DOI: 10.1111/cgf.13367

EUROGRAPHICS 2018 / D. Gutierrez and A. Sheffer
(Guest Editors)

Volume 37 (2018), Number 2

Hair Modeling and Simulation by Style

Seunghwan Jung and Sung-Hee Lee

Graduate School of Culture Technology, KAIST, Republic of Korea

Abstract

As the deformation behaviors of hair strands vary greatly depending on the hairstyle, the computational cost and accuracy of hair movement simulations can be significantly improved by applying simulation methods specific to a certain style. This paper makes two contributions with regard to the simulation of various hair styles. First, we propose a novel method to reconstruct simulatable hair strands from hair meshes created by artists. Manually created hair meshes consist of numerous mesh patches, and the strand reconstruction process is challenged by the absence of connectivity information among the patches for the same strand and the omission of hidden parts of strands due to the manual creation process. To this end, we develop a two-stage spectral clustering method for estimating the degree of connectivity among patches and a strand-growing method that preserves hairstyles. Next, we develop a hairstyle classification method for style-specific simulations. In particular, we propose a set of features for efficient classifications and show that classifiers trained with the proposed features have higher accuracy than those trained with naive features. Our method applies efficient simulation methods according to the hairstyle without specific user input, and thus is favorable for real-time simulation.

CCS Concepts

•***Computing methodologies*** $\rightarrow$ *Physical simulation; Clustering and classification; Shape analysis;*

1. Introduction

Although physical simulation is an effective means of generating realistic hair movement, the high computational cost of simulating hair, which consists of more than 100K strands with countless collisions with the body and other strands, has allowed only offline simulation mainly for the movie industry [HCL*07]. While real-time hair simulations, at least for unconstrained natural hairstyles, are becoming feasible for interactive games and virtual reality, for instance [HH12], due to recent advances in hardware and simulation techniques, the rapid simulation of hair in various styles remains a challenge in the field of computer animation research.

One notable feature of hair is its variety of styles, and the style defines the distinctive deformation behavior of the hair. For example, in a braided hairstyle, strands from the scalp to the beginning of the braid do not move, whereas strands in the braided part move in unison. Therefore, it would make sense not to simulate the former part of the strands, while the latter part may be simulated well using a wisp model [BKCN03]. This type of movement pattern is completely different than those used for strands of natural unconstrained hair, for which a regular strand-based simulation model is most appropriate. As such, a reasonable strategy for rapid hair simulations would be to apply different simulation models to different strands considering their style.

To this end, we present methods to generate simulatable hair models from hair mesh models created by artists and to simulate hair with appropriate simulation methods based on its style. There are two key challenges in relation to this. First, it is difficult to reconstruct strands from hair meshes created by artists, especially for complex styles such as braids. A mesh is typically composed of a soup of mesh patches. Estimating the degree of connectivity among patches is necessary for strand reconstruction, but the omission of hidden parts of strands and the spuriousness of manual creation pose difficulties. Our solution to this problem is to develop a two-stage spectral clustering method for estimating the degree of mesh connectivity, with the initial clustering dedicated to finding the connectivity of the middle part of the hair, especially for braids, and the second clustering for the entire set of meshes. In addition, a novel strand-growing method is developed to create hair strands considering the hairstyles.

The second challenge is to recognize hairstyles automatically and apply appropriate simulation methods with proper parameters, which, to our knowledge, has not been attempted in computer animation research. To this end, by analyzing the shapes of the strands in each cluster, we classify the clusters into one of four hairstyles: normal, fixed, short, and braided hair. Specifically, we develop a set of features for the efficient classification of styles. When simulating a hair model, we apply different properties for each cluster based on the classification results.

Through simulation experiments we demonstrate that our

© 2018 The Author(s)
Computer Graphics Forum © 2018 The Eurographics Association and John Wiley & Sons Ltd. Published by John Wiley & Sons Ltd.

delivered by EUROGRAPHICS DIGITAL LIBRARY
www.eg.org diglib.eg.org

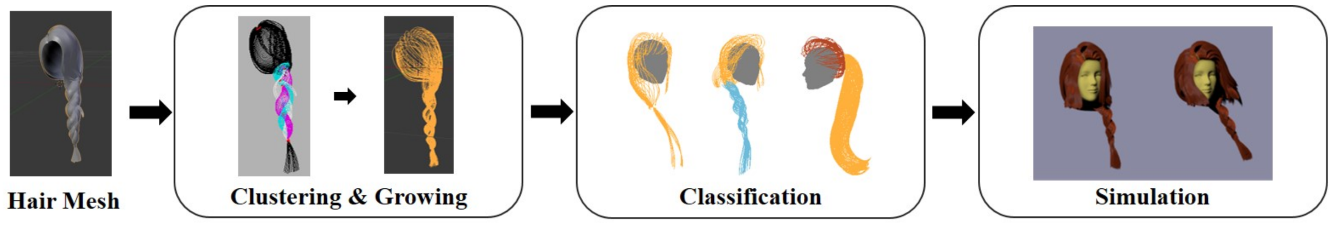

Figure 1: *Overview of our system. First, we convert triangle mesh-based hair models into strands which are clustered into those with similar shapes. Second, we classify the clusters into groups having one of four hairstyles. Finally, we apply style-specific simulation models to each cluster according to its style.*

method enables various hair simulations automatically, such as a ponytail or braids, without complicated manual intervention and that it improves the performance by reducing the computational costs.

2. Related Work

Due to its importance in modeling humans, hair has been researched intensively in the field of computer graphics with regard to modeling, simulation, and rendering [WBK*07]. This section reviews previous work on the modeling and simulating of hair.

