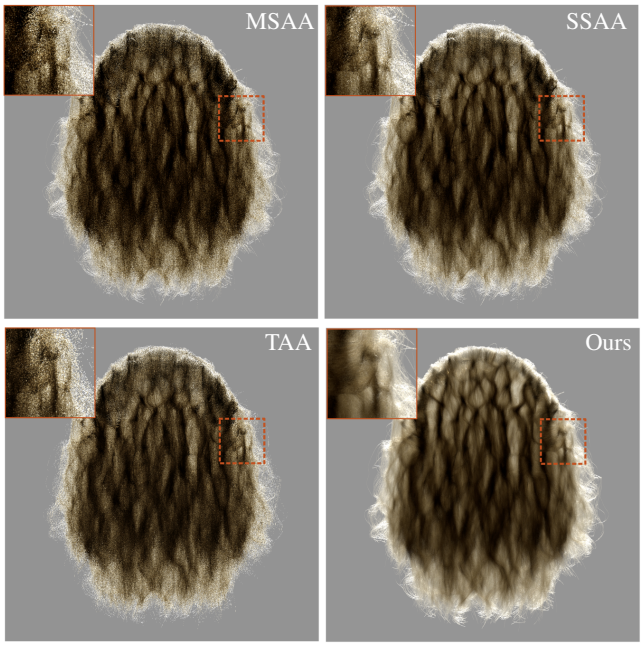

Figure 10: *Comparison of anti-aliasing techniques on complex geometry. Results from MSAA [FT20], SSAA [FvDFH90], and TAA [YLS20] are shown alongside our method. The prior methods struggle with aliasing artifacts on fine structures such as hair strands*

operations scale linearly with the number of segments processed. In the video, we demonstrate the efficiency of our method, where at this observation scale, the original 1spp shading for the curly hair scene runs at 45 fps, while after linking and filtering, it drops to 37 fps. For the straight hair scene, the original 1spp shading runs at 43 fps, while the performance drops to 34 fps after linking and filtering.

Resolution. All experiments were conducted at 1920×1080 resolution by default. The method is not resolution-limited; its performance mainly depends on screen resolution. The number of hair strands only indirectly affects the computational load. Under constant hair width and viewing distance, higher strand density tends to produce fewer missed (empty) pixels, while greater strand curvature increases the steps required during linking.

7. Conclusion and Discussion

Limitations. While our method demonstrates high fidelity in reconstructing strand-level hair geometry and robustly suppressing noise in real time, several limitations remain. First, in regions where dense hair structures occlude one another (e.g., multi-strand intersections or volumetric clustering), our current algorithm does not explicitly model complex inter-strand relationships. This simplification may lead to local inaccuracies in such high-complexity configurations. Additionally, although our screen-space extrapolation is generally effective, it can produce slight blurring near the main

