HairNet: Single-View Hair Reconstruction using Convolutional Neural Networks

Yi Zhou[1], Liwen Hu[1], Jun Xing[2], Weikai Chen[2], Han-Wei Kung[3], Xin Tong[4], and Hao Li[1,2,3]

[1] University of Southern California
zhou859@usc.edu
huliwenkidkid@gmail.com
[2] USC Institute for Creative Technologies
{junxnui,chenwk891}@gmail.com
[3] Pinscreen, Santa Monica USA
hanweikung@gmail.com
hao@hao-li.com
[4] Microsoft Research Asia, Beijing, China
xtong@microsoft.com

Abstract. We introduce a deep learning-based method to generate full 3D hair geometry from an unconstrained image. Our method can recover local strand details and has real-time performance. State-of-the-art hair modeling techniques rely on large hairstyle collections for nearest neighbor retrieval and then perform ad-hoc refinement. Our deep learning approach, in contrast, is highly efficient in storage and can run 1000 times faster while generating hair with 30K strands. The convolutional neural network takes the 2D orientation field of a hair image as input and generates strand features that are evenly distributed on the parameterized 2D scalp. We introduce a collision loss to synthesize more plausible hairstyles, and the visibility of each strand is also used as a weight term to improve the reconstruction accuracy. The encoder-decoder architecture of our network naturally provides a compact and continuous representation for hairstyles, which allows us to interpolate naturally between hairstyles. We use a large set of rendered synthetic hair models to train our network. Our method scales to real images because an intermediate 2D orientation field, automatically calculated from the real image, factors out the difference between synthetic and real hairs. We demonstrate the effectiveness and robustness of our method on a wide range of challenging real Internet pictures, and show reconstructed hair sequences from videos.

Keywords: Hair, Reconstruction, Real-time, DNN

1 Introduction

Realistic hair modeling is one of the most difficult tasks when digitizing virtual humans [3, 20, 25, 27, 14]. In contrast to objects that are easily parameterizable,

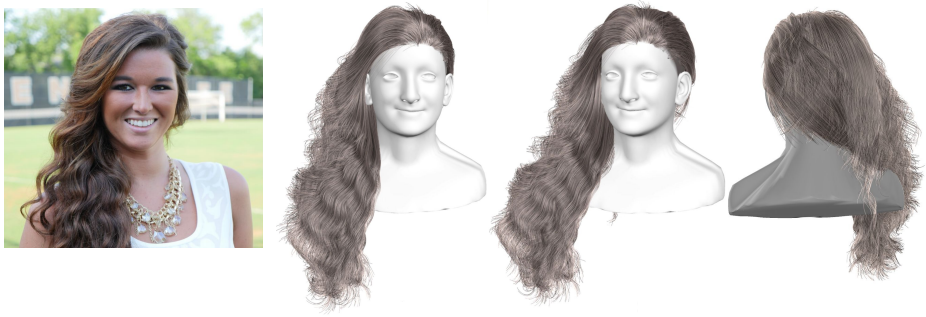

Fig. 1. Hair reconstruction from a single view image using HairNet.

like the human face, hair spans a wide range of shape variations and can be highly complex due to its volumetric structure and level of deformability in each strand. Although [28, 22, 2, 26, 38] can create high-quality 3D hair models, but they require specialized hardware setups that are difficult to be deployed and populated. Chai et al. [5, 6] introduced the first simple hair modeling technique from a single image, but the process requires manual input and cannot properly generate non-visible parts of the hair. Hu et al. [18] later addressed this problem by introducing a data-driven approach, but some user strokes were still required. More recently, Chai et al. [4] adopted a convolutional neural network to segment the hair in the input image to fully automate the modeling process, and [41] proposed a four-view approach for more flexible control.

However, these data-driven techniques rely on storing and querying a huge hair model dataset and performing computationally-heavy refinement steps. Thus, they are not feasible for applications that require real-time performance or have limited hard disk and memory space. More importantly, these methods reconstruct the target hairstyle by fitting the retrieved hair models to the input image, which may capture the main hair shape well, but cannot handle the details nor achieve high accuracy. Moreover, since both the query and refinement of hair models are based on an undirected 2D orientation match, where a horizontal orientation tensor can either direct to the right or the left, this method may sometimes produce hair with incorrect growing direction or parting lines and weird deformations in the z-axis.

To speed up the procedure and reconstruct hairs that preserve better style w.r.t the input image and look more natural, we propose a deep learning based approach to generate the full hair geometry from a single-view image, as shown in Figure 1. Different from recent advances that synthesize shapes in the form of volumetric grids [8] or point clouds [10] via neural networks, our method generates the hair strands directly, which are more suitable for non-manifold structures like hair and could achieve much higher details and precision.

Our neural network, which we call HairNet, is composed of a convolutional encoder that extracts the high-level hair-feature vector from the 2D orientation field of a hair image, and a deconvolutional decoder that generates 32×32 strand-

features evenly distributed on the parameterized 2D scalp. The hair strand-features could be interpolated on the scalp space to get higher (30K) resolution and further decoded to the final strands, represented as sequences of 3D points. In particular, the hair-feature vector can be seen as a compact and continuous representation of the hair model, which enables us to sample or interpolate more plausible hairstyles efficiently in the latent space. In addition to the reconstruction loss, we also introduce a collision loss between the hair strands and a body model to push the generated hairstyles towards a more plausible space. To further improve the accuracy, we uses the visibility of each strand based on the input image as a weight to modulate its loss.

