Adaptive Skinning for Interactive Hair-Solid Simulation

Menglei Chai, Changxi Zheng and Kun Zhou, *Fellow, IEEE*

Abstract—Reduced hair models have proven successful for interactively simulating a full head of hair strands, building upon a fundamental assumption that only a small set of guide hairs are needed for explicit simulation, and the rest of the hair move coherently and thus can be interpolated using guide hairs. Unfortunately, hair-solid interactions is a pathological case for traditional reduced hair models, as the motion coherence between hair strands can be arbitrarily broken by interacting with solids.

In this paper, we propose an adaptive hair skinning method for interactive hair simulation with hair-solid collisions. We precompute many eligible sets of guide hairs and the corresponding interpolation relationships that are represented using a compact strand-based hair skinning model. At runtime, we simulate only guide hairs; for interpolating every other hair, we adaptively choose its guide hairs, taking into account motion coherence and potential hair-solid collisions. Further, we introduce a two-way collision correction algorithm to allow sparsely sampled guide hairs to resolve collisions with solids that can have small geometric features. Our method enables interactive simulation of more than 150K hair strands interacting with complex solid objects, using 400 guide hairs. We demonstrate the efficiency and robustness of the method with various hairstyles and user-controlled arbitrary hair-solid interactions.

Index Terms—hair simulation, interactive method, reduced model, adaptivity, collision correction.

1 Introduction

Humans interact with their hair using solid objects every day, brushing or combing their hair daily and even playing with their hair while reading. In a virtual world, hair simulation is of vital importance in depicting computer animated characters. Despite its wide adoption in offline production, interactive hair simulation, especially with intricate hair-solid interactions, remains a great challenge, mainly because of the dense hair volume typically composed in excess of 100K individual strands. Indeed, virtually no hair-solid simulation of a full head of hair has achieved interactive performance yet. It is this interactive hair-solid simulation problem that we focus on in this paper.

A popular approach of interactive hair simulation is to use a reduced model: it simulates only a small set of representative hairs (or guide hairs) and interpolates the rest of the hairs (or normal hairs) in a way similar to the Linear Blend Skinning [1]. This method enjoys fast performance and plausible results, building on a fundamental assumption that there exists strong coherence between normal hairs and their guide hairs throughout.

Unfortunately, when used for simulating hair-solid interactions, current reduced hair models have significant deficiencies: because of the interactively user-controlled solids, any pair of hairs can move completely differently at any time. Thus, the coherence between guide and normal hairs can be disrupted in an unpredictable manner. Consequently, interpolating normal hairs using a fixed set of guide hairs leads to serious artifacts, such as ghost drifting and interpenetration (Figure 10 and the supplemental video). To make matter worse, when a solid has small geometric features such as the fine teeth of a comb, the sparsely selected guide hairs may not even touch those small features, and thus provide no clue to normal hairs for avoiding interpenetration (Figure 4).

Fig. 1: **Interactive hair salon.** We propose the first interactive hair-solid simulation method. Using 400 guide hairs, our method is able to simulate a full head of hairs with more than 150K hair strands interacting with complex solids. With our technique, a user-controlled virtual hair salon is made possible. See the supplemental video for animation results.

We notice that a hair strand always shares similar motions with some other hair strands at any moment, because of the dense distribution of hair strands. Interactions with solids break the hair's motion coherence only in local regions in a time-varying way, and thus dynamically change the groups of hairs that move coherently. This observation suggests that the set of guide hairs for every normal hair always exists, but dynamically varies under the hair-solid interactions.

In light of this, we propose an *adaptive hair skinning method* for interactive hair simulation with hair-solid collisions. In our method, every normal hair has multiple eligible sets of guide hairs, in contrast to a fixed single set of guide hairs in existing methods. This fundamental difference from existing reduced models allows our runtime simulation to exploit two key ideas: (i) when interpolating a normal hair, we take into account the current and potential hair-solid collisions, and adaptively choose

- *Menglei Chai and Kun Zhou are with the State Key Laboratory of CAD&CG, Zhejiang University, Hangzhou, China, 310058. Email: cmlatsim@gmail.com, kunzhou@acm.org.*
- *Changxi Zheng is with the Department of Computer Science, Columbia University, 616 Schapiro (CEPSR), New York, NY 10027. Email: cxz@cs.columbia.edu.*
- *Corresponding author: Kun Zhou.*

an optimal set of guide hairs for interpolation; and (ii) to enable sparsely distributed guide hairs to reliably detect collisions with solids, we exploit the densely distributed normal hairs to "propagate" hair-solid collisions to the guide hairs, leading to a novel two-way correction scheme for collision resolution at interactive rates.

While our runtime simulation method is independent from any particular guide-hair precomputation algorithm, we further propose a data-driven approach to precompute multiple sets of guide hairs and their interpolation weights. This approach learns a guide-hair skinning model from a set of hair animation data pre-simulated using a non-reduced hair model. With a single set of training data, we are able to simulate a wide range of hair-solid interactions at runtime, and the resulting hair animations are comparable to the full simulation results. Moreover, our precomputation and adaptive runtime simulation are made possible by a strand-based polynomial representation of hair skinning relationship. This representation is compact, significantly reducing memory footprint, and facilitates spatially coherent hair interpolation at runtime.

Our adaptive hair skinning model can account for hair friction, as long as both the training data and the runtime guide hairs are simulated with a friction model. Unfortunately, accurately modeling hair friction for a large hair volume remains computationally expensive. While an interactive hair friction model is out of our scope, we demonstrate that our interactive hair-solid simulation can obtain frictional effects by incorporating a simple and fast friction model (Section 7.1).

Contributions. In summary, we introduce three major technical contributions for interactive hair-solid simulation:

- We develop an adaptive algorithm for choosing guide hairs that affect a normal hair at runtime (Section 4).
- We design a two-way collision correction algorithm to avoid hair-solid intersections at runtime (Section 5).
- We propose a data-driven approach to precompute multiple eligible sets of guide hairs and their interpolation weights (Section 6).

We demonstrate the efficacy and robustness of our method simulating a full head of hair (with 150K strands) with various hairstyles at an interactive frame rate (Figure 1). Experiments show that our method produces similar results as produced by full-space simulations. To our knowledge, this kind of interactive hair-solid simulation has not yet been achieved.

2 Prior Work

There exist numerous work on creating realistic and efficient hair animation. Here we review topics most relevant to our interactive hair-solid simulation, while referring to the surveys [2], [3], [4] for more elaborate discussion.

Hair Dynamics. Hair has been modeled in a variety of ways. Perhaps the most classic approach, and the one commonly used in computer graphics, is to treat hair strands as particles interconnected with springs, starting from the pioneer work [5], [6]. Other successive work have focused on modeling bending and stretching effects of hair strands [7], [8]. Later, Selle et al. [9] augmented the mass-spring model to handle torsion and stretch-limiting for improved realism. More recently, Iben et al. [10] addressed the needs for artistic control in spring-mass hair simulation, and demonstrated its efficacy in production of feature films.

To accurately model hair strands' nonlinear bending and twisting deformations, *Kirchhoff rods* [11] has been introduced in graphics simulation [12]. Along this line of models, Bertails et al. [13] proposed Super-Helices to animate naturally shaped strands with high-order piecewise helical rods; Bergou et al. [14] used the twist-free Bishop frame to parameterize strands' material frame and facilitate the discretization of Kirchhoff equations. Later, Casati and Bertails-Descoubes [15] used a high-order rod element with affine curvature, and introduced an accurate integration scheme based on power series expansions to simulate highly curly strands.

Different from these methods, we do not propose a specific hair strand integrator. Rather, we focus on the interactive simulation of a large volume of hair strands interacting with solids. Our method can incorporate any of these hair models, as long as its computational cost is affordable for the interactive simulation of selected guide hairs. In our examples, we implement a spring-particle model, as it allows us to animate a full head of hair with 150K strands.

Hair-solid Interaction. Robustly simulating hair-hair and hair-solid interactions has long been a challenging problem. McAdams et al. [16] relied on a fluid solver to capture the spatial coherence of hair motions, while using spring-mass model to resolve detailed hair contacts and collisions. Other work have focused on accurate simulation of hair-hair interactions: Daviet et al. [17] introduced an iterative solver that computes hair dynamics in the presence of Coulomb friction for hair-hair contacts. Using the solver of [17] and the elastic rod model [14], the recent work of Kaufman et al. [18] introduced a collision response algorithm that adapts the degree of nonlinearity during the simulation.

To handle hair-solid interactions, common approaches are similar to those for cloth simulation, as they all involve small geometric features and require robust treatment of contacts and collisions. Bridson et al. [19] introduced a practical framework combining a fail-safe collision method with a fast repulsion force method. It detects collisions using a continuous collision detection method, which was improved in later works [20], [21]. This method was made more robust with globally-coupled geometric collision handling [22] and an improved impact zone method [23]. While robust for complex collision scenarios, these methods are generally expensive; it is not straightforward to apply them in interactive hair simulations. Another efficient solution is to represent solid objects implicitly using a signed distance field [24], [25]. Collisions are then detected in constant-time by querying distance values. Selle et al. [9] adopted this method to resolve collisions between hair and upper body by interpolating distance values using a semi-Lagrangian advection scheme. Our runtime simulation adopts this method for robust hair-solid collision resolution.

Reduced Hair Simulation. In parallel to the hair models focusing on realism, various reduced hair models have been devised. Exploiting the spatial coherence of hair motions, hair simulation has been reduced and simulated as continuum volume [16], [26] or coupled particles [27]. These models enjoy fast computational performance, but capture limited hair motion details, especially in the presence of complex hair-hair and hair-solid interactions.

We follow another widely adopted school of reduced hair

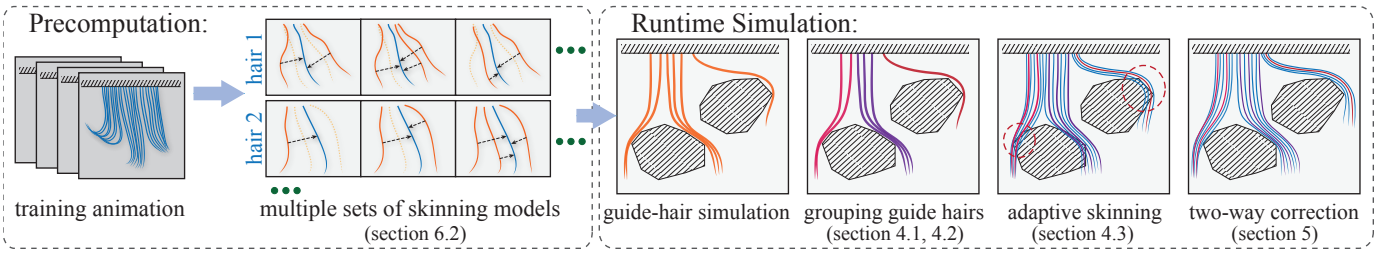


Fig. 2: **Overview of our pipeline.** We perform offline simulation to create a few sequences of training animation. Then we preform skinning precomputation, multiple sets of skinning models are assigned to each normal hair. At runtime, we simulate guide hairs to resolve large-scale collision. All guide hairs are dynamically grouped into many clusters. Each normal hair is then calculated using one of its adaptive skinning models that obey to current guide clusters. Finally, two-way correction is applied to further resolve remaining collisions.

models: among all hair strands, only a small subset of strands (i.e., guide hairs) are explicitly simulated, and the rest of the hairs are interpolated using guide hairs [28], [29]. A recent data-driven reduced hair model [30] can simulate up to 8000 strands in realtime, but neglects the entire hair-hair interactions.

