3D Hair Synthesis Using Volumetric Variational Autoencoders

SHUNSUKE SAITO, Pinscreen, University of Southern California, USC Institute for Creative Technologies
LIWEN HU, Pinscreen, University of Southern California
CHONGYANG MA, Snap Inc.
HIKARU IBAYASHI, University of Southern California
LINJIE LUO, Snap Inc.
HAO LI, Pinscreen, University of Southern California, USC Institute for Creative Technologies

Fig. 1. Our method automatically generates 3D hair strands from a variety of single-view inputs. Each panel from left to right: input image, volumetric representation with color-coded local orientations predicted by our method, and final synthesized hair strands rendered from two viewing points. Original images courtesy of @elstilespb, Alex Neman and Ron Armstrong for the wavy hairstyle photo, the back-view image and the dog photo. ©2018 Estate of Pablo Picasso / Artists Rights Society (ARS), New York for the portrait drawing.

Recent advances in single-view 3D hair digitization have made the creation of high-quality CG characters scalable and accessible to end-users, enabling new forms of personalized VR and gaming experiences. To handle the complexity and variety of hair structures, most cutting-edge techniques rely on the successful retrieval of a particular hair model from a comprehensive hair database. Not only are the aforementioned data-driven methods storage intensive, but they are also prone to failure for highly unconstrained input images, complicated hairstyles, and failed face detection. Instead of using a large collection of 3D hair models directly, we propose to represent the manifold of 3D hairstyles implicitly through a compact latent space of a volumetric variational autoencoder (VAE). This deep neural network is trained with volumetric orientation field representations of 3D hair models and can synthesize new hairstyles from a compressed code. To enable end-to-end 3D hair inference, we train an additional embedding network to predict the code in the VAE latent space from any input image. Strand-level hairstyles can then be generated from the predicted volumetric representation. Our fully automatic framework does not require any ad-hoc face fitting, intermediate classification and segmentation, or hairstyle database retrieval. Our hair synthesis approach is significantly more robust and can handle a much wider variation of hairstyles than state-of-the-art data-driven hair modeling techniques with challenging inputs, including photos that are low-resolution, overexposured, or contain extreme head poses. The storage requirements are minimal and a 3D hair model can be produced from an image in a second. Our evaluations also show that successful reconstructions are possible from highly stylized cartoon images, non-human subjects, and pictures taken from behind a person. Our approach is particularly well suited for continuous and plausible hair interpolation between very different hairstyles.

CCS Concepts: • **Shape modeling** → *Volumetric models*;

Additional Key Words and Phrases: hair synthesis, single-view modeling, deep generative model, volumetric variational autoencoder

ACM Reference Format:
Shunsuke Saito, Liwen Hu, Chongyang Ma, Hikaru Ibayashi, Linjie Luo, and Hao Li. 2018. 3D Hair Synthesis Using Volumetric Variational Autoencoders. *ACM Trans. Graph.* 37, 6, Article 208 (November 2018), 12 pages. https://doi.org/10.1145/3272127.3275019

Authors' addresses: Shunsuke Saito, Pinscreen, University of Southern California, USC Institute for Creative Technologies, saitos@usc.edu; Liwen Hu, Pinscreen, University of Southern California; Chongyang Ma, Snap Inc. Hikaru Ibayashi, University of Southern California; Linjie Luo, Snap Inc. Hao Li, Pinscreen, University of Southern California, USC Institute for Creative Technologies.

0730-0301/2018/11-ART208 $15.00
https://doi.org/10.1145/3272127.3275019

1 INTRODUCTION

The 3D acquisition of human hair has become an active research area in computer graphics in order to make the creation of digital humans more efficient, automated, and cost effective. High-end hair capture techniques based on specialized hardware [Beeler et al. 2012; Echevarria et al. 2014; Herrera et al. 2012; Jakob et al. 2009; Luo et al. 2013; Paris et al. 2008; Xu et al. 2014] can already produce high-quality 3D hair models, but can only operate in well-controlled studio environments. More consumer-friendly techniques, such as those that only require a single input image [Chai et al. 2015, 2016; Hu et al. 2015, 2017], are becoming increasingly popular and important as they can facilitate the mass adoption of new 3D avatar-driven applications, including personalized gaming, communication in VR [Li et al. 2015; Olszewski et al. 2016; Thies et al. 2018], and social media apps [FaceUnity 2017; itSeez3D: Avatar SDK 2017; Myidol 2017; Pinscreen 2017]. Existing single-view hair modeling methods all rely on a large database containing hundreds of 3D hairstyles, which is used as shape prior for further refinement and to handle the complex variations of possible hairstyles.

This paradigm comes with several fundamental limitations: (1) the large storage footprints of the hair model database prohibit their deployment on resource-constrained platforms such as mobile devices; (2) the search steps are usually slow and difficult to scale as the database grows to handle increasingly various hairstyles; (3) these techniques also rely on well-conditioned input photographs and are susceptible to the slightest failures during the image pre-processing and analysis step, such as failed face detection, incorrect head pose fitting, or poor hair segmentation. Furthermore, these data-driven algorithms are based on hand-crafted descriptors and do not generalize well beyond their designed usage scenarios. They often fail in practical scenarios, such as those with occluded face/hair, poor resolution, degraded quality, or artistically stylized input.

To address the above challenges, we propose an end-to-end single-view 3D hair synthesis approach using a deep generative model to represent the continuous space of hairstyles. We implicitly model the continuous space of hairstyles using a compact generative model so that plausible hairstyles can be effectively sampled and interpolated, and hence, eliminate the need for a comprehensive database. We also enable end-to-end training and 3D hairstyle inference from a single input image by learning deep features from a large set of unconstrained images.

To effectively model the space of hairstyles, we introduce the use of volumetric *occupancy and flow fields* to represent 3D hairstyles for our generative hair modeling framework. We present a variant of volumetric variational autoencoder (VAE) [Kingma and Welling 2014] to learn the mapping from a compact latent space to the space of hairstyles represented by a volumetric representation of a large database of hairstyles [Hu et al. 2015].

To achieve end-to-end 3D hair inference, we train an additional hair embedding neural network to predict the code in the learned VAE latent space from input images. Instead of direct prediction to the latent space, we perform Principled Component Analysis (PCA) in the latent space for an embedding subspace to achieve better generalization performance via prediction to this subspace. In addition, we apply Iterative Error Feedback (IEF) [Carreira et al. 2016] to our embedding network to further facilitate generalization.

We include an ablation study of different algorithmic components to validate our proposed architecture (Section 4). We show that our method can synthesize faithful 3D hairstyles from a wide range of input images with various occlusions, degraded image quality, extreme lighting conditions, uncommon hairstyles, and significant artistic abstraction (see Fig 1 and Section 5). We also compare our technique to the latest algorithm for single-view 3D hair modeling [Chai et al. 2016] and show that our approach is significantly more robust on challenging input photos. Using our learned generative model, we further demonstrate that plausible hairstyles can be interpolated effectively between drastically different ones, while the current state-of-the-art method [Weng et al. 2013] fails.

Our main contributions are:

- The first end-to-end framework for synthesis of 3D hairstyles from a single input image without requirement of face detection or hair segmentation. Our approach can handle a wider range of hairstyles and is significantly more robust for challenging input images than existing data-driven techniques.
- A variational autoencoder using a volumetric occupancy and flow field representation. The corresponding latent space is compact and models the wide range of possible hairstyles continuously. Plausible hairstyles can be sampled and interpolated effectively using this VAE-based generative model, and converted into a strand-based hair representation.
- A hair embedding network with robust generalization performance using PCA embedding and an iterative error feedback technique.

2 RELATED WORK

The creation of high-quality 3D hair models is one of the most time consuming tasks when modeling CG characters. Despite the availability of various design tools [Choe and Ko 2005; Fu et al. 2007; Kim and Neumann 2002; Wither et al. 2007; Yu et al. 2014; Yuksel et al. 2009] and commercial solutions such as XGen, Ornatrix and HairFarm, production of a single 3D hair model for a hero character can take hours or even days for professional character artists. A detailed discussion of seminal hair modeling techniques can be found in Ward et al. [2007].

