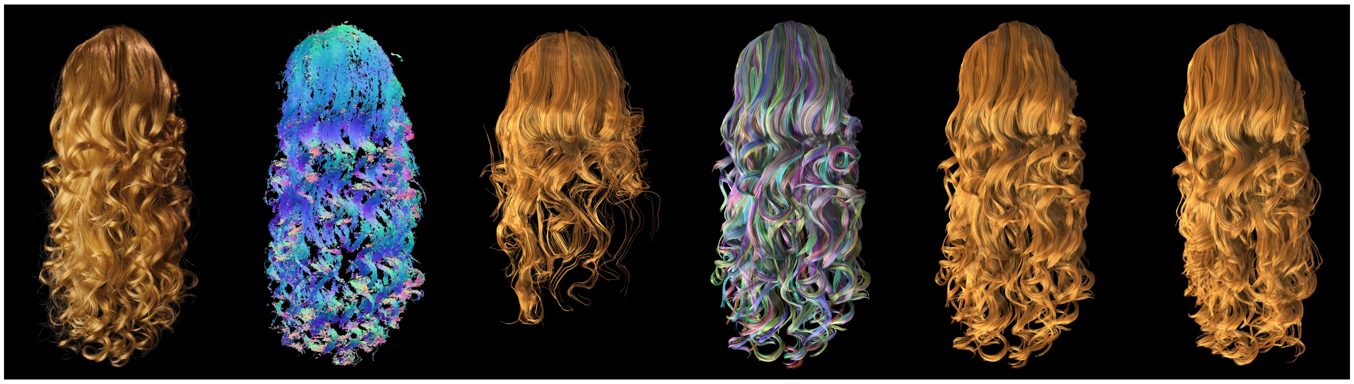
Structure-Aware Hair Capture

Linjie Luo[1] Hao Li[2] Szymon Rusinkiewicz[1]

[1]Princeton University
[2]Industrial Light & Magic / University of Southern California



(a) An image of captured hairstyle (b) Point cloud with 3D orientation field (c) Failure of previous method (d) Color-coded reconstructed wisps (e) Synthesized hair strands (f) A frame from hair simulation

***Figure 1:** Our system takes a collection of images as input (a) and reconstructs a point cloud with a 3D orientation field (b). In contrast to previous methods (e.g. [Paris et al. 2008]) that straightforwardly grow hair strands from the scalp following the orientation field and hence cannot reconstruct complex hairstyles with convoluted curl structures (c), we reconstruct complete, coherent and plausible wisps (d) aware of the underlying hair structures. The wisps can be used to synthesize hair strands (e) that are plausible for animation or simulation (f).*

Abstract

Existing hair capture systems fail to produce strands that reflect the structures of real-world hairstyles. We introduce a system that reconstructs coherent and plausible wisps aware of the underlying hair structures from a set of still images without any special lighting. Our system first discovers locally coherent wisp structures in the reconstructed point cloud and the 3D orientation field, and then uses a novel graph data structure to reason about both the connectivity and directions of the local wisp structures in a global optimization. The wisps are then completed and used to synthesize hair strands which are robust against occlusion and missing data and plausible for animation and simulation. We show reconstruction results for a variety of complex hairstyles including curly, wispy, and messy hair.

CR Categories: I.3.5 [Computer Graphics]: Computational Geometry and Object Modeling—Geometric algorithms;

Keywords: Hair reconstruction, hair modeling, multi-view stereo

Links: DL PDF WEB

ACM Reference Format
Luo, L., Li, H., Rusinkiewicz, S. 2013. Structure-Aware Hair Capture. ACM Trans. Graph. 32, 4, Article 76 (July 2013), 11 pages. DOI = 10.1145/2461912.2462026 http://doi.acm.org/10.1145/2461912.2462026.

1 Introduction

Shared between culture, nature, and sculpture, the hairstyle is a medium that creates a unique expression of self. A person's hairstyle is a vital component of his or her identity, and can provide strong cues about age, background, and even personality. The same is true of virtual characters, and the modeling and animation of hair occupy a large portion of the efforts of digital artists. Driven by increased expectations for the quality of lead and secondary characters, as well as digital doubles, this effort has only been increasing.

Acquiring complex hairstyles from the real world holds the promise of achieving higher quality with lower effort, much as 3D scanning has revolutionized the modeling of faces and other objects of high geometric complexity. This is especially true for digital doubles, which require high fidelity, and secondary animated characters, which may appear by the dozens and must receive less personalized attention from 3D modeling artists. However, even for lead characters that are modeled largely by hand, it is frequently easier to start with scanned data than to begin modeling from a blank canvas. This allows the digital hair to exploit the full talents of real-world stylists, who express their creativity through cutting, shearing, perming, combing, and waxing.

Despite the potential benefits of capturing real-world hairstyles, there is a large gap between the data produced by existing acquisition techniques and the form in which a hairstyle must end up to be incorporated into a production pipeline. Hair animation, whether done by hand or via physical simulation, typically operates on a collection of *guide strands*. Each of these is a curve through space, starting from a point on the scalp and going to the tip of the hair. The guide hairs must not intersect, and the entire collection must not be overly tangled. The hair model consists of tens of thousands of strands, whose motion is interpolated from the guide strands.

How close are existing hair capture systems to this representation? The raw output of 3D acquisition devices is typically an unstruc-

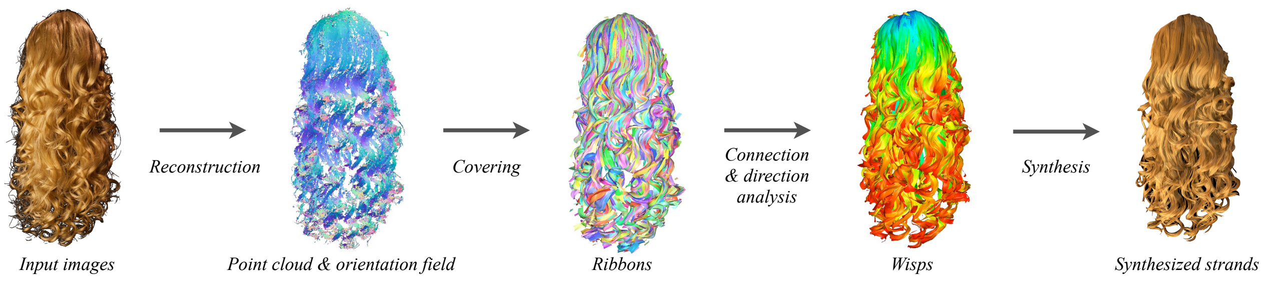


Figure 2: Overview of our method. We start with a collection of input images, reconstruct a point cloud with 3D orientation field, and cover the point cloud with ribbons that reveals the locally coherent wisp structures. A connection and direction analysis is then performed on the ribbons to determine their directions and connect them up into long wisps. Finally the wisps are used to synthesize the strands.

tured point cloud or a partial surface. Prior research on hair capture has typically augmented this geometry with a 3D orientation field, computed from orientations observed in a number of images. Given these two types of data, it is possible to grow a set of strands from the scalp whose position is constrained to the reconstructed geometry and whose direction follows the orientation field.

There are two major difficulties with this approach. First, the geometry will always suffer from missing regions and noise because of the complex occlusion patterns of hair. Second, the orientation field is necessarily extrapolated from what was observed on the outermost layer of hair. As a result, the grown hair strands exhibit a variety of undesirable artifacts: they may suddenly reverse direction and form implausible U-shapes, diverge wildly from their neighbors, exhibit variations in density, or simply fail to reach all parts of the visible hair volume (see, for example, Figure 1c). Moreover, these independently-grown strands have no natural and coherent grouping structure that allows them to be controlled by guide hairs. As a result, it is impossible to incorporate these systems, based on naive strand-growing, into animation pipelines.

We describe a system that reconstructs structured hair models plausible for hair animation and simulation (Figure 1) by performing a higher-level analysis of the hair's structure. The system incorporates four key insights. First, it grows groups or *wisps* of hair coherently. This not only matches the structure of real hairstyles, which frequently contain strands that run nearly parallel to many of their neighbors, but also allows each wisp to be associated with a guide strand for animation. Second, it builds up a *connection graph*, allowing strands to span regions of missing geometric data. As long as two curvature-compatible portions of hair have been acquired, our system is able to connect across the gap between them. Third, it explicitly resolves the 180-degree ambiguity of orientation fields by performing a global Markov Random Field (MRF) optimization on the directions associated with nodes in the graph. This keeps strands from performing sudden U-turns. Finally, it ensures that each portion of visible surface is connected back to the scalp along some path. This allows our system to work for arbitrarily complex hairstyles, even if a naive strand-growing method would have difficulty reaching the end of, e.g., a complicated curl, by strictly following the orientation field.

Our algorithm begins with a point cloud and orientation field obtained from a collection of still images (Sec. 4) and extracts a set of local strand segments. These are clustered into "ribbons" that cover sets of parallel strands (Sec. 5), and connected across gaps in the point cloud (Sec. 6.1). Following a global direction analysis (Sec. 6.2), we connect up the ribbons to each other, and to the scalp (Sec. 7.1), to obtain complete wisps with plausible topology. These not only drive the final strand synthesis (Sec. 7.3), but also can be used as guide strands for hair simulation (Sec. 8).