Obtaining a training set with real hair images and ground-truth 3D hair geometries is challenging. We can factor out the difference between synthetic and real hair data by using an intermediate 2D orientation field as network input. This enables our network to be trained with largely accessible synthetic hair models and also real images without any changes. For example, the 2D orientation field can be calculated from a real image by applying a Gabor filter on the hair region automatically segmented using the method of [42]. Specifically, we synthesized a hair data set composed of 40K different hairstyles and 160K corresponding 2D orientation images rendered from random views for training.

Compared to previous data-driven methods that could take minutes and terabytes of disk storage for a single reconstruction, our method only takes less than 1 second and 70 MB disk storage in total. We demonstrate the effectiveness and robustness of our method on both synthetic hair images and real images from the Internet, and show applications in hair interpolation and video tracking.

Our contributions can be summarized as follows:

1. We propose the first deep neural network to generate dense hair geometry from a single-view image. To the best of our knowledge, it is also the first work to incorporate both collision and visibility in a deep neural network to deal with 3D geometries.
2. Our approach achieves state-of-the-art resolution and quality, and significantly outperforms existing data-driven methods in both speed and storage.
3. Our network provides the first compact and continuous representation of hair geometry, from which different hairstyles can be smoothly sampled and interpolated.
4. We construct a large-scale database of around 40K 3D hair models and 160K corresponding rendered images.

2 Related Work

Hair Digitization. A general survey of existing hair modeling techniques can be found in Ward et.al [36]. For experienced artists, purely manual editing from scratch with commercial softwares such as XGen and Hairfarm is chosen for highest quality, flexibility and controllability, but the modeling of compelling and realistic hairstyles can easily take several weeks. To avoid tedious manipulations

on individual hair fibers, some efficient design tools are proposed in [7, 23, 11, 40, 37].

Meanwhile, hair capturing methods have been introduced to acquire hairstyle data from the real world. Most hair capturing methods typically rely on high-fidelity acquisition systems, controlled recording sessions, manual assistance such as multi-view stereo cameras[28, 2, 22, 26, 9, 38, 17], single RGB-D camera [19] or thermal imaging [16].

More recently, Single-view hair digitization methods have been proposed by Chai et.al [6, 5] but can only roughly produce the frontal geometry of the hair. Hu et.al [18] later demonstrated the first system that can model entire hairstyles at the strand level using a database-driven reconstruction technique with minimal user interactions from a single input image. A follow-up automatic method has been later proposed by [4], which uses a deep neural network for hair segmentation and augments a larger database for shape retrieval. To allow more flexible control of side and back views of the hairstyle, Zhang et.al [41] proposed a four-view image-based hair modeling method to fill the gap between multi-view and single-view hair capturing techniques. Since these methods rely on a large dataset for matching, speed is an issue and the final results depend highly on the database quality and diversity.

Single-View Reconstruction using Deep Learning. Generation of 3D data by deep neural networks has been attracting increasing attention recently. Volumetric CNNs [8, 12, 33, 21] use 3D convolutional neural networks to generate voxelized shapes but are highly constrained by the volume resolution and computation cost of 3D convolution. Although techniques such as hierarchical reconstruction [15] and octree [31, 32, 35] could be used to improve the resolution, generating details like hair strands are still extremely challenging.

On the other hand, point clouds scale well to high resolution due to their unstructured representation. [29, 30] proposed unified frameworks to learn features from point clouds for tasks like 3D object classification and segmentation, but not generation. Following the pioneering work of PointNet, [13] proposed the PCPNet to estimate the local normal and curvature from point sets, and [10] proposed a network for point set generation from a single image. However, point clouds still exhibit coarse structure and are not able to capture the topological structure of hair strands.

3 Method

The entire pipeline contains three steps. A preprocessing step is first adopted to calculate the 2D orientation field of the hair region based on the automatically estimated hair mask. Then, HairNet takes the 2D orientation fields as input and generates hair strands represented as sequences of 3D points. A reconstruction step is finally performed to efficiently generate a smooth and dense hair model.

3.1 Preprocessing

We first adopt PSPNet [42] to produce an accurate and robust pixel-wise hair mask of the input portrait image, followed by computing the undirected 2D orientation for each pixel of the hair region using a Gabor filter [26]. The use of undirected orientation eliminates the need of estimating the hair growth direction, which otherwise requires extra manual labeling [18] or learning [4]. However, the hair alone could be ambiguous due to the lack of camera view information and its scale and position with respect to the human body. Thus we also add the segmentation mask of the human head and body on the input image. In particular, the human head is obtained by fitting a 3D morphable head model to the face [20] and the body could be positioned accordingly via rigid transformation. All these processes could be automated and run in real-time. The final output is a $3 \times 256 \times 256$ image, whose first two channels store the color-coded hair orientation and third channel indicates the segmentation of hair, body and background.

3.2 Data Generation

Similar to Hu et. al [18], we first collect an original hair dataset with 340 3D hair models from public online repositories [1], align them to the same reference head, convert the mesh into hair strands and solve the collision between the hair and the body. We then populate the original hair set via mirroring and pair-wise blending.