Multi-View Hair Capture. Hair digitization techniques have been introduced in attempts to reduce and eliminate the laborious and manual effort of 3D hair modeling. Most high-end 3D hair capture systems [Beeler et al. 2012; Echevarria et al. 2014; Herrera et al. 2012; Jakob et al. 2009; Luo et al. 2013; Paris et al. 2004, 2008; Xu et al. 2014] maximize the coverage of hair during acquisition and are performed under controlled lighting conditions. The multi-view stereo technique of Luo et al. [2013] shows that, for the first time, highly complex real-world hairstyles can be convincingly reconstructed in 3D by discovering locally coherent wisp structures. Hu et al. [2014a] later proposes a data-driven variant using pre-simulated hair strands, which eliminates the generation of physically implausible hair strands. Their follow-up work [Hu et al. 2014b] solves the problem of capturing constrained hairstyles such as braids using procedurally generated braid structures. In this work they

used an RGB-D sensor (Kinect) that is swept around the subjects instead of a collection of calibrated cameras. [Zhang et al. 2017] recently proposes a generalized four-view image-based hair modeling method that does not require all views to be from the same hairstyle, which allows the creation of new hair models. These multi-view capture systems are not easily accessible to end-user, as they often require expensive hardware equipment, controlled capture settings, and professional manual clean-up.

Single-View Hair Modeling. With the availability of internet pictures and the ease of taking selfies, single-view hair modeling solutions are becoming increasingly important within the context of consumer-friendly 3D avatar digitization. Single-view hair modeling techniques were first introduced by Chai et al. [2013; 2012] for portrait manipulation purposes. These early geometric optimization methods are designed for reconstructing front-facing subjects and have difficulty approximating the geometry of the back of the hair.

Hu et al. [2015] proposes a data-driven method to produce entire hairstyles from a single input photograph and some user interactions. Their method assembles different hairstyles from a 3D hairstyle database developed for the purpose of shape reconstruction. Chai et al. [2016] later presents a fully automated variant using an augmented 3D hairstyle database and a deep convolutional neural network to segment hair regions. Hu et al. [2017] further improves the retrieval performance by introducing a deep learning-based hair attribute classifier, that increases the robustness for challenging input images from which local orientation fields are difficult to extract. However, these data-driven methods rely on the quality and diversity of the database, as well as a successful preprocessing and analysis of the input image. In particular, if a 3D hair model with identifiable likeness is not available in the database, the reconstructed hair model is likely to fail. Furthermore, handcrafted descriptors become difficult to optimize as the diversity or number of hair models increases. Recently, Zhou et al. [2018] presents a method for single-view hair modeling by directly inferring 3D strands from a 2D orientation field of segmented hair region.

Shape Space Embedding. Embedding a high-dimensional shape space into a compact subspace has been widely investigated for the shape modeling of human bodies [Anguelov et al. 2005; Loper et al. 2015] and faces [Blanz and Vetter 1999]. Since different subjects are anatomically compatible, it is relatively easy to present them into a continuous low dimensional subspace. However, it is not straightforward to apply these embedding techniques to hairstyles due to their complex volumetric and topological structures, and the difficulty of annotating correspondences between hairstyles.

3D Deep Learning. The recent success of deep neural networks for tasks such as classification and regression can be explained in part by their effectiveness in converting data into a high-dimensional feature representation. Because convolutional neural networks are designed to process images, 3D shapes are often converted into regular grid representations to enable convolutions. Multi-view CNNs [Qi et al. 2016; Su et al. 2015] render 3D point clouds or meshes into depth maps and then apply 2D convolutions to them. Volumetric CNNs [Maturana and Scherer 2015; Qi et al. 2016; Wu et al. 2015; Yumer and Mitra 2016] apply 3D convolutions directly on the voxels, which are converted from a 3D mesh or point cloud. PointNet [Qi et al. 2017a,b] presents a unified architecture that can directly take point clouds as input. Brock et al. [2016] applies 3D CNNs to variational autoencoder [Kingma and Welling 2014] in order to embed 3D volumetric objects into a compact subspace. These methods are limited to very low resolutions (e.g., $32 \times 32 \times 32$) and focus on man-made shapes, while our goal is to encode high-resolution ($128 \times 192 \times 128$) 3D orientation fields as well as volumes of hairstyles. Recently, Jackson et al. [2017] proposed to infer 3D face shape in the image space via direct volumetric regression from single-view input. While we are also embedding a volumetric representation, our hairstyle representation uses a 3D direction field in addition to an occupancy grid. Furthermore, we learn this embedding in a canonical space with fixed head size and position, which allows us to handle cropped images, as well as head models in arbitrary positions and orientations.

3 METHOD

In this section, we describe the entire pipeline of our algorithm for single-view 3D hair modeling (Figure 3). We first explain our hair data representation using volumetric *occupancy* and *flow* fields (Section 3.1). Using a dataset of more than two thousand different 3D hairstyles, we train a volumetric variational autoencoder to obtain a compact latent space, which encodes the immerse space of plausible 3D hairstyles (Section 3.2). To enable end-to-end single-view 3D hairstyle modeling, we train an additional embedding network to help predict the volumetric representation from an input image (Section 3.3). Finally, we synthesize hair strands by growing them from the scalp of a head model based on the predicted volume. If a face can be detected or manually fitted from the input image, we can optionally refine the output strands to better match the single-view input (Section 3.4).

3.1 Hair Data Representation

Our hair data representation is constrained by two factors. First, the data representation itself needs to be easily handled by neural networks for our training and inference algorithms. Second, our representation should be compatible with traditional strand-based representation for high-fidelity modeling and rendering. To achieve these two goals, we adopt a similar concept used in previous approaches [Paris et al. 2004, 2008; Wang et al. 2009; Wei et al. 2005] and convert hair strands into a representation of two components, i.e., a 3D occupancy field and the corresponding flow field, both defined on uniformly sampled grids of resolution $128 \times 192 \times 128$. We use a large resolution along the y-axis (vertical direction) to better accommodate longer hairstyles.

Specifically, given a hairstyle of 3D strands, we generate an occupancy field $\mathbf{O}$ using the outer surface extraction method proposed in Hu et al. [2017]. Each grid of $\mathbf{O}$ has a value of 1 if the grid center is inside of the hair volume and is set to 0 otherwise. We also generate a 3D flow field $\mathbf{F}$ from the 3D hair strands. We first compute the local 3D orientation for those grids inside the hair volume by averaging the orientations of nearby strands [Wang et al. 2009]. Then we smoothly diffuse the flow field into the entire volume as proposed by Paris et al. [2004]. Conversely, given an occupancy field $\mathbf{O}$ and

Fig. 2. Volumetric hairstyle representation. From left to right: original 3D hairstyle represented as strands; our representation using occupancy and flow fields defined on regular grids, with the visualization of the occupancy field boundary as mesh surface and encoding of local flow value ans surface color; regrown strands from our representation.

the corresponding flow field **F**, we can easily regenerate 3D strands by growing from hair roots on a fixed scalp. The hair strands are grown following the local orientation of the flow field **F** until hitting the surface boundary defined by the volume **O**. See Figure 2 for some concrete examples.

3.2 Volumetric Variational Autoencoder

Variational Autoencoder. Our approach is based on variational autoencoder (VAE), which has emerged as one of the most popular generative models in recent years [Kingma and Welling 2014; Rezende et al. 2014; Tan et al. 2018]. A typical VAE consists of an encoder $\mathcal{E}_\theta(\mathbf{x})$ and a decoder $\mathcal{D}_\phi(\mathbf{z})$. The encoder $\mathcal{E}_\theta$ encodes an input $\mathbf{x}$ into a latent code $\mathbf{z}$, and the decoder $\mathcal{D}_\phi$ generates an output $\mathbf{x}'$ from the latent code $\mathbf{z}$. The parameters θ and ϕ of the encoder and the decoder can be jointly trained so that the reconstruction error between $\mathbf{x}$ and $\mathbf{x}'$ is minimized. While a vanilla autoencoder [Bengio et al. 2009] uses a deterministic function for the encoder $\mathcal{E}_\theta(\mathbf{x})$, a variational autoencoder (VAE) [Kingma and Welling 2014] approximates $\mathcal{E}_\theta(\mathbf{x})$ as a posterior distribution $q(\mathbf{z}|\mathbf{x})$, allowing us to generate a new data $\mathbf{x}'$ by sampling $\mathbf{z}$ from a prior distribution. We train the encoding and decoding parameters θ and ϕ using stochastic gradient variational Bayes (SGVB) algorithm [Kingma and Welling 2014] as follows:

$$\theta^*, \phi^* = \underset{\theta,\phi}{\operatorname{argmin}} \mathbb{E}_{\mathbf{z}\sim\mathcal{E}_\theta(\mathbf{x})}\left[-\log p(\mathbf{x}|\mathbf{z})\right] + D_{kl}\big(\mathcal{E}_\theta(\mathbf{x})|p(\mathbf{z})\big), \quad (1)$$

where D_{kl} denotes the Kullback-Leibler divergence. Assuming a multivariate Gaussian distribution $\mathcal{E}_\theta(\mathbf{x}) \sim \mathcal{N}(\mathbf{z}_\mu, \mathrm{diag}(\mathbf{z}_\sigma))$ as a posterior and a standard isotropic Gaussian prior $p(\mathbf{z}) \sim \mathcal{N}(0, I)$, the Kullback-Leibler divergence D_{kl} is formulated as

$$D_{kl}\big(\mathcal{E}_\theta(\mathbf{x})|\mathcal{N}(0,I)\big) = \frac{1}{2}\sum_i \left(1 + 2\log \mathbf{z}_{\sigma,i} - \mathbf{z}_{\mu,i}^2 - \mathbf{z}_{\sigma,i}^2\right), \quad (2)$$