2 Related Work

Hair capture. Most existing hair capture methods generate a strand set model by growing hair strands independently, constrained by the captured hair orientations and geometry. Paris et al. [2004] proposed a method to estimate 3D hair orientations from highlights under a moving light source with known trajectory. They grow strands along the estimated orientations, starting at the scalp. Wei et al. [2005] introduced a technique to create a strand model from the visual hull constructed from many views. The strands are grown constrained by the orientation consistency across the views. Paris et al. [2008] introduced an active acquisition system capable of accurately capturing the positions of the exterior hair strands. A strand model is generated by growing the strands within the diffused orientation field from the scalp to the captured exterior hair layer. Jakob et al. [2009] proposed a system to capture fiber-by-fiber hair assemblies by growing hairs through the intersecting ribbons created by back-projecting the 2D strands with shallow depth-of-fields. Chai et al. [2012] showed how to create an approximate strand model from a single image using the inter-strand occlusion relationships and the head model fit to match the face in the image. Beeler et al. [2012] introduces a system to capture facial hairs using multi-view stereo matching. Their method employs a refinement method to improve the connections between the captured strand segments and remove outlier hairs. Luo et al. [2012] proposed a method to grow a strand model in the orientation field constrained by the geometry constructed from hair orientation fields. Herrera et al. [2012] applied thermal imaging to generate a strand model by growing strands on the boundary of the captured hairstyle. Their method joins the loose ends of nearby segments with smooth curvature and connects the strands to a user-defined ellipsoid as the scalp.

Geometry-based hair modeling. Structured models are commonly used in geometry-based hair modeling. A more complete survey on hair modeling techniques can be found in [Ward et al. 2006], here we only enumerate a few recent work relevant to ours. Ward et al. [2003] used a high level skeleton representation to control various low level representations for level-of-detail hair modeling. Kim and Neumann [2002] employed a hierarchy of generalized cylinders to model and edit hair in multi-resolution. Yuksel et al. [2009] introduced "Hair Meshes" to model hair using polygonal meshes with various topological operations. Wang et al. [2009] introduced a method to synthesize new hair styles from guide strands based on texture synthesis techniques.

Model-based reconstruction. Model fitting methods have been applied to the reconstruction of a variety of objects, e.g., architecture, furniture and trees. Livny et al. [2010] uses a tree skeletal model to reconstruct tree structures from point cloud input. Nan et al. [2010] introduced "SmartBoxes" to interactively reconstruct urban architectures with regular box-like structures. Li et al. [2010]

uses "Arterial Snakes" to reconstruct delicate interleaving man-made structures.

Comparison to our method. We build upon many of the ideas pioneered by the papers above, including matching hair strands to a reconstructed orientation field, exploiting a constant-curvature prior to hypothesize connections between partial strands, and connecting reconstructed strands to the scalp. However, we achieve a more plausible reconstruction that better matches the structures of real hair by focusing on *wisp* reconstruction, performing a global direction analysis, and using a connection graph data structure to find long connected paths between visible hair strands and points on the scalp. As a result, we are able to reconstruct complex hairstyles including curly and messy hair, and produce hair models that are plausible for animation and simulation.

3 Overview

Our method is shown in Figure 2. Its input is a set of images captured from multiple views for a real hairstyle (or, for some of our examples, a wig). We key the images to separate foreground and background, and use the Patch-based Multi-View Stereo (PMVS) [Furukawa and Ponce 2010] algorithm to reconstruct a raw point cloud with normals. We perform Moving Least Squares (MLS) fitting to filter the points and normals and compute the 3D orientation field on the points according to the 2D orientation maps (Sec. 4).

We then identify locally coherent wisp structures and group them into *ribbons* that cover the input point cloud. To achieve this goal, we first grow *strand segments* on the point cloud, following the 3D orientation field and stopping at discontinuities in orientation. (Sec. 5.1). Then we cover the grown strand segments with ribbons to expand the local regions into coherent wisp structures (Sec. 5.2).

Because of occlusion and missing data, the ribbons are disconnected from each other and do not form complete wisps. We discover missing connections between adjacent ribbons by trying to fit circular arcs to the covered strand segments of the ribbons (Sec. 6.1). Good fits indicate plausible connections between ribbons, which we encode in a *connection graph*. We also associate a growth direction with each ribbon by globally optimizing for compatible connections and local hints of ribbon direction using a Markov Random Field (MRF) formulation (Sec. 6.2). Following the connections and optimized directions, the ribbons can be connected up to form complete *wisps* (Sec. 6.3).

We attach the wisps to the scalp of a manually fit head model (Sec. 7.1), then generate interior wisps to fill in the empty regions inside the hair volume, which had been occluded in the input (Sec. 7.2). Finally we synthesize strands using the complete attached wisps and the interior wisps (Sec. 7.3).

4 Reconstruction

We begin by reconstructing an initial point cloud and 3D orientation field from a set of input images, acquired under unconstrained lighting. We semi-automatically segment out the hair from the background, then use PMVS [Furukawa and Ponce 2010], a state-of-the-art multi-view stereo algorithm, to reconstruct a point cloud. We use around $30 \sim 50$ input images for the reconstruction.

Filtering. The initial point cloud and the estimated normals from PMVS can be noisy, so we smooth them with Moving Least Squares (MLS) [Levin 1998]: for each point, we fit an optimal plane to the locally-weighted neighbors near that point. The normal of the plane and the point's projection on the plane are then used to update the point's original normal and position. We use a sigma of 2mm to achieve plausible filtering results.

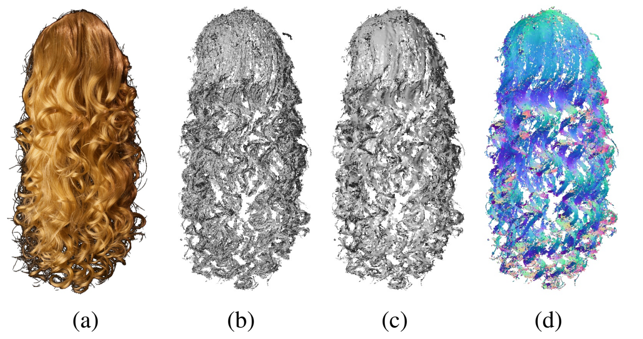


Figure 3: *Point cloud and orientation field generation. From a set of input images (a) we compute a point cloud (b) using PMVS [Furukawa and Ponce 2010]. We then apply moving least squares fitting to smooth out the reconstruction noise (c) followed by computing a 3D orientation field (d) based on the point cloud and the input images.*

2D orientation maps. We compute an orientation map for each input image using the method of Luo et al. [2012], which uses a bank of rotated filters to detect the dominant orientation at each pixel. The orientation map is then enhanced with 3 passes of iterative refinement for better signal-to-noise ratio, as proposed by Chai et al. [2012]. To further reduce noise in regions with low confidence, we apply the bilateral filtering method of Paris et al. [2004] to diffuse orientations from high-confidence regions.

3D orientation field. We use the 2D orientation maps to compute a 3D orientation field on the technique of Wei et al. [2005]. However, it is challenging to determine the visibility of each point in a point cloud to all the input views. We find the following scheme effective in our situation without having to apply the more sophisticated methods such as [Mehra et al. 2010].

For each point p in the point cloud with normal n, we find a reference view Y among all the views which has the minimum angle between n and the line-of-sight vector $v = \langle c - p \rangle$, where c is the view's projection center and $\langle \cdot \rangle$ is the normalization operator (i.e., $\langle A \rangle = A/\|A\|$ for any vector $A \neq 0$). We then compute a reference 3D orientation $o_Y = \langle v_Y \times d_Y \times n \rangle$, where d_Y is the orientation at p's projection on Y's orientation map and v_Y is Y's line-of-sight vector. We can determine a view V as one of the visible views $\mathcal{V}$ to p if V's derived 3D orientation $o_V = \langle v_V \times d_V \times n \rangle$ at p is compatible with o_Y, i.e., $|o_V \cdot o_Y| > T_o$, where v_V is V's line-of-sight vector and d_V is the orientation at p's projection on V's orientation map. $T_o = 0.5$ is a threshold to reject outlier views invisible to p. The final 3D orientation o at p can be computed by maximizing

$$\rho = \max_o \frac{\sum_{v \in \mathcal{V}} w_V (o \cdot o_V)^2}{\sum_{v \in \mathcal{V}} w_V}, \text{ subject to } \|o\| = 1, \quad (1)$$

where $w_V = \max(n \cdot v_V, 0)$. This can be solved efficiently by singular value decomposition. ρ is defined as the confidence of the 3D orientation o. Note that our formulation ensures that the 3D orientation o for each point p is normal to the point's normal n.

With the 3D orientation defined at each point in the point cloud, the 3D orientation $o(p)$ at any point p can be computed as:

$$o(p) = \arg\max_o \sum_{q \in \mathcal{N}(p)} \exp\left(\frac{\|p-q\|^2}{2\sigma_d}\right) \rho_q (o \cdot o_q)^2,$$

$$\text{subject to } \|o\| = 1, \quad (2)$$

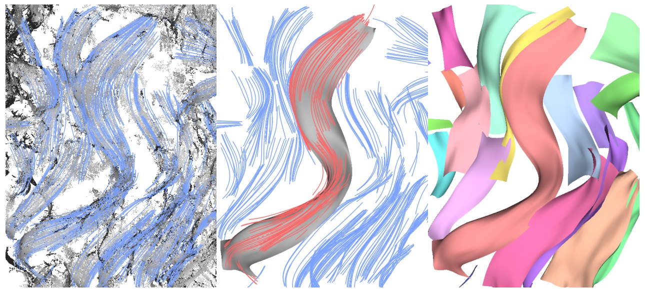


Figure 4: The steps of covering. Strand segments are first grown to cover the input point cloud (a). A ribbon is then expanded to cover adjacent strand segments (b). The covering continues until all the strand segments are covered by ribbons and thus reveals the locally coherent wisp structures (c).

where $\mathcal{N}(p)$ is the set of neighboring points around p and σ_d the parameter to control interpolation smoothness (2mm in our experiments). o_q and ρ_q are the orientation and confidence of a neighboring point q respectively. See Figure 3.