Different from AutoHair [4] which simply uses volume boundaries to avoid unnatural combinations, we separate the hairs into 12 classes based on styles shown in table 1 and blend each pair of hairstyles within the same class to generate more natural examples. In particular, we cluster the strands of each hair into five central strands, and each pair of hairstyles can generate $2^5 - 2$ additional combinations of central strands. The new central strands serve as a guidance to blend the detailed hairs. Instead of using all of the combinations, we randomly select the combination of them for each hair pair, leading to a total number over 40K hairs for our synthetic hair dataset.

XS_s	20	S_s	110	M_s	28	L_s	29	XL_s	27	XXL_s	4
XS_c	0	S_c	19	M_c	65	L_c	27	XL_c	23	XXL_c	1

Table 1. Hair classes and the number of hairs in each class. S refers to short, M refers to medium, L refers to long, X refers to very, s refers to straight and c refers to curly. Some hairs are assigned to multiple classes if its style is ambiguous.

In order to get the corresponding orientation images of each hair model, we randomly rotate and translate hair inside the view port of a fixed camera and render 4 orientation images at different views. The rotation ranges from -90° to +90° for the yaw axis and -15° to +15° for the pitch and roll axis. We also add Gaussian noises to the orientation to emulate the real conditions.

3.3 Hair Prediction Network

Hair Representation. We represent each strand as an ordered 3D point set $\zeta = \{s_i\}_{i=0}^{M}$, evenly sampled with a fixed number ($M = 100$ in our experiments) of points from the root to end. Each sample s_i contains attributes of position $\mathbf{p}_i$ and curvature c_i. Although the strands have large variance in length, curliness, and shape, they all grow from fixed roots to flexible ends. To remove the variance caused by root positions, we represent each strand in the local coordinate anchored at its root.

The hair model can be treated as a set of N strands $H = \zeta^N$ with fixed roots, and can be formulated as a matrix A_{N*M}, where each entry $A_{i,j} = (\mathbf{p}_{i,j}, c_{i,j})$ represents the jth sample point on the ith strand. In particular, we adopt the method in [34] to parameterize the scalp to a 32×32 grid, and sample hair roots at those grid centers ($N = 1024$).

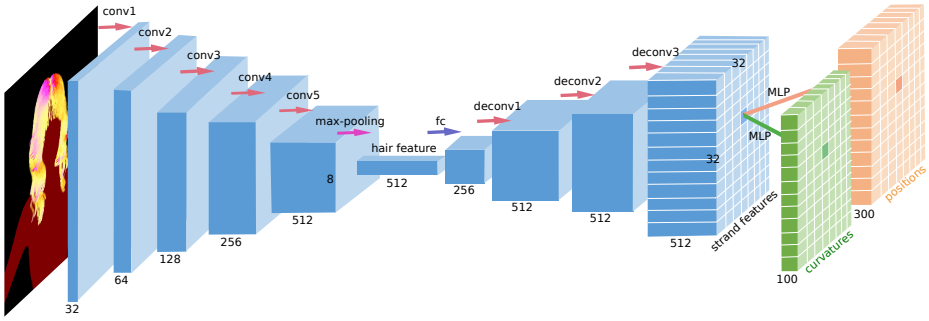

Fig. 2. Network Architecture. The input orientation image is first encoded into a high-level hair feature vector, which is then decoded to 32×32 individual strand-features. Each strand-feature is further decoded to the final strand geometry containing both sample positions and curvatures via two multi-layer perceptron (MLP) networks.

Network Architecture. As illustrated in Figure 2, our network first encodes the input image to a latent vector, followed by decoding the target hair strands from the vector. For the encoder, we use the convolutional layers to extract the high-level features of the image. Different from the common practices that use a fully-connected layer as the last layer, we use the 2D max-pooling to spatially aggregate the partial features (a total number of 8×8) into a global feature vector z. This greatly reduces the number of network parameters.

The decoder generates the hair strands in two steps. The hair feature vector z is first decoded into multiple strand feature vectors $\{z_i\}_{i=0}^{M}$ via deconvolutional layers, and each z_i could be further decoded into the final strand geometry ζ via the same multi-layer fully connected network. This multi-scale decoding mechanism allows us to efficiently produce denser hair models by interpolating

the strand features. According to our experiments, this achieves a more natural appearance than directly interpolating final strand geometry.

It is widely observed that generative neural networks often lose high frequency details, as the low frequency components often dominates the loss in training. Thus, apart from the 3D position $\{\mathbf{p}_i\}$ of each strand, our strand decoder also predicts the curvatures $\{c_i\}$ of all samples. With the curvature information, we can reconstruct the high frequency strand details.

Loss Functions. We apply three losses on our network. The first two losses are the L_2 reconstruction loss of the 3D position and the curvature of each sample. The third one is the collision loss between the output hair strand and the human body. To speed up the collision computation, we approximate the geometry of the body with four ellipsoids as shown in Figure 3.

Given a single-view image, the shape of the visible part of the hair is more reliable than the invisible part, e.g. the inner and back hair. Thus we assign adaptive weights to the samples based on their visibility — visible samples will have higher weights than the invisible ones.