2.1. Hair Modeling

To represent a variety of hairstyles effectively, researchers have developed many hairstyle modeling methods. Each has its own advantages, but not all of them can be used for simulations. Meshes or strips are simple and efficient and especially well supported by existing modeling software, but they are not appropriate for realistic simulations [KH00, KN00, LH03]. To reduce the modeling effort while also creating realistic hair models, vector-field-based modeling methods have been introduced [Yu01, CK05]. Recently, 3D hair-capture techniques or image-based hair-modeling methods have been explored [CWW*13,LLR13,HML*14,ZCW*17]. These methods can create highly realistic digital clones of existing hair models. The hairstyles that can be reconstructed by existing methods are mostly limited to unconstrained natural hairstyles, with the exception of one method [HML*14], which reconstructs various braided hair models using predefined braiding patterns. Our method can reconstruct braided hair models, in contrast to the aforementioned study [HML*14], without resorting to predefined patterns of braids.

Hu et al. [HMLL15] and Chai et al. [CSW*16] developed a method that converts mesh-based hair models into strand models. However, their method is limited to natural hairstyles without constraints. A procedural hair generation method developed by Yuksel et al. [YSK09] enables the generation of more complex styles such as knots and buns, from manually created hair meshes that provide clear information on the strand direction. However, hair meshes for complex styles, that are created by artists and shared through the Internet, often pose difficulty in estimating strand directions: These hair meshes may represent even a single hair strand with several disjoint strips, of which the connections may be elusive and unclear due to the manual creation process. Overcoming this limitation, our method is capable of constructing strand models of complex hairstyles from manually created mesh-based hair models.

2.2. Strand Model

Many methods have been developed for realistic hair animation. Among them, strand models for modeling individual hair strands and wisp models for modeling a chunk of hair strands allow for efficient simulations.

To simulate each hair strand, Rosenblum et al. [RCT91] proposed a mass spring model that represents a strand with a set of particles connected with springs. Due to its simplicity and computational efficiency, it is commonly used for real-time simulations but is vulnerable to stretching and is weak when used to express bending and torsion. The latter limitation can be improved by converting the line segment model into an aggregate tetrahedral structure by adding more edges between distant particles [SLF08], thus enabling the expression of bending, twists, and curls. Iben et al. [IMP*13] proposed a strand model that can control bending by adding two types of springs and thus maintain a curly hairstyle. Super helix [BAC*06] is another strand model derived from an elastic rod model based on the theories of Cosserat and Kirchhoff. Unlike a mass spring system, it can deal with the problems of bending, torsion, and stretching, but the required computation is somewhat heavy for a real-time simulation.

Articulated rigid-body models have also been used for simulating hair strands, especially a group of strands or wisps. By means of reduced-coordinate formulation, the degrees of freedom of a strand are reduced and the stretching artefacts are eliminated [Had06]. However, this approach complicates collision handling process. To alleviate this limitation, Choe et al. [CCK05] used a soft constraint method which involved the insertion of springs between rigid bodies. This method improves the simulation speed and is capable of controlling torsional motion.

2.3. Real-time Simulation

While hair simulation technique has advanced greatly to the extent to simulate interaction with other materials [FMB*17] for offline animation, compromise on quality is still inevitable for real-time simulation. Because there are in excess of tens of thousands

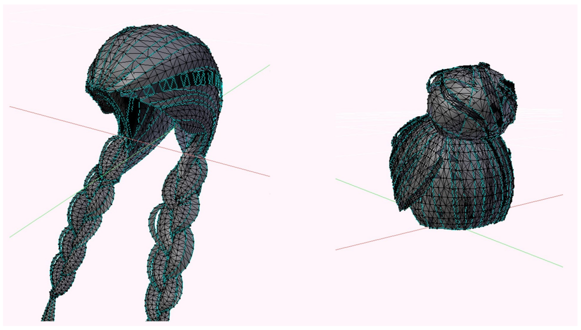

Figure 2: *Left: valid input mesh example. Right: Invalid input mesh, in which the direction of the hidden hair part is ambiguous.*

of hair strands, it is not possible to simulate all of the strands in real time. A reasonable method for reducing this level of complexity is initially to simulate only a small number of strands, called guide strands, and then to calculate the positions of the remaining strands by interpolating the guide strands [CJY02]. Chai et al. [CZZ14, CZZ17] developed a method that optimally selects guide strands that best preserves the fully-simulated hair motion. Other methods of real-time simulation include GPU acceleration [HH12, MKC12] and the example-based learning of hair movement behavior [GSRH12, HBLB17]. In contrast to the previous efforts, we take a different approach and apply a different simulation method depending on the hairstyle to improve the simulation speed.

3. Overview

Figure 1 shows an overview of our framework. Our input hair models are triangle meshes, and we initially convert them into strand-based hair models, with the strands clustered depending on their shapes (Section 4). We then examine each cluster and classify each into one of the four aforementioned hairstyles (Section 5.1). Lastly, we apply different simulation models to each cluster according to its style (Section 5.3).

Input mesh Even for identical hairstyles, the actual mesh models for the style differ significantly according to the artists. The variation usually becomes wider for complex hairstyles, such as braided hair. Our method assumes that the input mesh satisfies the following: First, the vertex indices in a hair mesh are set in increasing order from the proximal to the distal end of the hair. This eases identifying the direction of the extracted strands. Second, the hair mesh must be detailed enough so that the direction of hidden strands can be reasonably estimated from the mesh. Figure 2 shows an example of valid and invalid input meshes. The braided hair model (left) allows the estimation of the hidden hair parts because the mesh patches are arranged such that if one grows the strands from one patch, it is likely to meet the proper subsequent patch. In contrast, the omission of the large part of hair in Fig. 2 (right) makes such estimation extremely difficult, and thus it is considered an invalid input.

4. Strand Hair Generation

In order to create various hair models, Hu et al. [HMLL15] took a clever approach that collected hair mesh models from online repositories and converted them into strand models. To this end, they initially extracted strands from the edges of triangle meshes (dubbed *input strands* in our paper) and built a 3D orientation field that indicates the hair direction on a particular grid. Hair strands are then grown from the scalp using the orientation field. This method works well for uniform hairstyles but may fail to produce appropriate strands for complex hairstyles in which the meshes are overlapped and proceed in different directions or in cases where some parts of a strand are omitted. Because we deal with various hairstyles, such complexities arise often. Therefore, we developed a novel method to overcome the problems.