Table 1. Our volumetric VAE architecture. The last convolution layer in the encoder is duplicated for μ and σ for reparameterization trick. The decoders for occupancy field and orientation field are the same architecture except the last channel size (1 and 3 respectively). The weights on the decoders are not shared. All the convolutional layers are followed by batch normalization and ReLU activation except the last layer in both the encoder and the decoder.

Net	Type	Kernel	Stride	Output
enc.	conv.	4×4	2×2	$64\times96\times64\times4$
enc.	conv.	4×4	2×2	$32\times48\times32\times8$
enc.	conv.	4×4	2×2	$16\times24\times16\times16$
enc.	conv.	4×4	2×2	$8\times12\times8\times32$
enc.	conv.	4×4	2×2	$4\times6\times4\times64$
dec.	transconv.	4×4	2×2	$8\times12\times8\times32$
dec.	transconv.	4×4	2×2	$16\times24\times16\times16$
dec.	transconv.	4×4	2×2	$32\times48\times32\times8$
dec.	transconv.	4×4	2×2	$64\times96\times64\times4$
dec.	transconv.	4×4	2×2	$128\times192\times128\times\{1,3\}$

where $\mathbf{z}_\sigma, \mathbf{z}_\mu$ are the multidimentional output of $\mathcal{E}_\theta(\mathbf{x})$, representing the mean and standard deviation respectively, and D_{kl} is computed as summation over all the channels of $\mathbf{z}_\sigma$ and $\mathbf{z}_\mu$. To make all the operations differentiable for back propagation, the random variable $\mathbf{z}$ is sampled from the distribution $\mathcal{E}_\theta(\mathbf{x})$ via a reparameterization trick [Kingma and Welling 2014] as below:

$$\mathbf{z} = \mathbf{z}_\mu + \epsilon \odot \mathbf{z}_\sigma, \quad \epsilon \sim \mathcal{N}(0, I), \quad (3)$$

where $\odot$ is an element-wise matrix multiplication operator.

Hairstyle Dataset. To train the encoding and decoding parameters of a VAE using our volumetric representation, we first collect 816 portrait images of various hairstyles and use a state-of-the-art single-view hair modeling method [Hu et al. 2015] to reconstruct 3D hair strands as our dataset. For each portrait image, we manually draw $1 \sim 4$ strokes to model the global structure, the local strand shapes are then refined automatically. By adding the 343 hairstyles from the USC-HairSalon dataset[1], we have collected 1159 different 3D hairstyles in total. We further augment the data by flipping each hairstyle horizontally and obtain a dataset of 2318 different hairstyles. The hair geometry is normalized and aligned by fitting to a fixed head model. We randomly split the entire dataset into a training set of 2164 hairstyles and a test set of 154 hairstyles.

VAE Architecture. From the volumetric representation of the collected 3D hairstyle dataset, we train an encoder-decoder network to obtain a compact model for the space of 3D hairstyles. The architecture of our VAE model is shown in Table 1. The encoder concatenates the occupancy field **O** and flow field **F** together as volumetric input of resolution $128\times192\times128$ (see Section 3.1) and encode the input into a volumetric latent space $\mathbf{z}_\mu$ and $\mathbf{z}_\sigma$ of resolution $4\times6\times4$. Each voxel in the latent space has a feature vector of dimension 64. Then we sample a latent code $\mathbf{z} \in \mathbb{R}^{4\times6\times4\times64}$ from $\mathbf{z}_\mu$ and $\mathbf{z}_\sigma$ using the reparameterization trick [Kingma and Welling 2014]. The latent code $\mathbf{z}$ is used as input for two decoders. One of them generates a scalar field as a level-set representing the hair volume while the other is used to generate the 3D flow field.

[1] http://www-scf.usc.edu/~liwenhu/SHM/database.html

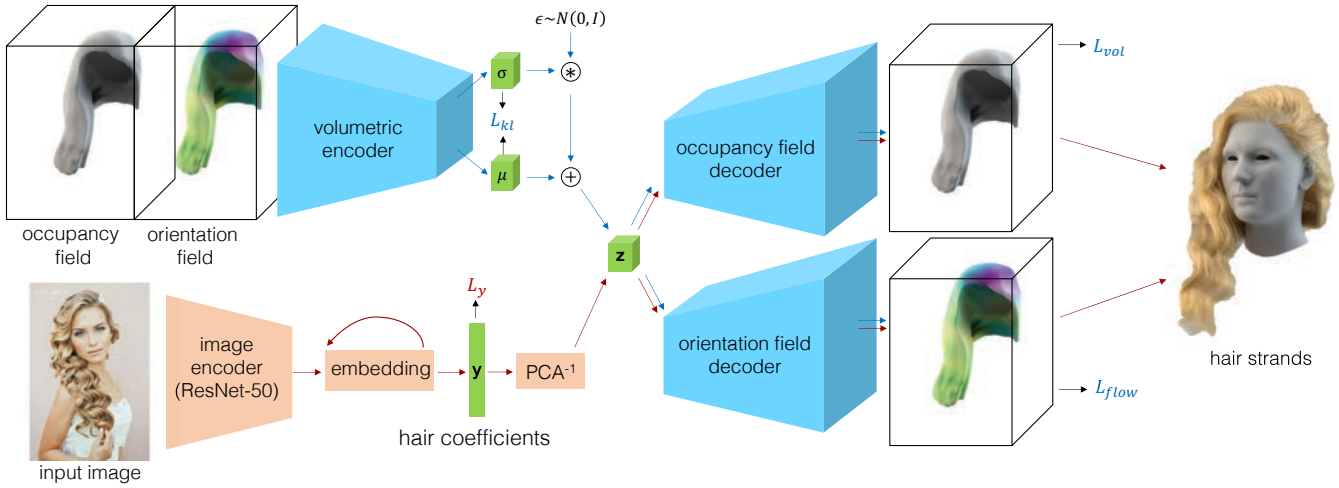

Fig. 3. Our pipeline overview. Our volumetric VAE consists of an encoder and two decoders (blue blocks) with blue arrows representing the related dataflow. Our hair embedding network (orange blocks) follows the red arrows to synthesize hair strands from an input image.

Loss Function. Our loss function to train the network weights consists of reconstruction errors for occupancy field and flow field, as well as KL divergence loss [Kingma and Welling 2014]. We use Binary Cross-Entropy (BCE) loss for the reconstruction of occupancy fields. The standard BCE loss is

$$\mathcal{L}_{BCE} = -\frac{1}{|\mathcal{V}|}\sum_{i\in\mathcal{V}}\left[O_i \log \hat{O}_i + (1-O_i)\log\left(1-\hat{O}_i\right)\right],$$

where $\mathcal{V}$ denotes the uniformly sampled grids, $|\mathcal{V}|$ is the total number of grids, $O_i \in \{0,1\}$ is the ground-truth occupancy field value at a voxel v_i, and $\hat{O}_i$ is the value predicted by the network and is in the range of $[0,1]$. Brock et al. [2016] modifies the BCE loss by setting the range of target value as $\{-1,2\}$ to prevent the gradient vanishing problem:

$$\mathcal{L}'_{BCE} = -\frac{1}{|\mathcal{V}|}\sum_{i\in\mathcal{V}}\left[\gamma O_i \log \hat{O}_i + (1-\gamma)(1-O_i)\log\left(1-\hat{O}_i\right)\right],$$