5 Covering

We begin the process of analyzing the hair's structure by proceeding bottom up: we first grow local *strands*, and then group coherent groups of strands into *ribbons*. In this way, we cover most of the point cloud with ribbons, omitting only those portions where we did not find coherent structures (Figure 4).

5.1 Covering by Strand Segments

We grow a number of strand segments $\mathcal{S}$ from a set of seed points. Typically we use all the points in the point cloud as seed points. A strand segment $S \in \mathcal{S}$ is a chain of vertices $(p_1, \ldots, p_N)$ connected by line segments in 3D space.

Growing. Starting from a seed point q we grow S in both directions $t(q) = \pm o(q)$, constrained by the 3D orientation field as well as the point cloud. For each growing direction, we repeatedly perform forward Euler steps to extend S from the current growing vertex p_i: $p'_i = p_i + t(p_i)\delta$, where δ is a small increment step (2mm) and $t(p_i)$ the current growing direction. To avoid drifting from the point cloud during the integration, we apply moving least squares to project p'_i onto some point p''_i within the point set surface defined by the point cloud. The growing is terminated, in either growing direction, if any of the following applies:

1. Incompatible growing direction: $|t(p''_i) \cdot t(p_i)| < T_t$, where $t(p''_i) = \mathrm{sgn}\big(t(p_i) \cdot o(p''_i)\big)\, o(p''_i)$ is the next growing direction from p''_i consistent with $t(p_i)$.
2. Being in holes or out of boundary: $|\mathcal{N}(p''_i)| < T_{\mathcal{N}}$.
3. Unreliable MLS projection: $|\langle p''_i - p'_i \rangle \cdot t(p_i)| > T_{\mathrm{MLS}}$. This happens where the estimated normals are unreliable.

We set $T_t = 0.9$, $T_{\mathcal{N}} = 5$ and $T_{\mathrm{MLS}} = 0.5$ in all our examples. The next growing vertex is obtained by $p_{i+1} = p''_i$ if none of these termination conditions apply.

Note that in the computation of the next orientation $o(p''_i)$ (Equation 2), we on-the-fly select $\mathcal{N}(p''_i)$ with orientations compatible with the current growing direction $t(p_i)$, i.e., $q \in \mathcal{N}(p''_i)$ only if $|o_q \cdot t(p_i)| > T_o$. This scheme avoids the orientational ambiguities at the crossings of multiple hair wisps with different orientations during strand growing. In contrast, many previous methods [Paris et al. 2008; Chai et al. 2012] precompute a diffused 3D orientation field, and the orientations at wisp crossings are problematic.

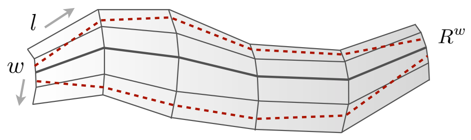


Figure 5: A ribbon $\mathcal{R}$ is a grid of 3D vertices. The two parametric directions $w \in [0,W]$ and $l \in [0,L]$ are shown with arrows. Isocurve R^w goes along the l direction (bold line). The ranges $g^-(l), g^+(l)$ of the ribbon is shown in red dashed lines (Sec. 7.3).

Smoothing. The initial grown strand may be noisy because of the MLS projection step and the noise in the input point cloud. We therefore smooth the strand by minimizing the following energy:

$$\mathcal{E} = \sum_i \alpha_0 \|p_i - p_i^{(0)}\|^2 + \alpha_1 \|p_{i+1} - p_i - t(p_i^{(0)})\delta\|^2 + \alpha_2 \|p_{i-1} - 2p_i + p_{i+1}\|^2, \qquad (3)$$

where p_i is a vertex on a strand, $p_i^{(0)}$ is the initial position of p_i before optimization, p_{i-1} and p_{i+1} are the predecessor and successor of p_i on the strand and α_0, α_1 and α_2 are weights that control positional, orientational, and curvature energy terms. We set $\alpha_0 = 0.1$, $\alpha_1 = 1$ and $\alpha_2 = 5$. (assuming the point cloud is in millimeters).

Covering. After a strand S is grown and smoothed, we remove the seed points that S covers from the original seed points to avoid repeated growing. To find if a seed point q is covered by a strand, we traverse every line segment (p_i, p_{i+1}) in the strand and compute the distance between q and (p_i, p_{i+1}) to find the minimum distance. If the minimum distance is smaller than a designated threshold Δ (set to 1.5mm), q is covered by the strand.

We repeat the growing, smoothing and covering steps above until all the seed points are covered.

Plausibility check. The strand growing may cause implausible strand segments, such as U-shaped strands with low turning points. To avoid this, we compute a height value $H(p) = p \cdot d_{down}$ for each point p in S, where d_{down} is a reference down direction, and check each height difference ΔH of every pair of consecutive local extrema. If the height difference $\Delta H > T_H$ (T_H is set to 50mm), then we split S at the extrema points. Note that the connections between these split strand segments will be re-discovered in the connection analysis (Sec. 6.1) for the subsequent connection graph (Sec. 6.2).

5.2 Covering by Ribbons

Intuitively, our goal is to group nearly-parallel strand segments into ribbons, which will later be connected into complete wisps. Thus, the coherence present in our final output is a function of the coherence we are able to find in the strand-to-ribbon grouping stage. We proceed greedily, by always working with the currently longest strand segment that has not been covered by a ribbon.

Ribbon. Formally, a ribbon $\mathcal{R}$ is a 2D grid of 3D vertices $\{R_i^j\}$ where $i = 0, 1, \ldots, L$ and $j = 0, 1, \ldots, W$. L is the length of the ribbon and W the width. The isocurves R^j along the length define the orientation of the ribbon. Isocurve $R^{W/2}$ is defined as the center isocurve of the ribbon (Figure 5).

After tessellating the grid of the ribbon, we can use $\mathcal{R}$ to define a local parameterization $R(w,l) : [0,W] \times [0,L] \mapsto \mathbb{R}^3$, where $w \in [0,W]$ and $l \in [0,L]$. Also, the inverse projection operator $R^{-1}(p) : \mathbb{R}^3 \mapsto [0,W] \times [0,L]$ can be defined for any point p by

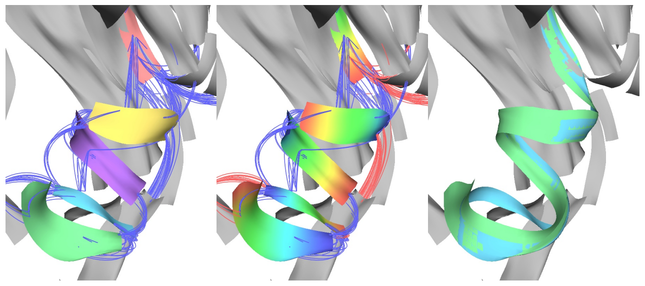

(a) (b) (c)

Figure 6: Connection and direction analysis on curly ribbons. We find possible missing connections between the ribbons by fitting circular arcs to the covered strands as shown in blue arcs (a). The direction analysis is performed to determine the directions of the ribbons in (b) (from blue to red). Notice that the resulting incompatible links are colored in red. Finally ribbons are connected up to form wisps (c). Note that the upper overlapped wisps are removed.

finding the closest point q on the tessellated ribbon and mapping back to the parametric domain.

We can define the orientation o_R of each vertex on the ribbon as $o_R(R_i^j) = \langle R_{i+1}^j - R_{i-1}^j \rangle$ and the normal as $n_R(R_i^j) = \langle (R_i^{j+1} - R_i^{j-1}) \times (R_{i+1}^j - R_{i-1}^j) \rangle$. These definitions can be extended to all the points on the ribbon by linear interpolation.

Expansion. Starting from a strand segment S, we add S to $\mathcal{R}$ as the first isocurve R^0, then we expand the width of $\mathcal{R}$ on both sides and fit to the input point cloud. The initial expansion offsets are computed as $b(R_i^0) = \pm o_R(R_i^0) \times n(R_i^0)$, where $n(R_i^0)$ is the normal at point R_i^0. $n(R_i^0)$ can be computed as the weighted average of the normals of the neighboring points $\mathcal{N}(R_i^0)$. We then initialize a new isocurve R' from R^0 by: $R'_i = R_i^0 + b(S_i)\Delta$. We apply MLS projection to every point of R' on the input point cloud and smooth R' as in Sec. 5.1.

Covering. Now we try to cover more adjacent strand segments with the expanded ribbon (Figure 4). For each adjacent strand segment S, we project each point p_i of S onto $\mathcal{R}$ as $q = R^{-1}(p_i)$ and classify p_i as a bad point if any of following applies:

1. Incompatible orientation: $|o_S(p_i) \cdot o_R(q)| < T_t$, where $o_S(p_i) = \langle p_{i+1} - p_{i-1} \rangle$ is the strand orientation at p_i.
2. Too far away: $\|p_i - q\| > \Delta$.