Multi-resolution hair representations have long been used in hair modeling and animation [31], [32]. These methods adaptively choose the local resolution of a control structure for efficient simulation and rendering. The adaptation is determined by the hair velocity, visibility, and the distance from the viewer, but not hair-solid contacts. In comparison, our method always animates and renders the entire hair model, but our adaptive selection of guide hairs is determined by hair-solid contacts.

Our algorithm of preparing multiple sets of guide hairs is relevant to the method of Chai et al. [33], which automatically selects guide hairs and builds a reduced hair model using precomputed animation data, also similar in spirit to other methods for cloth and deformation simulation [34], [35], [36]. Unlike our method, their method may not produce spatially coherent skinning weights along hair strands. More importantly, they focus on the fast simulation of hair-hair interactions, but are insufficient to simulate complex hair-solid interactions (details in Section 3.2), which we address by introducing two new techniques, namely adaptive runtime skinning and two-way collision correction.

Bertails et al. [13] also used their Super-Helix model to animate sparse guide strands. Their interpolation skinning weights vary smoothly along a hair strand. We also use a per-strand skinning model; we further use a polynomial representation of the per-strand skinning weights to reduce the memory cost. Simulating only guide hairs has fundamental limitations to capture detailed hair-solid interactions, as discussed in Section 3.2 and Section 7.1. It is those limitations that we focus to address in this paper.

Recently, Somasundaram [37] proposed a method to resolve hair-solid collisions in a reduced hair simulation. The main idea is detecting penetrations at runtime and updating interpolation weights to push interpolated normal strands away from solids. We found that a position-based correction approach can introduce temporal incoherence (Figure 5) and even fail to interpolate properly in certain cases (e.g., the bottom row of Figure 6).

Instead of using guide hairs, an alternative approach of Müller et al. [38] simulates hair and fur using one iteration per timestep while ensuring the hair's inextensibility. This is achieved using the algorithm of *Follow The Leader* [39] combined with position-based dynamics (PBD) [40]. Because of the nature of PBD, this method is not easy to control the hair's material properties such as stiffness. Moreover, it simplifies the collision volume using ellipsoids in exchange of performance. As a result, this method is not as accurate as physically based techniques. In the supplemental video, we compare this method with our simulation method.

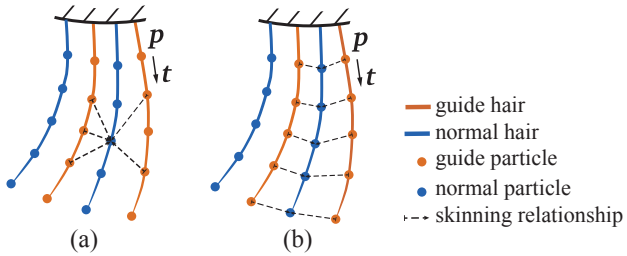


Fig. 3: **Particle-based and strand-based hair skinning.** (a) The particle-based hair skinning in [33], in which every normal particle can be affected by multiple guide particles on a guide hair (Section 3.1). (b) In our strand-based hair skinning, every normal particle is affected by one guide particle on a guide hair, enabling a compact polynomial representation of the skinning weights (Section 6.1).

3 Motivation and Overview

We start by briefly reviewing the standard reduced hair simulation method (Section 3.1), and then discuss its fundamental limitations for simulating hair-solid interactions (Section 3.2). Lastly, we overview the major steps of our method (Section 3.3) and highlight the key differences from the traditional reduced method.

3.1 Background on Reduced Hair Model

The standard reduced hair model simulates a small set of guide hairs and interpolates a large set of normal hairs. Here we follow the notation of Chai et al. [33]. Consider hair strands attached to a rigid head model. Every hair strand is represented using a B-spline curve defined by a chain of particles. The state of a hair particle is described by a tuple of $\boldsymbol{s} = (\boldsymbol{p}, \boldsymbol{t})$ indicating its position $\boldsymbol{p} \in \mathbb{R}^3$ and tangential direction $\boldsymbol{t}$ in the world frame of reference (Figure 3). Further, let $\bar{\boldsymbol{s}} = (\bar{\boldsymbol{p}}, \bar{\boldsymbol{t}})$ denote a hair particle position in the head's local frame of reference. In other words, if the head's rigid transformation is T, then $\boldsymbol{s}$ relates to $\bar{\boldsymbol{s}}$ through $\boldsymbol{s} = \mathsf{T}\bar{\boldsymbol{s}}$, in which $\boldsymbol{p}$ is translated and $\boldsymbol{t}$ is rotated by

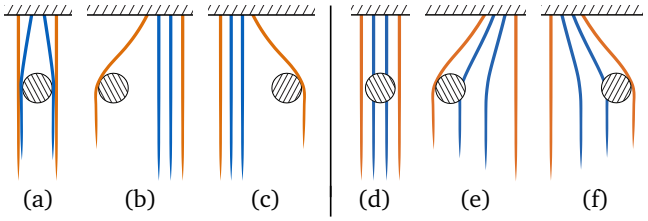


Fig. 4: **A simple solid-hair interaction.** Two normal hairs (in blue) are interpolated using two guide hairs (in orange). **(a–c)** In our proposed simulation method, the normal hairs are interpolated with adaptively selected guide hairs, leading to physically correct results. **(d–f)** In previous reduced hair simulation, normal hairs are always interpolated using fixed sets of guide hairs, resulting in unnatural hair shapes and intersections with the solid object.

T. In addition, when the head is static, every particle has a rest state $\boldsymbol{s}^*$ in the world space and a corresponding rest state $\bar{\boldsymbol{s}}^*$ in the head's local space.

A reduced hair model consists of a small group of hair strands as *guide hairs* and the rest as *normal hairs*. At runtime, only the states of the guide hairs are simulated, while the states of the normal hairs are interpolated using guide hairs. In particular, for every normal particle i (i.e., a particle on a normal hair), it associates with a set $\mathscr{C}_i$ of guide particles (i.e., the hair particles of some guide hairs). At each simulation step, after updating the states of all guide particles, this method interpolates the states of every normal particle using

$$\boldsymbol{s}_i = \mathsf{T}\left(\sum_{g\in\mathscr{C}_i} w_{ig}\mathsf{B}_g\bar{\boldsymbol{s}}_i^*\right), \tag{1}$$

where B_g is a linear transformation that transforms a guide particle g from its rest state $\bar{\boldsymbol{s}}_g^*$ to its current state $\bar{\boldsymbol{s}}_g$ in the head's local space (i.e., $\bar{\boldsymbol{s}}_g = \mathsf{B}_g\bar{\boldsymbol{s}}_g^*$), T is the current rigid transformation of the head, and w_{ig} are hair skinning weights. This interpolation scheme is similar to the Linear Blend Skinning models [1] widely used in deforming meshes [34], so we refer it as a *hair skinning* model.

3.2 Fundamental Limitations for Hair-Solid Interaction

Existing reduced hair models are able to simulate plausible hair motions interactively (e.g., [29], [33]). However, when simulating hairs interacting with solid bodies — especially when the solids are controlled by the user at runtime and thus have no predictable motions, the existing methods are inherently limited:

1) While the interpolation weights (i.e., w_{ig} in (1)) can be spatially varying, they, once computed, typically stay unchanged all the time in a simulation. The assumption here is that for every normal hair, its entire set of guide hairs manifest similar motions throughout the runtime simulation, so the normal hairs can always be interpolated as some states in between. However, this assumption breaks when a user-controlled object interacts with hairs. As shown in Figure 4.(a-c), a solid object can split a group of guide hairs (e.g., the orange hairs in Figure 4), causing completely different motions among them. In fact, when hair strands interact with solids, it is almost impossible to use fixed guide hairs and skinning weights all the time; the set of guide hairs that affect a normal hair ought to be dynamically changed.
2) Guide hairs, which are sparsely selected over the entire volume of hairs, can not fully resolve hair-solid collisions at runtime. Because of the guide hair sparsity, it is quite possible that the geometric features of solid objects are smaller than the granularity of guide hairs (Figure 4.(d)). Consequently, guide hairs can be unaware of the collisions between normal hairs and solids, and thus unable to "guide" the normal hairs to avoid collisions. Increasing the density of guide hairs would allow guide hairs to capture more collisions, but also drastically sacrifice the interactive performance. The position-based hair correction method in [33] designed for resolving hair-hair collisions may also be used to resolve hair-solid collisions. Unfortunately, it suffers from flickering artifacts due to the lack of temporal coherence, because its position-based correction disregards hair strands' velocity continuity with respect to the solids (see Figure 5 and supplemental video).

3.3 Method Overview

Runtime Simulation with Multiple Sets of Guide Hairs. To tackle these limitations, we first augment the reduced hair model by introducing *multiple* sets of guide hairs for every normal hair, in contrast to a fixed single set of guide hairs in previous methods. This augmentation in turn allows us to implement two key ideas:

- **Adaptive hair skinning (Section 4).** To address the first limitation above, we adaptively choose a set of guide hairs when interpolating the state of a normal hair, taking into account possible hair-solid collisions. As a result, guide hairs and the interpolation weights are time-varying during the simulation.
- **Two-way collision correction (Section 5).** We further correct hairs to avoid hair-solid intersections using a new two-way collision correction algorithm. In this algorithm, not only the normal hairs are interpolated using guide hairs, but guide hairs are also adjusted according to the intersections between normal hairs and solids. Thereby, we are able to resolve hair-solid collisions with only guide hairs explicitly simulated, and thus retain the interactive simulation performance.

Guide-Hair Precomputation. Our method does not depend critically on any particular scheme for selecting guide hairs and computing interpolation weights. We demonstrate that even with a naïve skinning model that computes interpolation weights at run-time, our adaptive hair-skinning method produces significantly better results (see Figure 11 and video). Further, we propose a data-driven method to automatically compute exhaustively many eligible sets of guide hairs for every normal hair and estimate corresponding skinning weights (Section 6).