The final loss function is given by:

$$L = L_{pos} + \lambda_1 L_{curv} + \lambda_2 L_{collision}. \tag{1}$$

L_{pos} and L_{curv} are the loss of the 3D positions and the curvatures respectively, written as:

$$\begin{aligned} L_{pos} &= \frac{1}{NM} \sum_{i=0}^{N-1} \sum_{j=0}^{M-1} w_{i,j} ||\mathbf{p}_{i,j} - \mathbf{p}^*_{i,j}||_2^2 \\ L_{curv} &= \frac{1}{NM} \sum_{i=0}^{N-1} \sum_{j=0}^{M-1} w_{i,j} (c_{i,j} - c^*_{i,j})^2 \\ w_{i,j} &= \begin{cases} 10.0 & s_{i,j} \textit{ is visible} \\ 0.1 & \text{otherwise} \end{cases} \end{aligned} \tag{2}$$

where $\mathbf{p}^*_{i,j}$ and $c^*_{i,j}$ are the corresponding ground truth position and curvature to $\mathbf{p}_{i,j}$ and $c_{i,j}$, and $w_{i,j}$ is the visibility weight.

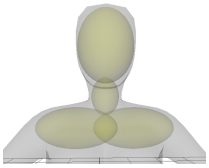

Fig. 3. Ellipsoids for Collision Test.

The collision loss L_{col} is written as the sum of each collision error on the four ellipsoids:

$$L_{col} = \frac{1}{NM} \sum_{k=0}^{3} C_k \tag{3}$$

Each collision error is calculated as the sum of the distance of each collided point to the ellipsoid surface weighted by the length of strand that is inside the ellipsoid, written

$$C_k = \sum_{i=0}^{N-1} \sum_{j=1}^{M-1} \|\mathbf{p}_{i,j} - \mathbf{p}_{i,j-1}\| max(0, Dist_k) \tag{4}$$

$$Dist_k = 1 - \frac{(\mathbf{p}_{i,j,0} - x_k)^2}{a_k^2} - \frac{(\mathbf{p}_{i,j,1} - y_k)^2}{b_k^2} - \frac{(\mathbf{p}_{i,j,2} - z_k)^2}{c_k^2} \tag{5}$$

where $\|\mathbf{p}_{i,j} - \mathbf{p}_{i,j-1}\|$ is the L_1 distance between two adjacent samples on the strand. x_k, y_k, z_k, a_k, b_k, and d_k are the model parameters of the ellipsoid.

Training Details. The training parameters of Equation 1 are fixed to be $\lambda_1 = 1.0$ and $\lambda_2 = 10^{-4}$. During training, we resize all the hair so that the hair is measured in the metric system. We use Relu for nonlinear activation, Adam [24] for optimization, and run the training for 500 epochs using a batch size of 32 and learning rate of 10^{-4} divided by 2 after 250 epochs.

Fig. 4. The orientation image (b) can be automatically generated from a real image (a), or from a synthesized hair model with 9K strands. The orientation map and a down-sampled hair model with 1K strands (c) are used to train the neural network.

3.4 Reconstruction

The output strands from the network may contain noise, and sometimes lose high-frequency details when the target hair is curly. Thus, we further refine the smoothness and curliness of the hair. We first smooth the hair strands by using a Gaussian filter to remove the noise. Then, we compare the difference between the

predicted curvatures and the curvatures of the output strands. If the difference is higher than a threshold, we add offsets to the strands samples. In particular, we first construct a local coordinate frame at each sample with one axis along the tangent of the strand, then apply an offset function along the other two axises by applying the curve generation function described in the work of Zhou et. al [39].

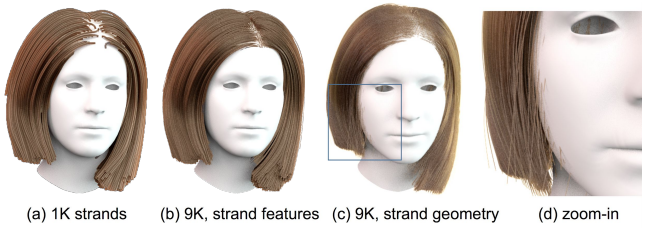

Fig. 5. Hair strand upsampling in the space of (b) the strand-features and (c) the final strand geometry. (d) shows the zoom-in of (c).

The network only generates 1K hair strands, which is insufficient to render a high fidelity output. To obtain higher resolution, traditional methods build a 3D direction field from the guide strands and regrows strands using the direction field from a dense set of follicles. However, this method is time consuming and cannot be used to reconstruct an accurate hair model. Although directly interpolating the hair strands is fast, it can also produce an unnatural appearance. Instead, we bilinearly interpolate the intermediate strand features z_i generated by our network and decode them to strands by using the perceptron network, which enables us to create hair models with arbitrary resolution.

Figure 5 demonstrates that by interpolating in strand-feature space, we can generate a more plausible hair model. In contrast, direct interpolation of the final strands could lead to artifacts like collisions. This is easy to understand, as the strand-feature could be seen as a non-linear mapping of the strand, and could fall in a more plausible space.

Fig. 6. Reconstruction with and without using curliness.

Figure 6 demonstrates the effectiveness of adding curliness in our network. Without using the curliness as an extra constraint, the network only learns the dominant main growing direction while losing the high-frequency details. In this paper, we demonstrate all our results at a resolution of 9K to 30K strands.