S is covered by $\mathcal{R}$ if and only if the number of bad points is less than T_{bad} (set to 10).

We continue to expand $\mathcal{R}$ if there are new strand segments covered by $\mathcal{R}$. Otherwise, we mark this expansion as failed. If we have 2 consecutive failed expansions, we terminate the expansion on the current side and try the other side if it has not been expanded.

Note that when the expansion on one side is terminated, we may have excessive expanded isocurves for covering the strand segments. We then repeatedly remove the outermost isocurves on both sides from $\mathcal{R}$ given that the set of covered strand segments are unaffected by the removal. Also, after the first expansion, we can replace $n(R_i^j)$ with $n_R(R_i^j)$ to evaluate the expansion offset $b(R_i^j)$.

6 Connection and Direction Analysis

So far, our bottom-up analysis of the hairstyle has proceeded by local agglomeration. However, occlusions and missing data force us to look for more distant connections between ribbons (Sec. 6.1).

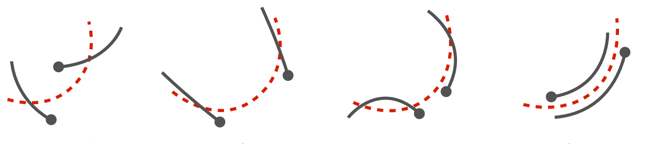

(a) (b) (c) (d)

Figure 7: Rejection criteria in connection analysis. Adjacent strand segments (solid curves) with end vertices (dots) are fit with circular arcs (dashed curves). (a) Large fitting error. (b) Incompatible curvatures. (c) Large torsion. (d) Large overlap.

These connections, however, may be less reliable, and in fact may be inconsistent with each other.

This inconsistency becomes apparent when we attempt to assign a *direction* to each ribbon: which way strands should traverse the ribbon as they go from scalp to tip. This direction will be necessary for final strand synthesis (Sec. 7.3), yet is difficult or impossible to infer locally from the input, especially for challenging hairstyles in which hair strands can go in all directions such as the messy hairstyle illustrated in Figure 13 and the curly hairstyle with helical wisp structures illustrated in Figure 6.

Therefore, we encode the hypothesized ribbon connections in a graph, and perform a global direction analysis (Sec. 6.2) to discover the most consistent direction assignments. We drop any connections that are inconsistent with the assigned directions, and connect the ribbons into our final wisps (Sec. 6.3) as illustrated in Figure 6.

6.1 Connection Analysis

We analyze the possible connections between two ribbons by fitting circular arcs between the strand segments covered by these ribbons. We use circular arcs as the fitting model because hair strands exhibit low variation in curvature, as suggested by the hair simulation model [Bergou et al. 2008]. Also, efficient methods [Chernov 2011] exist to fit circles robustly in the presence of noise and missing data. Note that one could use helix fitting [Savadjiev et al. 2006] to explicitly account for hair torsion in the fitting model, however we found that helix fitting is much more expensive to compute and tends to overfit for noisy data.

Thus, we test each pair of adjacent strand segments S and S' covered by $\mathcal{R}$ and $\mathcal{R}'$, respectively, and having two end vertices p and p' within a distance of 30mm. We extract K (set to 10) closest vertices on S and S' to p and p' denoted as $\mathcal{P} = \{p_1, \ldots, p_K\}$ and $\mathcal{P}' = \{p'_1, \ldots, p'_K\}$ ($p = p_1$ and $p' = p'_1$), from the closest to the farthest. We also denote $\mathcal{Q} = \{q_i\} = \{p_K, \ldots, p_1, p'_1, \ldots, p'_K\}$ as the concatenation of the input points. We adopt a method in [Chernov 2011] to fit a 3D circle to $\mathcal{Q}$. First, we fit a plane to these points so that we can project the points on the plane and apply robust 2D circle fitting methods to initialize the fit. We find that Taubin's algebraic fit [Taubin 1991] works well even in the cases with very small curvature. We then refine the circle center $\hat{c}$, normal $\hat{n}$ and radius $\hat{r}$ jointly using Levenberg-Marquardt.

We use a set of criteria to reject bad fits (Figure 7):

(a) Large fitting error. We reject the fit if the Root Mean Square (RMS) fitting error is larger than 2mm.

(b) Incompatible curvatures. We also compute the curvatures κ and κ' of $\mathcal{P}$ and $\mathcal{P}'$ by fitting circles and compare with the curvature $\hat{\kappa}$ of the fit circle. We reject the fitting if $|\kappa - \kappa'|/|\kappa + \kappa'| > T_{curv}$ or $|2\hat{\kappa} - \kappa - \kappa'|/|\kappa + \kappa'| > T_{curv}$, where T_{curv} is set to 0.4.

(c) Large torsion. We reject the fitting if the angle between any two of the normals n_1, n_2 and $\hat{n}$ is larger than 90 degrees, where n_1 and n_2 are the normals of the fit circles to $\mathcal{P}$ and $\mathcal{P}'$.

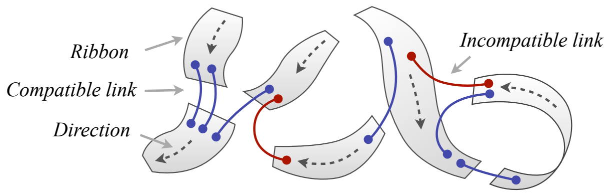


Figure 8: *A connection graph consists of ribbons connected by links. The ribbons are assigned with directions (dashed line). The links can be compatible (blue) or incompatible (red) with the directions of the incident ribbons.*

(d) Large overlap of the input points. We compute the overlap of the input points as $\eta = 1 - |V/V_0|$, where the signed sweeping volume $V = \sum_i (q_{i+1} - \hat{c}) \times (q_i - \hat{c}) \cdot \hat{n}$ and the unsigned sweeping volume $V_0 = \sum_i |(q_{i+1} - \hat{c}) \times (q_i - \hat{c}) \cdot \hat{n}|$. We reject the fitting if $\eta > 0.1$.

If we accept the fitting, we project p and p' onto $\mathcal{R}$ and $\mathcal{R}'$ and define the pair of projected points $(R^{-1}(p), R'^{-1}(p'))$ as the *link points* of a *link* ℓ. We refer to the link points $R^{-1}(p)$ and $R'^{-1}(p')$ as $\ell(\mathcal{R})$ and $\ell(\mathcal{R}')$. See example fitting results in Figure 6.

6.2 Direction Analysis

Connection graph. The major data structure used in direction analysis is the connection graph (Figure 8). The connection graph consists of ribbons as vertices and connections between ribbons as edges. Each connection contains one or more links derived from the connection analysis (Sec. 6.1).

The direction $D(\mathcal{R})$ of a ribbon $\mathcal{R}$ is either consistent with the parametric direction along the length $0 \to L$ or the opposite $L \to 0$. We denote that a strand segment $S = \{p_1, \ldots, p_n\}$ is compatible with the ribbon's direction $D(\mathcal{R})$ as $S \sim D(\mathcal{R})$, if the following expression is *false* for the majority of the vertices in S: $o_S(p) \cdot o_R(R^{-1}(p)) > 0 \oplus D(\mathcal{R}) = 0 \to L$, where p is a vertex of S and $\oplus$ the *exclusive or* operator.

We define that a link ℓ between $\mathcal{R}$ and $\mathcal{R}'$ is compatible with $D(\mathcal{R})$ and $D(\mathcal{R}')$ if the following expression is *true* for strand segments $\mathcal{P} = \{p_1, ..., p_K\}$ in $\mathcal{R}$ and $\mathcal{P}' = \{p'_1, ..., p'_K\}$ in $\mathcal{R}'$ (Sec. 6.1) used to fit ℓ: $\mathcal{P} \sim D(\mathcal{R}) \oplus \mathcal{P}' \sim D(\mathcal{R}')$. Intuitively, a link is compatible with the directions of two ribbons if a strand can grow from one ribbon to the other through the link without having an incompatible direction at the other, as illustrated in Figure 8.

We can then define the set of compatible links between $\mathcal{R}$ and $\mathcal{R}'$ as $\mathcal{C}(\mathcal{R}, \mathcal{R}')$ and the set of incompatible links as $\overline{\mathcal{C}}(\mathcal{R}, \mathcal{R}')$.

The strength $\Psi(\mathcal{L})$ of a set of links $\mathcal{L}$ between two ribbons $\mathcal{R}$ and $\mathcal{R}'$ is defined as the smaller one of the numbers of different strand segments in $\mathcal{R}$ and $\mathcal{R}'$ used to fit the links in $\mathcal{L}$. Note that this definition down-weights the outlier case where only one or a few strand segments in a ribbon are connected by many strand segments in another ribbon, as often occurs in practice.

For ribbons that overlap each other, we define the overlap $\eta(\mathcal{R}, \mathcal{R}')$ of $\mathcal{R}$ and $\mathcal{R}'$ as the ratio of the number of overlapped vertices over the total number of vertices in $\mathcal{R}$ and $\mathcal{R}'$. A ribbon vertex p is overlapped with a vertex p' in another ribbon if $\|p - p'\| < \Delta$. The directions of $\mathcal{R}$ and $\mathcal{R}'$ at p and p' are compatible if the following expression is *false*: $o_R(p) \cdot o_{R'}(p') > 0 \oplus D(\mathcal{R}) = D(\mathcal{R}')$. We denote that $D(\mathcal{R})$ is compatible with $D(\mathcal{R}')$ as $D(\mathcal{R}) \sim D(\mathcal{R}')$ if the directions are compatible at the majority of the overlapped vertices or as incompatible $D(\mathcal{R}) \nsim D(\mathcal{R}')$ otherwise.