4 Adaptive Hair Skinning

We first present our runtime method for simulating hair-solid interactions. Different from previous methods, it requires for every normal hair s multiple eligible sets of guide hairs, denoted as $\boldsymbol{G}_\mathsf{s}$. Every element $\mathscr{G} \in \boldsymbol{G}_\mathsf{s}$ is a set of guide hairs, whose hair skinning weights $w_{\mathsf{g}\to\mathsf{s}}, \forall \mathsf{g} \in \mathscr{G}$ need to be precomputed before the simulation. We defer the details of preparing guide hairs and weights until Section 6 and assume they are given in this section.

Like previous reduced hair models, to obtain interactive performance, we simulate only guide hairs and use them to

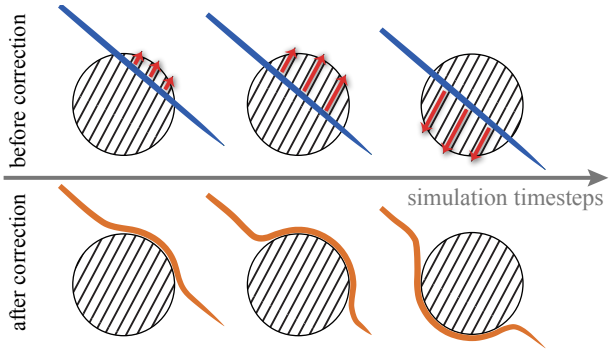


Fig. 5: **Temporal incoherence of position-based hair correction.** (top) Hair interpolation is unaware of the hair's collision states; (bottom) position-based correction at individual frame lacks the temporal coherence and can introduce flickering artifacts.

interpolate normal hairs. While there exist many hair simulation models, we choose the one of [9] and handle hair-object and inter-hair collision and friction as described in Section 7.1.

Adaptive Skinning in Brief. After simulating guide hairs at each timestep, we choose for every normal hair s an element $\mathscr{G}$ of its eligible guide-hair sets $\boldsymbol{G}_\mathsf{s}$. We then use the guide hairs in $\mathscr{G}$ to interpolate the state of s (recall (1)). When choosing $\mathscr{G}$, we take into account two criteria:

- **Hair motion similarity (Section 4.1).** Guide hairs in $\mathscr{G}$ must have similar states and velocity. If their motions are radically different, they are likely affected by external objects. Interpolating using their linear combination makes little sense physically.
- **Avoiding collisions with solids (Section 4.2).** Interpolations using guide hairs in $\mathscr{G}$ must avoid collisions with solids. For instance, if the guide hairs in $\mathscr{G}$ are separated by a solid in-between (Figure 4.(e-f)), they should not be used together for interpolating a normal hair, as the resulting normal hair would likely colliding with the solid object.

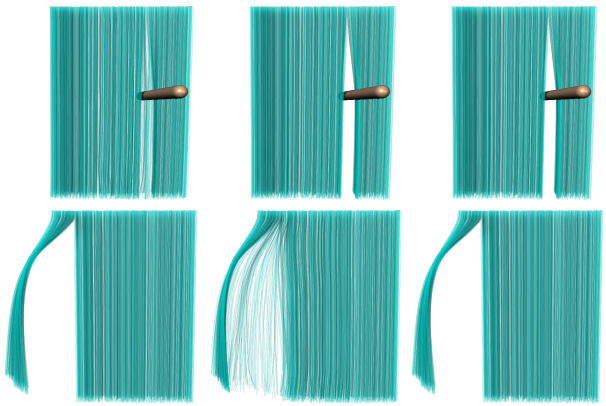
In brief, we cluster all potential guide hairs into groups using a graph representation: a node in the graph represents a guide hair; two nodes are directly connected only when the corresponding guide hairs satisfy the two criteria above. Finally, we choose the best $\mathscr{G} \in \boldsymbol{G}_\mathsf{s}$ that is also consistent with the clustered groups. We also note that both criteria are complementary to each other, rendering the guide-hair interpolation more robust (Figure 6).

4.1 Motion Similarity

When not affected by external objects, two guide hairs that are in proximity have similar shapes and moving velocities. This similarity is disrupted when a solid interacts with guide hairs. We therefore define a measure of the runtime motion similarity between two guide hairs i and j:

$$d(\mathsf{i},\mathsf{j}) = \frac{1}{A_{\mathsf{ij}}} \left\| \boldsymbol{p}_\mathsf{i} - \boldsymbol{p}_\mathsf{j} \right\|^2 + \frac{\sigma}{B_{\mathsf{ij}}} \left\| \boldsymbol{v}_\mathsf{i} - \boldsymbol{v}_\mathsf{j} \right\|^2, \qquad (2)$$

where $\boldsymbol{p}_\mathsf{i}$ and $\boldsymbol{p}_\mathsf{j}$ stack current positions of all the particles on the hairs i and j respectively; $\boldsymbol{v}_\mathsf{i}$ and $\boldsymbol{v}_\mathsf{j}$ respectively stack their velocities; σ is a user-controlled scalar to balance both terms, and A_{ij} and B_{ij} are constants at runtime to normalize the position and velocity discrepancies respectively. In our implementation, σ is set to 5, and A_{ij} and B_{ij} are automatically chosen based on



(a) (b) (c)

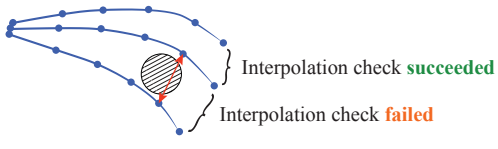
Fig. 6: **Comparison of guide-hair selection criteria**. We compare different selection criteria in two situations: (top) the hair strands are split by a thin and solid rod; and (bottom) after picking up a small group of hair, the solid rod is removed and the hair starts to drop down. **(a)** We select guide hairs based on only motion similarity; the interpolated normal hairs intersect with the solid rod (a-top). **(b)** We use only collision avoidance to select guide hairs; and the interpolated normal hairs spread unnaturally when dropping down (b-bottom). **(c)** Combing both criteria together for guide-hair selection, we obtain plausible hair motion in both situations.

provided training data. We defer this implementation detail until Section 7.1.

4.2 Avoiding Collisions with Solids

Next, we check for every pair of guide hairs if the linear combination of their states possibly intersects with solids. This check is complementary to the motion similarity measure introduced above, because even if the shapes and velocities of two guide hairs are close to each other, their interpolation can still collide with a solid body (Figure 4(d)). A normal hair in essence is a linear combination of a set, $\mathscr{G}$, of guide hairs. Thus, as long as the linear interpolation between any pair of guide hairs in $\mathscr{G}$ is collision-free from solids, the interpolated normal hair is most likely collision-free as well.

To maintain interactive simulation performance, we perform the check approximately and keep it lightweight. Consider two guide hairs i and j which have N_i and N_j hair particles respectively. Let $p_i(t)$ and $p_j(t)$ denote respectively the t-th particles on both hairs. Assuming hair particles are uniformly distributed on the hair strands in our setup, we simply check if the line segment connecting $p_i(t)$ and $p_j(t)$ intersects with any solid body for all $t = 1 \ldots \min(N_\mathsf{i}, N_\mathsf{j})$. This check is a standard continuous collision detection for which fast algorithms exist [20]. In the rest of the



presentation, we refer these checks as the guide hairs' *interpolation check*.

4.3 Adaptation of Guide Hairs

With the two criteria depicted, we now present our algorithm for runtime guide hair adaptation. Our idea is to cluster guide

Algorithm 1 Runtime guide-hair adaption and interpolation

Require: guide-hair graph $\mathscr{M} = (\mathscr{V}, \mathscr{E})$
eligible sets of guide hairs, $\boldsymbol{G}_\mathsf{s}$ for each normal hair s
1: $\boldsymbol{C} \leftarrow \emptyset$
2: **for** each guide-hair node $g \in \mathscr{V}$ **do**
3: $\mathscr{T} \leftarrow \emptyset$
4: **for** each adjacent node t of g on $\mathscr{M}$, $t \in \mathscr{V}$ **do**
5: $\mathscr{T} \leftarrow \mathscr{T} \cup \{t\}$
6: **end for**
7: **while** $\exists a, b \in \mathscr{T}$, a and b are not connected on $\mathscr{M}$ **do**
8: **if** $d(g,a) > d(g,b)$ **then** ▷ Keep the closer guide hair
9: $\mathscr{T} \leftarrow \mathscr{T} \setminus \{a\}$
10: **else**
11: $\mathscr{T} \leftarrow \mathscr{T} \setminus \{b\}$
12: **end if**
13: **end while**
14: $\mathscr{T} \leftarrow \mathscr{T} \cup \{g\}$; $\boldsymbol{C} \leftarrow \boldsymbol{C} \cup \{\mathscr{T}\}$
15: **end for**
16: **for** each normal hair s **do**
17: $\boldsymbol{T}_\mathsf{s} \leftarrow \emptyset$
18: **for** each eligible guide-hair set $\mathscr{G} \in \boldsymbol{G}_\mathsf{s}$ **do**
19: **for** each guide-hair cluster $\mathscr{C} \in \boldsymbol{C}$ **do**
20: $\boldsymbol{T}_\mathsf{s} \leftarrow \boldsymbol{T}_\mathsf{s} \cup \{\mathscr{G} \cap \mathscr{C}\}$
21: **end for**
22: **end for**
23: $\mathscr{T}_\mathsf{s} \leftarrow$the best set of guide hairs from $\boldsymbol{T}_\mathsf{s}$ ▷ Eq. (3)
24: interpolate state of s using guide hairs in $\mathscr{T}_\mathsf{s}$ ▷ Eq. (5)
25: **end for**

hairs into groups such that the guide hairs in each group can be used together for interpolating normal hairs. Taking into account these groups and a normal hair's eligible sets of guide hairs, we select the normal hair's best set of guide hairs, from which we interpolate its state (Algorithm 1). Figure 7 visualizes the guide-hair adaptation using one of our examples.

Guide-Hair Graph. We create a guide-hair graph $\mathscr{M} = (\mathscr{V}, \mathscr{E})$ consisting of a set of nodes $\mathscr{V}$ and edges $\mathscr{E}$. Every node in $\mathscr{V}$ corresponds to a guide hair, and thus we refer it as a guide-hair node in Algorithm 1. An edge e_{ij} connects nodes i and j when two conditions are satisfied: (i) the motion similarity $d(\mathsf{i},\mathsf{j})$ between two guide hairs i and j that correspond to the two nodes is less than a threshold δ ($\delta = 2$ in all our examples); and (ii) the guide hairs i and j pass the interpolation test described in Section 4.2. In short, if two nodes i and j are connected, potentially we can use the linear combination of the corresponding guide hairs to interpolate a normal hair.