where γ is a relative weight to penalize more on false negatives [Brock et al. 2016]. Although the modified loss function above improves the overall reconstruction accuracy, the details around the hair volume boundary from a typical encoder-decoder network are usually over-smoothed, which may change the hairstyle structure unnaturally (see Figure 4). To address this issue, we introduce a boundary-aware weighting scheme by changing the loss function into:

$$\mathcal{L}_{vol} = -\frac{1}{n(\alpha-1)+|\mathcal{V}|}\sum_{i\in\mathcal{V}} w_i\left[\gamma O_i \log \hat{O}_i + (1-\gamma)(1-O_i)\log\left(1-\hat{O}_i\right)\right], \quad (4)$$

$$w_i = \begin{cases}\alpha & \text{iff } v_i \in \mathcal{N}(B_t)\\ 1 & \text{otherwise}\end{cases}$$

where w_i takes a constant weight α larger than 1 when the voxel v_i belongs to the one-ring neighbor $\mathcal{N}(B_t)$ of any boundary voxel B_t inside the ground-truth hair volume, and n is the number of voxels $\{v_i \in \mathcal{N}(B_t)\}$. For 3D orientation field, we use L1 loss because L2 loss is known to produce over smoothed prediction results [Isola et al. 2017]:

$$\mathcal{L}_{flow} = \sum_{i\in\mathcal{V}} O_i \left\|\mathbf{f}_i - \hat{\mathbf{f}}_i\right\|_1 \Big/ \sum_{i\in\mathcal{V}} O_i, \quad (5)$$

where $\mathbf{f}_i$ and $\hat{\mathbf{f}}_i$ are the ground-truth and predicted flow vectors at voxel v_i respectively. Our KL-divergence loss is defined as:

$$\mathcal{L}_{kl} = D_{kl}\Big(q(z|O,\mathbf{f})\big|\mathcal{N}(0,I)\Big). \quad (6)$$

where q is the Gaussian posterior $\mathcal{E}_\theta(O,\mathbf{f})$. Then our total loss becomes

$$\mathcal{L} = \mathcal{L}_{vol} + w_{flow}\mathcal{L}_{flow} + w_{kl}\mathcal{L}_{kl}, \quad (7)$$

where w_{flow}, w_{kl} are relative weights for the orientation field reconstruction loss and the KL divergence loss respectively.

Implementation Details. For both the encoder and decoder networks, we use kernel size of 4, stride of 2 and padding of 1 for all the convolution operations. All the convolutional layers are followed by batch normalization and ReLU activation except the last layer in both networks. We use *sigmoid* and *tanh* as the nonlinear activations for the occupancy and flow fields respectively. The training parameters are fixed to be $\gamma = 0.97$, $\alpha = 50$, $w_{flow} = 1.0$ and $w_{kl} = 2\times10^{-5}$ based on cross validation. We minimize the loss function for 400 epochs using Adam solver [Kingma and Ba 2015]. We use a batch size of 4 and learning rate of 1×10^{-3}.

3.3 Hair Embedding Network

To achieve end-to-end single-view 3D hair synthesis, we train an embedding network to predict the hair latent code $\mathbf{z}$ in the latent space from input images. We use the collected dataset of portrait photos and the corresponding 3D hairstyles as training data (see Section 3.2).

Since our training data is limited, it is desirable to reduce the number of unknowns to be predicted for more robust training of the

Table 2. Evaluation of training loss functions in terms of reconstruction accuracy for occupancy field (IOU, precision and recall) and flow field (*L2* loss). We evaluate the effectiveness of our proposed loss function by comparing it with (1) our loss function without the KL-divergence loss term denoted as "Ours (AE)", (2) a state-of-the-art volumetric generative model using VAE [Brock et al. 2016] and (3) vanilla VAE [Kingma and Welling 2014].

Training Loss	IOU	Precision	Recall	L2 (flow)
Ours (VAE)	**0.8243**	**0.8888**	**0.9191**	**0.2118**
Ours (AE)	0.8135	0.8832	0.9116	0.2403
[Brock et al. 2016]	0.6879	0.8249	0.8056	0.2308
Vanilla VAE	0.5977	0.7672	0.7302	0.2341

embedding. We assume that the latent space of 3D hairstyles can be well-approximated in a low-rank linear space. Based on this assumption, we compute the PCA embedding of the volumetric latent space and use 512-dimensional PCA coefficients $\mathbf{y}$ as a compact feature representation of the feasible space of 3D hairstyles. Then the goal of the embedding task is to match predicted hair coefficients $\hat{\mathbf{y}}$ to the ground-truth coefficients $\mathbf{y}$ by minimizing the following L2 loss:

$$\mathcal{L}_{\mathbf{y}} = \left\| \mathbf{y} - \hat{\mathbf{y}} \right\|_2. \tag{8}$$

Note that we use $\mathbf{z}_\mu$ instead of stochastically sampled latent code $\mathbf{z} \sim \mathcal{N}(\mathbf{z}_\mu, \mathbf{z}_\sigma)$ to eliminate randomness in the embedding process. Our hair embedding pipeline is shown in Figure 3 (bottom part).

We use a ResNet-50 model [He et al. 2016] pretrained on ImageNet [Deng et al. 2009] and fine tune the model as an image encoder. We apply average pooling in the last convolution layer and take the output vector as an image feature vector $\mathbf{I} \in \mathbb{R}^{2048}$. Then we apply the process of Iterative Error Feedback (IEF) [Carreira et al. 2016; Kanazawa et al. 2018] to train our hair embedding network. The embedding network $\mathcal{P}$ takes the image feature vector $\mathbf{I}$ together with the current hair coefficients $\mathbf{y}_t$ as input and predicts the updated coefficients $\mathbf{y}_{t+1}$ as below:

$$\hat{\mathbf{y}}_{t+1} = \hat{\mathbf{y}}_t + \mathcal{P}\left(\mathbf{I}, \hat{\mathbf{y}}_t\right). \tag{9}$$

IEF is known to have better generalization performance compared to direct embedding in a single shot, which usually overfits the ground-truth training data (see Section 4 for some evaluations). We run three iterations of IEF since no further performance improvement is observed afterwards.

Our hair embedding network consists of two 1024-dimensional fully connected layers with ReLU and dropout layers in-between, followed by an output layer with 512 neurons. The learning rate is set to 10^{-5} and 10^{-4} for the image encoder and the hair embedding network respectively. We train the network with 1000 epochs using Adam solver [Kingma and Ba 2015] on our collected hairstyle dataset (Section 3.2). We use a batch size of 16 and learning rate of $1{\times}10^{-4}$. To make our embedding network more robust against input variations, we augment our image dataset by applying different random image manipulations, including Gaussian noise (of standard deviation 0.15), Gaussian blur (of standard deviation 0.15), rotation (maximum 20 degrees), scaling (within the range of $[0.5, 1.5]$), occlusion (maximum 40% with random color [Saito et al. 2016]) and color jittering (brightness, contrast, hue and saturation).

Table 3. Evaluation of different embedding methods. First row: our linear PCA embedding. Second row: a single-vector VAE (in contrast to our volumetric VAE). Third row: a non-linear embedding with fully connected layers and ReLU activations. The dimension of latent space is 512 for all the three methods here.

Embedding	IOU	Precision	Recall	L2 (flow)
PCA	**0.8127**	**0.8797**	**0.9143**	**0.2170**
Single-vector VAE	0.6278	0.7214	0.7907	0.2223
Non-Linear	0.6639	0.7784	0.8186	0.2637

Discussions about PCA Embedding. Unlike most VAE based approaches, which reshape the encoding result into one long vector and apply fully connected layers to further reduce dimension, we use a volumetric latent space to preserve the spatial dimensions. We argue that reshaping into a one-dimensional vector will limit the expressiveness of the network significantly because it is difficult to fully cover hair local variation in the training process. See Section 4 for the results of our ablation study. Zhang et al. [2016] shows that PCA is the optimal solution for low-rank approximation in linear cases. We have also experimented with a non-linear embedding using multilayer perceptron (MLP) [Rumelhart et al. 1985]. Although MLP should be more general than PCA with its nonlinear layers, we have found that using MLP will overfit our training data and lead to larger generalization errors on the test set.