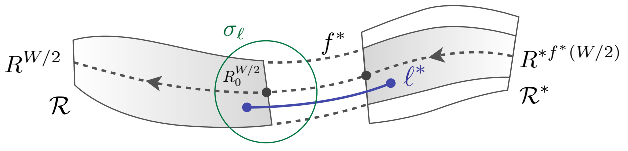


Figure 9: *A ribbon $\mathcal{R}$ is connected up to a predecessor ribbon $\mathcal{R}^*$. The feasible link ℓ^* that minimizes the mapping distortion in Equation (10) within the distance σ_ℓ of vertex $R_0^{W/2}$ defines a stitching mapping f^* to map each isocurve of $\mathcal{R}$ to that of $\mathcal{R}^*$. In particular, $\mathcal{R}$'s center isocurve $R^{W/2}$ is mapped to $R^{*f^*(W/2)}$.*

MRF formulation. We use a Markov Random Field (MRF) to optimize the set of directions $\mathcal{D}$ for all the ribbons by minimizing the following energy:

$$E = \sum_{\mathcal{R}} E_{\text{ribb}}(\mathcal{R}) + \sum_{\mathcal{R},\mathcal{R}'} \alpha_{\text{link}} E_{\text{link}}(\mathcal{R}, \mathcal{R}') + \alpha_{\text{close}} E_{\text{close}}(\mathcal{R}, \mathcal{R}'). \quad (4)$$

The ribbon energy E_{ribb} accounts for the fact that the ribbon's direction is more likely to be falling down due to gravity or going farther away from scalp since it originates from the scalp.

$$E_{\text{ribb}}(\mathcal{R}) = \frac{1}{2h}\Big(H(R_{l_1}^{W/2}) - H(R_{l_2}^{W/2}) + \zeta(R_{l_1}^{W/2}) - \zeta(R_{l_2}^{W/2})\Big), \quad (5)$$

where H is the height function defined in Sec. 5.1, $\zeta(p)$ is the distance from a point p to the closest point on the scalp (please refer to Sec. 7.1), $l_1 = 0$ and $l_2 = L$ if $D(\mathcal{R}) = 0 \to L$ and $l_1 = L$ and $l_2 = 0$ if $D(\mathcal{R}) = L \to 0$. h is a height threshold (set to 100mm) to adjust the sensitivity of ribbon's direction to its height difference or the scalp distance difference.

The link energy E_{link} minimizes incompatible links:

$$E_{\text{link}}(\mathcal{R}, \mathcal{R}') = \frac{\Psi\big(\overline{\mathcal{C}}(\mathcal{R}, \mathcal{R}')\big)}{\max_{\mathcal{R},\mathcal{R}'} \Psi\big(\overline{\mathcal{C}}(\mathcal{R}, \mathcal{R}')\big)}. \quad (6)$$

Finally, the closeness energy E_{close} penalizes incompatible directions between overlapped ribbons:

$$E_{\text{close}}(\mathcal{R}, \mathcal{R}') = \begin{cases} 0, & D(\mathcal{R}) \sim D(\mathcal{R}') \\ \eta(\mathcal{R}, \mathcal{R}'), & D(\mathcal{R}) \nsim D(\mathcal{R}') \end{cases}. \quad (7)$$

The total energy E can be effectively minimized using the method in [Boykov et al. 2001].

6.3 Connecting Ribbons into Wisps

After we determine the ribbon directions, we connect the ribbons into wisps following the compatible links. For the sake of brevity, we now assume that each ribbon $\mathcal{R}$ is consistent to its optimized direction, i.e., $D(\mathcal{R}) = 0 \to L$.

For each ribbon $\mathcal{R}$, we connect "up" from the beginning ($l = 0$) of the ribbon to the end ($l = L$) of a *predecessor* ribbon $\mathcal{R}^*$. To simplify the problem, we use the center isocurve $R^{W/2}$ as the delegate to the ribbon when we are looking for the predecessor ribbon to connect up (Figure 9).

We first define a *feasible link* that can be used to connect the beginning of $R^{W/2}$ to another ribbon. We require the feasible link to be close enough to $R_0^{W/2}$. To be specific, a link ℓ is *feasible* for $\mathcal{R}$ if $\|\ell(\mathcal{R}) - R_0^{W/2}\| < \sigma_\ell$, where σ_ℓ is a distance threshold (10mm) to define closeness between the link point and the center isocurve. We can then define the set of feasible links between $\mathcal{R}$ and $\mathcal{R}'$ as

$\mathcal{F}(\mathcal{R}, \mathcal{R}')$. The predecessor ribbon $\mathcal{R}^*$ is the ribbon that maximizes the strength of compatible and feasible links between $\mathcal{R}$ and $\mathcal{R}^*$:

$$\mathcal{R}^* = \arg\max_{\mathcal{R}'} \Psi\big(\mathcal{F}(\mathcal{R}, \mathcal{R}') \cap \mathcal{C}(\mathcal{R}, \mathcal{R}')\big). \tag{8}$$

Given a feasible link $\ell \in \mathcal{F}(\mathcal{R}, \mathcal{R}^*) \cap \mathcal{C}(\mathcal{R}, \mathcal{R}^*)$, we can define a stitching mapping $f^\ell : [0, W] \mapsto [0, W^*]$, where $[0, W]$ is the width domain of $\mathcal{R}$ and $[0, W^*]$ of $\mathcal{R}^*$ as follows:

$$f^\ell(w) = \min(\max(w - \ell(\mathcal{R}).w + \ell(\mathcal{R}^*).w, 0), W^*), \tag{9}$$

where $\ell(\mathcal{R}).w$ is the w parameter of $\ell(\mathcal{R})$ and $\ell(\mathcal{R}^*).w$ of $\ell(\mathcal{R}^*)$.

Finally, we choose the link $\ell^* \in \mathcal{F}(\mathcal{R}, \mathcal{R}^*) \cap \mathcal{C}(\mathcal{R}, \mathcal{R}^*)$ for the stitching mapping f^* that minimizes the mapping distortion:

$$\ell^* = \arg\min_{\ell} \left(1 - \frac{f^\ell(W) - f^\ell(0)}{W}\right). \tag{10}$$

Using f^* we can extend every isocurve $R^w \in \mathcal{R}$ to $R^{*f^*(w)} \in \mathcal{R}^*$. To generate a smooth transition between the two isocurves, we first connect them with a straight line using a discretization step of δ, then we perform the smoothing step described in Sec. 5.1 according to the orientation field. Note that during the smoothing, we fix the two end points of the line and use a larger σ_d to compute the orientation since the transition region lacks data points.

We replace the center isocurve $R^{W/2}$ with $R^{*f^*(W/2)}$ and repeat the connection process above until we cannot find any predecessor ribbons to connect.

However, in order to maximize the chance of finding a predecessor ribbon and connecting up, when no predecessor ribbons can be found directly from $\mathcal{R}$, we also look for feasible links in the adjacent overlapped ribbons $\mathcal{R}'$ with $\eta(\mathcal{R}, \mathcal{R}') > 0$. We compute the R^* and f^* with respect to $\mathcal{R}'$ but change the definition of a feasible link to any link ℓ that $\|\ell(\mathcal{R}') - R_0^{W/2}\| < \sigma_\ell$ as well as replacing the role of $\mathcal{R}'$ with $\mathcal{R}$ in Equations 9 and 10.

To avoid cycles when connecting up, we store all the ribbons connected and avoid connecting to them again. Also, we try to avoid local cycles by rejecting connections to a overlapped ribbon $\mathcal{R}'$ with $\eta(\mathcal{R}, \mathcal{R}') > 0.1$

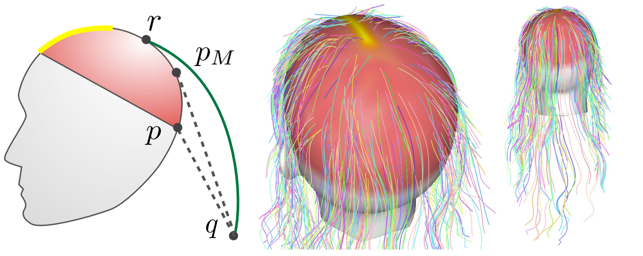
After all the ribbons are connected up into wisps, we remove all wisps that are covered by others. To see if one ribbon is covered, we check if every vertex of the wisp is overlapped by the vertices of other wisps using the method in Sec. 6.2. We also remove outlier ribbons that fail to connect to at least one other ribbon, which often arise from outlier points of the initial point cloud.

7 Synthesis

Once we have found the wisps, we attach them to the scalp. Since the interior of the hair is occluded, and hence we have no data there, we need to generate interior wisps to fill in the empty region. Finally, we use the attached exterior wisps and interior wisps to generate plausible strands according to the input point cloud and orientation field.