Guide-Hair Groups. With the guide-hair graph, we cluster guide hairs into groups so that hairs in each group can be used together for interpolating a normal hair. If $\mathscr{G}$ is such a guide-hair group, then any two hairs in this group should be connected on the graph $\mathscr{M}$, owing to the construction of the graph. In other words, this group needs to be *complete* (or fully connected) on $\mathscr{M}$. For robust interpolation of normal hairs, we wish to identify guide-hair groups whose sizes are as large as possible. This is a typical *maximum clique problem*, a well-known NP-complete problem [41].

To obtain interactive performance, we propose an approximate algorithm (Line 1-15 of Algorithm 1). Because of the bijective mapping between graph nodes and guide hairs, we refer to them interchangeably herein. For every guide hair g, we identify a complete subgraph which contains g using a simple algorithm: we start from a set $\mathscr{T}$ consisting of nodes that are adjacent to g (Line 3-6 of Algorithm 1). Then we repeatedly check all pairs of nodes in $\mathscr{T}$. If there exists two nodes $\mathsf{a}, \mathsf{b} \in \mathscr{T}$ that are not connected, we simply remove one of the two nodes from $\mathscr{T}$ and repeat the check (Line 7-13 of Algorithm 1) until all notes in $\mathscr{T}$ are connected on $\mathscr{M}$. In particular, we remove the node who has larger motion difference from the guide hair g (i.e., $\arg\max_{\mathsf{i}=\mathsf{a},\mathsf{b}} d(\mathsf{i},\mathsf{g})$). Finally, the resulting $\mathscr{T}$ becomes one guide-hair group. After adding the guide hair g into $\mathscr{T}$, we add $\mathscr{T}$ into a set $\boldsymbol{C}$ (Line 14 of Algorithm 1).

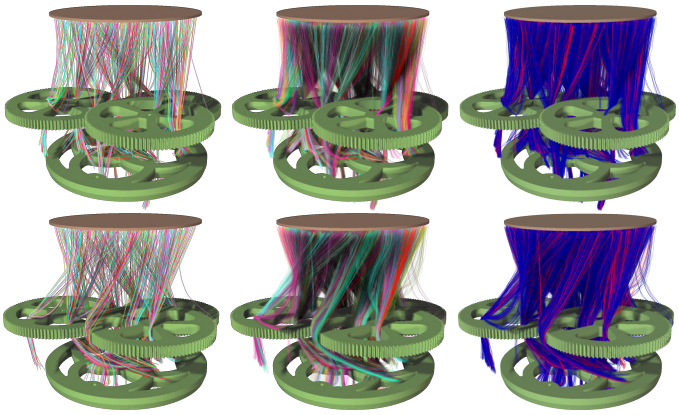


Fig. 7: **Adaptive hair skinning.** We visualize the adaptivity of hair skinning using two simulation snapshots in the top and bottom row, respectively. **(left)** We visualize guide hairs each with a different color. **(middle)** The hair interpolation weights are color-mapped at the two simulation frames. The weight distribution dynamically changes. **(right)** The current effective guide hairs (in red) that are serving for interpolation. Because of the adaptivity, the sets of guide hairs change during the simulation.

Guide-Hair Assignment. Now we have a set of guide-hair groups, denoted by $\boldsymbol{C}$, that can be used for normal-hair interpolation. On the other hand, a normal hair s has many eligible sets of guide hairs, $\boldsymbol{G}_\mathsf{s}$, assembled at the precomputation stage (Section 6). We therefore combine $\boldsymbol{C}$ and $\boldsymbol{G}_\mathsf{s}$ to select the guide hairs of s and meanwhile maximize the temporal coherence of the interpolated state of s. Namely, let $\mathscr{G}_\mathsf{s}$ be the set of guide hairs used for interpolating s in the *previous* timestep. We select its updated guide hairs $\mathscr{T}_s$ by computing

$$\mathscr{C}^*, \mathscr{G}^* = \arg\min_{\substack{\mathscr{C}\in\boldsymbol{C} \\ \mathscr{G}\in\boldsymbol{G}_\mathsf{s}}} \|\boldsymbol{s}(\mathscr{C}\cap\mathscr{G}) - \boldsymbol{s}(\mathscr{G}_\mathsf{s})\|^2 \quad \text{and} \quad \mathscr{T}_s = \mathscr{C}^* \cap \mathscr{G}^*, \tag{3}$$

where $\boldsymbol{s}(\cdot)$ is a vector stacking the interpolated states of hair particles on s using the provided set of guide hairs. In short, we choose among all eligible and valid guide hairs a set that leads to an interpolation with the best temporal coherence. As a result, $\mathscr{T}_\mathsf{s}$ consists of the guide hairs that are most suitable for interpolating s and unlikely introduce collisions. The skinning weights of guide hairs in $\mathscr{T}_\mathsf{s}$ have been estimated at the precomputation stage. Thus, we set $\mathscr{T}_\mathsf{s}$ as the current guide hairs of s, and update its state by computing $\boldsymbol{s}(\mathscr{T}_s)$ using (1).

5 Two-Way Collision Correction

In this section, we address the second limitation discussed in Section 3.2. Namely, it is insufficient to fully resolve hair-solid col-

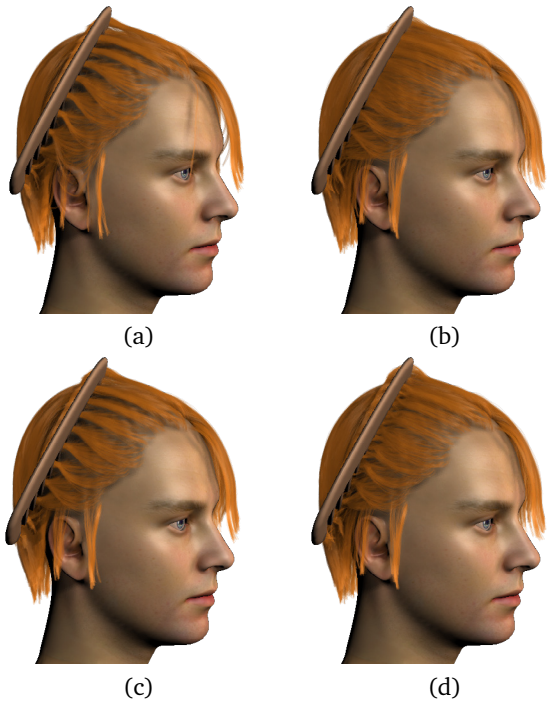


Fig. 8: **Comparison of two-way collision correction**. We compare four different schemes to resolve collisions: (a) full-space simulation, (b) guide-hair interpolation only, (c) our two-way collision correction, (d) the position-based collision correction as used in [33]. Our two-way correction (c) produces results closely matching the full-space simulation, while the other two (b,d) both show artifacts.

lisions by simulating only guide hairs that are sparsely sampled. The interpolated normal hairs can still suffer from intersections with solids (Figure 8). Adaptively updating guide hairs for every normal hair can alleviate the solid-hair intersections, but never eliminate them completely. This is because in the construction of guide-hair graph for updating guide hairs (Section 4.3), we perform the guide hairs' interpolation check approximately in exchange for performance. Further, when interacting with small solids, sparsely selected guide hairs are fundamentally limited for resolving the collisions.

One natural attempt is to resolve collisions with solids using the position-based hair correction method introduced in [33]. This method has proven fast and effective for resolving hair-hair collisions, and we follow it to resolve the hair-hair collisions in our system. However, when used for resolving hair-solid collisions, it can result in hair flickering. This is because the position-based method is a post-processing approach, disregarding the hair dynamics and the temporal coherence of velocities (see Figure 8 and the supplementary video).

Key Idea. Inspired by the schemes of applying forces from fine levels back to coarse levels [42], [43] in deformable simulations, we propose a two-way collision correction approach to resolve solid-hair collisions. Our approach is different from previous reduced hair simulation approaches, wherein after integrating guide-hair states at a simulation step, guide hairs are fixed for interpolating normal hairs. In other words, normal hairs are *one-way coupled* with guide hairs in previous approaches. In contrast, we allow the normal hairs that are in collision with solids to further affect the states of guide hairs, which in turn help normal hairs to avoid collisions. Thus, in our simulation, normal and guide hairs are *two-way* coupled. Through this two-way correction scheme, guide hairs are aware of the collisions even if they are not directly colliding with solid bodies.

5.1 Collision Detection

To enable fast collision detection between hair and solids, we precompute an object-space signed distance field for each solid body [24], and store it on a uniform grid [44]. Points outside of the object have positive distance values, and negative values are assigned inside. With this data structure, we can quickly detect collisions of hair particles against solid bodies by querying their distance values. Similar to other hair models (e.g., [13], [45]), we adopt a penalty method to compute collision forces for pushing hair particles away from interpenetration. By the end of the collision detection stage, we get a set of hairs that are in collision with solids, as well as the penalty forces on colliding hair particles. In the rest of this section, we use a vector $\boldsymbol{I}_{\mathsf{s}}$ to denote the penalty forces on the hair s, so $\boldsymbol{I}_{\mathsf{s}}(i)$ indicates the force on the i-th particle. If the hair s is not in collision, $\boldsymbol{I}_{\mathsf{s}}$ is zero.

5.2 Two-Way Correction via Collision Forces

The collision forces can be quickly computed by checking the distance values without degenerating the interactive performance. However, we can not afford to directly apply these forces to all hairs, as that would require computing internal forces and simulating the dynamics of all hairs. To maintain the interactive performance, we propose a two-way correction: we estimate effective collision forces on guide hairs for integrating the guide-hair dynamics. Afterwards, we interpolate normal hairs.

We reuse the skinning relationships between normal and guide hairs. Let $\mathscr{G}$ be the set of all guide hairs, and $\mathscr{N}$ be the set of all normal hairs. We estimate the effective collision forces $\boldsymbol{I}^*_{\mathsf{g}}, \mathsf{g} \in \mathscr{G}$ on guide hairs such that the interpolated collision forces on normal hairs approximate the forces computed at the collision detection step. This amount to solving a least-squares problem,

$$\min_{\boldsymbol{I}^*_{\mathsf{g}}, \mathsf{g}\in\mathscr{G}} \sum_{\mathsf{s}\in\mathscr{N}} \sum_{i=1}^{N_{\mathsf{s}}} \left\| \sum_{\mathsf{g}\in\mathscr{G}_{\mathsf{s}}} w_{\mathsf{g}\to\mathsf{s}}(i)\boldsymbol{I}^*_{\mathsf{g}}(i) - \boldsymbol{I}_{\mathsf{s}}(i) \right\|_2^2, \tag{4}$$

where N_{s} denote the number of hair particles on the hair s. $\mathscr{G}_{\mathsf{s}}$ is the set of guide hairs of s. $\boldsymbol{I}_{\mathsf{s}}(i)$ is the penalty force applied on the i-th particle of s. If s is collision free, then $\boldsymbol{I}_{\mathsf{s}}$ vanishes.

