

Avatar Digitization From a Single Image For Real-Time Rendering

LIWEN HU*, Pinscreen, University of Southern California

SHUNSUKE SAITO*, Pinscreen, University of Southern California

LINGYU WEI*, Pinscreen, University of Southern California

KOKI NAGANO, Pinscreen

JAEWOO SEO, Pinscreen

JENS FURSUND, Pinscreen

IMAN SADEGHI, Pinscreen

CARRIE SUN, Pinscreen

YEN-CHUN CHEN, Pinscreen

HAO LI, Pinscreen, University of Southern California, USC Institute for Creative Technologies

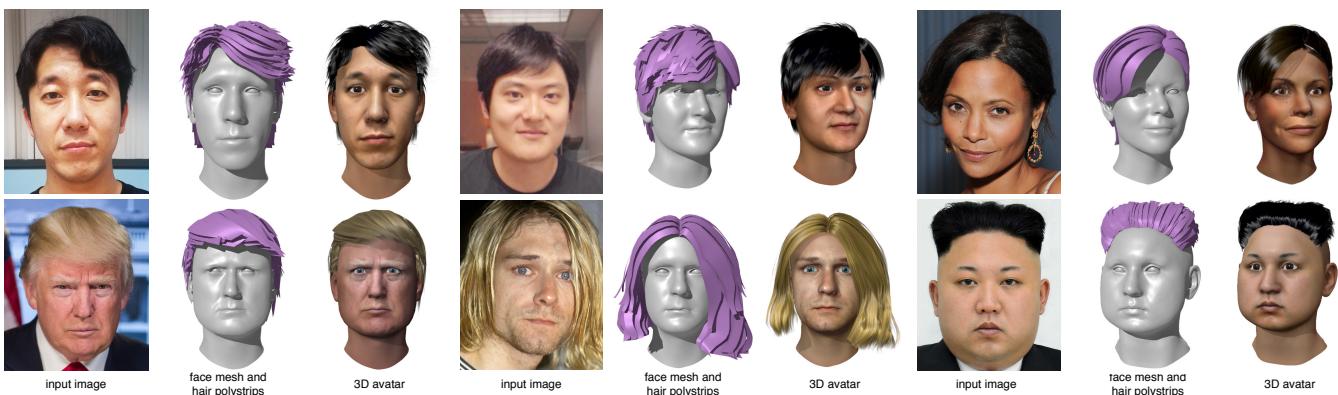


Fig. 1. We introduce an end-to-end framework for modeling a complete 3D avatar from a single input image for real-time rendering. We infer fully rigged textured faces models and polygonal strips for hair. Our flexible and efficient mesh-based hair representation is suitable for a wide range of hairstyles and can be readily integrated into existing real-time game engines. All of the illustrations are rendered in realtime in Unity. President Trump's picture is obtained from whitehouse.gov and Kim Jong-un's photograph was published in the Rodong Sinmun. The other celebrity pictures are used with permission from Getty Images.

We present a fully automatic framework that digitizes a complete 3D head with hair from a single unconstrained image. Our system offers a practical and consumer-friendly end-to-end solution for avatar personalization in gaming and social VR applications. The reconstructed models include secondary components (eyes, teeth, tongue, and gums) and provide animation-friendly blendshapes and joint-based rigs. While the generated face is a high-quality textured mesh, we propose a versatile and efficient polygonal strips (polystrips) representation for the hair. Polystrips are suitable for an extremely wide range of hairstyles and textures and are compatible with existing game engines for real-time rendering. In addition to integrating state-of-the-art advances in facial shape modeling and appearance inference, we propose a novel single-view hair generation pipeline, based on

3D-model and texture retrieval, shape refinement, and polystrip patching optimization. The performance of our hairstyle retrieval is enhanced using a deep convolutional neural network for semantic hair attribute classification. Our generated models are visually comparable to state-of-the-art game characters designed by professional artists. For real-time settings, we demonstrate the flexibility of polystrips in handling hairstyle variations, as opposed to conventional strand-based representations. We further show the effectiveness of our approach on a large number of images taken in the wild, and how compelling avatars can be easily created by anyone.

CCS Concepts: • Computing methodologies → Mesh geometry models; • Theory of computation → Machine learning theory;

Additional Key Words and Phrases: dynamic avatar, face, hair, digitization, modeling, rigging, polystrip, texture synthesis, data-driven, deep learning, deep convolutional neural network

ACM Reference Format:

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, Article 1 (November 2017), 14 pages.
<https://doi.org/10.1145/3130800.3130887>

* indicates equal contribution

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

© 2017 Association for Computing Machinery.
0730-0301/2017/11-ART1 \$15.00
<https://doi.org/10.1145/3130800.3130887>

1 INTRODUCTION

The onset of virtual reality (VR) and its entertainment applications have highlighted how valuable and captivating the immersion of alternate universes can be. VR and its democratization have the potential to revolutionize 3D face-to-face communication and social interactions through compelling digital embodiments of ourselves, as demonstrated lately with the help of VR head mounted displays with facial sensing capabilities [Li et al. 2015; Olszewski et al. 2016; Thies et al. 2016b] or voice-driven technology demonstrated at Oculus Connect 3. In addition to enabling personalized gaming experiences, faithfully individualized 3D avatars could facilitate natural telepresence and interactions between remote participants in virtual worlds, and potentially, one day, displace physical travels. Meanwhile, companies such as Facebook and Snap are popularizing the use of augmented reality filters to alter selfie videos and emerging tech startups such as Pinscreen [2017], FaceUnity [2017], Loom.ai [2017], and itSeez3D [2017], are exploring the automatic creation of 3D avatars for virtual chatting applications.