4 Evaluation

4.1 Quantitative Results and Ablation Study

In order to quantitatively estimate the accuracy of our method, we prepare a synthetic test set with 100 random hair models and 4 images rendered from random views for each hair model. We compute the reconstruction errors on both the visible and invisible part of the hair separately using the mean square distance between points and the collision error using equation 3. We compare our result with Chai et al.'s method [4]. Their method first queries for the nearest neighbor in the database and then performs a refinement process which globally deforms the hair using the 2D boundary constraints and the 2D orientation constraints based on the input image. To ensure the fairness and efficiency of the comparison, we use the same database in our training set for the nearest neighbor query of [4] based on the visible part of the hair, and set the resolution at 1000 strands. We also compare with Hu et al.'s method [18] which requires manual strokes for generating the 3D hair model. But drawing strokes for the whole test set is too laborious, so in our test, we use three synthetic strokes randomly sampled from the ground-truth model as input. In Table 2, we show the error comparison with the nearest neighbor query results and the methods of both papers. We also perform an ablation test by respectively eliminating the visibility-adaptive weights, the collision loss and the curvature loss from our network.

From the experiments, we observe that our method outperforms all the ablation methods and Chai et al.'s method. Without the visibility-adaptive weights, the reconstruction error is about the same for both the visible and invisible parts, while the reconstruction error of the visible hair decreased by around 30% for all the networks that applies the visibility-adaptive weights. The curvature loss also helps decrease the mean square distance error of the reconstruction. The experiment also shows that using the collision loss will lead to much less error in collision. The nearest-neighbor method results have 0 collision error because the hairs in the database have no collisions.

In Table 3, we compare the computation time and hard disk usage of our method and the data-driven method at the resolution of 9K strands. It can be seen that our method can be about three magnitude faster faster and only uses a small amount of storage space. The reconstruction time differs from straight hair styles and curly hair styles because for straight hair styles which have less curvature difference, we skip the process of adding curves.

	Visible Pos Error	Invisible Pos Error	Collision Error
HairNet	0.017	0.027	2.26×10^{-7}
HairNet - VAW	0.024	0.026	3.5×10^{-7}
HairNet - Col	0.019	0.027	3.26×10^{-6}
NairNet - Curv	0.020	0.029	3.3×10^{-7}
NN	0.033	0.041	0
Chai et al.[4]	0.021	0.040	0
Hu et al.[18]	0.023	0.028	0

Table 2. Reconstruction Error Comparison. The errors are measured in metric. The Pos Error refers to the mean square distance error between the ground-truth and the predicted hair. "-VAW" refers to eliminating the visibility-adaptive weights. "-Col" refers to eliminating the collision loss, "-Curv" refers to eliminating the curvature loss. "NN" refers to nearest neighbor query based on the visible part of the hair.

ours	preprocessing	inference	reconstruction	total time	total space
	0.02 s	0.01 s	0.01 - 0.05 s	0.04 - 0.08 s	70 MiB
Chai et al.[4]	preprocessing	NN query	refinement	total time	total space
	3 s	10 s	40 s	53 s	1 TiB

Table 3. Time and space complexity.

4.2 Qualitative Results

To demonstrate the generality of our method, we tested with different real portrait photographs as input, as shown in the supplementary materials. Our method can handle different overall shapes (e.g. short hairstyles and long hairstyles). In addition, our method can also reconstruct different levels curliness within hairstyles (e.g. straight, wavy, and very curly) efficiently, since we learn the curliness as curvatures in the network and use it to synthesize our final strands.

In Figure 9 and Figure 8, we compare our results of single-view hair reconstruction with autohair [4]. We found that both methods can make rational inference of the overall hair geometry in terms of length and shape, but the hair from our method can preserve better local details and looks more natural, especially for curly hairs. This is because Chai et al.'s method depends on the accuracy and precision of the orientation field generated from the input image, but the orientation field generated from many curly hair images is noisy and the wisps overlap with each other. In addition, they use helix fitting to infer the depth of the hair, but it may fail for very curly hairs, as shown in the second row of Figure 8. Moreover, Chai et al.'s method can only refine the visible part of the hair, so the reconstructed hair may look unnatural from views other than the view of the input image, while the hair reconstructed with our method looks comparatively more coherent from additional views.

Figure 7 shows the interpolation results of our method. The interpolation is performed between four different hair styles and the result shows that our method can smoothly interpolate hair between curly or straight and short or long hairs. We also compare interpolation with Weng et al.'s method [37]. In Figure 7,

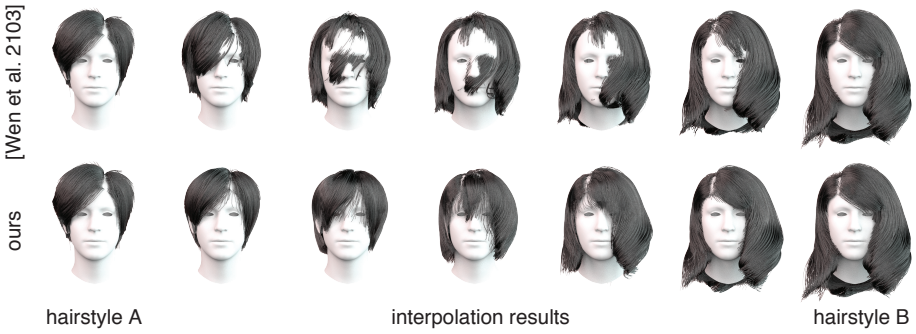

Fig. 7. Interpolation comparison.