First, we collect hair models from online repositories and extract input strands from the meshes, as originally proposed by Hu et al. [HMLL15]. Subsequently, for disconnected input strands, we estimate the connections among the strands via the following three steps. (i) Grouping the input strands with those from the same mesh. (ii) Type identification of the groups to identify the root, middle, and tip parts (Section 4.1). (iii) Applying two-level spectral clustering with the groups to gather groups to be connected into the same cluster (Section 4.2). With the estimated connection information from the clustering step, we grow a strand using the vectors of the input strand and the 3D orientation field (Section 4.4). Our method has two advantages. It can generate hair strands even when the base hair models contain overlapped or disconnected meshes. Second, by means of spectral clustering, it can generate strands of complex hair models such as braided hair and ponytails.

4.1. Strand Type Identification

Given the input strands from meshes, we initially group the strands with those from the same mesh patch to simplify and reduce the number of nodes for clustering.

Subsequently, we employ spectral clustering [NJW01], which is widely used for image segmentation, as a basic tool for the clustering of the input strands. However, a naive application of spectral clustering for the entire set of strand groups did not perform well, especially for complex styles such as braids, as the intertwined pattern of strand streams makes it extremely difficult to separate them out when they are compared on par with other parts of hair.

Our strategy to confront this challenge is to develop two-level spectral clustering, in which the middle region in the hair is clustered first to resolve the clustering of intertwined hair groups, with this followed by the overall clustering step. This strategy necessitates the identification of the middle part of the hair. Thus, for each strand group, at the outset we identify the group type as the root, tip, or middle type according to its distance from the scalp.

The type of each strand group g^i is determined as follows:

$$Type(g^i) = \begin{cases} \text{root}, & \text{if } |R(g^i)| = 0 \\ \text{tip}, & \text{else if } |T(g^i)| = 0 \\ \text{middle}, & \text{else}, \end{cases}$$

where $R(g^i)$ and $T(g^i)$ denote other groups located close to the

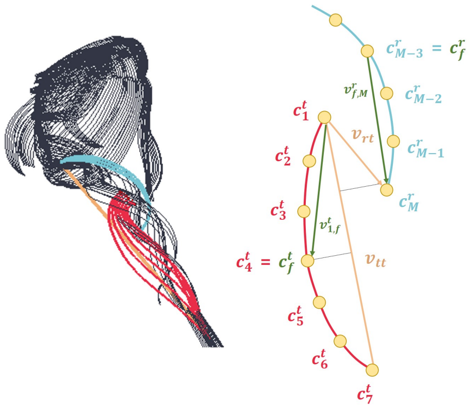

Figure 3: *To calculate the edge weights between two groups, we create center strands and compare the two center strands.*

beginning and end of group g^i, respectively. The root and tip types indicate that there is no preceding group and no subsequent group. The middle type is neither a root nor a tip (Algorithm 1).

For the root-type group, we further distinguish it into two subgroups. These are a connected-root-type group that is connected with either middle- or tip-type groups, and a free-root-type group that stands alone. The degree of connectedness of a root-type group with a nearby group is determined by the flow directions of the two groups: If the direction of the distal end of a root-type group is similar to that of the proximal end of a nearby group, it is identified as a connected-root-type group. The threshold of similarity is determined by the distribution of the angles, i.e., the direction differences, of all pairs of root groups and nearby groups.

Algorithm 1 Strand Type Identification

```
for each strand group $i$ do
    Find other groups located close to the beginning and end of $i$
for each strand group $i$ do
    if there is no group close to the beginning of $i$ then
        if there is no connectable group close to the end of $i$ then
            $Type(g^i)$ = root-free
        else
            $Type(g^i)$ = root-connected
    else if there is no group close to the end of $i$ then
        $Type(g^i)$ = tip
    else
        $Type(g^i)$ = middle
```

4.2. Middle Group Clustering

After the type identification of each strand group, we apply spectral clustering over the middle-type groups to connect the input strands naturally and to gather similar strands into the same cluster. For spectral clustering, we build a graph in which each node indicates a middle-type group and each edge is weighted by the appropriateness of the connection between two groups. We calculate the edge weights with the following assumption: Two groups will be connected (i) if they are close at the connection parts, and (ii) if their tangential directions at the connection part are similar.

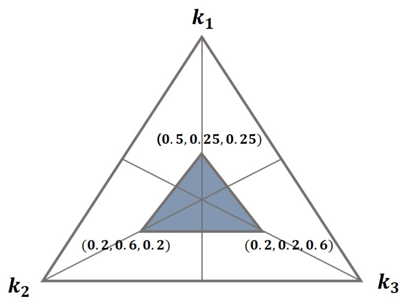

Figure 4: *The range of valid coefficients are shown in the shaded triangle.*

Before calculating the edge weights between g^i and g^j, we initially create center strands $c^i = (c^i_1, \ldots, c^i_{M^i})$, $c^j = (c^j_1, \ldots, c^j_{M^j})$ by calculating the average of the input strand for each group and then use the center strands instead of the input strands to simplify the examination. By comparing the distances from the first particle of one group to the last particle of another group, we determine the order of two groups and refer to the group located near the root side as c^r and the group located near the tip side as c^t. Next, we calculate the following three values (see Figure 3):

$$f_1 = \min_{\frac{M^r}{2} \leq m \leq M^r, 1 \leq n \leq \frac{M^t}{2}} \frac{||\mathbf{p}(c^r_m) - \mathbf{p}(c^t_n)||}{H} \tag{1}$$

$$f_2 = \frac{|\mathbf{v}_{rt} \cdot \mathbf{v}_{tt}|}{||\mathbf{v}_{tt}||^2} \tag{2}$$