Recent progress in data-driven methods and deep learning research have catalyzed the development of high-quality 3D face modeling techniques from a single image [Cao et al. 2014b; Saito et al. 2017; Thies et al. 2016a]. Even the generation of realistic strand-level hair models is possible from an image fully automatically [Chai et al. 2016]. However, despite efforts in real-time simulation [Chai et al. 2014], strand-based representations are still very difficult to integrate into game environments due to their rendering and simulation complexity. Furthermore, strands are not efficient representations for short hairstyles and ones with highly stochastic structures, such as for curly hair. Cao et al. [2016] have recently introduced a system that uses a versatile image-based mesh representation, but it requires the usage of multiple photographs and manual intervention, and the volumetric structure of hair is not captured. Despite substantial advances in making avatar creation as easy as possible, the barriers to entry are still too high for commodity user adoption.

In this paper, we present the first automatic framework that generates a complete 3D avatar from a single unconstrained image, using high-quality optimized polygonal strips (polystrips or poly cards) for real-time hair rendering. By eliminating the need of multiple photographs and a controlled capture environment, we provide a practical and consumer-friendly solution for digitizing ourselves or others, such as celebrities, from any photograph. Our digitized models are fully rigged with intuitive animation controls such as blendshapes and joint-based skeletons, and can be readily integrated into existing game engines.

We first address the challenge of predicting the 3D shape and appearance of entire heads from partially visible 2D input data. We carefully integrate multiple cutting edge techniques into a comprehensive facial digitization framework. An accurate 3D face model is estimated using a modified dense analysis-through-synthesis approach [Thies et al. 2016a] with visibility constraints on a pre-segmented input image, which is obtained from a convolutional neural network for segmentation [Saito et al. 2016]. Subsequently, a complete high-quality facial texture is synthesized using a deep learning-based inference technique introduced by Saito et al. [2017].

While a straightforward incorporation of an existing single-view hair modeling technique is possible [Chai et al. 2016; Hu et al. 2015],

we focus on a method that produces highly efficient polystrips rather than strands. The use of polystrips is particularly suitable for real-time rendering and integration with existing game engines. For games, hair models rarely exceed 100K triangles, especially when a large number of characters need to be on screen at any given time. With appropriate textures and alpha masks, this representation also supports for a much larger variety of hairstyles than strands. Though widely used in cutting edge games (e.g., Uncharted 4), the creation of visually compelling hair polystrips is typically associated with a tedious and time-consuming modeling and texture painting process by skilled artists.

We introduce an automatic hair digitization pipeline for modeling polystrip-based hairstyles. Critical to reconstructing high-quality hair meshes are convincing shapes and structures, such as fringes, which are laid out manually by a modeler. We propose a deep learning-based framework to first extract semantical hair attributes that characterizes the input hairstyle. A tractable subset of candidate hairstyles with compatible traits is then selected from a large hair model database. A closest hairstyle is then retrieved from this hairstyle collection and refined to match the input. Our deep neural network also identifies hair appearance attributes, that describe the local structure and styling with the corresponding shading properties. Though a small set of local hairstyle textures can generalize well for different hair models, the associated alpha masks often introduce severe transparency artifacts and alter the overall look of the hair model significantly. In production, the crafting of hair polystrips typically involves a complex iterative design process of mesh adjustments, UV layout, texturing, as well as polystrip duplication and perturbation. To this end, we develop a novel iterative optimization technique for polystrip patching, placement, and shape refinement based on a scalp visibility metric. For visually pleasing animations, we also rig our hair model to the head skeleton using inverse distance skinning [Jacobson et al. 2014].

We show the effectiveness of our approach on a wide range of subjects and hairstyles, and also demonstrate compelling animations of our avatars with simulated hair dynamics. The output quality of our framework is comparable to state-of-the-art game characters, as well as cutting-edge avatar modeling systems that are based on multiple input photographs [Cao et al. 2016; Ichim et al. 2015]. The proposed pipeline also produces superior results than existing commercial single view-based solutions such as Loom.ai and itSeez3D.

Contributions:

- We present a fully automatic framework for complete 3D avatar modeling and rigging, from a single unconstrained image that is suitable for real-time rendering in game and VR environments. Our facial digitization pipeline integrates the latest advances in facial segmentation, shape modeling, and high-fidelity appearance inference.
- We develop a new single-view hair digitization pipeline that produces highly efficient and versatile polystrip models. Our system captures both hair shape and appearance properties.
- To ensure high-quality output hair meshes, we present a hair attributes classification framework based on deep learning. Furthermore, an iterative optimization algorithm for polystrip patching is introduced to ensure a flawless scalp coverage and correct hair shape likeness to the input.