7.1 Attaching Wisps to the Scalp

Head model and scalp. We manually fit a head model to each dataset and paint on the model to indicate the possible scalp region. We also paint a parting line to specify the vertices on the scalp that parts the hair into different directions according to the captured hairstyle (Figure 10). Painting on the model can be done in minutes using any 3D mesh texture painting software and the scalp region is reused for all the examples.



(a) (b) (c)

Figure 10: *(a) Attaching a point q to the scalp (the red area). p is the closest point to q on the scalp and p_M is a point on p's closest path to the parting line (the yellow line) that maximizes the directional compatibility. r is chosen as the root point to attach and generate the interior strand (green curve). A real attaching example with internal strands is shown in (b) with a zoom-out view in (c).*

Hair growing directions. Human hair has natural growing directions perpendicular to the radial line from the hair whorl on the scalp. However, in practice it is very difficult to determine the growing directions for non-trivial hairstyles. Existing methods either simplify the growing directions to normal directions on the scalp [Yagyu et al. 2006] or specify the growing directions manually [Sobottka et al. 2006].

We notice the fact that most common hairstyles have a distinctive parting line to part the hair into different growing directions. We compute the reference growing directions with the drawn parting line on the scalp. We first compute shortest paths from the vertices on the parting line to every vertex p on the scalp. For each p we denote the shortest path from the parting line to p as $\Gamma(p) = \{p_1, p_2, ..., p_N\}$, where p_1 is the vertex on the parting line and $p_N = p$. Then the reference growing direction $d_s(p)$ at p is computed as $d_s = p_N - p_{N-1}$. We can compute the reference growing directions at every point on the scalp by interpolation.

Root point assignment. To attach a wisp $\mathcal{R}$ to the scalp, we first need to find a root point r and a growing direction d_g on the scalp for $\mathcal{R}$. Here we again use the center isocurve $R^{W/2}$ as the delegate for $\mathcal{R}$ to simplify the problem. For notational simplicity, we denote the vertex $R_1^{W/2}$ at the beginning of $\mathcal{R}$ as the attaching point q. We find the closest scalp vertex p for q and search along p's shortest path from the parting line $\Gamma(p) = \{p_1, p_2, ..., p_N\}$ and find p_M with the maximum directional compatibility:

$$p_M = \arg\max_{p_i \in \Gamma(p)} \langle q - p_i \rangle \cdot d_s(p_i). \tag{11}$$

To avoid cluttering of the root points we randomly select a point p_k from $\{p_1, p_2, ...p_M\}$ and sample the root point r around a neighborhood of p_k on the scalp. The final growing direction d_g is computed as:

$$d_g = \begin{cases} d_s(r), & n(r) \cdot d_s(r) > n(r) \cdot \langle q - r \rangle \\ \langle q - r \rangle, & n(r) \cdot d_s(r) < n(r) \cdot \langle q - r \rangle, \end{cases} \tag{12}$$

where $n(r)$ is the normal at root point r on the scalp. The second case adjusts d_g when the attaching points are above the root points on the scalp.

Interior strand generation. We use a technique similar to that in Sec. 6.3 to generate the interior strand between the attaching point $R_0^{W/2}$ at the wisp and the root point r. That is, we first initialize the strand as a straight line and then smooth it. However, it is desirable to ensure that interior strands are hidden in the interior region, so

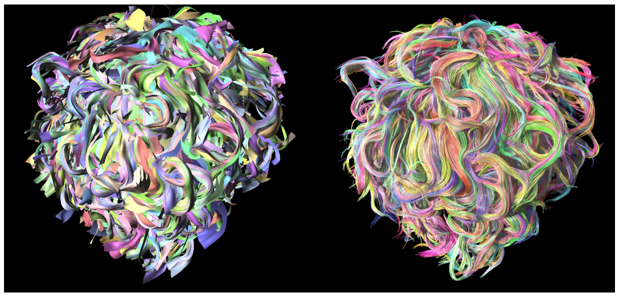


Figure 11: The synthesized strands (right) with the wisps (left).

that they have minimum impact on the exterior appearance. To this end, we use the *interior field* to regularize the interior strands in the interior region.

The interior field I is a level set field that smoothly transitions from 0 to 1 from the scalp points to the input data points. Using a regular grid, we first set the voxels containing input points to 1 and the voxels on the scalp to 0. Then we apply diffusion constrained by the values of these set voxels to smoothly generate the interior values between the input points and the scalp. We add the following interior energy term E_{int} to Equation 3 for the interior field:

$$E_{\text{int}} = \sum_i \alpha_{\text{int}} B \left(\frac{i}{N} - I(p_i) \right)^2, \tag{13}$$

where N is the number of vertices of the interior strand and B the diagonal length of the bounding box of the input point cloud. The weight for interior field α_{int} is set to 0.005. We assume the direction of the interior strand is from the scalp to the input data points.

To keep the growing direction on the root point of the interior strand during smoothing with Equation 3, we fix two points nearest to the root point, to align the strand with the growing direction.

After we compute and attach the interior strand for the center isocurve $R^{W/2}$, we offset and attach the interior strand to all the other isocurves in $\mathcal{R}$ to form a complete wisp from the scalp.

7.2 Interior Wisp Generation

Interior wisps are generated in a similar way as the interior strands in Sec. 7.1. Specifically, we sample the points from the input point cloud that have greater distances to the scalp than a specified threshold. For each, we generate an interior strand as in Sec. 7.1. To make the strand completely hidden in the hair volume, we truncate the farther end of the curve.

We then expand the strand into a ribbon with a designated width. Note that the initial normals of the ribbon can be derived from the gradient of the interior field ∇I.

7.3 Strand Synthesis

We introduce the *ranges* of a wisp $\mathcal{R}$ to fit the input data more accurately when we synthesize the strands. To be specific, the ranges of $\mathcal{R}$ are two functions $g^-, g^+ : [0,L] \mapsto [0,W]$. g^-, g^+ are the lower and upper bound of the width parameter defined on $[0,L]$.

To compute the ranges of each wisp $\mathcal{R}$ for the input data points, we project each data point p to $\mathcal{R}$ and check if p is covered by $\mathcal{R}$. Uncovered points are not used to compute the ranges of $\mathcal{R}$. We then update the ranges with the covered point p as follows:

$$\begin{aligned} g^-(l_p) &= \min(g^-(l_p), R^{-1}(p).w) \\ g^+(l_p) &= \max(g^+(l_p), R^{-1}(p).w) \end{aligned} \tag{14}$$

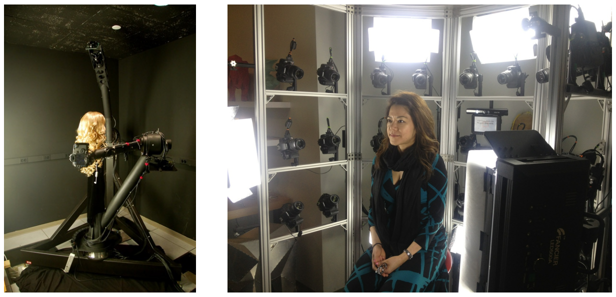


Figure 12: We use a robotic gantry to position an SLR camera at 50 views to capture the images for the wigs (left). For real hairstyles, we use a camera array of 30 SLR cameras (right).

where $l_p = \lfloor R^{-1}(p).l + 0.5 \rfloor$ is rounded to the nearest integer length parameter. We then smooth $g^-(l), g^+(l)$ by fitting smooth 1D curves to them on $[0,L]$. To improve realism, we also taper the ranges towards the strand tips using a quadratically decreasing function as the tapering factor.

For the interior wisps where no data points can be used to compute the ranges, we simply set the ranges to the maximum: $[0,W]$.

We can interpolate the ranges by defining the range interpolation function $g(l,t) = g^-(l) \cdot (1-t) + g^+(l) \cdot t$ and a strand S can be synthesized from $\mathcal{R}$ by $R_l^{g(l,t)}$ for all $l \in [0,L]$.

Now S is synthesized on the surface of the wisp (Figure 11). To add thickness to the wisps, we offset S in $\mathcal{R}$'s inverse normal direction by a random amount between 0 and T_{thick}. We find that T_{thick} = 3mm works well for all the examples in the paper. Finally, we perform smoothing on S using Equation 3.

To control the density of the synthesized strands, we record the number of strands covering the vertices within the ranges of the wisps during strand synthesis. If the number of covering strands for a vertex is smaller than a preset threshold $T_{density}$, the vertex can initiate the synthesis of one new strand covering it from its wisp, otherwise the vertex is skipped. We iterate on each vertex until no vertex can initiate the synthesis of new strands.

8 Results

We present results of our pipeline on five datasets. Three of these are of wigs, and were captured using a single digital SLR camera mounted on a motorized spherical gantry and moved to 50 positions (Figure 12 left). These results are shown in Figure 1 and the first two rows of Figure 13. Two other results (Figure 13, last two rows) are of real hair, and were captured using a rig containing 30 cameras (Figure 12 right).

As shown in Figure 1, our system is capable of reconstructing challenging hair styles. Our connection graph is able to establish correct correspondences among the partially-visible portions of curls, and hence our reconstructed wisps are, in most cases, able to follow the curls all the way from the scalp to the tips. Figure 14 compares the reconstruction details to the input hairstyle and shows that our reconstructed hair model can faithfully reveal the intricate hair structures in the input hairstyle. Figure 13, first row, contains many crossing wisps that present a challenge to competing algorithms, while the second row illustrates a complicated and disordered hairstyle. Though we are not able to reconstruct each curl perfectly, many specific curls are captured, and the general impression of the hairstyle makes our reconstructed data suitable for virtual characters.