3.4 Post-Processing

After we predict the hair volume with local orientations using our hair embedding and decoding networks, we can synthesize hair strands by growing them from the scalp, following the orientation inside the hair volume. Since we represent the 3D hair geometry in a normalized model space, the synthesized strands may not align with the head pose in the input image. If the head pose is available (e.g., via manual fitting or face landmark detection) and the segmentation/orientation can be estimated reliably from the input image, we can optionally apply several post-processing steps to further improve the modeling results, following some prior methods with similar data representation [Chai et al. 2016; Hu et al. 2015, 2017]. Starting from the input image, we first segment the pixel-level hair mask and digitize the head model [Hu et al. 2017]. Then we run a spatial deformation step as proposed in Hu et al. [2017] to fit our hairstyle to the personalized head model. Next we apply the mask-based deformation method [Chai et al. 2016] to improve alignment with the hair segmentation mask. Finally, we adopt the 2D orientation deformation and the depth estimation method, from Hu et al. [2015], to match the local details of synthesized strands to the 2D orientation map from the input image.

4 EVALUATION

In this section, we evaluate the design options of several algorithm components by comparing our method with alternative approaches.

Loss Functions. We run an ablation study on the proposed loss function in Eqn 4 by comparing it with three other functions: non-variational autoencoder with our reconstruction loss function, a state-of-the-art volumetric generative model using VAE [Brock et al. 2016] and vanilla VAE [Kingma and Welling 2014]. We refer to

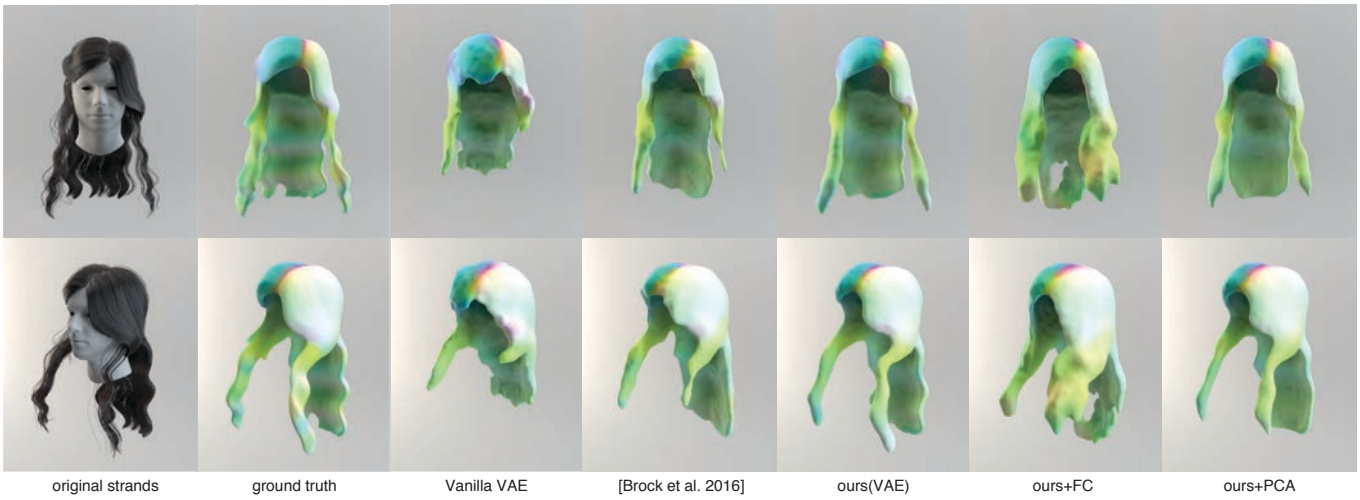

Fig. 4. Comparison of different training schemes. From left to right, we show (1) the original strands, (2) the ground-truth volumetric representation, the reconstruction results using (3) vanilla VAE [Kingma and Welling 2014], (4) a volumetric VAE [Brock et al. 2016], (5) our proposed VAE, (6) our VAE with non-linear embedding and (7) our VAE with PCA embedding respectively.

Table 4. Evaluation of prediction methods. We compare our embedding method based on Iterative Error Feedback (IEF) [Carreira et al. 2016] with direct prediction in a single shot for hair coefficients and end-to-end training where the network directly predicts the volumetric representation given an input image.

Method	IOU	Precision	Recall	L2 (flow)
IEF	**0.6487**	**0.8187**	0.7565	**0.1879**
Direct prediction	0.6346	0.8063	0.7374	0.2080
End-to-end	0.4914	0.5301	**0.8630**	0.3844

vanilla VAE as a VAE trained using naive Binary Cross-Entropy loss for occupancy fields, with the rest remaining the same. For fair comparison, we use the same architecture and same parameters, with the exception of the loss function used for training. Table 2 shows that our proposed loss function achieves the greatest intersection of union (IOU), the greatest precision and recall for reconstruction of occupancy field, and smallest error for reconstruction of flow field. Fig 4 demonstrates that our reconstruction results match the ground-truth data closely, whereas alternative approaches lead to over-smoothed output. Although we use the same flow term defined in Eqn 5 for all the comparisons in Table 2, our training scheme achieves superior reconstruction accuracy for flow field as well.

PCA Embedding. We compare our PCA embedding to a non-linear embedding with fully connected layers, which is commonly used for convolutional variational training [Brock et al. 2016]. For a fully connected VAE with a latent space of resolution $4 \times 6 \times 4 \times 64$, the output of our encoder is reshaped into a long vector and is passed to a multilayer perceptron (MLP). The dimension of each layer of the MLP is 1024, 512, 1024 and 6144 respectively. Each layer is followed by batch normalization and ReLU activation, except the layer with 512 neurons for variational training. The output of the MLP is connected to the first layer of the decoder by reshaping it back into $4 \times 6 \times 4 \times 64$. Table 3 shows that PCA embedding achieves significantly better reconstruction accuracy compared to a non-linear embedding VAE with fully connected layers and ReLU activations (the second row in Table 3). We also compare our linear PCA embedding to non-linear embedding, using the same MLP architecture as above. The MLP is trained to obtain a low dimensional embedding by minimizing the L2 reconstruction loss of the volumetric latent variables from the proposed volumetric VAE on the dataset used in the PCA embedding. Due to our limited number of training data, we have observed poor generalization in the test set (the third row in Table 3). Additionally, compared to our VAE model (the first row in Table 2), our PCA embedding (the first row in Table 3) has led to very little increase in reconstruction errors. This observation validates our low-rank assumption of the hair latent space.

Embedding Method. We compare our embedding network using IEF with two alternative approaches. The first method directly predicts parameters in a single shot [Olszewski et al. 2016; Tran et al. 2017], and the second one, end-to-end training, directly predicts the volumetric representation given an input image. The numbers in Table 4 show that compared to direct prediction, our IEF based embedding network achieves better performance in terms of IOU, precision and recall for occupancy field, as well as lower L2 error for prediction of flow field. Moreover, end-to-end training has substantially worse reconstruction accuracy for both occupancy field and flow field. These comparison results show that our two-step approach improves the stability of the training process by separately learning the generative model and the embedding network.

5 RESULTS

Single-View Hair Modeling. We show single-view 3D hairstyle modeling results from a variety of input images in Figures 1 and 5. For each image, we show the predicted occupancy field with color-coded local orientation as well as synthesized strands with manually