2 RELATED WORK

Facial Modeling and Capture. Over the past two decades, a great amount of research has been dedicated to the modeling and animation of digital faces. We refer to [Parke and Waters 2008] for a comprehensive introduction and overview. Though artist-friendly digital modeling tools have significantly evolved over the years, 3D scanning and performance capture technologies provide an attractive way to scale content creation and improve realism through accurate measurements from the physical world. While expensive and difficult to deploy, sophisticated 3D facial capture systems [Beeler et al. 2010, 2011; Bradley et al. 2010; Debevec et al. 2000; Ghosh et al. 2011; Li et al. 2009; Ma et al. 2007; Weise et al. 2009] are widely adopted in high-end production and have proven to be a critical component for creating photoreal digital actors. Different rigging techniques such as joint-based skeletons, blendshapes [Li et al. 2010; von der Pahlen et al. 2014], or muscle-based systems [Sifakis et al. 2005; Terzopoulos and Waters 1990] have been introduced to ensure intuitive control in facial animation and high-fidelity retargeting for performance capture. Dedicated systems for capture, rigging, and animation have also emerged for the treatment of secondary components such as eyes [Bérard et al. 2016; Miller and Pinskiy 2009], lips [Garrido et al. 2016b], and teeth [Wu et al. 2016]. Despite high-fidelity output, these capture and modeling systems are too complex for mainstream adoption.

The PCA-based linear face models of [Blanz and Vetter 1999] have laid the foundations for the modern treatment of image-based 3D face modeling, with extensions to multi-view stereo [Blake et al. 2007], large-scale internet pictures [Kemelmacher-Shlizerman 2013; Liang et al. 2016], massive 3D scan datasets [Booth et al. 2016], and the use of shading cues [Kemelmacher-Shlizerman and Basri 2011]. Blanz and Vetter have demonstrated in their original work that compelling facial shapes and appearances with consistent parameterization can be extracted reliably from a single input image. Recent progress in single-view face modeling demonstrate improved detail reconstruction [Richardson et al. 2016], component separation [Kim et al. 2017; Tewari et al. 2017], and manipulation capabilities [Shu et al. 2017] using deep convolutional neural networks. To handle facial expressions, vector spaces based on visemes and expressions have been proposed [Blanz et al. 2003], which led to the development of PCA-based multi-linear face models [Vlasic et al. 2005] and the popularization of FACS-based blendshapes [Cao et al. 2014b]. The low dimensionality and effectiveness in representing faces have made linear models particularly suitable for instant 3D face modeling and robust facial performance capture in monocular settings using depth sensors [Bouaziz et al. 2013; Hsieh et al. 2015; Li et al. 2013; Weise et al. 2011, 2009], as well as RGB video [Cao et al. 2014a; Garrido et al. 2013, 2016a; Saito et al. 2016; Shi et al. 2014; Thies et al. 2016a]. When modeling a 3D face automatically from an image, sparse 2D facial landmarks [Cootes et al. 2001; Cristinacce and Cootes 2008; Saragih et al. 2011; Xiong and De la Torre 2013] are typically used for robust initialization during fitting. State-of-the-art landmark detection methods achieve impressive efficiency by using explicit shape regressions [Cao et al. 2013; Kazemi and Sullivan 2014; Ren et al. 2014].

While linear models can estimate entire head models from a single view, the resulting textures are typically crude approximations of

the subject, especially in the presence of details such as facial hair, complex skin tones, and wrinkles. In order to ensure likeness to the captured subject, existing 3D avatar creation systems often avoid the use of a purely linear appearance model, but rely on acquisitions from multiple views to build a more accurate texture map. Ichim et al. [2015] introduced a comprehensive pipeline for video-based avatar reconstruction in uncontrolled environments. They first produce a dense point cloud using multi-view stereo and then estimate a 3D face model using non-rigid registration. An integrated albedo texture map is then extracted using a combination of Poisson blending and light factorization via spherical harmonics. Their method is limited to a controlled acquisition procedure based on a semi-circular sweep of a hand-held sensor, and hair modeling is omitted. Chai et al. [2015] presented a single-view system for high-quality 2.5D depth map reconstruction of both faces and hair, using structural hair priors, silhouette, and shading cues. However, their technique is not suitable for avatars, as a full head cannot be produced nor animated. More recently, Cao et al. [2016] developed an end-to-end avatar creation system that can produce compelling face and hair models based on an image-based mesh representation. While their system can handle very large variations of hairstyles and also produce high-quality facial animations with fine-scale details, they require up to 32 input images and some manual guidance for segmentation and labeling. Instead of a controlled capture procedure with multiple photographs, we propose a fully automatic system that only needs a single image as input.

Notice that proprietary technologies for single-view avatar modeling have emerged recently in the commercial world, such as Pin-screen’s demonstration at *SIGGRAPH Real Time Live!* show [Li et al. 2017] and FaceUnity’s photo-to-avatar preview [FaceUnity 2017]. In Section 6, we compare our proposed solution with two other recent avatar creation solutions, Loom.ai [2017] and itSeez3D [2017].

Hair Modeling and Capture. Hair is an essential component of life-like avatars and CG characters. In studio settings, human hair is traditionally modeled, simulated, and rendered using sophisticated design tools [Choe and Ko 2005; Kim and Neumann 2002; Weng et al. 2013; Yuksel et al. 2009]. We refer to the survey of Ward et al. [2006] for an extensive overview. 3D hair capture techniques, analogous to those used for face capture, have been introduced to digitize hair from physical inputs. High-fidelity acquisition systems typically involve controlled recording sessions, manual assistance, and complex hardware equipments, such as multi-view stereo rigs [Beeler et al. 2012; Echevarria et al. 2014; Jakob et al. 2009; Luo et al. 2013; Paris et al. 2008] or even thermal imaging [Herrera et al. 2012].

Hu et al. [2014a] demonstrated a highly robust multi-view hair modeling approach using a data-collection of pre-simulated hair strands, which can fully eliminate the need for manual hair segmentation. Since physically simulated hair strands are used as shape priors, their method can only handle unconstrained hairstyles. The same authors later introduced a procedural method for hair patch generation [Hu et al. 2014b] to handle highly convoluted hairstyles such as braids. They also proposed a more accessible acquisition approach based on a single RGB-D camera, that is swept around the subject. Single-view hair digitization methods have been pioneered by Chai et al. [2013; 2012] but rely on high-resolution input photographs and can only produce the frontal geometry of the hair.