$$f_3 = \frac{angle(\mathbf{v}^r_{f,M}, \mathbf{v}^t_{1,f})}{\pi} \tag{3}$$

where $\mathbf{v}_{rt} = \mathbf{p}(c^r_M) - \mathbf{p}(c^t_1)$, $\mathbf{v}_{tt} = \mathbf{p}(c^t_M) - \mathbf{p}(c^t_1)$, with $\mathbf{p}(c^p_m)$ indicating the position of particle c^p_m. The head height H serves as a normalization term. The vector $\mathbf{v}^i_{m,n}$ indicates the vector from particle c^i_m to c^i_n, and c^i_f is the farthest particle from an edge connecting c^i_1 and c^i_M. Even if two groups are close, they should not be connected if the groups overlap greatly. Therefore, we check the distance using (1), and the overlap ratio using (2). Equations (1) and (2) are used for assumption (i), and (3) is for assumption (ii).

The edge weights w_{ij} are determined as a weighted sum of f_i values using coefficient k_i. In our experiment, we used the following formula

$$w_{ij} = \begin{cases} k_1 f_1 + k_2 f_2 + k_3 f_3 \text{ if } f_1 < \frac{H}{4}, f_2 < 0.5, f_3 < 0.3 \\ \infty \text{ otherwise,} \end{cases}$$

where $\sum_i k_i = 1$ and the bounds for f_i define the valid region of each criterion. In addition, we found that adding a heuristic emphasizing the local similarity improves the clustering quality by scaling the weight w_{ij} to half between g^i and g^j if w_{ij} is the minimum among the values of w_{ik} where g^k denotes the tip side groups of g^i. Figure 4 shows the approximate range of the coefficients k_i for the edge

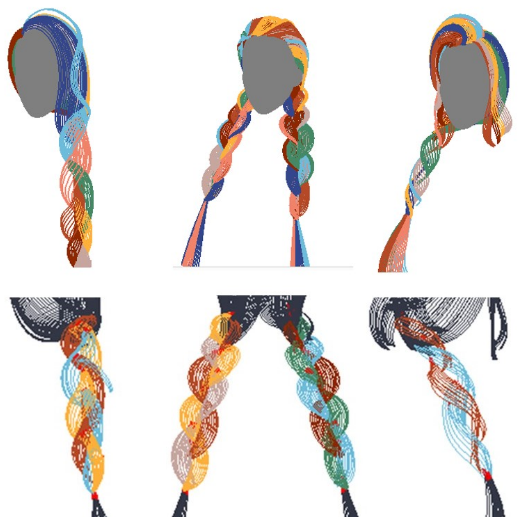
Figure 5: *Top: Strand groups. Bottom: Results after middle group clustering.*

weights that succeeded in clustering for the example shown in Figure 5 (middle).

Figure 5 shows the clustering results of the middle groups. We set the desired number of clusters for each hair model for spectral clustering. Each cluster obtained in this stage is regarded as a single node in the next stage.

Algorithm 2 Clustering

```
if Middle groups exist then
    for each middle type group $i$ do
        for each middle type group $j$ do
            Calculate the connectivity edge weights
    Do spectral clustering for middle groups
for each strand group $i$ do
    for each strand group $j$ do
        if Both groups are root type then
            Calculate the similarity edge weight
        else
            Calculate the connectivity edge weight
Do spectral clustering
```

4.3. Hair Clustering

This stage performs clustering over every strand group of the root and tip types and every cluster of the middle type. We divide groups and clusters into two sets, S_1 and S_2, and apply different schemes for determining edge weights. Free-root groups belong to S_1 and all other groups belong to S_2. Edge weights between groups in S_1 are based on the similarity with respect to the distance and direction between two groups, defined as follows:

$$||\mathbf{p}(c_1^i) - \mathbf{p}(c_1^j)|| \quad (4)$$

$$||\mathbf{p}(c_M^i) - \mathbf{p}(c_M^j)|| \quad (5)$$

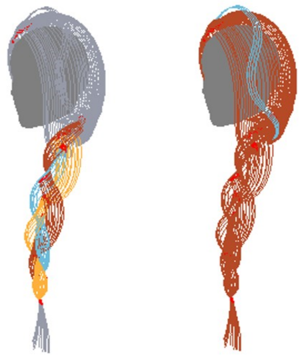
Figure 6: *Left: Middle group clustering result. Right: Hair clustering result.*

Edge weights between elements in S_2 are based on the connectivity of the two elements, by using (1), (2), and (3). Edge weights between the S_1 and S_2 groups are set to infinity. Algorithm 2 summarizes the two-level clustering method. Figure 6 shows the case how the strands in a braided hair style are clustered through the middle group clustering and final clustering.

After clustering, we divide clusters at positions that change from the root type to another type as the proximal and distal parts of the positions may have different styles.

4.4. Hair Growing

With input strands and their cluster information, we grow hair strands starting from the scalp. For this, we use two types of growing strategies. One is to grow the strands using the orientation fields constructed by the input strands as has been done in [CSW*16]. This method effectively grows strands even when the input strands are sparse because the orientation field interpolates the strand directions into the region where input strands do not exist. In contrast to [CSW*16] that constructs only one orientation field, we build separate field per cluster. This prevents that hair directions are averaged out in the region two different clusters meet. We also found that the orientation field-based method is not very effective for the middle and tip-type groups as it has a tendency to extrapolate the hair directions outside where a cluster is non-existent. Therefore, for these types of input strands, we grow the strands in the direction of the closest input strand. When the strand reaches the end of the input strand, a search is attempted to find a new input strand from the same cluster and then from other clusters. Figure 7 shows the process of creating a strand-based hair model from a mesh-based hair model.