In practice, there is no need to solve the least-squares problem (4) as a single system. Very often, hair-solid collisions occur locally, and the guide hairs influence only normal hairs in their local regions. Therefore, in our implementation, we split this least-squares problem into multiple smaller subsystems, and solve them in parallel.

After estimating the effective collision forces $\boldsymbol{I}^*_{\mathsf{g}}, \mathsf{g} \in \mathscr{G}$, we apply these forces on guide hairs together with their direct penalty forces from collision detection, and then integrate the dynamics of guide hairs. It is possible that after the two-way correction there still exist collisions. One can repeat this step, and iterate a few times to eliminate collisions. In our implementation, we do not iterate inside each timestep, but simply use a smaller timestep size in the runtime simulation ($\Delta t = 10ms$ in our examples).

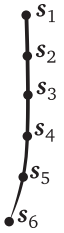
One additional advantage of using effective collision forces to correct hairs is the ability to eliminate flickering artifacts. This is because we essentially correct the acceleration of hairs and adjust their positions using time integrations, which is able to eliminate small discontinuities and produce temporally coherent motions.

6 Precomputation of Hair Skinning Model

Our runtime hair simulation relies on multiple sets of eligible guide hairs prepared for every normal hair, but does not depend on any specific algorithm to select those guide hairs and compute the interpolation weights. Even with a naïve skinning model that computes interpolation weights at runtime, our adaptive skinning method can improve the simulation results (see Section 7.2).

In this section, we present a new method to prepare a guide-hair skinning model. Our method is built on the method of Chai et al. [33], and offers two major advantages comparing to previous approaches: our skinning representation is memory-efficient and guarantees spatially coherent skinning weights; automatically learned from a provided full simulation data, our skinning model produces hair animations comparable to the full simulation, even with complex hair-solid interactions.

Hair Strand Notation. We first introduce some notation. Recall that every hair strand is represented by a chain of particles. When constructing the hair strands, we sample hair particles uniformly over the hair strands, so the distance between two adjacent particles of a hair is constant. We index these particles with an integer number. Consider a hair strand s with N_s particles. Following the same notation in Section 3.1, we use $\boldsymbol{s}_i, i = 1...N_\mathsf{s}$ to denote the states (i.e., $\boldsymbol{s}_i = (\boldsymbol{p}_i, \boldsymbol{t}_i)$) of the i-th particles starting from the hair root on a head scalp. Following the notation in Section 3.1, we put a bar over the letter (e.g., $\bar{\boldsymbol{s}}_i$) to distinguish the coordinates in the head's local frame of reference from the coordinates in the world frame.



6.1 Spatially Coherent Representation of Hair Skinning

Similar to [13], we use a strand-based hair skinning representation, in which all the particles of a normal hair are affected by the same set of guide hairs, and the skinning weights vary continuously along hair strands.

Hair skinning representation. For every normal hair s, it has a set $\mathscr{G}_\mathsf{s}$ of guide hairs. Our skinning model interpolates the state of every normal particle $\boldsymbol{s}_i$ using

$$\boldsymbol{s}_i = \mathsf{T}\left(\sum_{\mathsf{g}\in\mathscr{G}_\mathsf{s}} w_{\mathsf{g}\to\mathsf{s}}(i)\mathsf{B}_\mathsf{g}(i)\bar{\boldsymbol{s}}_i^*\right),\ i = 1...N_\mathsf{s}, \tag{5}$$

where $\bar{\boldsymbol{s}}_i^*$ is the rest-state coordinate of the i-th particle in the head's local frame, $\mathsf{B}_\mathsf{g}(i)$ linearly transforms the i-th particle of the guide hair g from its rest state to its current state in the head's local frame, and $w_{\mathsf{g}\to\mathsf{s}}(i)$ is the skinning weights for the i-th particle of g affecting the i-th particle of s. As indicated in (5), in our skinning model, all the particles of a normal hair are affected by the same set of guide hairs. The i-th particle of a normal hair depends only on the i-th particles of its guide hairs (Figure 3(b)). If a normal hair is longer than one of its guide hairs, then the normal particles that exceed the length of the guide hair are affected by the closest particles on guide hairs.

Algorithm 2 Precomputation of skinning for a normal hair

Require: a normal hair s and a set $\mathscr{G}$ of eligible guide hairs
procedure SKINNINGPRECOMPUTATION(s, $\mathscr{G}$)
  $\boldsymbol{G}_\mathsf{s} \leftarrow \emptyset$
  for each $\mathscr{G}' \subseteq \mathscr{G}$ and $\mathscr{G}' \neq \emptyset$ in ascending order of $|\mathscr{G}'|$ **do**
    for each guide hair $\mathsf{g} \in \mathscr{G}'$ **do**
      Estimate coefficients of $w_{\mathsf{g}\to\mathsf{s}}(i)$ in (6) ▷ Section 6.2
      if $\max w_{\mathsf{g}\to\mathsf{s}} < \epsilon$ **then**
        goto next_set ▷ Skip this guide hair set
      end if
    end for
    $\boldsymbol{G}_\mathsf{s} \leftarrow \boldsymbol{G}_\mathsf{s} \cup \{\mathscr{G}'\}$ ▷ s gets one more guide hair set
  next_set:
  end for
  return $\boldsymbol{G}_\mathsf{s}$
end procedure

Furthermore, we guarantee the spatial continuity of the skinning weights along hair strands by representing $w_{\mathsf{g}\to\mathsf{s}}(i)$ using polynomials. In our experiment, we found that second-order polynomials are sufficient to enable the skinning model to closely fit the training data (within 5% error, while in contrast linear fitting results in about 5× larger error) and also work well at runtime. Thus, $w_{\mathsf{g}\to\mathsf{s}}(i)$ takes a quadratic form,

$$w_{\mathsf{g}\to\mathsf{s}}(i) = ai^2 + bi + c, \tag{6}$$

where the coefficients a, b and c depend on a specific pair of guide and normal hairs, and are precomputed based on the training data (details in Section 6.2). We also note that, besides being able to largely reduce over-fitting, the polynomial expressions of $w_{\mathsf{g}\to\mathsf{s}}$ form a compact representation of hair skinning. It stores three scalars to describe the skinning relationship between a normal hair and a guide hair, consuming much less memory in comparison to the previous method which stores the weight per particle (about 25× reduction). This compactness is particularly beneficial for our adaptive skinning model, as we need to prepare multiple sets of guide hairs for every normal hair. In our experiments, the memory footprints are always less than 400MB, even for hair models with more than 100K strands.

Remark. Notable differences exist between our hair skinning model and the previous method [33]. Previously, the skinning is defined in terms of particles: a normal particles has a group of guide particles; the distribution of these guide particles can be arbitrary, whether they are on a single guide hair or multiple guide hairs. This scheme offers a flexibility to allow a normal particle to have its optimal set of guide particles, but fails to ensure the spatial coherence of guide-particle distributions. For instance, two adjacent normal particles, a and b, on a single normal hair can have guide particles from many different guide hairs (Figure 3(a)). When the motion discrepancy among those guide hairs becomes large, the interpolated positions of a and b start losing spatial coherence. In the case of hair-solid interaction, it is very likely for those guide hairs to have distinct motions; and the loss of spatial coherence leads to unpleasant artifacts.

6.2 Precomputation of Skinning Weights

Training data. We construct the hair skinning model based on provided hair animation data. While our focus is to simulate hair-solid interactions, it is neither practical nor necessary to

include hair-solid interactions in the training data. Despite infinitely possible hair-solid interactions, our skinning model (5) interpolates states of normal hairs purely based on the states of guide hairs. Therefore, all we need to extract from the training data is the correlations between different pairs of normal and guide hairs. This allows us to simply follow the previous method [33] to prepare the training data, including complete hair geometry in a rest state, several sequences of rigid head motions, and the resulting hair motions simulated using a full-space simulation method. We refer to their paper for more details. Throughout the training, no solid object is involved, and inter-hair friction is handled in the same way as Section 7.1.

Multiple Combinations of Guide Hairs. Our goal at the precomputation stage is to prepare multiple sets of guide hairs for every normal hair (see an overview in Algorithm 2). Consider one normal hair s. We first select a set $\mathscr{G}$ of eligible guide hairs, from which we build multiple sets of guide hairs for runtime adaptation. The selection of $\mathscr{G}$, based on the training data, follows the method in [33]. We further improve its efficacy using a sparse coding method, which we postpone until Section 6.3. Given a set $\mathscr{G}$ of eligible guide hairs, we check exhaustively all the (nonempty) subsets $\mathscr{G}' \subseteq \mathscr{G}$ (Line 3 of Algorithm 2), and estimate the skinning weights of the guide hairs in $\mathscr{G}'$ for interpolating s. If the weights $w_{\mathsf{g}\to\mathsf{s}}$ for a guide hair $\mathsf{g} \in \mathscr{G}'$ is too small, then g is unnecessary, and $\mathscr{G}'$ is excessive and thus discarded (Line 6-8 of Algorithm 2). Otherwise, we add $\mathscr{G}'$ as one of the potential guide hair sets for runtime selection. In the rest of this paper, we denote all the eligible guide hair sets of a normal hair s as $\boldsymbol{G}_\mathsf{s}$. Every element of $\boldsymbol{G}_\mathsf{s}$ is a set of guide hairs that can potentially serve for runtime interpolation of s. In practice, $\mathscr{G}$ typically consists of less than 10 guide hairs, so the total number of combinations of guide hairs in $\mathscr{G}$ is no more than 511. Discarding excessive combinations, we obtain about 100 different sets of guide hairs (i.e., $|\boldsymbol{G}_\mathsf{s}| \approx 100$) for every normal hair.

Estimation of $w_{\mathsf{g}\to\mathsf{s}}$. Now the step in Algorithm 2 that remains to be described is the estimation of $w_{\mathsf{g}\to\mathsf{s}}$ (Line 5 of Algorithm 2). Given a normal hair s and one of its guide hair set $\mathscr{G} \in \boldsymbol{G}_\mathsf{s}$, we first estimate the skinning weights for every normal particle on s, and then fit the weights along every guide hair using a quadratic polynomial (6).