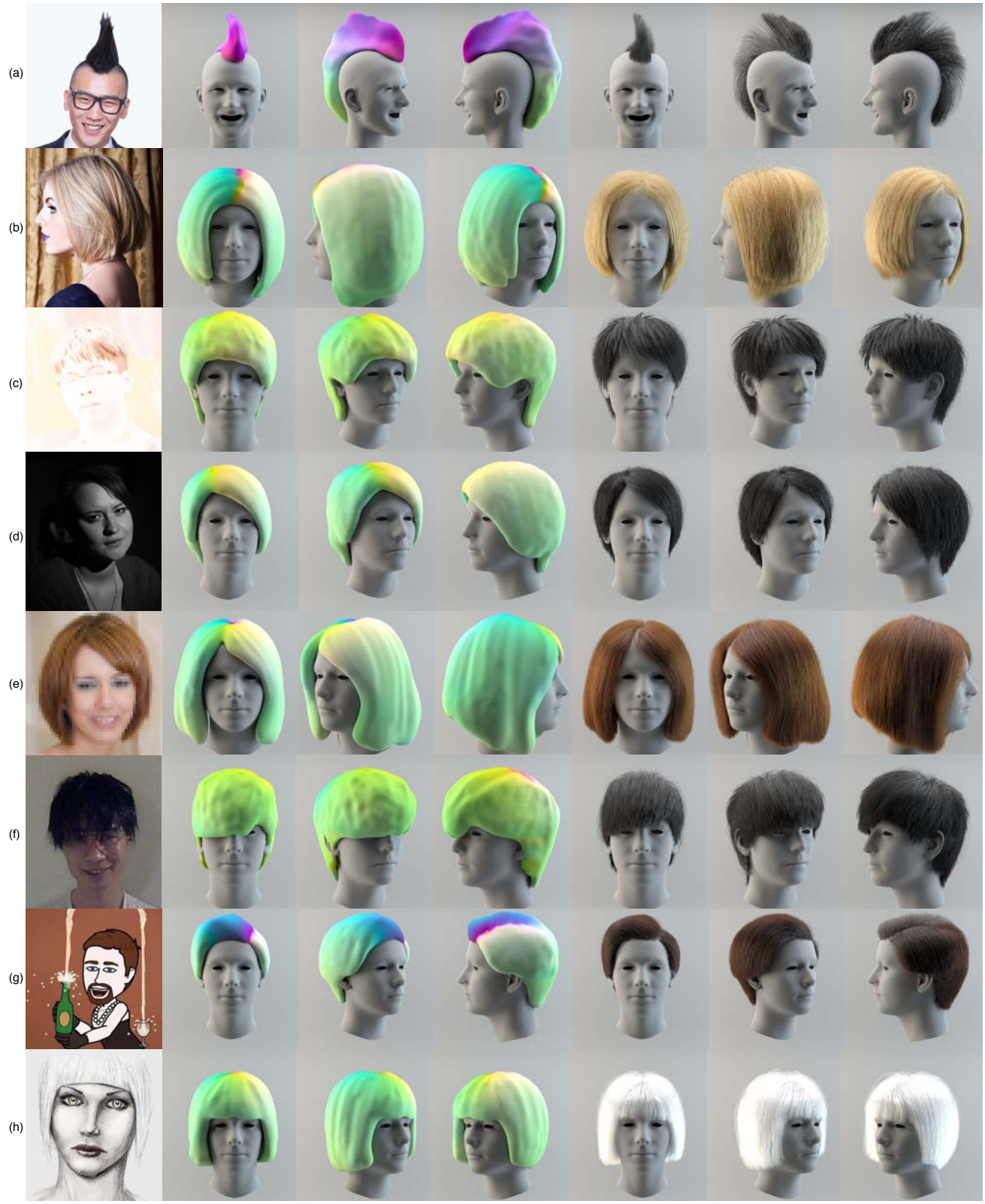

Fig. 5. Modeling results of 3D hairstyle from single input image. From left to right, we show the input image, occupancy field with color-coded local orientations predicted by our single-view hair modeling pipeline, as well as the synthesized output strands. None of these input images has been used for training of our embedding network. Original images in (a), (b), (d), (e) and (h) are courtesy of XiXinXing, Laura D'Alessandro, Dmitriy Yurchenko, Eric Dush and Ivan, respectively.

Fig. 6. Comparisons between our method with AutoHair [Chai et al. 2016]. From left to right, we show the input image, the result from AutoHair, the volumetric output of our VAE network, and our final strands. We achieve comparable results on input images of typical hairstyles (a)-(f)(l), and can generate results closer to the modeling target on more challenging examples (g)-(k). The inset images of (g) and (h) show the intermediate segmentation masks generated by AutoHair. Original images in (a), (b), (c), (d), (f), (g), (h), (i), (k) and (l) are courtesy of Joakim Berndes, Vittorio Zunino Celotto, Ernest von Rosen, Christen Harper, Bob Harris, Jason Merritt, gregoire2002, Philip Ramey Photography, Kris Krüg and Bob Harris, respectively.

specified color. Note that none of these test images are used to train our hair embedding network. Our method is end-to-end and does not require any user interactions such as manually fitting a head model and drawing guiding strokes. Moreover, several input images in Figure 5 are particularly challenging, because they are either over-exposed (the third row), have low contrast between the hair and the background (the fourth row), have low resolution (the fifth row and the sixth row), or are illustrated in a cartoon style (the last two rows). Although our training dataset for the hair embedding network only consists of examples modeled from normal headshot photographs without any extreme cases (e.g. poorly illuminated images or pictures of dogs), our method generalizes very well due to the robustness of deep image features. A typical face detector will fail to detect a human face from the third, the fifth and the sixth input images in Figure 5, which will prevent existing automatic hair modeling method [Hu et al. 2017] from generating any meaningful results. In Figure 5, only the first image can be handled by the system

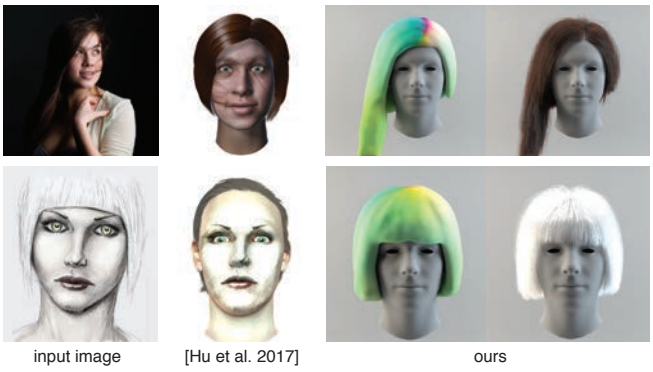

Fig. 7. Comparisons between our method with the state-of-the-art avatar digitization method [Hu et al. 2017] using the same input images. The first photo is a courtesy of Ian Norman.

proposed by Chai et al. [2016], since their algorithm requires both successful face detection and high-quality hair segmentation.

In Figure 6, we compare our method to a state-of-the-art automatic single-view hair modeling technique [Chai et al. 2016] on a variety of input images. Our results are comparable to those by Chai et al. [2016] on those less challenging input of typical hairstyles (Figure 6(a) - (f) and (l)). For these challenging cases (Figure 6(g) - (k)), we can generate more faithful modeling output, since the method of Chai et al. [2016] relies on accurate hair segmentation which can be difficult to achieve with partial occlusions or less typical hairstyles.

We also compare our method with another recent automatic avatar digitization method [Hu et al. 2017] in Figure 7. Their hair attribute classifier can successfully identify the long hairstyle for the first image, but fails to retrieve a proper hairstyle from the database because the hair segmentation is not accurate enough. For the second input image in Figure 7, their method generates a less faithful result because the classifier cannot correctly identify the target hairstyle as "with fringe".

In all of our results, we have only applied the post-processing step (Section 3.4) to the top-left example in Figure 1, the first one in Figure 5 and all those in Figure 6. All the other results are generated by growing strands directly from the fields predicted by our network.

Hair Interpolation. Our compact represention of latent space for 3D hairstyles can be easily applied to hair interpolation. Given multiple input hairstyles and the normalized weights for interpolation, we first compute the corresponding hair coefficients of PCA embedding in the latent space for each hairstyle. Then we obtain the interpolated PCA coefficients for the output by averaging the coefficients of input hairstyles based on the weights. Finally we generate the interpolated hairstyle via our decoder network.

In Figure 8, we show interpolation results of multiple hairstyles. The four input hairstyles are shown at the corners. All the interpolation results are obtained by bi-linearly interpolating the hair coefficients of PCA embedding computed from the four input hairstyles. We also compare our hairstyle interpolation results to the output using a state-of-the-art method [Weng et al. 2013]. As

Fig. 8. Interpolation results of multiple hairstyles. The four input hairstyles are shown at the corners while all the interpolation results are shown in-between based on bi-linear interpolation weights.

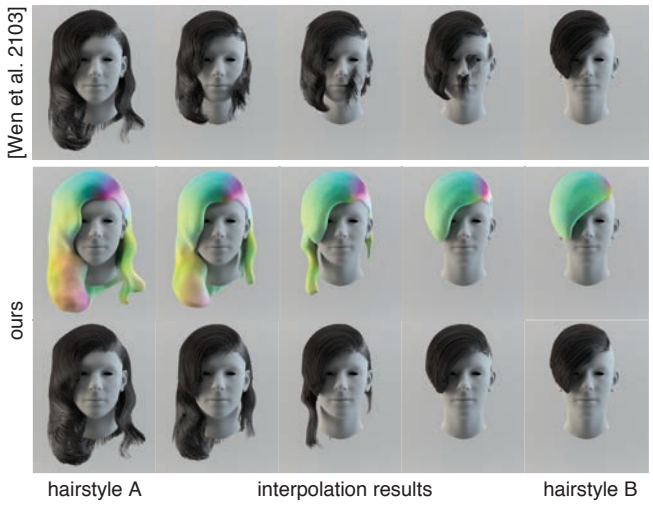

Fig. 9. Comparison between direct interpolation of hair strands [Weng et al. 2013] and our latent space interpolation results.

shown in Figure 9, our compact representation of hair latent space leads to much more plausible interpolation results from the two input hairstyles with drastically different structures.