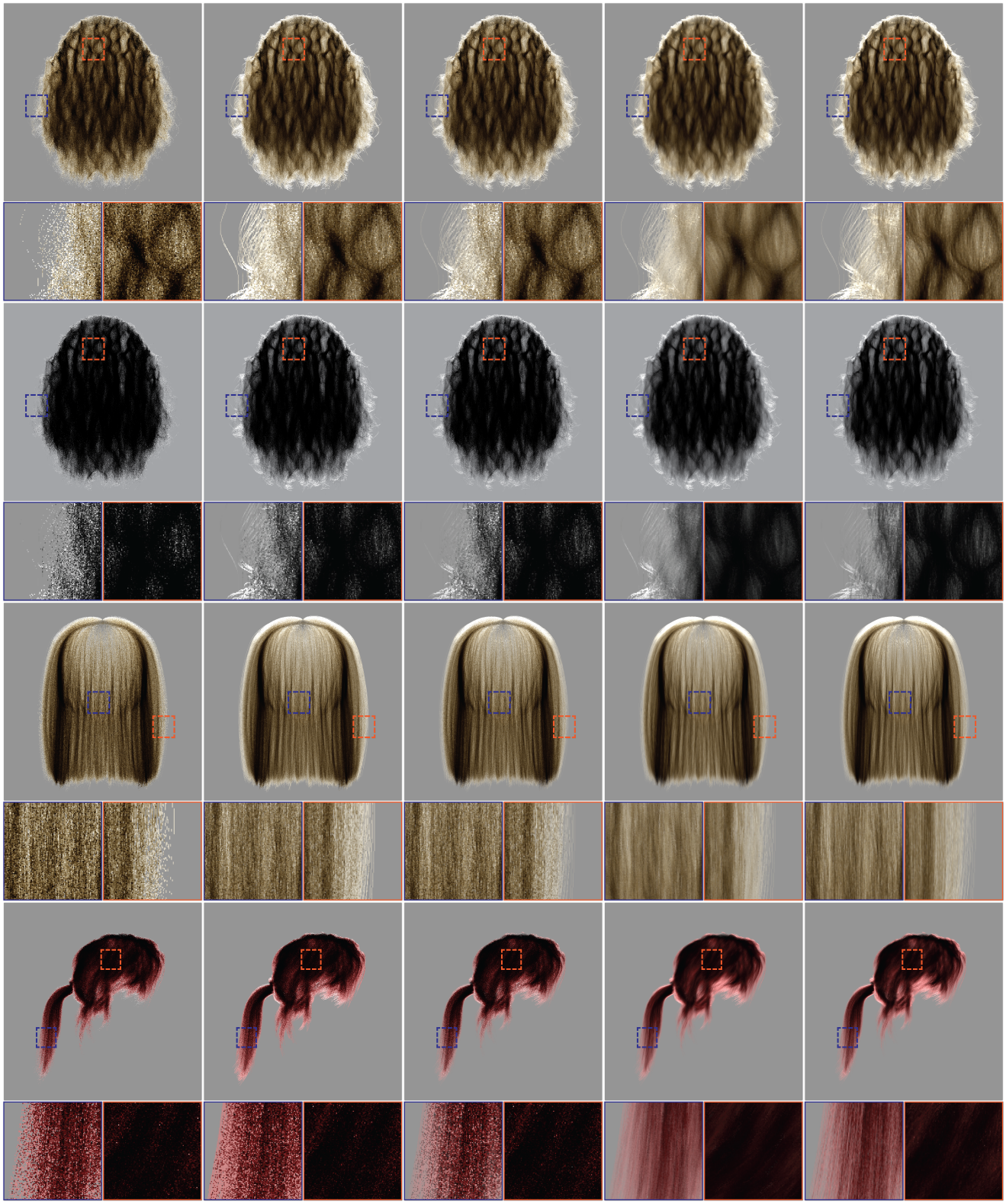

Figure 11: ***Step-by-Step Evaluation of Our Method.*** *We demonstrate the effectiveness of our pipeline across various hair types, including curly, straight and ponytail hair; blonde, black and red hair. Each column shows the progressive improvement through key stages of our method, culminating in the final output. The corresponding reference is shown for comparison.*

body of hair masses, especially under strong perspective foreshortening. Nonetheless, these artifacts are visually subtle and largely acceptable in real-time applications.

Future Work. A natural direction for future work is to integrate our strand reconstruction with learned denoisers in a joint optimization framework. This would allow for end-to-end training and could improve both segmentation robustness and reconstruction accuracy. Another promising extension is to incorporate volumetric or physically-based hair priors to better handle dense or tangled hair structures where multiple strands overlap in projection. Finally, developing multi-resolution strategies or temporal filtering schemes could further stabilize results in animated sequences or under fast camera motion.

Conclusion. We present a real-time, strand-level reconstruction framework for hair that is both visually similar to the reference and efficient. By leveraging a three-stage pipeline—candidate strand extrapolation, unbiased pixel coverage approximation, and hair-aware filtering—we achieve high-fidelity hair reconstruction with significant noise reduction. Our approach preserves fine-grained details while remaining computationally efficient, making it suitable for real-time rendering and interactive editing tasks. The modularity of our design allows it to work in tandem with existing denoisers, and our results closely match ground truth even under challenging conditions.

Acknowledgments. We thank the reviewers for their valuable comments. This work has been partially supported by the National Natural Science Foundation of China (No. 62272275, No. 61972194, No. 62032011), Natural Science Foundation of Jiangsu Province (No. BK20211147), the Taishan Scholars Program (No. tsqn202312231), Qilu University of Technology (Shandong Academy of Sciences) Faculty of Computer Science and Technology Pairing Program (No. 2024JDJH13).

References

[BEJM15] BAUSZAT P., EISEMANN M., JOHN S., MAGNOR M.: Sample-based manifold filtering for interactive global illumination and depth of field. In *Computer Graphics Forum* (2015), vol. 34, Wiley Online Library, pp. 265–276. 3

[BMDY24] BHOKARE G., MONTALVO E., DIAZ E., YUKSEL C.: Real-time hair rendering with hair meshes. In *ACM SIGGRAPH 2024 Conference Papers* (2024), pp. 1–10. 2

[BP21] BENAMIRA A., PATTANAIK S.: A combined scattering and diffraction model for elliptical hair rendering. In *Computer Graphics Forum* (2021), vol. 40, Wiley Online Library, pp. 163–175. 2