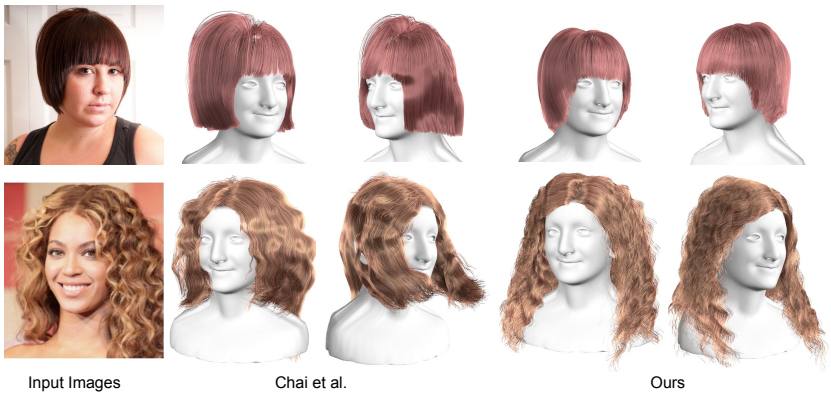

Fig. 8. Comparison with Autohair in different views [4].

Weng et al.'s method produces a lot of artifacts while our method generates more natural and smooth results. The interpolation results indicate the effectiveness of our latent hair representation. Please refer to the supplemental materials for more interpolation results.

We also show video tracking results (see Figure 10 and supplemental video). It shows that our output may fail to achieve sufficient temporal coherence.

5 Conclusion

We have demonstrated the first deep convolutional neural network capable of performing real-time hair generation from a single-view image. By training an end-to-end network to directly generate the final hair strands, our method can capture more hair details and achieve higher accuracy than current state-of-the-art. The intermediate 2D orientation field as our network input provides flexibility, which enables our network to be used for various types of hair representations, such as images, sketches and scans given proper preprocessing. By adopting a multi-scale decoding mechanism, our network could generate hairstyles

Fig. 9. Comparison with Autohair for local details. [4].

of arbitrary resolution while maintaining a natural appearance. Thanks to the encoder-decoder architecture, our network provides a continuous hair representation, from which plausible hairstyles could be smoothly sampled and interpolated.

6 Limitations and Future Work

We found that our approach fails to generate exotic hairstyles like kinky, afro or buzz cuts as shown in Figure 11. We think the main reason is that we do not have such hairstyles in our training database. Building a large hair dataset that covers more variations could mitigate this problem. Our method would also fail when the hair is partially occluded. Thus we plan to enhance our training in the future by adding random occlusions. In addition, we use face detection to estimate the pose of the torso in this paper, but it can be replaced by using deep learning to segment the head and body. Currently, the generated hair model is insufficiently temporally coherent for video frames. Integrating temporal smoothness

as a constraint for training is also an interesting future direction. Although our network provides a more compact representation for the hair, there is no semantic meaning of such latent representation. It would be interesting to concatenate explicit labels (e.g. color) to the latent variable for controlled training.

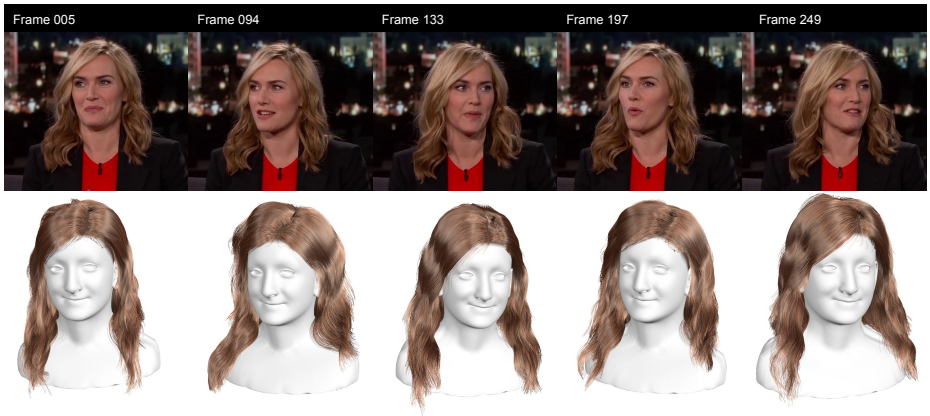

Fig. 10. Hair tracking and reconstruction on video.

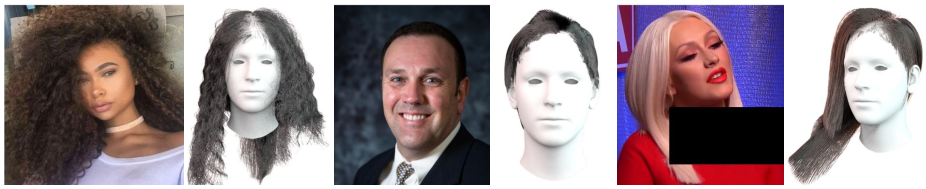

Fig. 11. Failure Cases.