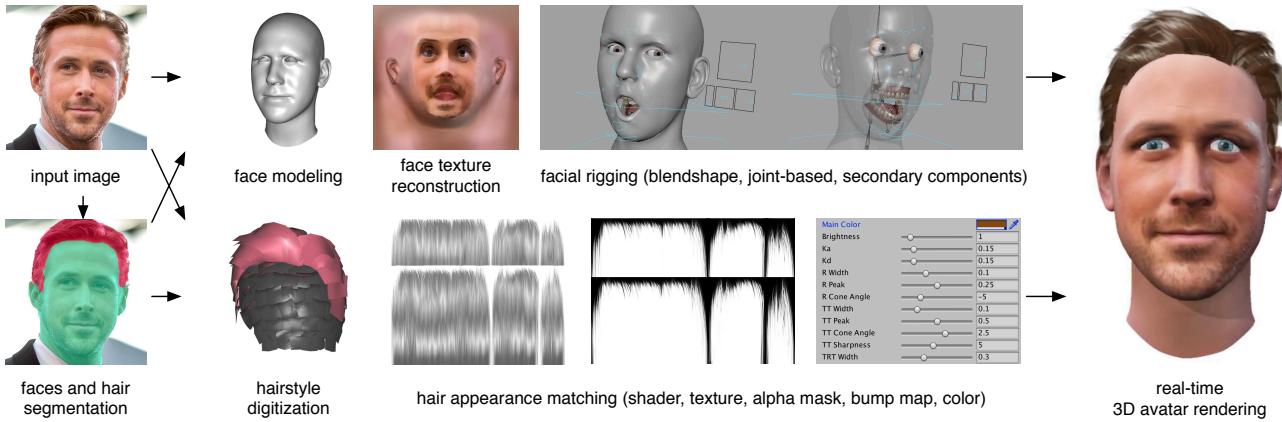


Fig. 2. Our single-view avatar creation framework is based on a pipeline that combines both complete face digitization and hair polystrip digitization—both geometry and appearance are captured. Original image courtesy of Getty Images.

A database-driven approach by Hu et al. [2015] later showed that the modeling of complete strand-level hairstyles is possible from a single image, with the help of very few user strokes as guidance. A similar, but fully automatic approach has been furthered by Chai et al. [2016] using a larger database for shape retrieval and a deep learning-technique for hair segmentation. While a wide range of high-quality hair models can be digitized, many hairstyles with multiple layers or stochastic structures—such as afros or messy hair—are difficult to capture and not suitable for strand-based representations. Furthermore, strand-based hair models are still difficult to integrate into real-time game environments, due to their complexity in real-time hair rendering and simulation. We introduce a new hair digitization framework based on highly efficient and flexible polystrips, which are widely adopted in modern games. Hair polystrips are more efficient for rendering than hair strands, and also can also achieve believable volumetric structures through textures with alpha masks and cut-off techniques as opposed to the opaque textured mesh representation used by Cao et al. [2016].

3 AVATAR MODELING FRAMEWORK

Our end-to-end pipeline for face and hair digitization is illustrated in Figure 2. An initial pre-processing step computes pixel-level segmentation of the face and hair regions. We then produce a fully rigged avatar based on textured meshes and hair polystrips from this image. We decouple the digitization of face and hair since they span entirely different spaces for shape, appearance, and deformation. While the full head topology of the face is anatomically consistent between subjects and expressions, the mesh of the hair model will be unique for each person.

Image Pre-Processing. Segmenting the face and hair regions of an input image improves the accuracy of the 3D model fitting process, as only relevant pixels are used as constraints. It also provides additional occlusion areas, that need to be completed during texture reconstruction, especially when the face is covered by hair. For the hair modeling step, the silhouette of the segmented hair region will provide important matching cues.

We adopt the real-time and automatic semantic segmentation technique of [Saito et al. 2016] which uses a two-stream deconvolution network to predict face and hair regions. This technique produces accurate and robust pixel-level segmentations for unconstrained photographs. While the original implementation is designed to process face regions, we repurpose the same convolutional neural network to segment hair. In contrast to the image pre-processing step of [Cao et al. 2016], ours is fully automatic.

To train our convolutional neural network, we collected 9269 images from the public LFW face dataset [Huang et al. 2007] and produce the corresponding binary segmentation masks for both faces and hair via Amazon Mechanical Turk (AMT) as illustrated in Figure 3. We detect the face in each image using the popular Viola-Jones face detector [2001] and normalize their positions and scales to a 128×128 image. To avoid overfitting, we augment the training dataset with random Gaussian-distributed transformation perturbations and produce 83421 images in total. The standard deviations are 10° for rotations, 5 pixels for translations, and 0.1 for scale, and the means are 0, 0, and 1.0 respectively. We further use a learning rate of 0.1, a momentum of 0.9, and weight decay of 0.0005 for the training. The optimization uses 50,000 stochastic gradient descent (SGD) iterations which take roughly 10 hours on a machine with 16GB RAM and NVIDIA GTX Titan X GPU. We refer to the work of [Saito et al. 2016] for implementation details. Once trained, the network outputs a multi-class probability map (for face and hair) from an arbitrary input image. A post-hoc inference algorithm based on dense conditional random field (CRF) [Krähenbühl and Koltun 2011] is then used to extract the resulting binary mask. Successful results and failure cases are presented in Figure 3.