Memory and Timing Statistics. Our hair model takes about 14MB to store in main memory, including PCA embedding and weights of decoders for both occupancy and flow fields. The training of our VAE model and hair embedding network takes about eight hours and one day respectively. The prediction of PCA coefficients takes less than one second for an input image of resolution 256×256 while decoding into volumetric representation of occupancy and flow fields only takes about two milliseconds on GPU. The generation of final strands from occupancy and flow fields takes $0.7 \sim 1.7$ seconds, depending on the strand lengths. All the timing statistics are measured on a single PC with an Intel Core i7 CPU, 64GB of memory, and an NVIDIA GeForce GTX 1080 Ti graphics card.

6 CONCLUSION

We have presented a fully automatic single-view 3D hair reconstruction method based on a deep learning framework that is trained end-to-end using a combination of artistically created and synthetically digitized hair models. Convolutions are made possible by converting 3D hair strands into a volumetric occupancy grid and a 3D orientation field. We also show that our volumetric variational autoencoder is highly effective in encoding the immense space of possible hairstyles into a compact feature embedding. Plausible hairstyles can be sampled and interpolated from the latent space of this VAE. We further show the effectiveness of using a PCA embedding and iterative error feedback technique to improve the hairstyle embedding network for handling difficult input images. Compared to state-of-the-art data-driven techniques, our approach is significantly faster and more robust, as we do not rely on successful image pre-processing, analysis, or database retrieval. In addition to our ability to produce hairstyles that were not included in the training data, we can also handle extremely challenging cases, such as in inputs including occluded faces, poorly-lit subjects, and stylized pictures. Due to its minimal storage requirements and superior robustness compared to existing methods, our 3D hair synthesis framework is particularly well-suited for next generation avatar digitization solutions. While we focus on the application of 3D hair digitization, we believe that our volumetric VAE-based synthesis algorithm can be extended to reconstruct a broad range of non-trivial shapes such as clothing, furry animals, and facial hair.

Limitations and Future Work. Figure 10 shows two failure cases of our current system. As with using any grid-based volumetric representation, the demand of GPU memory imposes a considerable limitation in terms of hair geometry resolution and bounding volume. As a data-driven approach, the effectiveness of our method is also determined by the available training data. Our modeling results may be biased towards examples in the dataset which are close to but different from the target hairstyles. Furthermore, while the latent space of our volumetric VAE can compactly describe the space of feasible hairstyles, there is no semantic meaning associated to each sample. For many graphics applications, it would be advantageous to provide high-level controls to a user for intuitive analysis and manipulation.

As our hair synthesis algorithm is currently limited to strand-based hairstyles, it would be worth exploring more generalized representations that can also handle very short hair, Afro hairstyles, or even more rendering efficient polystrip models [Hu et al. 2017; Yuksel et al. 2009]. In the future, we plan to explore ways to provide semantic controls for intuitive hairstyle modeling, such as learning a manifold of hairstyle space [Campbell and Kautz 2014; Umetani 2017]. Another interesting direction is to investigate methods of inference for high-frequency details of a greater variety of hair structures, such as those found in curly and voluminous hairstyles.

Fig. 10. Failure cases. Our method cannot faithfully handle hairstyles of fine-scale details smaller than the grid resolution, or extremely long strands outside the uniformly sampled grids in model space. The photos are courtesy of hairstopandshop.com and Hairfreaky Long Hair.

ACKNOWLEDGMENTS

We would like to thank Menglei Chai for the comparisons, as well as Aletta Hiemstra and Carrie Sun for proofreading 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

Dragomir Anguelov, Praveen Srinivasan, Daphne Koller, Sebastian Thrun, Jim Rodgers, and James Davis. 2005. SCAPE: Shape Completion and Animation of People. *ACM Trans. Graph.* 24, 3 (2005), 408–416.

Thabo Beeler, Bernd Bickel, Gioacchino Noris, Paul Beardsley, Steve Marschner, Robert W. Sumner, and Markus Gross. 2012. Coupled 3D Reconstruction of Sparse Facial Hair and Skin. *ACM Trans. Graph.* 31, 4 (2012), 117:1–117:10.

Yoshua Bengio et al. 2009. Learning deep architectures for AI. *Foundations and trends® in Machine Learning* 2, 1 (2009), 1–127.

Volker Blanz and Thomas Vetter. 1999. A Morphable Model for the Synthesis of 3D Faces. In *SIGGRAPH '99*. 187–194.

Andrew Brock, Theodore Lim, James M Ritchie, and Nick Weston. 2016. Generative and Discriminative Voxel Modeling with Convolutional Neural Networks. In *3D Deep Learning Workshop, Advances in neural information processing systems (NIPS)*. 1–9.

Neill D. F. Campbell and Jan Kautz. 2014. Learning a Manifold of Fonts. *ACM Trans. Graph.* 33, 4 (2014), 91:1–91:11.

Joao Carreira, Pulkit Agrawal, Katerina Fragkiadaki, and Jitendra Malik. 2016. Human pose estimation with iterative error feedback. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 4733–4742.

Menglei Chai, Linjie Luo, Kalyan Sunkavalli, Nathan Carr, Sunil Hadap, and Kun Zhou. 2015. High-quality hair modeling from a single portrait photo. *ACM Trans. Graph.* 34, 6 (2015), 204:1–204:10.

Menglei Chai, Tianjia Shao, Hongzhi Wu, Yanlin Weng, and Kun Zhou. 2016. Autohair: Fully automatic hair modeling from a single image. *ACM Trans. Graph.* 35, 4 (2016), 116:1–116:12.

Menglei Chai, Lvdi Wang, Yanlin Weng, Xiaogang Jin, and Kun Zhou. 2013. Dynamic hair manipulation in images and videos. *ACM Trans. Graph.* 32, 4 (2013), 75.

Menglei Chai, Lvdi Wang, Yanlin Weng, Yizhou Yu, Baining Guo, and Kun Zhou. 2012. Single-view hair modeling for portrait manipulation. *ACM Trans. Graph.* 31, 4 (2012), 116:1–116:8.

Byoungwon Choe and Hyeong-Seok Ko. 2005. A statistical wisp model and pseudophysical approaches for interactive hairstyle generation. *IEEE Transactions on Visualization and Computer Graphics* 11, 2 (2005), 160–170.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. 2009. Imagenet: A large-scale hierarchical image database. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 248–255.

Jose I Echevarria, Derek Bradley, Diego Gutierrez, and Thabo Beeler. 2014. Capturing and stylizing hair for 3D fabrication. *ACM Trans. Graph.* 33, 4 (2014), 125.

FaceUnity. 2017. http://www.faceunity.com/p2a-demo.mp4.

Hongbo Fu, Yichen Wei, Chiew-Lan Tai, and Long Quan. 2007. Sketching hairstyles. In *Proceedings of the 4th Eurographics Workshop on Sketch-based Interfaces and Modeling*. 31–36.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 770–778.

Tomas Lay Herrera, Arno Zinke, and Andreas Weber. 2012. Lighting hair from the inside: A thermal approach to hair reconstruction. *ACM Trans. Graph.* 31, 6 (2012), 146:1–146:9.

Liwen Hu, Chongyang Ma, Linjie Luo, and Hao Li. 2014a. Robust hair capture using simulated examples. *ACM Trans. Graph.* 33, 4 (2014), 126:1–126:10.

Liwen Hu, Chongyang Ma, Linjie Luo, and Hao Li. 2015. Single-view hair modeling using a hairstyle database. *ACM Trans. Graph.* 34, 4 (2015), 125:1–125:9.

Liwen Hu, Chongyang Ma, Linjie Luo, Li-Yi Wei, and Hao Li. 2014b. Capturing braided hairstyles. *ACM Trans. Graph.* 33, 6 (2014), 225:1–225:9.

Liwen Hu, Shunsuke Saito, Lingyu Wei, Koki Nagano, Jaewoo Seo, Jens Fursund, Iman Sadeghi, Carrie Sun, Yen-Chun Chen, and Hao Li. 2017. Avatar Digitization from a Single Image for Real-time Rendering. *ACM Trans. Graph.* 36, 6 (2017), 195:1–195:14.

Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros. 2017. Image-to-image translation with conditional adversarial networks. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*. 5967–5976.

itSeez3D: Avatar SDK. 2017. https://avatarsdk.com.

Aaron S Jackson, Adrian Bulat, Vasileios Argyriou, and Georgios Tzimiropoulos. 2017. Large Pose 3D Face Reconstruction from a Single Image via Direct Volumetric CNN Regression. In *Proceedings of International Conference on Computer Vision*. 1031–1039.

Wenzel Jakob, Jonathan T Moon, and Steve Marschner. 2009. Capturing hair assemblies fiber by fiber. *ACM Trans. Graph.* 28, 5 (2009), 164:1–164:9.

Angjoo Kanazawa, Michael J. Black, David W. Jacobs, and Jitendra Malik. 2018. End-to-end Recovery of Human Shape and Pose. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 7122–7131.

Tae-Yong Kim and Ulrich Neumann. 2002. Interactive Multiresolution Hair Modeling and Editing. *ACM Trans. Graph.* 21, 3 (2002), 620–629.

Diederik Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *Proceedings of International Conference on Learning Representations (ICLR)*.

Diederik P Kingma and Max Welling. 2014. Auto-encoding variational bayes. In *Proceedings of International Conference on Learning Representations (ICLR)*.

Hao Li, Laura Trutoiu, Kyle Olszewski, Lingyu Wei, Tristan Trutna, Pei-Lun Hsieh, Aaron Nicholls, and Chongyang Ma. 2015. Facial Performance Sensing Head-Mounted Display. *ACM Trans. Graph.* 34, 4 (2015), 47:1–47:9.

Matthew Loper, Naureen Mahmood, Javier Romero, Gerard Pons-Moll, and Michael J. Black. 2015. SMPL: A Skinned Multi-Person Linear Model. *ACM Trans. Graph.* 34, 6 (2015), 248:1–248:16.

Linjie Luo, Hao Li, and Szymon Rusinkiewicz. 2013. Structure-aware hair capture. *ACM Trans. Graph.* 32, 4 (2013), 76:1–76:12.

D. Maturana and S. Scherer. 2015. VoxNet: A 3D Convolutional Neural Network for real-time object recognition. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 922–928.

Myidol. 2017. http://en.faceii.com/.

Kyle Olszewski, Joseph J. Lim, Shunsuke Saito, and Hao Li. 2016. High-Fidelity Facial and Speech Animation for VR HMDs. *ACM Trans. Graph.* 35, 6 (2016), 221:1–221:14.

Sylvain Paris, Hector M Briceño, and François X Sillion. 2004. Capture of hair geometry from multiple images. *ACM Trans. Graph.* 23, 3 (2004), 712–719.

Sylvain Paris, Will Chang, Oleg I Kozhushnyan, Wojciech Jarosz, Wojciech Matusik, Matthias Zwicker, and Frédo Durand. 2008. Hair photobooth: geometric and photometric acquisition of real hairstyles. *ACM Trans. Graph.* 27, 3 (2008), 30:1–30:9.

Pinscreen. 2017. http://www.pinscreen.com.

Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. 2017a. PointNet: Deep learning on point sets for 3d classification and segmentation. In *IEEE Conference on Computer Vision and Pattern Recognition*. 77–85.

Charles R Qi, Hao Su, Matthias Nießner, Angela Dai, Mengyuan Yan, and Leonidas J Guibas. 2016. Volumetric and Multi-View CNNs for Object Classification on 3D Data. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*. 5648–5656.

Charles R Qi, Li Yi, Hao Su, and Leonidas J Guibas. 2017b. PointNet++: Deep Hierarchical Feature Learning on Point Sets in a Metric Space. In *Advances in Neural Information Processing Systems*. 5099–5108.

Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. 2014. Stochastic backpropagation and approximate inference in deep generative models. In *Proceedings of International Conference on International Conference on Machine Learning (ICML)*. 1278–1286.

David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. 1985. *Learning internal representations by error propagation.* Technical Report. California Univ San Diego La Jolla Inst for Cognitive Science.

Shunsuke Saito, Tianye Li, and Hao Li. 2016. Real-Time Facial Segmentation and Performance Capture from RGB Input. In *Proceedings of the European Conference on Computer Vision*. 244–261.

Hang Su, Subhransu Maji, Evangelos Kalogerakis, and Erik Learned-Miller. 2015. Multi-view Convolutional Neural Networks for 3D Shape Recognition. In *Proceedings of the IEEE International Conference on Computer Vision*. 945–953.

Qingyang Tan, Lin Gao, Yu-Kun Lai, and Shihong Xia. 2018. Variational Autoencoders for Deforming 3D Mesh Models. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*. 5841–5850.

Justus Thies, Michael Zollhöfer, Marc Stamminger, Christian Theobalt, and Matthias Nießner. 2018. Facevr: Real-time facial reenactment and eye gaze control in virtual reality. *ACM Trans. Graph.* 37, 2 (2018), 25:1–25:15.

Anh Tuan Tran, Tal Hassner, Iacopo Masi, and Gerard Medioni. 2017. Regressing Robust and Discriminative 3D Morphable Models with a very Deep Neural Network. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*. 1493–1502.

Nobuyuki Umetani. 2017. Exploring Generative 3D Shapes Using Autoencoder Networks. In *SIGGRAPH Asia 2017 Technical Briefs*. 24:1–24:4.

Lvdi Wang, Yizhou Yu, Kun Zhou, and Baining Guo. 2009. Example-based Hair Geometry Synthesis. *ACM Trans. Graph.* 28, 3 (2009), 56:1–56:9.

Kelly Ward, Florence Bertails, Tae-Yong Kim, Stephen R Marschner, Marie-Paule Cani, and Ming C Lin. 2007. A survey on hair modeling: Styling, simulation, and rendering. *IEEE Transactions on Visualization and Computer Graphics* 13, 2 (2007), 213–234.

Yichen Wei, Eyal Ofek, Long Quan, and Heung-Yeung Shum. 2005. Modeling Hair from Multiple Views. *ACM Trans. Graph.* 24, 3 (2005), 816–820.

Yanlin Weng, Lvdi Wang, Xiao Li, Menglei Chai, and Kun Zhou. 2013. Hair interpolation for portrait morphing. *Computer Graphics Forum* 32, 7 (2013), 79–84.

Jamie Wither, Florence Bertails, and Marie-Paule Cani. 2007. Realistic hair from a sketch. In *IEEE International Conference on Shape Modeling and Applications*. 33–42.

Zhirong Wu, Shuran Song, Aditya Khosla, Fisher Yu, Linguang Zhang, Xiaoou Tang, and Jianxiong Xiao. 2015. 3D ShapeNets: A deep representation for volumetric shapes. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*. 1912–1920.

Zexiang Xu, Hsiang-Tao Wu, Lvdi Wang, Changxi Zheng, Xin Tong, and Yue Qi. 2014. Dynamic Hair Capture Using Spacetime Optimization. *ACM Trans. Graph.* 33, 6 (2014), 224:1–224:11.

Xuan Yu, Zhan Yu, Xiaogang Chen, and Jingyi Yu. 2014. A hybrid image-CAD based system for modeling realistic hairstyles. In *Proceedings of the ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games (I3D)*. 63–70.

Cem Yuksel, Scott Schaefer, and John Keyser. 2009. Hair meshes. *ACM Trans. Graph.* 28, 5 (2009), 166:1–166:7.

M Ersin Yumer and Niloy J Mitra. 2016. Learning Semantic Deformation Flows with 3D Convolutional Networks. In *Proceedings of the European Conference on Computer Vision*. 294–311.

Meng Zhang, Menglei Chai, Hongzhi Wu, Hao Yang, and Kun Zhou. 2017. A data-driven approach to four-view image-based hair modeling. *ACM Trans. Graph.* 36, 4 (2017), 156:1–156:11.

Xiangyu Zhang, Jianhua Zou, Kaiming He, and Jian Sun. 2016. Accelerating very deep convolutional networks for classification and detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 38, 10 (2016), 1943–1955.

Yi Zhou, Liwen Hu, Jun Xin, Weikai Chen, Han-Wei Kung, Xin Tong, and Hao Li. 2018. HairNet: Single-View Hair Reconstruction using Convolutional Neural Networks. In *Proceedings of the European Conference on Computer Vision*. 235–251.