7 Acknowledgement

We thank Weiyue Wang, Haoqi Li, Sitao Xiang and Tianye Li for giving us valuable suggestions in designing the algorithms and writing the paper. This work was supported in part by the ONR YIP grant N00014-17-S-FO14, the CONIX Research Center, one of six centers in JUMP, a Semiconductor Research Corporation (SRC) program sponsored by DARPA, the Andrew and Erna Viterbi Early Career Chair, the U.S. Army Research Laboratory (ARL) under contract number W911NF-14-D-0005, Adobe, and Sony. The content of the information does not necessarily reflect the position or the policy of the Government, and no official endorsement should be inferred.

References

1. Arts, E.: The sims resource (2017), http://www.thesimsresource.com/
2. Beeler, T., Bickel, B., Noris, G., Marschner, S., Beardsley, P., Sumner, R.W., Gross, M.: Coupled 3d reconstruction of sparse facial hair and skin. ACM Trans. Graph. **31**, 117:1–117:10 (August 2012). https://doi.org/10.1145/2185520.2185613, http://graphics.ethz.ch/publications/papers/paperBee12.php
3. Cao, X., Wei, Y., Wen, F., Sun, J.: Face alignment by explicit shape regression. International Journal of Computer Vision **107**(2), 177–190 (2014)
4. Chai, M., Shao, T., Wu, H., Weng, Y., Zhou, K.: Autohair: Fully automatic hair modeling from a single image. ACM Transactions on Graphics (TOG) **35**(4), 116 (2016)
5. Chai, M., Wang, L., Weng, Y., Jin, X., Zhou, K.: Dynamic hair manipulation in images and videos. ACM Trans. Graph. **32**(4), 75:1–75:8 (Jul 2013). https://doi.org/10.1145/2461912.2461990, http://doi.acm.org/10.1145/2461912.2461990
6. Chai, M., Wang, L., Weng, Y., Yu, Y., Guo, B., Zhou, K.: Single-view hair modeling for portrait manipulation. ACM Trans. Graph. **31**(4), 116:1–116:8 (Jul 2012). https://doi.org/10.1145/2185520.2185612, http://doi.acm.org/10.1145/2185520.2185612
7. Choe, B., Ko, H.: A statistical wisp model and pseudophysical approaches for interactivehairstyle generation. IEEE Trans. Vis. Comput. Graph. **11**(2), 160–170 (2005). https://doi.org/10.1109/TVCG.2005.20, http://dx.doi.org/10.1109/TVCG.2005.20
8. Choy, C.B., Xu, D., Gwak, J., Chen, K., Savarese, S.: 3d-r2n2: A unified approach for single and multi-view 3d object reconstruction. CoRR **abs/1604.00449** (2016), http://arxiv.org/abs/1604.00449
9. Echevarria, J.I., Bradley, D., Gutierrez, D., Beeler, T.: Capturing and stylizing hair for 3d fabrication. ACM Trans. Graph. **33**(4), 125:1–125:11 (Jul 2014). https://doi.org/10.1145/2601097.2601133, http://doi.acm.org/10.1145/2601097.2601133
10. Fan, H., Su, H., Guibas, L.J.: A point set generation network for 3d object reconstruction from a single image. CoRR **abs/1612.00603** (2016), http://arxiv.org/abs/1612.00603
11. Fu, H., Wei, Y., Tai, C.L., Quan, L.: Sketching hairstyles. In: Proceedings of the 4th Eurographics Workshop on Sketch-based Interfaces and Modeling. pp. 31–36. SBIM '07, ACM, New York, NY, USA (2007). https://doi.org/10.1145/1384429.1384439, http://doi.acm.org/10.1145/1384429.1384439
12. Girdhar, R., Fouhey, D.F., Rodriguez, M., Gupta, A.: Learning a predictable and generative vector representation for objects. CoRR **abs/1603.08637** (2016), http://arxiv.org/abs/1603.08637
13. Guerrero, P., Kleiman, Y., Ovsjanikov, M., Mitra, N.J.: Pcpnet: Learning local shape properties from raw point clouds. Computer Graphics Forum (Eurographics) (2017)
14. 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. pp. 1–150. ACM (2007)
15. Häne, C., Tulsiani, S., Malik, J.: Hierarchical surface prediction for 3d object reconstruction. CoRR **abs/1704.00710** (2017), http://arxiv.org/abs/1704.00710