Face Digitization. We first fit a PCA-based linear face model for shape and appearance to the segmented face region. Next, a variant of the efficient pixel-level analysis-through-synthesis optimization method of [Thies et al. 2016a] is adopted to solve for the PCA coefficients of the 3D face model and an initial low-frequency albedo map. We use our own artist-created head topology (front and back head) with identity shapes transferred from [Blanz and Vetter 1999]



Fig. 3. Hair segmentation training data, successful results, and failure cases.

and expressions from [Cao et al. 2014b]. A visibility constraint is incorporated into the model fitting process to improve occlusion handling and non-visible regions. A PCA-based appearance model is constructed for the textures of the full head, using artist-painted skin textures in missing regions of the original data samples. We then infer high-frequency details to the frontal face regions even if they are not visible in the capture using a feature correlation analysis approach based on deep neural networks [Saito et al. 2017]. Finally, we eliminate the expression coefficients of our linear face model to neutralize the face. The resulting model is then translated and scaled to fit the eye-balls using the average pupillary distance of an adult human of 66 mm. We then translate and scale the teeth/gum to fit pre-selected vertices of the mouth region. We ensure that these secondary components do not intersect the face using a penetration test for all the FACS expressions of our custom animation rig.

Hair Digitization. Our hair digitization pipeline produces a hair mesh model and infers appearance properties for the hair shader. We first use a state-of-the-art deep convolutional neural network based on residual learning [He et al. 2016] to extract semantic hair attributes such as hair length, level of baldness, and the existence of hairlines and fringes. These hair attributes are compared with a large hairstyle database containing artist created hair polystrip models. We then form a reduced hairstyle dataset that only contains relevant models with compatible hair attributes. We then search for the closest hairstyle to our input image based on the silhouette of its segmentation and the orientation field of the hair strands. As the retrieved hairstyle may not match the input exactly, we further perform a mesh fitting step to deform the retrieved hairstyle to the input image using the silhouette and the input orientation field. We incorporate collision handling between the deformed hair and the personalized face model to avoid hair meshes intersecting the face mesh. The classification network for hair attribute classification also identifies hair appearance properties for proper rendering such as hair color, texture and alpha maps, various shader parameters, etc. Polystrip duplication is necessary, since the use of alpha masks for the hair texture can cause a loss of scalp coverage during rendering. Consequently, we iteratively identify the incomplete hair regions using multi-view visibility map and patch them with interpolated hair strips. The hair polystrips are alpha blended using an efficient rendering algorithm based on order-independent transparency with depth peeling [Bavoil and Myers 2008].

Rigging and Animation. Since our linear face model is expressed by a combination of identity and expression coefficients [Saito et al. 2017], we can easily obtain the neutral pose. Using an example-based approach, we can compute the face input’s corresponding FACS-based expressions (including high-level controls) via transfer from a generic face model [Li et al. 2010]. Our generic face is also equipped with skeleton joints based on linear blend skinning (LBS) [Parke and Waters 2008]. The face and secondary components (eyes, teeth, tongue, and gums) also possess blendshapes. Eye colors (black, brown, blue, and green) are detected using the same deep convolutional neural network used for hair attribute classification [He et al. 2016] and the appropriate texture is used. Our model consists of 71 blendshapes, and 16 joints in total. Our face rig also abstracts the low-level deformation parameters with a smaller and more intuitive set of high-level controls as well as manipulation handles. We implemented our rig in both the animation tool, Autodesk Maya, and the real-time game engine, Unity. We can rig our hair model directly with the skeleton joints of the head in order to add a minimal amount of dynamics for simple head rotations. For more complex hair dynamics, we also demonstrate a simple real-time physical simulation of our polystrip hair representation using mass-spring models with rigid body chains and hair-head collisions [Selle et al. 2008].

4 FACE DIGITIZATION

We first build a fully textured head model using a multi-linear PCA face model. Given a single unconstrained image and the corresponding segmentation mask, we compute a shape V , a low-frequency facial albedo map I , a rigid head pose (R, t), a perspective transformation $\Pi_P(V)$ with the camera intrinsic matrix P , and illumination L , together with high-frequency textures from the visible skin region. Since the extracted high-frequency texture is incomplete from a single-view, we infer the complete texture map using a facial appearance inference method based on deep neural networks [Saito et al. 2017].

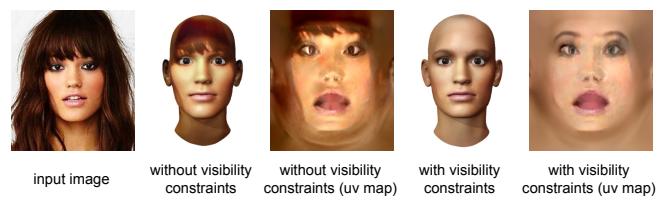


Fig. 4. Our facial modeling pipeline with visibility constraints produces plausible facial textures when there are occlusions such as hair.

3D Head Modeling. To obtain the unknown parameters $\chi = \{V, I, R, t, P, L\}$, we adopt the pipeline of [Thies et al. 2016a] which is based on morphable face models [Blanz and Vetter 1999] extended with a PCA-based facial expression model and an efficient optimization based on pixel color constraints. We further incorporate pixel-level visibility constraints using our segmentation mask obtained using the method of [Saito et al. 2016].