5. Style-Specific Hair Simulation

With strand based hair models where the strands are clustered and gathered depending on the degree of similarity, we apply style-specific simulation methods to each cluster. For this, each cluster is first classified into one of four hairstyles: normal, fixed, short, and braided. A cluster of hair that should be static with respect

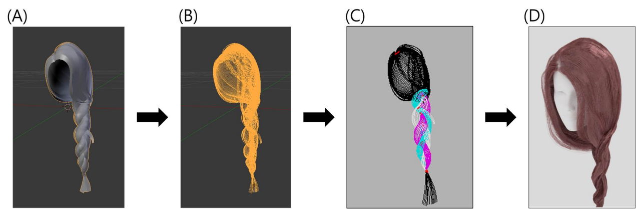

Figure 7: *(A) A mesh-based hair model. (B) Extracted input strands. (C) Clustering of the input strands. (D) Grown strands.*

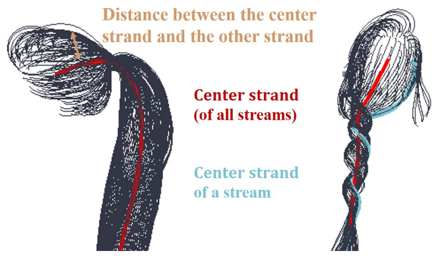
Distance between the center strand and the other strand
Center strand (of all streams)
Center strand of a stream

Figure 8: *Left: We determine whether strands are gathered by calculating distances between the center strand and the other strands. The center strand is colored red. Right: To determine whether strands are twisted around the center strand, we calculate distance between the center strands of each stream and the center strand of all streams. The former is colored blue and the latter is colored red.*

to the head is classified as the fixed style. The short style is assigned to short hair clusters that should exhibit stiffer deformation behavior than normal hair. Braided parts of hair are identified as the braided style, and these are simulated with a wisp model. A cluster that does not belong to any of these three styles is classified as the normal style. Normal and short hairstyles are simulated with the mass-spring model.

5.1. Classification

We train a support vector machine and the random forest method to classify hairstyles. To obtain accurate classifiers, we develop a set of features for each style as explained below.

First, two features are used to determine a short hairstyle: the average length and the maximum length of all hair strands in a cluster.

Second, a cluster is determined as a fixed hairstyle if 1) the starting particles of the cluster are root particles, 2) the strands gather at the end, and 3) all parts of the strands are close to the scalp. To examine these conditions, we initially designate the center strand c as the average of strands $s = (s^1, ..., s^S)$ in the cluster where S is the number of strands, and then compute the following two features. The first measures the second condition by comparing the average distance between a strand and the center strand at the terminal position (indexed as M) and at its maximum position (Fig. 8).

$$\frac{d_M}{\max_j d_j} \text{ where } d_j = \frac{1}{S}\sum_i ||\mathbf{p}(s^i_j) - \mathbf{p}(c_j)||. \tag{6}$$

This feature value becomes smaller as the strands gather more at the end. The second feature is the distance from the farthest particle to the scalp.

Third, the classification of the braided hairstyle is performed against clusters of middle type groups. We check whether the cluster consists of more than one node (i.e., a cluster after middle-group clustering) and whether they are braided. To examine this, the center strand c^i of each node and the center strand c^{total} of all nodes are then computed, after which the distances from the c^i to c^{total} are measured. The feature value for this is

$$\max_{i,j} ||\mathbf{p}(c^i_j) - \mathbf{p}(c^{total}_j)|| \tag{7}$$

Training From a total of 159 clusters from 46 hair models, we collect a dataset of the above features and train classifiers, one with the support vector machine with the pearson VII function-based universal kernel [B06] and the other with a random forest algorithm with 100 trees. With ten-fold cross validation, the accuracy levels of the trained classifiers are 84.8% and 86.7%, respectively. The training data that we used for our experiment can be downloaded from https://github.com/data1087/HairData/blob/master/ConvertedObj.zip

5.2. Simulation

For the simulation, we use the altitude mass-spring model proposed by Selle et al. [SLF08]. To overcome the limitation of the conventional mass-spring model, which cannot control bending and torsion, the altitude mass-spring model constructs tetrahedral structures using additional particles and edges. Hair-body collisions are handled based on the method proposed by Bridson et al. [BMF03], which uses a signed distance field (SDF). Specifically, distance of each particle to the near-body surface is calculated using the SDF, and in the event of a collision with the body, it is repositioned toward the closest surface of the body. Hair-hair interactions are handled by the method of Muller et al. [MKC12], which uses a density field that applies repulsive force to each particle to a lower density direction. For a rapid simulation, we simulate only a subset of strands, referred to as guide strands, and the deformation of the remaining strands is achieved by interpolating the guide strands.

Depending on the classification results, we apply certain properties to each cluster. Fixed hair clusters are excluded from the simulation because they are fixed to the head. Short hair clusters are simulated with the method used for normal hair clusters, but with fewer particles per length than normal hair in order to make the result stiffer than normal hair so as to maintain the original hair style.

For a braided hair cluster, we use the wisp model in which only a center strand is physically simulated and the shapes of actual strands are determined kinematically from the center strand. To do this, in the default pose, each particle's relative position with respect to a coordinate frame defined on the closest tetrahedron of a center strand is stored, and its global position is updated as if the particle is fixed to the reference coordinate frame of the tetrahedron, which is dynamically simulated. The center strand is connected to preceding strands by adding edges between the start of