We estimate the weights $w_{\mathsf{g}\to\mathsf{s}}(i), \forall \mathsf{g} \in \mathscr{G}$ by solving a constrained least-squares problem,

$$\min_{w_{\mathsf{g}\to\mathsf{s}}(i)} \sum_{f=1}^{F} \left\| \sum_{\mathsf{g}\in\mathscr{G}} w_{\mathsf{g}\to\mathsf{s}}(i) \mathsf{B}_\mathsf{g}(i,f) \bar{\mathbf{s}}_i^* - \bar{\mathbf{s}}_i(f) \right\|_2^2, \quad s.t.\ w_{\mathsf{g}\to\mathsf{s}}(i) \geq 0, \text{ and } \sum_{\mathsf{g}\in\mathscr{G}} w_{\mathsf{g}\to\mathsf{s}}(i) = 1. \tag{7}$$

Here we account for all animation frames $1...F$ in the training data; the notations are the same as those in (5); the parameter f indicates an input frame f. This least-squares problem is a zero-extended version of that in [33]. It solves for a single normal particle indexed by i and all guide hairs in $\mathscr{G}$, so we solve it for every particle on s (i.e., $i = 1...N_\mathsf{s}$) using a linearly constrained quadratic programming solver [46].

In contrast to the previous method, we further compress the weights by fitting the values of $w_{\mathsf{g}\to\mathsf{s}}(i)$ using a quadratic polynomials. This amounts to solving another small least-squares problem for every pair of normal and guide hairs,

$$\min_{a,b,c} \sum_{i=1}^{N_\mathsf{s}} [ai^2 + bi + c - w_{\mathsf{g}\to\mathsf{s}}(i)]^2. \tag{8}$$

We note that because we use a low-order (quadratic) polynomial regression, overfitting never happens in our examples; the resulting polynomials always produce positive weights for $i = 1...N_\mathsf{s}$. But it can not guarantee the unit constraint in (7). Therefore, we simply normalize the weights when using them at runtime.

Because of the preparation of multiple sets of guide hairs, the precomputation cost is certainly larger than that in the previous reduced hair simulation method. Nevertheless, we note that both least-squares problems (7) and (8) are lightweight, and can be solved in parallel for all pairs of guide and normal hairs.

In summary, at the end of the precomputation step, every normal hair has multiple eligible sets of guide hairs, whose hair skinning weights are also estimated. The weight function for every pair of normal and guide hairs is represented by a quadratic polynomial and stored using its three coefficients.

Fig. 9: **Comparison of guide hair precomputation.** A snapshot of a full simulation (left) is compared against the reduced simulation with guide hairs selected using our sparse coding method (middle) and the previous graph-based method [33] (right). Our results with sparse coding are visually closer to the full-space simulation (which are absent from the training data).

6.3 Guide Hair Selection using Sparse Coding

The precomputation of skinning weights of a normal hair s in Algorithm 2 requires a set, $\mathscr{G}$, of eligible guide hairs as input. $\mathscr{G}$ can be chosen using the method in [33], wherein it clusters all hairs based on their motion similarities in the training data and selects one guide hair from each individual cluster. While this method is directly applicable in our pipeline, its graph-based algorithm using a heuristic measure of hair motion similarity renders the selected guide hairs suboptimal.

We extend our method to further improve the guide hair selection, formulating it as a problem of sparse basis selection. In the hair skinning model, guide hairs serve as *basis functions*, whose linear combinations represent the states of normal hairs. Every normal hair is affected by only a small subset of guide hairs. In other words, the coefficients of the linear combinations need to be sparse, suggesting that we can formulate the guide hair selection as a best-subset-selection problem, one that is ubiquitous in many disciplines and a central topic in the field of Compressed Sensing [47].

Specifically, in the training data, we represent the state of a hair strand at a time frame f by stacking the states of the hair particles along that strand, that is, we denote a hair's state in the head's local frame of reference as $\bar{\mathbf{s}}(f) = [\bar{\mathbf{s}}_1(f)\ \bar{\mathbf{s}}_2(f)\ ...\ \bar{\mathbf{s}}_{N_\mathsf{s}}(f)]^T$. Again, we put a bar over the variables to indicate states in the head's local frame of reference. For a

normal hair s, we further stacking its states at every frame into a long vector $\boldsymbol{u}_s = [\bar{\boldsymbol{s}}(1)\ \bar{\boldsymbol{s}}(2); ...\ \bar{\boldsymbol{s}}_{N_f}]^T$, where N_f is the total number of frames in the training data. $\boldsymbol{u}_s$ therefore represents the state of a hair s in the training data. We then choose a set of basis vectors to represent hair states $\boldsymbol{u}_s$ for all hairs $\mathscr{S}$ in the training data. Our plan is to use the basis vectors to select guide hairs afterwards. Thus, we seek basis vectors whose linear combination can closely approximate all $\boldsymbol{u}_s$, and meanwhile the basis coefficients stay sparse. Formally, we solve a minimization problem :

$$\arg\min_{\mathsf{G},\boldsymbol{\alpha}} \sum_{s\in\mathscr{S}} \left(\|\boldsymbol{u}_s - \mathsf{G}\boldsymbol{\alpha}_s\|_2^2 + \beta \|\boldsymbol{\alpha}_s\|_1 \right), \tag{9}$$

where G is a matrix consisting of the basis vectors as its column vectors; and $\boldsymbol{\alpha}_s$ is a vector of coefficients for representing $\boldsymbol{u}_s$ using the basis in G. This is a well-known sparse coding problem (also known as *basis pursuit* [48]): the first term of (9) penalizes the approximation error; the second term penalizes the L_1 norm of $\boldsymbol{\alpha}_s$, leading to a sparse solution of $\boldsymbol{\alpha}_s$. The scalar β balances the weights of both terms; in practice we use 0.15. The user also specifies the number of basis vectors (i.e., the number of columns of G). In our case, it is the total number of guide hairs. In our implementation, we solve the problem (9) using the method of [49] and their released library.

After we solve (9), the resulting basis vectors in G also depict a picture of the "ideal" guide hairs: if there exist hairs whose state vector (i.e., $\boldsymbol{u}_s$) in the training data agree with the basis vectors, then they are the best choice of guide hairs under the metric used in (9). Therefore, for every column vector $\boldsymbol{g}$ of G, we find a hair g that is closest to $\boldsymbol{g}$, that is

$$\mathsf{g} = \arg\min_{s\in\mathscr{S}} \|\boldsymbol{u}_s - \boldsymbol{g}\|_2^2. \tag{10}$$

We repeat this process for every column vector $\boldsymbol{g}$ of G, obtaining the set, $\mathscr{G}$, of guide hairs. Lastly, we stack the state vectors $\boldsymbol{u}_g, \forall g \in \mathscr{G}$ in a matrix $\tilde{\mathsf{G}}$. We replace the basis matrix G in (9) with $\tilde{\mathsf{G}}$, and solve for $\boldsymbol{\alpha}_s$. The resulting $\boldsymbol{\alpha}_s$ is sparse, indicating skinning weights for interpolating a normal hair s using guide hairs in $\mathscr{G}$. After discarding the guide hairs whose corresponding coefficients in $\boldsymbol{\alpha}_s$ are less than a threshold ϵ ($\epsilon = 0.01$ in our examples), we obtain the set of eligible guide hairs of a normal hair s.

As shown in Figure 9, our formulation selects guide hairs that lead to much smaller fitting error and higher quality of runtime interpolation in comparison to the previous method.

7 Implementation and Results

7.1 Implementation Details

Motion Similarity. Within motion-similarity calculation (Section 4.1), A_{ij} and B_{ij} are constants at runtime to normalize the position and velocity discrepancies respectively. We precomptue their values using their averages in training animations:

$$A_{ij} = \frac{1}{F}\sum_{f=1}^{F} \left\| \tilde{\boldsymbol{p}}_i(f) - \tilde{\boldsymbol{p}}_j(f) \right\|^2 \ \text{ and } \ B_{ij} = \frac{1}{F}\sum_{f=1}^{F} \left\| \tilde{\boldsymbol{v}}_i(f) - \tilde{\boldsymbol{v}}_j(f) \right\|^2.$$

Here F is the total number of animation frames in the training data, and $\tilde{\boldsymbol{p}}_i(f)$ and $\tilde{\boldsymbol{v}}_i(f)$ are respectively the particle positions and velocities of the hair i at frame f of the training data.

Friction. Hair-hair and hair-object frictions are vital for realistic hair simulation. Following the method of Bridson et al. [19], we use an approximation of the Coulomb friction model [24] to handle frictions in both precomputation and runtime simulation. Specifically, for every detected collision, we estimate the repulsion force f_r along the normal direction of the contact point. The friction impulse is then approximated as $I_f = \mu f_r \Delta t$, where μ is the friction coefficient and Δt is the timestep. We determine the sticking-slipping status by comparing the velocity change caused by friction (i.e., $\Delta v = I_f/m$) with the current relative tangential velocity magnitude $|v_t|$ at the contact point. If $|v_t|$ is smaller than Δv, the contact is strictly stuck, otherwise slipping is allowed with relative tangential velocity $(1 - \Delta v/|v_t|)v_t$.

Note that our friction solution is a simplified approximation of the Coulomb friction model. While a more accurate solution is available for generating higher fidelity simulation [17], it is unaffordable in real-time simulation due to the high computation cost of iteratively solving the complex hair dynamics. The simplified solution fits well in our real-time system and is able to produce visually plausible results in practice (see video[1]).

Testing Environment. We implemented our adaptive skinning method in C++. The code was run on a commodity PC with a quad-core Intel i7 CPU (32GB memory) and a Nvidia GTX 760 graphics card. Under this setting, with about 400 guide hairs, our adaptive skinning method is able to interactively simulate and render solid-hair interactions in a variety of simulation scenes for as many as 150K normal hair strands (see video).

In the following sub-sections, we verify our method by comparing it with a full-space simulation and previous (or alternative) methods (Section 7.2), and then present our main results with a range of hair styles and solid-object setups.

7.2 Validation and Comparison

Robustness. To verify the robustness of our method for simulating complex solid-hair interactions, we compare our method with the previous work on interactive hair simulation [33], and use a full-space simulation as a reference. Figure 10 shows one snapshot of 80K simulated hairs. Both our method and the method of Chai et al. [33] are able to simulate them interactively. However, their method is unable to avoid complex hair-solid interactions; hairs can penetrate into the solids, leading to visually incorrect hair motions (insets of Figure 10). In contrast, our method is able to robustly avoid collisions while maintaining the interactive performance. By comparing to the reference, our simulation indeed produces similar motion as obtained using the full-space simulation (see video for the animated comparisons).