We use a multi-linear PCA model to represent the low-frequency facial albedo I and the facial geometry V with $n = 10,822$ vertices

and 21,510 faces:

$$\begin{aligned} V(\alpha_{id}, \alpha_{exp}) &= \bar{V} + A_{id}\alpha_{id} + A_{exp}\alpha_{exp}, \\ I(\alpha_{al}) &= \bar{I} + A_{al}\alpha_{al}. \end{aligned}$$

Here $A_{id} \in \mathbb{R}^{3n \times 40}$, $A_{exp} \in \mathbb{R}^{3n \times 40}$, and $A_{al} \in \mathbb{R}^{3n \times 40}$ are the basis of a multivariate normal distribution for identity, expression, and albedo with the corresponding mean: $\bar{V} = \bar{V}_{id} + \bar{V}_{exp} \in \mathbb{R}^{3n}$, and $\bar{I} \in \mathbb{R}^{3n}$, and the corresponding standard deviation: $\sigma_{id} \in \mathbb{R}^{40}$, $\sigma_{exp} \in \mathbb{R}^{40}$, and $\sigma_{al} \in \mathbb{R}^{40}$. A_{id} , A_{al} , \bar{V} , and \bar{I} are based on the Basel Face Model database [Payson et al. 2009] and A_{exp} is obtained from FaceWarehouse [Cao et al. 2014b]. We assume Lambertian surface reflectance and approximate the illumination using second order Spherical Harmonics (SH).

First, we detect 2D facial landmarks $f_i \in \mathcal{F}$ using the method of Kazemi et al. [Kazemi and Sullivan 2014] in order to initialize the face fitting by minimizing the following energy:

$$E_{lan}(\chi) = \frac{1}{|\mathcal{F}|} \sum_{f_i \in \mathcal{F}} \|f_i - \Pi_P(RV_i + t)\|_2^2.$$

We further refine the shape and optimize the low-frequency albedo, as well as the illumination, by minimizing the photometric difference between the input image and a synthetic face rendering. The objective function is defined as:

$$E(\chi) = w_c E_c(\chi) + w_{lan} E_{lan}(\chi) + w_{reg} E_{reg}(\chi), \quad (1)$$

with energy term weights $w_c = 1$, $w_{lan} = 10$, and $w_{reg} = 2.5 \times 10^{-5}$ for the photo-consistency term E_c , the landmark term E_{lan} , and the regularization term E_{reg} . Following [Saito et al. 2017], we also ensure that the photo-consistency term E_c is only evaluated for visible face regions:

$$E_c(\chi) = \frac{1}{|\mathcal{M}|} \sum_{p \in \mathcal{M}} \|C_{input}(p) - C_{synth}(p)\|_2,$$

where C_{input} is the input image, C_{synth} the rendered image, and $p \in \mathcal{M}$ a visibility pixel given by the facial segmentation mask. The regularization term E_{reg} is defined as:

$$E_{reg}(\chi) = \sum_{i=1}^{40} \left[\left(\frac{\alpha_{id,i}}{\sigma_{id,i}} \right)^2 + \left(\frac{\alpha_{al,i}}{\sigma_{al,i}} \right)^2 \right] + \sum_{i=1}^{40} \left(\frac{\alpha_{exp,i}}{\sigma_{exp,i}} \right)^2.$$

This term encourages the coefficients of the multi-linear model to conform a normal distribution and reduces the chance to converge into a local minimum. We use an iteratively reweighted Gauss-Newton method to minimize the objective function (1) using three levels of image pyramids. In our experiments, 30, 10, and 3 Gauss-Newton steps were sufficient for convergence from the coarsest level to the finest one. After this optimization, a high-frequency albedo texture is obtained by factoring out the shading component consisting of the illumination L and the surface normal from the input image. The resulting texture map is stored in the uv texture map and used for the high-fidelity texture inference.

Face Texture Reconstruction. After obtaining the low-frequency albedo map and a partially visible fine-scale texture, we can infer a complete high-frequency texture map, as shown in Figure 5, using a deep learning-based transfer technique and a high-resolution face database [Ma et al. 2015]. The technique has been recently introduced in [Saito et al. 2017] and is based on the concept of feature

correlation analysis using convolutional neural networks [Gatys et al. 2016]. Given an input image I and a filter response $F^l(I)$ on the layer l of a convolutional neural network, the feature correlation can be represented by a normalized Gramian matrix $G^l(I)$:

$$G^l(I) = \frac{1}{M_l} F^l(I) (F^l(I))^T$$

Saito et al. [2017] have found that high-quality facial details (e.g., pores, moles, etc.) can be captured and synthesized effectively using Gramian matrices. Let I_0 be the low-frequency texture map and I_h be the high-frequency albedo map with the corresponding visibility mask M_h . We aim to represent the desired feature correlation G_h as a convex combination of $G(I_i)$, where I_1, \dots, I_K are the high-resolution images in the texture database:

$$G_h^l = \sum_k w_k G^l(I_k), \forall l \quad s.t. \quad \sum_{k=1}^K w_k = 1.$$

We compute an optimal blending weight $\{w_k\}$ by minimizing the difference between the feature correlation of the partial high-frequency texture I_h and the convex combination of the feature correlations in the database under the same visibility. This is formulated as the following problem:

$$\begin{array}{ll} \min_w & \sum_l \left\| \sum_k w_k G_M^l(I_k, M_h) - G_M^l(I_h, M_h) \right\|_F \\ \text{s.t.} & \sum_{k=1}^K w_k = 1 \\ & w_k \geq 0 \quad \forall k \in \{1, \dots, K\} \end{array}, \quad (2)$$

where $G_M(I, M)$ is the Gramian Matrix computed from only the masked region M . This allows us to transfer multi-scale features of partially visible skin details to the complete texture. We refer to [Saito et al. 2017] for more detail.