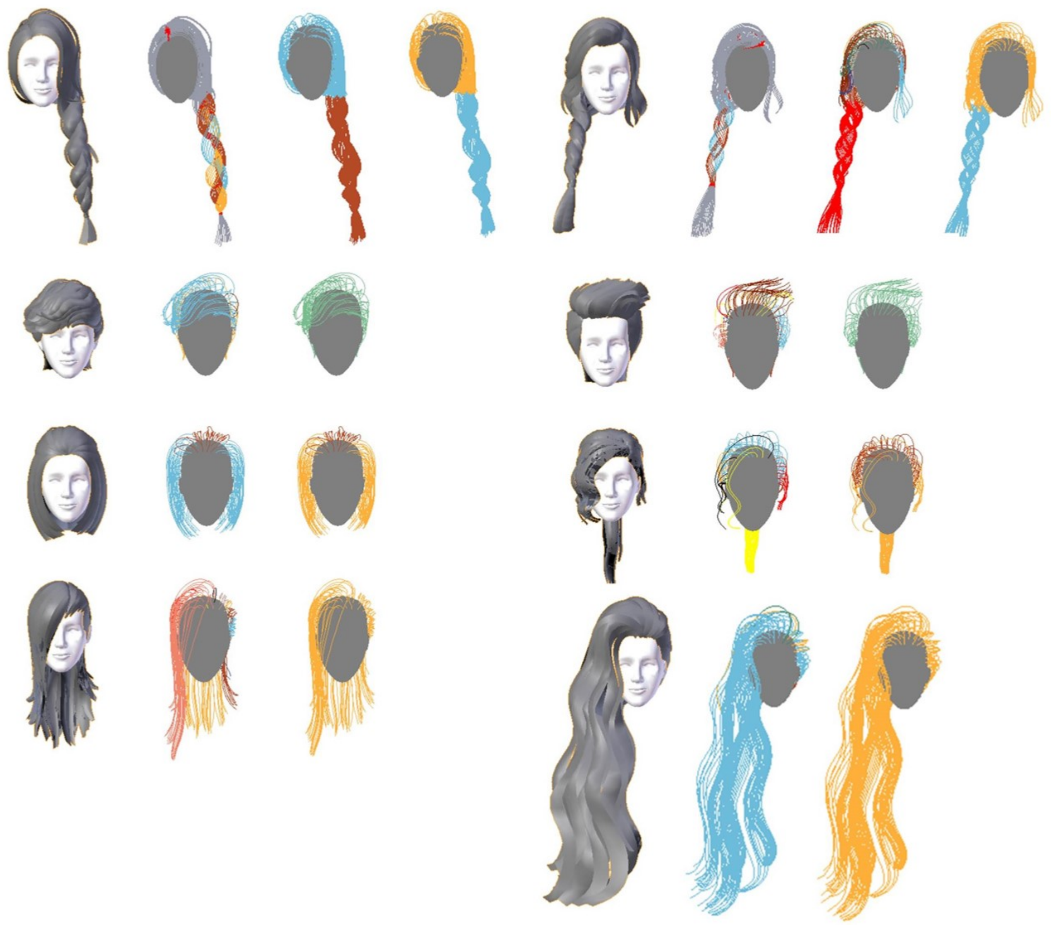

Figure 9: *Input meshes (left), clustered strands (middle), Classification results (right). Classified strands are colored as follows. Brown: fixed hair. Blue: braided hair. Green: short hair. Yellow: normal hair. In the case of braided hairs (top row), middle group clustering results are shown next to the input meshes.*

the center strand and the end of the preceding strands, and the mass of each particle is scaled according to the number of preceding strands.

Interpolation The shapes of the interpolated strands are updated by interpolating the guide strands. To accomplish this, we use both barycentric interpolation, which constructs each interpolated strand using three nearby guide strands, and clump interpolation, where an interpolated strand follows only one guide strand. Barycentric interpolation is used when the three nearby strands belong to the same cluster and when their directions are similar. Otherwise, the clump interpolation method is used.

6. Experiments

This section shows the results of our strand reconstruction method and the performance of our hairstyle classifiers. Our method can convert various styles of hair mesh models into strand models and then perform a dynamic simulation. Figure 9 shows the input mesh, its strand reconstruction, and the classification results.

6.1. Classification Performance

Our classification features are selected somewhat heuristically to reflect the characteristics of each hairstyle. To examine the effectiveness of the features, we compare them with other features that use basic geometric and statistical information computed from the input hair model, specifically the mean and variance of the positions of the strand particles, the number of strands, the mean and variance of the distances from each particle to the skin, the mean and variance of the distances from each particle to the average strand, the mean of edge vectors, the mean of the direction differences between adjacent edges, and the position of the average strand, all of which constitute 190-dimensional vectors. To unify the dimension of the feature vectors, particles of each strand are resampled to obtain $N-1$ particles from N equidistant intervals.

We trained a support vector machine and a random forest classifier with these raw features against the same hair database, and obtained 79.2% accuracy for the support vector machine and 72.4% accuracy for the random forest. The classification result using the original features shows at least 5% greater accuracy than the raw

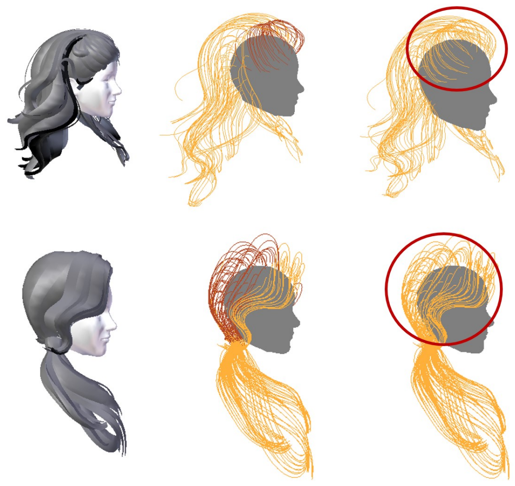

Figure 10: *Left: Hair meshes. Middle: Ideal classification that identifies fixed hair parts (brown) due to hair bands. Right: Classification failure cases. All hairs are classified as the normal type.*

features, which suggests that our features have reasonably good discriminative power.

Failure cases Sometimes our classification method results in sub-optimal results. Most of the failures occur in the normal and fixed types. Figure 10 shows the cases that our method did not correctly classify the styles. The end of strands in Fig. 10 (top) are not gathered close enough, and some strands in Fig. 10 (bottom) are located too far from the scalp to be classified as the fixed type. As a result, they were classified as the normal type.