16. Herrera, T.L., Zinke, A., Weber, A.: Lighting hair from the inside: A thermal approach to hair reconstruction. ACM Trans. Graph. **31**(6), 146:1–146:9 (Nov 2012). https://doi.org/10.1145/2366145.2366165, http://doi.acm.org/10.1145/2366145.2366165
17. Hu, L., Ma, C., Luo, L., Li, H.: Robust hair capture using simulated examples. ACM Transactions on Graphics (Proceedings SIGGRAPH 2014) **33**(4) (July 2014)
18. Hu, L., Ma, C., Luo, L., Li, H.: Single-view hair modeling using a hairstyle database. ACM Transactions on Graphics (TOG) **34**(4), 125 (2015)
19. Hu, L., Ma, C., Luo, L., Wei, L.Y., Li, H.: Capturing braided hairstyles. ACM Transactions on Graphics (Proceedings SIGGRAPH Asia 2014) **33**(6) (December 2014)
20. Hu, L., Saito, S., Wei, L., Nagano, K., Seo, J., Fursund, J., Sadeghi, I., Sun, C., Chen, Y.C., Li, H.: Avatar digitization from a single image for real-time rendering. ACM Transactions on Graphics (TOG) **36**(6), 195 (2017)
21. Jackson, A.S., Bulat, A., Argyriou, V., Tzimiropoulos, G.: Large pose 3d face reconstruction from a single image via direct volumetric CNN regression. International Conference on Computer Vision (2017)
22. Jakob, W., Moon, J.T., Marschner, S.: Capturing hair assemblies fiber by fiber. ACM Trans. Graph. **28**(5), 164:1–164:9 (Dec 2009). https://doi.org/10.1145/1618452.1618510, http://doi.acm.org/10.1145/1618452.1618510
23. Kim, T.Y., Neumann, U.: Interactive multiresolution hair modeling and editing. ACM Trans. Graph. **21**(3), 620–629 (Jul 2002). https://doi.org/10.1145/566654.566627, http://doi.acm.org/10.1145/566654.566627
24. Kingma, D., Ba, J.: Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014)
25. Li, H., Trutoiu, L., Olszewski, K., Wei, L., Trutna, T., Hsieh, P.L., Nicholls, A., Ma, C.: Facial performance sensing head-mounted display. ACM Transactions on Graphics (TOG) **34**(4), 47 (2015)
26. Luo, L., Li, H., Rusinkiewicz, S.: Structure-aware hair capture. ACM Transactions on Graphics (Proceedings SIGGRAPH 2013) **32**(4) (July 2013)
27. Olszewski, K., Lim, J.J., Saito, S., Li, H.: High-fidelity facial and speech animation for vr hmds. ACM Transactions on Graphics (Proceedings SIGGRAPH Asia 2016) **35**(6) (December 2016)
28. Paris, S., Chang, W., Kozhushnyan, O.I., Jarosz, W., Matusik, W., Zwicker, M., Durand, F.: Hair photobooth: Geometric and photometric acquisition of real hairstyles. ACM Trans. Graph. **27**(3), 30:1–30:9 (Aug 2008). https://doi.org/10.1145/1360612.1360629, http://doi.acm.org/10.1145/1360612.1360629
29. Qi, C.R., Su, H., Mo, K., Guibas, L.J.: Pointnet: Deep learning on point sets for 3d classification and segmentation. arXiv preprint arXiv:1612.00593 (2016)
30. Qi, C.R., Yi, L., Su, H., Guibas, L.J.: Pointnet++: Deep hierarchical feature learning on point sets in a metric space. arXiv preprint arXiv:1706.02413 (2017)
31. Riegler, G., Ulusoy, A.O., Geiger, A.: Octnet: Learning deep 3d representations at high resolutions. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. vol. 3 (2017)
32. Tatarchenko, M., Dosovitskiy, A., Brox, T.: Octree generating networks: Efficient convolutional architectures for high-resolution 3d outputs. CoRR, abs/1703.09438 (2017)

33. Tulsiani, S., Zhou, T., Efros, A.A., Malik, J.: Multi-view supervision for single-view reconstruction via differentiable ray consistency. CoRR **abs/1704.06254** (2017), http://arxiv.org/abs/1704.06254
34. Wang, L., Yu, Y., Zhou, K., Guo, B.: Example-based hair geometry synthesis. ACM Trans. Graph. **28**(3), 56:1–56:9 (2009)
35. Wang, P.S., Liu, Y., Guo, Y.X., Sun, C.Y., Tong, X.: O-cnn: Octree-based convolutional neural networks for 3d shape analysis. ACM Trans. Graph. **36**(4), 72:1–72:11 (Jul 2017). https://doi.org/10.1145/3072959.3073608, http://doi.acm.org/10.1145/3072959.3073608
36. Ward, K., Bertails, F., yong Kim, T., Marschner, S.R., paule Cani, M., Lin, M.C.: A survey on hair modeling: styling, simulation, and rendering. In: IEEE TRANSACTION ON VISUALIZATION AND COMPUTER GRAPHICS. pp. 213–234 (2006)
37. Weng, Y., Wang, L., Li, X., Chai, M., Zhou, K.: Hair Interpolation for Portrait Morphing. Computer Graphics Forum (2013). https://doi.org/10.1111/cgf.12214
38. Xu, Z., Wu, H.T., Wang, L., Zheng, C., Tong, X., Qi, Y.: Dynamic hair capture using spacetime optimization. ACM Trans. Graph. **33**(6), 224:1–224:11 (Nov 2014). https://doi.org/10.1145/2661229.2661284, http://doi.acm.org/10.1145/2661229.2661284
39. Yu, Y.: Modeling realistic virtual hairstyles. In: Computer Graphics and Applications, 2001. Proceedings. Ninth Pacific Conference on. pp. 295–304. IEEE (2001)
40. Yuksel, C., Schaefer, S., Keyser, J.: Hair meshes. ACM Trans. Graph. **28**(5), 166:1–166:7 (Dec 2009). https://doi.org/10.1145/1618452.1618512, http://doi.acm.org/10.1145/1618452.1618512
41. 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), 156:1–156:11 (Jul 2017). https://doi.org/10.1145/3072959.3073627, http://doi.acm.org/10.1145/3072959.3073627
42. Zhao, H., Shi, J., Qi, X., Wang, X., Jia, J.: Pyramid scene parsing network. In: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2017)