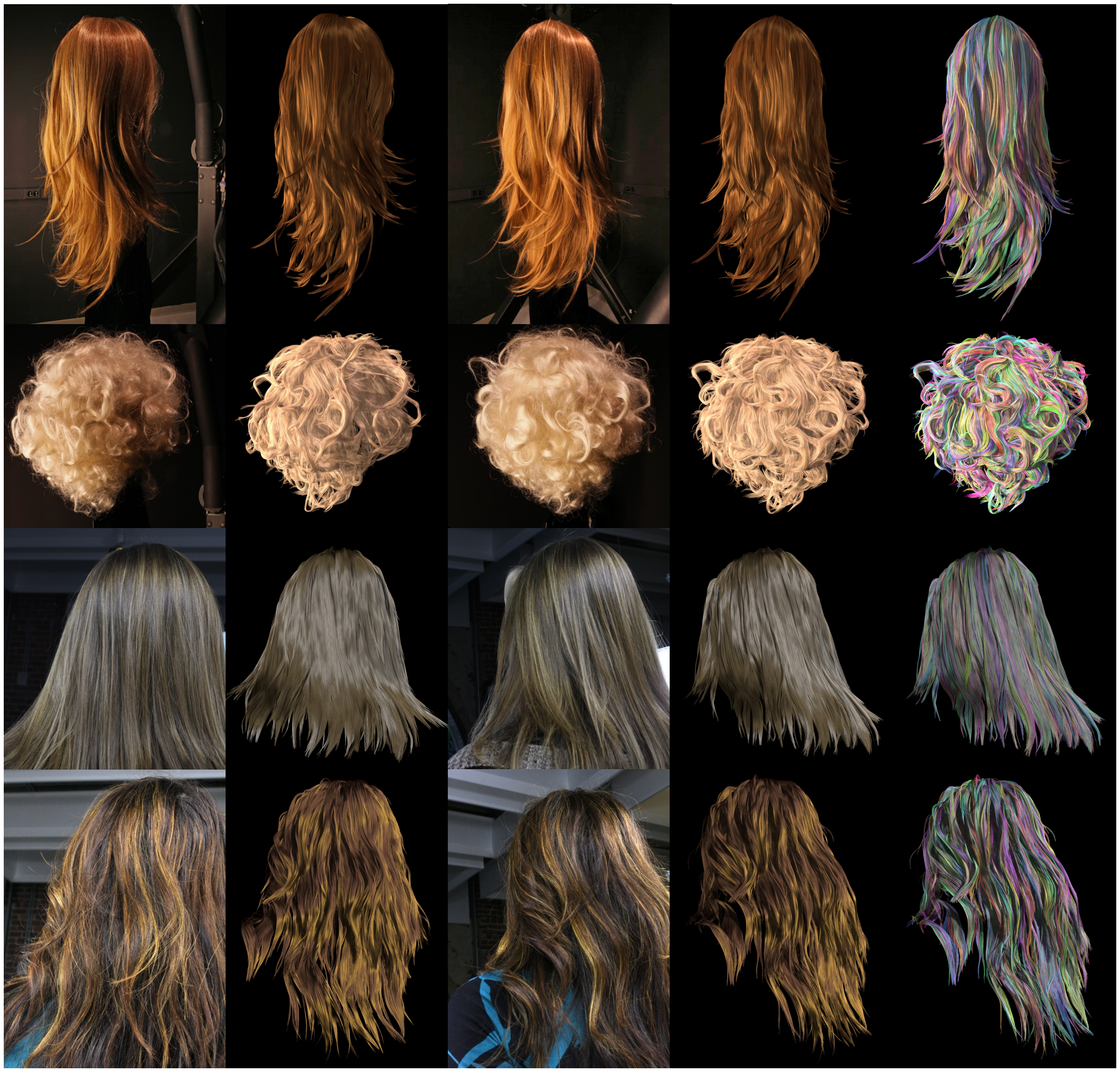


Figure 13: *Examples of our pipeline applied to four datasets. For each, we show two views of the reference input images and the synthesized hairs as well as a color-coded visualization of the reconstructed wisps, where synthesized strands in the same wisp are in the same color.*

The third and fourth rows of Figure 13 illustrate hair capture for a "digital double" scenario. Though the required acquisition apparatus is nontrivial, it consists solely of digital (still) cameras. Acquisition is thus completely passive and one-shot, meaning that it would be practical to incorporate a hair capture session into a movie special-effects workflow (and budget). These hair styles are simpler than the wigs (which were chosen to illustrate challenging cases), and our reconstruction is relatively accurate. Though we have chosen rendering parameters by hand (for the Marschner et al. [2003] hair scattering model) to roughly match the input images, a closer match might be obtained using a method such as that of Bonneel et al. [2009]. Of course, situations in which hair is colored or highlighted, such as the fourth row of Figure 13, would require even more sophisticated estimation of hair appearance; we believe that this is an interesting and necessary direction of future work to allow hair capture for digital doubles to become practical.

To evaluate the plausibility of our reconstruction results, we created animations of growing the synthesized strands from the scalp to the tip to visualize the internal coherent topology of the hair model. Please see the accompanying video for the results.

Robustness. We evaluate the robustness of our method using inputs of varying quality. We apply our pipeline on an input point cloud of 230K points, about 10 times less than the full resolution 2M points which we used to reconstruct the hair model shown in Figure 1. We compare our reconstructed hair models for two differ-

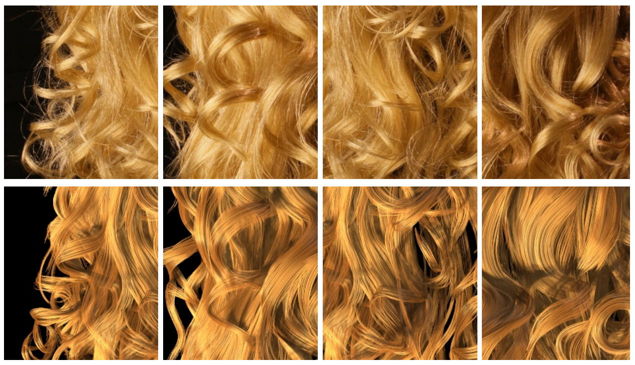


Figure 14: Close-up comparison of the hair details between the reference hairstyle and our reconstructed hairstyle.

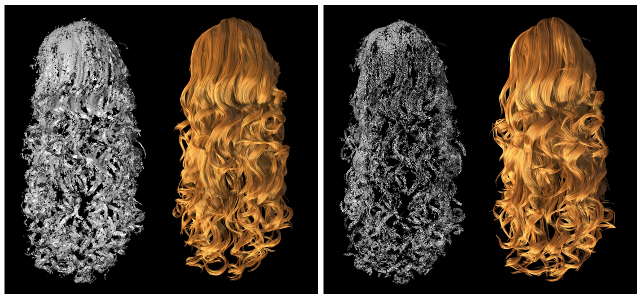


Figure 15: We evaluate the robustness of our method by reconstructing hairstyles (b) and (d) from the input point clouds of 2M points (a) and 230K points (c), respectively. Notice the little visual difference of our results from the inputs of very different quality.

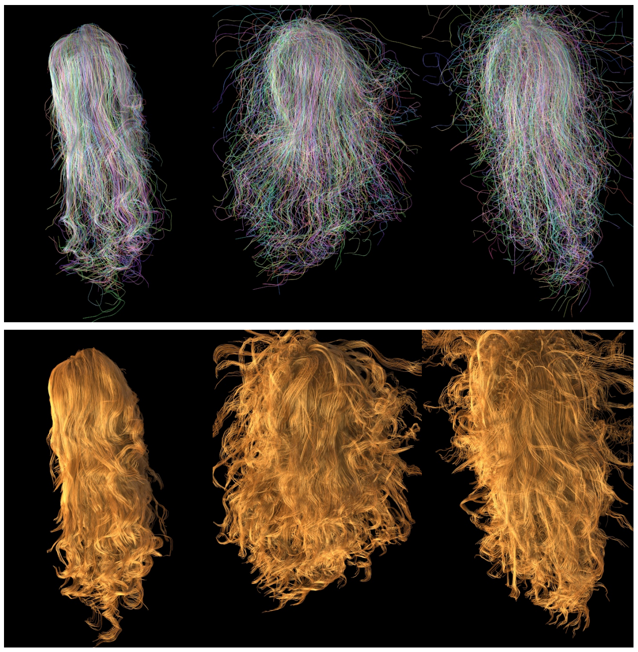


Figure 16: Three frames from a physical simulation in which hair is tousled through rapid head motion. As with most hair simulations, this one uses sparse guide strands (About 1K in this simulation, shown at top), which naturally arise from the reconstructed wisps. The motion of the guide strands is interpolated onto the full set of strands (about 30K shown at bottom).

ent inputs in Figure 15. Note that the result from low quality input faithfully recovers all key hair wisp structures and has little difference on the visual quality compared to the full resolution result.

Simulation. Our wisp-based hair models are plausible for hair simulations. We demonstrate a simulation setup using the center isocurve of each wisp as the guide strands in a hair simulator based on articulated rigid curves. The hair motion is driven by a pre-defined scalp movement and the full motion of the synthesized strands is then interpolated from the guide strands. We use about 1K guide strands to drive around 30K synthesized strands. Three frames of the simulation result are shown in Figure 16, and more results can be found in the accompanying video. More advanced simulation-specific processing and computation are important to improve the realism of the simulation result, including pretensioning for inverse statics and collision correction for complex hair structures.