Once the desired G_h is computed, we update the albedo map I so that the resulting correlation $G(I)$ is similar to G_h , while preserving the low frequency spatial information $F^l(I_0)$ (i.e., position of eye brows, mouth, nose, and eyes):

$$\min_I \sum_{l \in L_F} \left\| F^l(I) - F^l(I_0) \right\|_F^2 + \alpha \sum_{l \in L_G} \left\| G^l(I) - G_h^l \right\|_F^2, \quad (3)$$

where L_G is a set of high-frequency preserving layers and L_F a set of low-frequency preserving layers in VGG-19 [Simonyan and Zisserman 2014]. A weight α balances the influence of high frequency and low frequency and $\alpha = 2000$ is used for all our experiments. Following Gatys et al. [2016], we solve Equation 3 using an L-BFGS solver. Since only frontal faces are available in the database, we can only enhance frontal face regions. To obtain a complete texture, we combine the results with the PCA-based low-frequency textures of the back of the head using Poisson blending [Pérez et al. 2003].

Secondary Components. To enhance the realism of the reconstructed avatar, we insert template models for eyes, teeth, gums, and tongue into the reconstructed head model. The reconstructed face model is rescaled and translated to fit a standardized pair of eye balls so that each avatar is aligned as to avoid scale ambiguity during the single-view reconstruction. The mouth-related template models are aligned based on pre-selected vertices on the facial template model. After the initial alignment, we test for intersections between the face and the secondary components for each activated blendshape

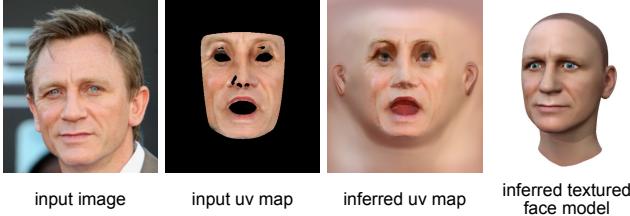


Fig. 5. We produce a complete and high-fidelity texture map from a partially visible and low resolution subject using a deep learning-based inference technique. Original image courtesy of Getty Images.

expression. The secondary models for the mouth region are then translated by the minimal offset where no intersection is present. The eye color texture (black, brown, green, blue) is computed using a similar convolutional neural network for semantic attribute inference as the one used for hair color classification. The input to this network is a cropped image of the face region based on the bounding box around the 2D landmarks from [Kazemi and Sullivan 2014], where non-face regions are set to black and the image centered between the two eyes.

5 HAIR DIGITIZATION

Hairstyle Database. Starting from the *USC-HairSalon* database for 3D hairstyles, introduced in [Hu et al. 2015], and 89 additional artist created models, we align all the hairstyle samples to the PCA mean head model \bar{V} used in Section 4. Inspired by [Chai et al. 2015], we also increase the number of samples in our database using a combinatorial process, which is necessary to span a sufficiently large variation of hairstyles. While the online model generation approach of [Hu et al. 2015] is less memory consuming, it requires some level of user interaction.

To extend the number of models, we first group each sample of the *USC-HairSalon* database into 5 clusters via k -means clustering using the root positions and the strand shapes as in [Wang et al. 2009]. Next, for every pair of hairstyles, we randomly pick a pair of strands among the cluster centroids and construct a new hairstyle using these two sampled strands as a guide using the volumetric combination method introduced in [Hu et al. 2015]. We further augment our database by flipping each hairstyle w.r.t. the x -axis plane, forming a total of 100,000 hairstyles.

For each hair model, the set of all particles forms the outer surface of the entire hair by considering each hair strand as a chain of particles. This surface can be constructed using a signed distance field obtained by volumetric points samples [Zhu and Bridson 2005]. By using the surface normal of this mesh, we compose close and nearly parallel hair strands into a hair polystrip, which is a parametric piece-wise linear patch. This thin surface structure can carry realistic looking textures that provide additional variations of hair, such as curls, crossings, or thinner tips. Additionally, the transparency of the texture allows us to see through the overlay of different polystrips and provide an efficient way to achieve volumetric hair renderings.

Luo et al. [2013] proposed a method to group short hair segments into a ribbon structure. Adopting a similar approach, we start from the longest hair strand in the hairstyle as the center strand of the