[BVM*17] BAKO S., VOGELS T., MCWILLIAMS B., MEYER M., NOVÁK J., HARVILL A., SEN P., DEROSE T., ROUSSELLE F.: Kernel-predicting convolutional networks for denoising monte carlo renderings. *ACM Trans. Graph. 36*, 4 (2017), 97–1. 3

[CAS22] CURRIUS R. R., ASSARSSON U., SINTORN E.: Real-time hair filtering with convolutional neural networks. *Proc. ACM Comput. Graph. Interact. Tech. 5*, 1 (May 2022). URL: https://doi.org/10.1145/3522606, doi:10.1145/3522606. 3, 7, 8

[CJL*16] CHIANG M., JIANG Y., LACEWELL D., BALA K., BURLEY B.: Efficient gpu path rendering. In *High-Performance Graphics* (2016), pp. 1–10. doi:10.1145/2945078.2945080. 2

[CKS*17] CHAITANYA C. R. A., KAPLANYAN A. S., SCHIED C., SALVI M., LEFOHN A., NOWROUZEZAHRAI D., AILA T.: Interactive reconstruction of monte carlo image sequences using a recurrent denoising autoencoder. *ACM Transactions on Graphics (TOG) 36*, 4 (2017), 1–12. 3

[CML11] CHAJDAS M. G., MCGUIRE M., LUEBKE D.: Subpixel reconstruction antialiasing for deferred shading. In *Symposium on interactive 3D graphics and games* (2011), pp. 15–22. 3

[Cor24] CORPORATION N.: NVIDIA Real-time Denoisers. https://github.com/NVIDIA-RTX/NRD, 2024. 2

[Cor25a] CORPORATION N.: DLSS 4: Transforming Real-Time Graphics with AI. https://research.nvidia.com/labs/adlr/DLSS4/, 2025. Technical Report. 2

[Cor25b] CORPORATION N.: *NVIDIA OptiX™ AI-Accelerated Denoiser*. Tech. rep., NVIDIA Corporation, 2025. URL: https://developer.nvidia.com/optix-denoiser. 2

[dFH*11] D'EON E., FRANCOIS G., HILL S., LETTERI J., AUBRY J.-M.: An energy-conserving hair reflectance model. In *Eurographics Symposium on Rendering* (2011), pp. 1181–1187. doi:10.1111/j.1467-8659.2011.01976.x. 2

[DSHL10] DAMMERTZ H., SEWTZ D., HANIKA J., LENSCH H. P.: Edge-avoiding a-trous wavelet transform for fast global illumination filtering. In *Proceedings of the Conference on High Performance Graphics* (2010), pp. 67–75. 3

[FT20] FRIDVALSZKY A., TÓTH B.: Multisample anti-aliasing in deferred rendering. In *Eurographics (Short Papers)* (2020), pp. 21–24. 3, 8

[FvDFH90] FOLEY J. D., VAN DAM A., FEINER S. K., HUGHES J. F.: *Computer graphics: principles and practice (2nd ed.)*. Addison-Wesley Longman Publishing Co., Inc., USA, 1990. 8

[GO12] GASTAL E. S., OLIVEIRA M. M.: Adaptive manifolds for real-time high-dimensional filtering. *ACM Transactions on Graphics (TOG) 31*, 4 (2012), 1–13. 3

[HHH22] HUANG W., HULLIN M. B., HANIKA J.: A microfacet-based hair scattering model. In *Computer Graphics Forum* (2022), vol. 41, Wiley Online Library, pp. 79–91. 2

[HZL*24] HUANG T., ZHOU Y., LIN D., ZHU J., YAN L.-Q., WU K.: Real-time level-of-detail strand-based hair rendering, 2024. URL: https://arxiv.org/abs/2405.10565, arXiv:2405.10565. 2

[Int21] INTEL CORPORATION: Intel open image denoise. https://www.openimagedenoise.org, 2021. Version 1.4.0. 1, 8

[JESG12] JIMENEZ J., ECHEVARRIA J. I., SOUSA T., GUTIERREZ D.: Smaa: Enhanced subpixel morphological antialiasing. In *Computer Graphics Forum* (2012), vol. 31, Wiley Online Library, pp. 355–364. 3, 7

[JME*18] JIMENEZ J., MASIA B., ECHEVARRIA J. I., NAVARRO F., GUTIERREZ D.: Practical morphological antialiasing. In *GPU Pro 360 Guide to Rendering*. AK Peters/CRC Press, 2018, pp. 135–153. 3

[Kar13] KARIS B.: Real shading in unreal engine 4. In *SIGGRAPH Course Notes* (2013). https:https://cdn2.unrealengine.com/Resources/files/2013SiggraphPresentationsNotes-26915738.pdf. 2

[KB83] KOREIN J., BADLER N.: Temporal anti-aliasing in computer generated animation. In *Proceedings of the 10th annual conference on Computer graphics and interactive techniques* (1983), pp. 377–388. 2