Parameter choice. Although our method involves a large number of parameters, we find most of the parameters insensitive to the input datasets and we use the provided fixed values throughout our experiments. One parameter we do find useful to improve the results for specific hairstyles is the curvature weight α_2 for strand smoothing in Equation (3). For straight hairstyles, we set $\alpha_2 = 30$ to provide stronger regularization against the stereo noise in the reconstructed point cloud to compensate for the weak regularization nature of PMVS.

Timings. All the examples are computed on a quad-core Intel i7 machine with 16GB memory. For datasets with 50 views, the reconstruction of initial point cloud takes about one hour using PMVS. The computation for 3D orientation field takes 3 minutes. All the subsequent processing steps are implemented single threaded. For an input point cloud with 2M points, the covering step takes 3 minutes. The connection analysis takes 5 minutes and direction analysis 1 minute. The final strand synthesis takes 2 minutes.

9 Limitations, Future Work and Conclusion

Although our method can be successfully applied to a variety of challenging hairstyles, the ribbon-based representation and certain smoothness constraints prevent our method from capturing very fine-scale stray hairs or extremely disordered hairstyles (This becomes clear by looking at the result of messy hairstyle in Figure 13). The effective amount of regularization, however, could be reduced by starting with a more accurate initial point cloud obtained using one of the recent methods such as [Jakob et al. 2009] or [Herrera et al. 2012]. These more accurate hair reconstruction methods can also help the cases with smooth hairstyles, for which we find larger perceivable reconstruction errors in the point cloud due to increased hair specularity and the weak regularization nature of PMVS. Other improvements to our method might include making our connection and direction analysis more physically based, by including gravity, contact forces, and hair growth models (with estimated stiffness and "curliness" parameters) as priors. Further image-based analysis can also be done to estimate the thickness of the wisps automatically.

Another especially difficult class of hairstyles is those in which the hair does not hang freely but is constrained by braids, dreadlocks, clips, ties, or support against the body. We believe that most existing methods, including the one presented here, would have difficulty in generating topologically-correct hair strands in these cases. Handling these styles may require coupling the hair acquisition process with physical simulation, and possibly matching to a database of exemplars.

Looking beyond geometry, a full system for hair capture should also include measurement of appearance and motion. As mentioned above, researchers have already investigated the problem of estimating the parameters of hair appearance models, but handling variation from wisp to wisp is likely to require a combination of inverse rendering and data-driven techniques.

Because our system is one-shot, it could be generalized to video input just by running independently on each frame. This is likely to result in flicker, so some method of ensuring temporal coherence would be needed. This may require coupling a hair simulator with the reconstruction system, simultaneously using the data to constrain the simulation and using the simulation to provide temporal coherence and fill in parts of the data that could not be observed.

Overall, the results in Figure 13 suggest that our system can generate hair models of the sort needed for current production pipelines. For digital doubles and secondary animated characters, only modest manual editing might be necessary to achieve the required quality. For primary characters, of course, there are considerably greater requirements on quality and controllability, but the captured results may still serve as a reference for hand-modeling by skilled artists.

Acknowledgments

We thank Steve Marschner from Cornell University, Stephane Negri, Oytun Akman and Tatjana Dzambazova from Autodesk and Lee Perry Smith for their help with the data acquisition. We also thank Karin Cooper and Stephen Bowline from ILM and Chris Twigg from 3Gear for helping with the physical hair simulation. This work is partially supported by NSF grant CCF-1012147.

References

BEELER, T., BICKEL, B., NORIS, G., MARSCHNER, S., BEARDSLEY, P., SUMNER, R. W., AND GROSS, M. 2012. Coupled 3d reconstruction of sparse facial hair and skin. *ACM Trans. Graph. 31*, 4, 117:1–117:10.

BERGOU, M., WARDETZKY, M., ROBINSON, S., AUDOLY, B., AND GRINSPUN, E. 2008. Discrete elastic rods. *ACM Trans. Graph. 27*, 3, 63:1–63:12.

BONNEEL, N., PARIS, S., PANNE, M. V. D., DURAND, F., AND DRETTAKIS, G. 2009. Single photo estimation of hair appearance. *Computer Graphics Forum (Proc. EGSR) 28*, 4.

BOYKOV, Y., VEKSLER, O., AND ZABIH, R. 2001. Fast approximate energy minimization via graph cuts. *IEEE Trans. PAMI 23*, 11, 1222–1239.

CHAI, M., WANG, L., WENG, Y., YU, Y., GUO, B., AND ZHOU, K. 2012. Single-view hair modeling for portrait manipulation. *ACM Trans. Graph. 31*, 4, 116:1–116:8.

CHERNOV, N. 2011. *Circular and linear regression : fitting circles and lines by least squares*. Monographs on statistics and applied probability. CRC Press/Taylor & Francis, Boca Raton.

FURUKAWA, Y., AND PONCE, J. 2010. Accurate, dense, and robust multiview stereopsis. *IEEE Trans. PAMI 32*, 1362–1376.

HERRERA, T. L., ZINKE, A., AND WEBER, A. 2012. Lighting hair from the inside: A thermal approach to hair reconstruction. *ACM Trans. Graph. 31*, 6, 146:1–146:9.

JAKOB, W., MOON, J. T., AND MARSCHNER, S. 2009. Capturing hair assemblies fiber by fiber. *ACM Trans. Graph. 28*, 5, 164:1–164:9.

KIM, T.-Y., AND NEUMANN, U. 2002. Interactive multiresolution hair modeling and editing. *ACM Trans. Graph. 21*, 3, 620–629.

LEVIN, D. 1998. The approximation power of moving least-squares. *Mathematics of Computation 67*, 224, 1517–1531.

LI, G., LIU, L., ZHENG, H., AND MITRA, N. J. 2010. Analysis, reconstruction and manipulation using arterial snakes. *ACM Trans. Graph. 29*, 5, 152:1–152:10.

LIVNY, Y., YAN, F., OLSON, M., CHEN, B., ZHANG, H., AND EL-SANA, J. 2010. Automatic reconstruction of tree skeletal structures from point clouds. *ACM Trans. Graph. 29*, 6, 151:1–151:8.

LUO, L., LI, H., PARIS, S., WEISE, T., PAULY, M., AND RUSINKIEWICZ, S. 2012. Multi-view hair capture using orientation fields. In *Proc. CVPR*.

MARSCHNER, S., JENSEN, H. W., ANDS. WORLEY, M. C., AND HANRAHAN, P. 2003. Light scattering from human hair fibers. *ACM Trans. Graph. 22*, 3, 780–791.

MEHRA, R., TRIPATHI, P., SHEFFER, A., AND MITRA, N. J. 2010. Visibility of noisy point cloud data. *Computers and Graphics 34*, 3, 219–230.

NAN, L., SHARF, A., ZHANG, H., COHEN-OR, D., AND CHEN, B. 2010. SmartBoxes for interactive urban reconstruction. *ACM Trans. Graph. 29*, 4, 93:1–93:10.

PARIS, S., BRICEÑO, H., AND SILLION, F. 2004. Capture of hair geometry from multiple images. *ACM Trans. Graph. 23*, 3, 712–719.

PARIS, S., CHANG, W., KOZHUSHNYAN, O. I., JAROSZ, W., MATUSIK, W., ZWICKER, M., AND DURAND, F. 2008. Hair Photobooth: Geometric and photometric acquisition of real hairstyles. *ACM Trans. Graph. 27*, 3, 30:1–30:9.

SAVADJIEV, P., CAMPBELL, J. S., PIKE, G. B., AND SIDDIQI, K. 2006. 3d curve inference for diffusion mri regularization and fibre tractography. *Medical Image Analysis 10*, 5, 799–813.

SOBOTTKA, G., KUSAK, M., AND WEBER, A. 2006. In *Proc. CGIV*, 365–371.

TAUBIN, G. 1991. Estimation of planar curves, surfaces, and nonplanar space curves defined by implicit equations with applications to edge and range image segmentation. *IEEE Trans. PAMI 13*, 11, 1115–1138.

WANG, L., YU, Y., ZHOU, K., AND GUO, B. 2009. Example-based hair geometry synthesis. *ACM Trans. Graph. 28*, 3, 56:1–56:9.

WARD, K., LIN, M. C., LEE, J., FISHER, S., AND MACRI, D. 2003. Modeling hair using level-of-detail representations. In *Proc. CASA*, p. 41.

WARD, K., BERTAILS, F., YONG KIM, T., MARSCHNER, S. R., PAULE CANI, M., AND LIN, M. C. 2006. A survey on hair modeling: Styling, simulation, and rendering. *TVCG 13*, 2, 213–234.

WEI, Y., OFEK, E., QUAN, L., AND SHUM, H.-Y. 2005. Modeling hair from multiple views. *ACM Trans. Graph. 24*, 3, 816–820.

YAGYU, K., HAYASHI, K., AND CHANG, S. 2006. Orientation of multi-hair follicles in nonbald men: perpendicular versus parallel. *Dermatologic Surgery 32*, 5, 651–660.

YUKSEL, C., SCHAEFER, S., AND KEYSER, J. 2009. Hair meshes. *ACM Trans. Graph. 28*, 5, 166:1–166:7.