Adaptive Guide Hairs. We then validate our adaptive guide-hair selection scheme. Since we use two criteria for selecting guide-hairs, namely the motion similarity and collision avoidance (recall Section 4), we test their efficacy by comparing three combinations of the criteria: (i) motion similarity only, (ii) collision avoidance only; and (iii) the combination of both. Figure 6 illustrates the comparison. Using either the motion similarity or collision avoidance, the simulator shows artifacts at different situations. In fact, these criteria are complementary, so their combination produces much satisfying results (right most column of Figure 6).

Two-way Correction. To validate our two-way correction scheme, we compare four different methods (Figure 8): (i) a full-space simulation; (ii) guide hair interpolation; (iii) position-based hair correction as in [33], and (iv) our two-way correction.

1. A high-resolution version of the video can be downloaded from http://gaps-zju.org/hairsimulation-hd.mp4.

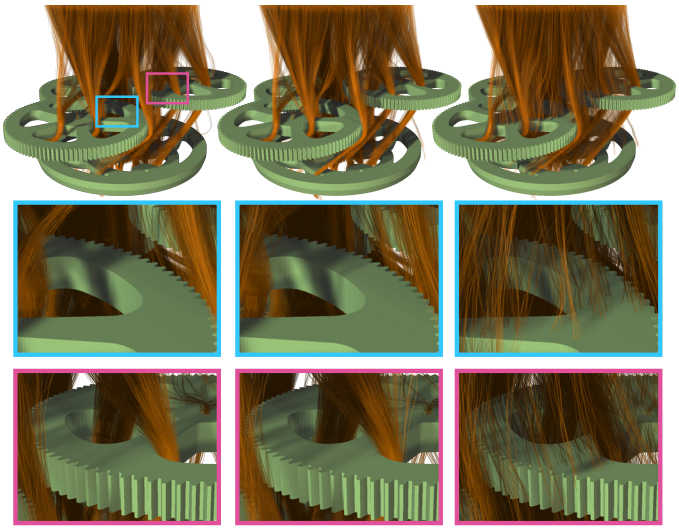


Fig. 10: **Comparison of reduced hair model.** We use a full-space simulation as a reference (left column), comparing it to our method (middle) and the previous reduced hair simulation method [33] (right). Our method resolves the hair-solid collisions well, while in the previous method, collisions remain.

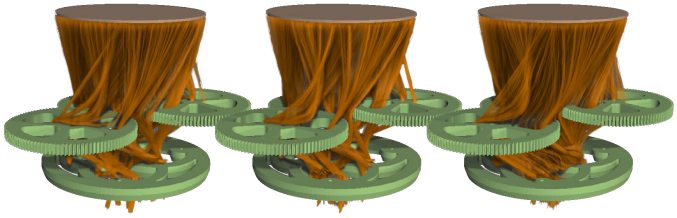


Fig. 11: **A naïve skinning model.** We test our method with a naïve skinning model. (left) Full-space simulation. (middle) The naïve interpolation result using our adaptive skinning method. (right) The naïve interpolation result without using our method.

Guide-hair interpolation disregards collisions and thus results in inter-penetrations. The position-based correction method is less natural as our two-way correction scheme, and more importantly it produces flickering artifacts (see video).

Working with Other Skinning Methods. As aforementioned, our adaptive skinning method can be incorporated with any guide-hair based reduced model. As an example, we tested our method with a naïve skinning model that randomly chooses a set of guide hairs. For each normal hair, a fixed number of closest guide hairs (ten in our experiments) are selected as its eligible guide hairs. At runtime, we use our adaptive skinning method to find the best guide hairs for each normal hair and computes the per-particle interpolation weights:

$$w_{\mathsf{g}\to\mathsf{s}}(i) = \frac{d_{\mathsf{g}\to\mathsf{s}}(i)}{\sum_{\mathsf{g}'\in\mathscr{G}_\mathsf{s}} d_{\mathsf{g}'\to\mathsf{s}}(i)}, \tag{11}$$

where $d_{\mathsf{g}\to\mathsf{s}}(i) = 1/\left\|\boldsymbol{p}_\mathsf{s}(i) - \boldsymbol{p}_\mathsf{g}(i)\right\|$ is the reciprocal of the distance between normal hair s and guide hair g at the ith particle. Even for such a naïve skinning model, our adaptive skinning method manages to resolve hair-object penetrations and improve the results. (see Figure 11). On the other hand, by comparing with our data-driven skinning model, our approach can produce results more closely to full simulation (see video).

Comparison with Alternative Methods. Bertails et al. [13] introduced a technique to effectively avoid collisions between the hair and upper-body during guide-hair interpolation. It defines a distance threshold d_{max} (typically the minimal penetration distance of target objects) and prevents interpolation between two guide hairs if their tip distance is greater than d_{max}. While this technique works well for relatively large objects such as head and body, it could introduce artifacts when handling intricate hair-solid interactions shown in our paper. This is because for objects having small features such as the fine teeth of a comb, d_{max} needs to be very small to avoid interpolation across objects. In this case, a large portion of guide pairs may have tip distances greater than d_{max}, and could be mistakenly prevented from being used in interpolation. Furthermore, tip distances of free-moving hair strands may jump across the threshold frequently during simulation, leading to temporal jittering due to the sudden change of the interpolation relationship. Please see the video for a comparison between the technique and our method.

We also compared our method with the real-time hair simulation method of Müller et al. [38]. Due to the known limitation of this PBD-based model (Section 2) to control physical properties such as stiffness values within limited iterations (one iteration per frame), it can result in excessive softness and unnatural dynamic behaviors especially for long hairs (see video).

Type	Stretch	Bend	Twist	Damping	Friction
straight	3×10^5	30	15	10	0.05
wavy	2×10^6	200	100	10	0.1

TABLE 1: **Dynamic parameters of our hair models**. From left to right: type of hair geometry, stretch spring stiffness, bending spring stiffness, twisting spring stiffness, spring damping, and friction.

Scene	Normal	Guide	Seg.	Time(o)	Time(f)	Memory
gears	80K	300	30	46ms	20s	120MB
hairy ball	60K	200	30	32ms	15s	80MB
hair salon	150K	400	25	50ms	60s	400MB

TABLE 2: **Statistics of experiment scenes**. From left to right: names of the scenes, number of normal hairs, number of guide hairs, averaged number of hair particles on every strand, average simulation time per frame with our method and full simulation, and memory footprint of the entire reduced model.

7.3 Main Results

We tested a variety of simulation scenes with the number of hairs ranging from 60K to 150K. The timestep size is set to 10ms for all scenes. For all our examples, the offline training takes less than two hours to precompute the skinning model, including training data generation with full-space simulation, guide selection using sparse coding and skinning estimation. The full-space simulation dominates the precomputation time (more than 80%). The runtime simulation takes 40ms–55ms per frame, within which, guide hair simulation costs 20-25%, guide hair grouping and assignment cost about 30%, adaptive interpolation costs less than 10%, and two-way collision correction costs about 30%. The dynamic parameters of our hair models are listed in Table 1 and the major statistics of these experiment scenes are listed in Table 2.

Gear Set. We place a set of complex gear combinations under 80K long hair strands. The gears have cavities such that strands can drop through the holes and fully contact with these detailed

Fig. 12: **Hairy sphere.** The user controls a hairy sphere to interact with a set of 3D solids.

Fig. 13: **Hair salon.** Interactive manipulation of hair strands with comb and animated hand. We demonstrate the simulation using two different hair styles, including both the straight hairs (left) and the highly curly hairs (right), with a friction model.

solids. We animate gears with non-uniform rotation, which forces strands into highly tangled shapes.

Toys. A hairball has 60K hairs and is interactively controlled by the user in a virtual environment to interact with 3D models (Figure 12). The strands experience collisions with such solid objects as smooth surface (e.g., the back of the horse) as well as sharp features (e.g., the armadillo's fingers).

Hair Interaction. We interact with a full head of hairs of virtual characters. To realistically depict virtual characters, we use 150K hairs interacting with a comb and pre-animated hand models. We also tested different hair styles including both the straight hairs and highly curvy hairs (Figure 13). Our method is able to well resolve collisions between hairs and the densely spaced comb teeth; it is also stable to form the hair strands into a plausible stacking shape when pushed aside.

8 Conclusion

We have introduced an adaptive hair skinning method for simulating a full head of hairs with complex hair-solid interactions interactively. Our method is featured with an adaptive algorithm for choosing guide hairs that interpolate a normal hair at runtime, a two-way collision correction algorithm to avoid hair-solid intersections, and a data-driven approach to precompute the eligible sets of guide hairs and their interpolation weights. Large scale interactive hair-solid simulation has not yet been achieved before. We therefore hope our method further advances the deployment of hair simulation in interactive applications.

Our data-driven approach to precompute the eligible sets of guide hairs can supply results visually comparable to the full simulation, but also has several limitations: i) The runtime simulated hairs need to share the same geometry and simulation parameters (e.g., the bending and twisting coefficients). The skinning relationship can not be reused in a completely new hair model. ii) The sets of eligible guide hairs precomputated using full-space simulation remain unchanged at runtime. If solid bodies break the motion coherence between a normal hair and *all* guide hairs, that normal hair will no longer have guide hairs for interpolation. Employing more densely distributed guide hairs could alleviate the problem, but also increases the runtime computation cost. Therefore, an effective algorithm to dynamically add new guide hairs into the eligible guide hair sets and remove unnecessary guide hairs at runtime is an interesting future direction. iii) We focus on hair-solid interactions, and adopt the hair collision correction method of Chai et al. [33] to handle hair-hair collisions, which is not able to completely avoid hair-hair penetrations and is frictionless. Extending our method to account for hair-hair interactions would be a promising future direction to further improve the animation quality.

Acknowledgments

This work is partially supported by the NSF of China (No. 61272305), the National High-tech R&D Program of China (No. 2012AA010903), the National Science Foundation of U.S. (CAREER-1453101), National Program for Special Support of Eminent Professionals of China, Lenovo's Program for Young Scientists, and generous gifts from Intel and Adobe. Any opinions, findings and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of funding agencies or others.

References

[1] N. Magnenat-Thalmann, R. Laperrire, and D. Thalmann, "Joint-dependent local deformations for hand animation and object grasping," in *Proceedings on Graphics interface'88*, 1988.

[2] K. Ward, F. Bertails, T.-Y. Kim, S. R. Marschner, M.-P. Cani, and M. C. Lin, "A survey on hair modeling: Styling, simulation, and rendering," *IEEE TVCG*, vol. 13, no. 2, pp. 213–234, 2007.

[3] F. Bertails, S. Hadap, M.-P. Cani, M. Lin, T.-Y. Kim, S. Marschner, K. Ward, and Z. Kačić-Alesić, "Realistic hair simulation: animation and rendering," in *ACM SIGGRAPH 2008 classes*, 2008, p. 89.