[KBS15] KALANTARI N. K., BAKO S., SEN P.: A machine learning approach for filtering monte carlo noise. *ACM Trans. Graph. 34*, 4 (2015), 122–1. 3

[KDB16] KELLER A., DAHM K., BINDER N.: Path space filtering. In *Monte Carlo and Quasi-Monte Carlo Methods: MCQMC, Leuven, Belgium, April 2014*. Springer, 2016, pp. 423–436. 3

[KJA*23] KT A., JARABO A., ALIAGA C., CHIANG M. J.-Y., MAURY O., HERY C., NARAYANAN P., NAM G.: Accelerating hair rendering

by learning high-order scattered radiance. In *Computer graphics forum* (2023), vol. 42, Wiley Online Library, p. e14895. 2

[Lot09] LOTTES T.: Fxaa-whitepaper. *Nvidia* (2009). 3

[MJC*03] MARSCHNER S. R., JENSEN H. W., CAMMARANO M., WORLEY S., HANRAHAN P.: Light scattering from human hair fibers. *ACM Transactions on Graphics 22*, 3 (2003), 780–791. doi:10.1145/882262.882345. 2, 3, 7

[MZV*20] MENG X., ZHENG Q., VARSHNEY A., SINGH G., ZWICKER M.: Real-time monte carlo denoising with the neural bilateral grid. In *EGSR (DL)* (2020), pp. 13–24. 2

[NVI25] NVIDIA: Optix™ ai-accelerated denoiser, 2025. Accessed: 2025-04-05. URL: https://developer.nvidia.com/optix-denoiser. 1, 8

[Res09] RESHETOV A.: Morphological antialiasing. In *Proceedings of the Conference on High Performance Graphics 2009* (2009), pp. 109–116. 3

[SDJ19] SAADE A., DOGGETT M., JOHANSSON TOMAS AKENINE-MÖLLER U.: Strand-based hair rendering in frostbite. In *ACM SIGGRAPH 2019 Courses* (New York, NY, USA, 2019), SIGGRAPH '19, ACM. URL: https://doi.org/10.1145/3305366.3328092, doi:10.1145/3305366.3328092. 7

[SZR*15] SEN P., ZWICKER M., ROUSSELLE F., YOON S.-E., KALANTARI N. K.: Denoising your monte carlo renders: recent advances in image-space adaptive sampling and reconstruction. *ACM Siggraph 2015 Courses* (2015), 1–255. 3

[TN23] TAKAGI Y., NISHIMOTO S.: High-resolution image reconstruction with latent diffusion models from human brain activity. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (2023), pp. 14453–14463. 2

[V*18] VOORHUIS V., ET AL.: *Beyond Spatiotemporal Variance-Guided Filtering: Temporally Stable Filtering of Path-Traced Reflections in Real-Time*. Master's thesis, 2018. 2

[WLL*21] WANG P., LIU L., LIU Y., THEOBALT C., KOMURA T., WANG W.: Neus: Learning neural implicit surfaces by volume rendering for multi-view reconstruction. *arXiv preprint arXiv:2106.10689* (2021). 2

[YJR17] YAN L.-Q., JENSEN H. W., RAMAMOORTHI R.: An efficient and practical near and far field fur reflectance model. *ACM Transactions on Graphics 36*, 4 (2017). doi:10.1145/3072959.3073685. 2

[YLS20] YANG L., LIU S., SALVI M.: A survey of temporal antialiasing techniques. In *Computer graphics forum* (2020), vol. 39, Wiley Online Library, pp. 607–621. 3, 4, 8

[YTJR15] YAN L.-Q., TSENG C.-W., JENSEN H. W., RAMAMOORTHI R.: Physically-accurate fur reflectance: Modeling, measurement and rendering. *ACM Transactions on Graphics 34*, 6 (2015). doi:10.1145/2816795.2818076. 2

[ZWW*20] ZENG Z., WANG L., WANG B.-B., KANG C.-M., XU Y.-N.: Denoising stochastic progressive photon mapping renderings using a multi-residual network. *Journal of Computer Science and Technology 35* (2020), 506–521. 3

[ZYWK08] ZINKE A., YUKSEL C., WEBER A., KEYSER J.: Dual scattering approximation for fast multiple scattering in hair. In *ACM SIGGRAPH 2008 papers*. 2008, pp. 1–10. 2, 7

[ZZW*22] ZHU J., ZHAO S., WANG L., XU Y., YAN L.-Q.: Practical level-of-detail aggregation of fur appearance. *ACM Transactions on Graphics 41*, 4 (2022), 1–17. doi:10.1145/3528223.3530064. 2