6.2. Simulation Performance

Figure 11 shows that our method generates reasonable hair simulation for a number of styles. Each hair model consists of approximately 10k strands, including 169 guide strands. The supplementary video shows the quality of modeling and simulation. As can be seen in the video, in the example of Fig. 11 (bottom left), the strands spread somewhat excessively at the junction of the fixed and normal styles. This particular model was challenging because there was a large gap between the fixed hair part and the ponytail part and between the two parts existed a hair band. When hair strands were grown, they followed the orientation field of the fixed hair part and then the direction of the input strands of the ponytail part. During the transition, the strands spread to some degree because of the large gap.

We tested simulation with a computer with an Intel I7 950 processor, 4GB of memory, and an NVidia GeForce GTS 450 graphics card. The simulation required 20-30 ms per frame, with 15-20 ms for the guide strand simulation and 5-10 ms for interpolation. We made use of multi-threaded parallel computing, without GPU acceleration for the simulation. We used an offline hair rendering tool of Blender for creating the animation shown in Fig. 11.

7. Discussion and Future Work

This paper presented modeling and simulation methods for realizing hair simulations depending on the hairstyle. Starting with a hair mesh model, we reconstructed a strand hair model by finding the degree of connectivity information among mesh patches through spectral clustering and by growing strands while preserving the style of the clusters. The style of each hair cluster is then classified and appropriate simulation methods with proper parameters are identified. Our hair simulation shows realistic hair movement performed at interactive computational speed.

There are a number of limitations in our method that can be improved by further research. First, more hairstyles need be included in our framework. The four styles considered here are the most common styles, yet the set is not complete. Including other styles such as natural African hair or even a bun remains as interesting future work. This extension of the number of styles will naturally necessitate a further improvement of the hair classification and simulation methods.

Second, the simulation method for each style can be improved. We modeled a braid with a wisp model, but the true deformation characteristics of braids are more complex, with irregular and non-isotropic material properties which depend on how the strands are braided. Modeling more accurate physical properties will allow for more realistic hair simulations. The hair interpolation method for the normal style based on barycentric interpolation has a limitation when used to generate natural-looking hair. Therefore this can be improved as well.

Acknowledgement

This work was supported by Giga Korea Project (GK17P0200) and Basic Science Research Program (NRF-2017R1A2B2006160) funded by MSIT, Korea.

References

[BAC*06] BERTAILS F., AUDOLY B., CANI M.-P., QUERLEUX B., LEROY F., LÉVÊQUE J.-L.: Super-helices for predicting the dynamics of natural hair. *ACM Trans. Graph. 25*, 3 (July 2006), 1180–1187. 2

[BKCN03] BERTAILS F., KIM T.-Y., CANI M.-P., NEUMANN U.: Adaptive Wisp Tree - a multiresolution control structure for simulating dynamic clustering in hair motion. In *Proc. Symposium on Computer Animation* (2003), The Eurographics Association. 1

[BMF03] BRIDSON R., MARINO S., FEDKIW R.: Simulation of clothing with folds and wrinkles. In *Proc. Symposium on Computer Animation* (2003), SCA '03, Eurographics Association, pp. 28–36. 6

[CCK05] CHOE B., CHOI M. G., KO H.-S.: Simulating complex hair with robust collision handling. In *Proc. Symposium on Computer Animation* (2005), SCA '05, ACM, pp. 153–160. 2

[CJY02] CHANG J. T., JIN J., YU Y.: A practical model for hair mutual interactions. In *Proc. Symposium on Computer Animation* (2002), SCA '02, ACM, pp. 73–80. 3

[CK05] CHOE B., KO H.-S.: A statistical wisp model and pseudophysical approaches for interactive hairstyle generation. *IEEE Transactions on Visualization and Computer Graphics 11*, 2 (Mar. 2005), 160–170. 2

[CSW*16] CHAI M., SHAO T., WU H., WENG Y., ZHOU K.: Autohair: Fully automatic hair modeling from a single image. *ACM Trans. Graph. 35*, 4 (July 2016), 116:1–116:12. 2, 5

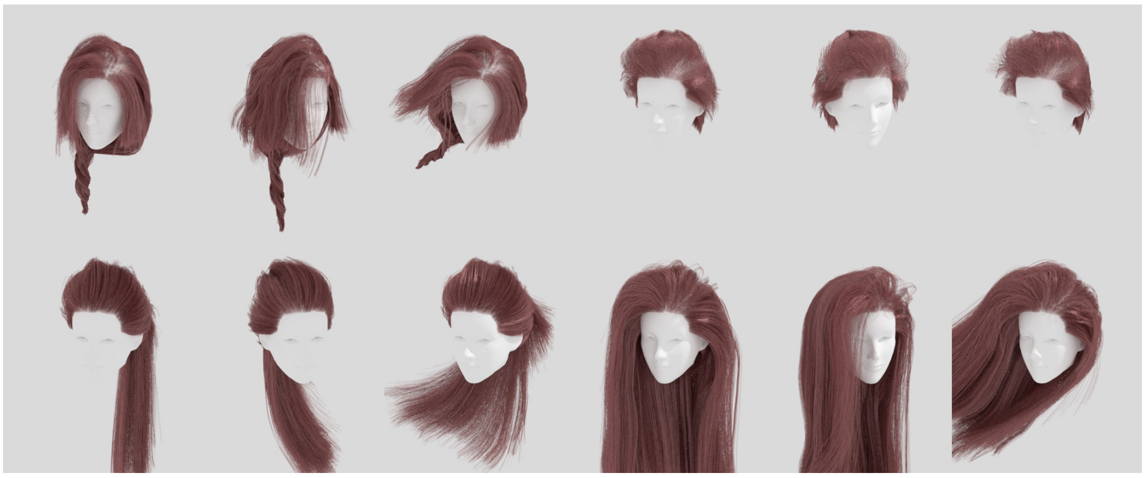

Figure 11: *Simulation results using our method. Top left: Mixture of the normal and braided styles. Top right: Short hairstyle. Bottom left: the ponytail style is simulated as a combination of the fixed and normal hairstyles. Bottom right: Normal hairstyle.*