[4] C. Yuksel and S. Tariq, "Advanced techniques in real-time hair rendering and simulation," in *ACM SIGGRAPH 2010 Courses*, 2010.

[5] R. E. Rosenblum, W. E. Carlson, and E. Tripp, "Simulating the structure and dynamics of human hair: modelling, rendering and animation," *The Journal of Visualization and Computer Animation*, vol. 2, no. 4, pp. 141–148, 1991.
[6] K.-i. Anjyo, Y. Usami, and T. Kurihara, "A simple method for extracting the natural beauty of hair," *Computer Graphics (SIGGRAPH)*, vol. 26, no. 2, pp. 111–120, 1992.
[7] L. Petrovic, M. Henne, and J. Anderson, "Volumetric methods for simulation and rendering of hair," *Pixar Animation Studios*, 2005.
[8] R. Gupta, M. Montagnol, P. Volino, and N. Magnenat-Thalmann, "Optimized framework for real time hair simulation," in *Advances in Computer Graphics*, 2006, pp. 702–710.
[9] A. Selle, M. Lentine, and R. Fedkiw, "A mass spring model for hair simulation," *ACM Trans. Graph.*, vol. 27, no. 3, pp. 64:1–64:11, 2008.
[10] H. Iben, M. Meyer, L. Petrovic, O. Soares, J. Anderson, and A. Witkin, "Artistic simulation of curly hair," in *Proceedings of SCA*, 2013.
[11] C. Shells, *Cosserat theories: shells, rods and points*. Springer, 2000.
[12] D. K. Pai, "Strands: Interactive simulation of thin solids using cosserat models," *Computer Graphics Forum*, vol. 21, no. 3, pp. 347–352, 2002.
[13] F. Bertails, B. Audoly, M.-P. Cani, B. Querleux, F. Leroy, and J.-L. Lévêque, "Super-helices for predicting the dynamics of natural hair," *ACM Trans. Graph.*, vol. 25, no. 3, pp. 1180–1187, 2006.
[14] M. Bergou, M. Wardetzky, S. Robinson, B. Audoly, and E. Grinspun, "Discrete elastic rods," *ACM Trans. Graph.*, vol. 27, no. 3, pp. 63:1–63:12, 2008.
[15] R. Casati and F. Bertails-Descoubes, "Super space clothoids," *ACM Trans. Graph.*, vol. 32, no. 4, pp. 48:1–48:12, 2013.
[16] A. McAdams, A. Selle, K. Ward, E. Sifakis, and J. Teran, "Detail preserving continuum simulation of straight hair," *ACM Trans. Graph.*, vol. 28, no. 3, pp. 62:1–62:6, 2009.
[17] G. Daviet, F. Bertails-Descoubes, and L. Boissieux, "A hybrid iterative solver for robustly capturing Coulomb friction in hair dynamics," *ACM Trans. Graph.*, vol. 30, no. 6, pp. 139:1–139:12, 2011.
[18] D. M. Kaufman, R. Tamstorf, B. Smith, J.-M. Aubry, and E. Grinspun, "Adaptive nonlinearity for collisions in complex rod assemblies," *ACM Trans. Graph.*, vol. 33, no. 4, pp. 123:1–123:12, 2014.
[19] R. Bridson, R. Fedkiw, and J. Anderson, "Robust treatment of collisions, contact and friction for cloth animation," *ACM Trans. Graph.*, vol. 21, no. 3, pp. 594–603, 2002.
[20] T. Brochu, E. Edwards, and R. Bridson, "Efficient geometrically exact continuous collision detection," *ACM Trans. Graph.*, vol. 31, no. 4, pp. 96:1–96:7, 2012.
[21] M. Tang, R. Tong, Z. Wang, and D. Manocha, "Fast and exact continuous collision detection with bernstein sign classification," *ACM Trans. Graph.*, vol. 33, no. 6, pp. 186:1–186:8, 2014.
[22] E. Sifakis, S. Marino, and J. Teran, "Globally coupled collision handling using volume preserving impulses," in *Proceedings of SCA*, 2008.
[23] D. Harmon, E. Vouga, R. Tamstorf, and E. Grinspun, "Robust treatment of simultaneous collisions," *ACM Trans. Graph.*, vol. 27, no. 3, pp. 23:1–23:4, 2008.
[24] E. Guendelman, R. Bridson, and R. Fedkiw, "Nonconvex rigid bodies with stacking," *ACM Trans. Graph.*, vol. 22, no. 3, 2003.
[25] R. Bridson, S. Marino, and R. Fedkiw, "Simulation of clothing with folds and wrinkles," in *Proceedings of SCA*, 2003, pp. 28–36.
[26] S. Hadap and N. Magnenat-Thalmann, "Modeling dynamic hair as a continuum," *Computer Graphics Forum*, vol. 20, no. 3, 2001.
[27] Y. Bando, B.-Y. Chen, and T. Nishita, "Animating hair with loosely connected particles," *Computer Graphics Forum*, vol. 22, no. 3, pp. 411–418, 2003.
[28] B. Choe, M. G. Choi, and H.-S. Ko, "Simulating complex hair with robust collision handling," in *Proceedings of SCA*, 2005, pp. 153–160.
[29] S. Tariq and L. Bavoil, "Real time hair simulation and rendering on the GPU," in *ACM SIGGRAPH 2008 talks*, 2008, p. 37.
[30] P. Guan, L. Sigal, V. Reznitskaya, and J. K. Hodgins, "Multi-linear data-driven dynamic hair model with efficient hair-body collision handling," in *Proceedings of SCA*, 2012, pp. 295–304.
[31] F. Bertails, T.-Y. Kim, M.-P. Cani, and U. Neumann, "Adaptive wisp tree: A multiresolution control structure for simulating dynamic clustering in hair motion," in *Proceedings of SCA*, 2003, pp. 207–213.
[32] K. Ward and M. C. Lin, "Adaptive grouping and subdivision for simulating hair dynamics," in *Proceedings of Pacific Graphics*, 2003, pp. 234–243.
[33] M. Chai, C. Zheng, and K. Zhou, "A reduced model for interactive hairs," *ACM Trans. Graph.*, vol. 33, no. 4, pp. 124:1–124:11, 2014.
[34] D. L. James and C. D. Twigg, "Skinning mesh animations," *ACM Trans. Graph.*, vol. 24, no. 3, pp. 399–407, 2005.
[35] L. Kavan, D. Gerszewski, A. W. Bargteil, and P.-P. Sloan, "Physics-inspired upsampling for cloth simulation in games," *ACM Trans. Graph.*, vol. 30, no. 4, pp. 93:1–93:10, 2011.
[36] F. Hahn, B. Thomaszewski, S. Coros, R. W. Sumner, F. Cole, M. Meyer, T. DeRose, and M. Gross, "Subspace clothing simulation using adaptive bases," *ACM Trans. Graph.*, vol. 33, no. 4, pp. 105:1–105:9, 2014.
[37] A. Somasundaram, "Dynamically controlling hair interpolation," in *ACM SIGGRAPH 2015 Talks*, 2015, pp. 36:1–36:1.
[38] M. Müller, T.-Y. Kim, and N. Chentanez, "Fast simulation of inextensible hair and fur." *VRIPHYS*, vol. 12, pp. 39–44, 2012.
[39] J. Brown, J.-C. Latombe, and K. Montgomery, "Real-time knot-tying simulation," *The Visual Computer*, vol. 20, no. 2-3, pp. 165–179, 2004.
[40] M. Müller, B. Heidelberger, M. Hennix, and J. Ratcliff, "Position based dynamics," *Journal of Visual Communication and Image Representation*, vol. 18, no. 2, pp. 109–118, 2007.
[41] E. Balas and C. S. Yu, "Finding a maximum clique in an arbitrary graph," *SIAM Journal on Computing*, vol. 15, pp. 1054–1068, 1986.
[42] E. Sifakis, T. Shinar, G. Irving, and R. Fedkiw, "Hybrid simulation of deformable solids," in *Proceedings of SCA*, 2007, pp. 81–90.
[43] Y. Fan, J. Litven, D. I. W. Levin, and D. K. Pai, "Eulerian-on-Lagrangian simulation," *ACM Trans. Graph.*, vol. 32, no. 3, pp. 22:1–22:9, 2013.
[44] S. Osher and R. Fedkiw, *Level Set Methods and Dynamic Implicit Surfaces*. Springer, 2003.
[45] S. Hadap, "Oriented strands: Dynamics of stiff multi-body system," in *Proceedings of SCA*, 2006, pp. 91–100.
[46] K. Schittkowski, "QL: A Fortran code for convex quadratic programming-user's guide, version 2.11," *Report, Department of Mathematics, University of Bayreuth*, 2005.
[47] E. J. Candes and T. Tao, "Decoding by linear programming," *IEEE Transactions on Information Theory*, vol. 51, no. 12, 2005.
[48] S. S. Chen, D. L. Donoho, and M. A. Saunders, "Atomic decomposition by basis pursuit," *SIAM journal on scientific computing*, vol. 20, no. 1, pp. 33–61, 1998.
[49] J. Mairal, F. Bach, J. Ponce, and G. Sapiro, "Online learning for matrix factorization and sparse coding," *The Journal of Machine Learning Research*, vol. 11, pp. 19–60, 2010.

Menglei Chai received the bachelor's degree in computer science from Zhejiang University in 2011. Currently, he is working toward the PhD degree at the State Key Lab of CAD&CG, Zhejiang University. His research interests include image-based modeling and interactive image manipulation.

Changxi Zheng is an Assistant Professor in the Computer Science Department at Columbia University. Prior to joining Columbia, he received his M.S. and Ph.D. from Cornell University and his B.S. from Shanghai Jiaotong University. His research spans computer graphics, physically-based simulation, computational design, computational acoustics, scientific computing and robotics. He has been serving as an Associated Editor of ACM Transactions on Graphics, and won the NSF CAREER Award and the Cornell CS Best Dissertation award in 2012.

Kun Zhou is a Cheung Kong Professor in the Computer Science Department of Zhejiang University, and the Director of the State Key Lab of CAD&CG. Prior to joining Zhejiang University in 2008, Dr. Zhou was a Leader Researcher of the Internet Graphics Group at Microsoft Research Asia. He received his B.S. degree and Ph.D. degree in computer science from Zhejiang University in 1997 and 2002, respectively. His research interests are in visual computing, parallel computing, human computer interaction, and virtual reality. He currently serves on the editorial/advisory boards of ACM Transactions on Graphics and IEEE Spectrum. He is a Fellow of IEEE.