[CWW*13] CHAI M., WANG L., WENG Y., JIN X., ZHOU K.: Dynamic hair manipulation in images and videos. *ACM Trans. Graph. 32*, 4 (July 2013), 75:1–75:8. 2

[CZZ14] CHAI M., ZHENG C., ZHOU K.: A reduced model for interactive hairs. *ACM Trans. Graph. 33*, 4 (July 2014), 124:1–124:11. 3

[CZZ17] CHAI M., ZHENG C., ZHOU K.: Adaptive skinning for interactive hair-solid simulation. *IEEE Transactions on Visualization and Computer Graphics 23*, 7 (July 2017), 1725–1738. 3

[FMB*17] FEI Y. R., MAIA H. T., BATTY C., ZHENG C., GRINSPUN E.: A multi-scale model for simulating liquid-hair interactions. *ACM Trans. Graph. 36*, 4 (July 2017), 56:1–56:17. 2

[GSRH12] GUAN P., SIGAL L., REZNITSKAYA V., HODGINS J. K.: Multi-linear data-driven dynamic hair model with efficient hair-body collision handling. In *Proc. Symposium on Computer Animation* (2012), SCA '12, Eurographics Association, pp. 295–304. 3

[Had06] HADAP S.: Oriented strands: Dynamics of stiff multi-body system. In *Proc. Symposium on Computer Animation* (2006), SCA '06, Eurographics Association, pp. 91–100. 2

[HBLB17] HU L., BRADLEY D., LI H., BEELER T.: Simulation-ready hair capture. *Computer Graphics Forum 36*, 2 (May 2017), 281–294. 3

[HCL*07] HADAP S., CANI M.-P., LIN M., KIM T.-Y., BERTAILS F., MARSCHNER S., WARD K., KAČIĆ-ALESIĆ Z.: Strands and hair: Modeling, animation, and rendering. In *ACM SIGGRAPH 2007 Courses* (2007), SIGGRAPH '07, ACM, pp. 1–150. 1

[HH12] HAN D., HARADA T.: Real-time Hair Simulation with Efficient Hair Style Preservation. In *Workshop on Virtual Reality Interaction and Physical Simulation* (2012), The Eurographics Association. 1, 3

[HML*14] HU L., MA C., LUO L., WEI L.-Y., LI H.: Capturing braided hairstyles. *ACM Trans. Graph. 33*, 6 (Nov. 2014), 225:1–225:9. 2

[HMLL15] HU L., MA C., LUO L., LI H.: Single-view hair modeling using a hairstyle database. *ACM Trans. Graph. 34*, 4 (July 2015), 125:1–125:9. 2, 3

[IMP*13] IBEN H., MEYER M., PETROVIC L., SOARES O., ANDERSON J., WITKIN A.: Artistic simulation of curly hair. In *Proc. Symposium on Computer Animation* (2013), SCA '13, ACM, pp. 63–71. 2

[KH00] KOH C. K., HUANG Z.: *Real-Time Animation of Human Hair Modeled in Strips*. Springer Vienna, 2000. 2

[KN00] KIM T.-Y., NEUMANN U.: A thin shell volume for modeling human hair. In *Proceedings Computer Animation 2000* (2000), pp. 104–111. 2

[LH03] LIANG W., HUANG Z.: An enhanced framework for real-time hair animation. In *Proceedings of the 11th Pacific Conference on Computer Graphics and Applications* (2003), PG '03, IEEE Computer Society, pp. 467–. 2

[LLR13] LUO L., LI H., RUSINKIEWICZ S.: Structure-aware hair capture. *ACM Trans. Graph. 32*, 4 (July 2013), 76:1–76:12. 2

[MKC12] MÜLLER M., KIM T.-Y., CHENTANEZ N.: Fast Simulation of Inextensible Hair and Fur. In *Workshop on Virtual Reality Interaction and Physical Simulation* (2012), The Eurographics Association. 3, 6

[NJW01] NG A. Y., JORDAN M. I., WEISS Y.: On spectral clustering: Analysis and an algorithm. In *Proceedings of the 14th International Conference on Neural Information Processing Systems: Natural and Synthetic* (2001), NIPS'01, MIT Press, pp. 849–856. 3

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

[SLF08] SELLE A., LENTINE M., FEDKIW R.: A mass spring model for hair simulation. *ACM Trans. Graph. 27*, 3 (Aug. 2008), 64:1–64:11. 2, 6

[WBK*07] WARD K., BERTAILS F., KIM T.-Y., MARSCHNER S. R., CANI M.-P., LIN M. C.: A survey on hair modeling: Styling, simulation, and rendering. *IEEE Transactions on Visualization and Computer Graphics 13*, 2 (2007), 213–234. 2

[YSK09] YUKSEL C., SCHAEFER S., KEYSER J.: Hair meshes. *ACM Trans. Graph. 28*, 5 (Dec. 2009), 166:1–166:7. 2

[Yu01] YU Y.: Modeling realistic virtual hairstyles. In *Proceedings Ninth Pacific Conference on Computer Graphics and Applications. Pacific Graphics 2001* (2001), pp. 295–304. 2

[ZCW*17] ZHANG M., CHAI M., WU H., YANG H., ZHOU K.: A data-driven approach to four-view image-based hair modeling. *ACM Trans. Graph. 36*, 4 (July 2017), 156:1–156:11. 2

[B06] ÜSTÜN B., MELSSEN W., BUYDENS L.: Facilitating the application of support vector regression by using a universal pearson vii function based kernel. *Chemometrics and Intelligent Laboratory Systems 81*, 1 (2006), 29 – 40. 6