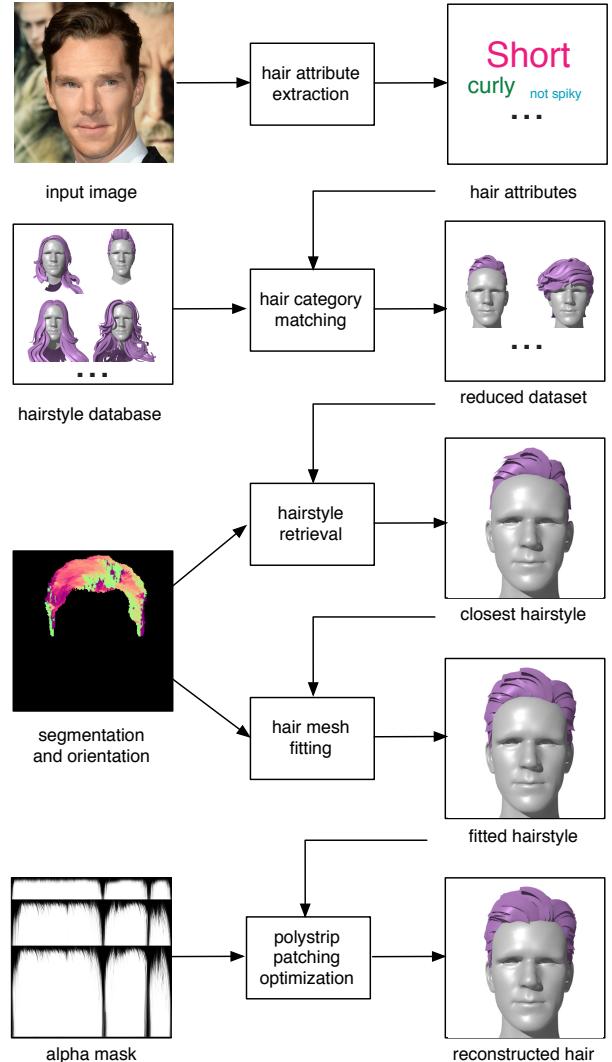


Fig. 6. Our hair mesh digitization pipeline. Original image courtesy of Getty Images.

polystrip. By associating the normal of each vertex on the strand to the closest point on the hair surface, we can expand the center strand on both sides of the binormal as well as its opposite direction. We compute the coverage of all hair strands by the current polystrip, and continue to expand the polystrip until no more strands are covered. Once a polystrip is generated, we remove all the covered strands in the hairstyle, and reinitiate process from the longest strand in the remaining hair strand subset. Finally, we obtain a complete hair polystrip model, once all the hair strands are removed from the hairstyle. We refer to [Luo et al. 2013] for more details.

Hair Attribute Classification. We use 40K images from the CelebA dataset [Liu et al. 2015] with various hairstyles and collect their hair attributes using AMT (see Table 1 for the list of hair attributes). Similarly, we manually label all the hair models in our database

using high level semantic attributes. We also actively ensure that we have roughly the same quantity of images for each attribute by resampling the training data.

These annotations are then fed into a state-of-the-art classification network, ResNet [He et al. 2016], to train multiple classifiers predicting each hair attribute given an input image. We use the 50-layer ResNet pre-trained with ImageNet [Deng et al. 2009], and fine-tune it using our training data under learning rate 10^{-4} , weight decay 10^{-4} , momentum 0.9, batch size 32, and 90 epochs using the stochastic gradient descent method. The images are augmented for the training based on perturbations suggested by He et al. [2016] (variations in cropping, brightness, contrast, and saturation).

During test time, input images are resized so that the maximum width or height is 256, center-cropped to 224×224 , and fed into the trained classifiers. Each classifier returns a normalized n-dimensional vector, where $n = 2$ for binary attributes and $n = m$ for m-class attributes. The predictions of all classifiers are then concatenated into a multi-dimensional descriptor. Nearest neighbor search is then performed to find the k -closest matching hair with smallest Euclidean distance in the descriptor space. If the classifier detects a bald head, the following hairstyle matching process is skipped.

Hairstyle Matching. After obtaining a reduced hair model subset based on the semantic attributes, we compare the segmentation mask and hair orientations at the pixel level using pre-rendered thumbnails to retrieve the most similar hairstyle [Chai et al. 2016]. Following Chai et al. [2016], we organize our database as thumbnails and adopt the binary edge-based descriptor from [Zitnick 2010] to increase matching efficiency. For each hairstyle in the database, we pre-render the mask and the orientation map as thumbnails from 35 different views, where 7 angles are uniformly sampled in $[-\pi/4, \pi/4]$ as yaw and 5 angles in $[-\pi/4, \pi/4]$ as pitch. If the hair segmentation mask has multiple connected components due to occlusion or if the hair is partially cropped, then the segmentation descriptor may not be reliable; in this case, we find the most similar hairstyle using the classifiers.

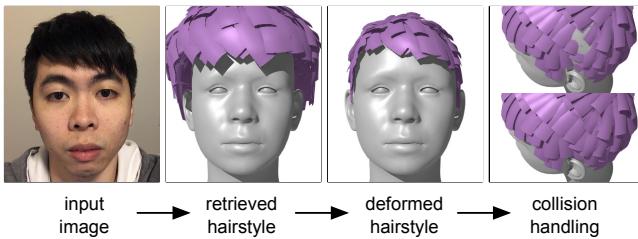


Fig. 7. Our hair mesh fitting pipeline.

Hair Mesh Fitting. In order to match the retrieved model with the silhouette and orientation of the input, we extend the hair fitting algorithm for strands [Chai et al. 2016; Hu et al. 2015] to the polystrip meshes. First, we perform spatial deformation in order to fit the hair model to the personalized head model, using an as-rigid-as-possible graph-based deformation model [Li et al. 2009]. We represent the displacement of each vertex on the hair mesh as a linear combination

of the displacements of k-nearest vertices on the head mesh using the following inversely weighted Gaussian approximation:

$$dp_i = \sum_{j \in N_i} (1 + \|p_i - q_j\|_2 + \|p_i - q_j\|_2^2)^{-1} dq_j,$$

where p and q are vertices on the hair and mean head mesh respectively. This allows the hair model to follow the head deformation without causing intersection. Once the scalp and the hair mesh is aligned, we compute a smooth warping function $\mathcal{W}(\cdot)$ mapping vertices on the 3D model's silhouette to the closest points on the input's 2D silhouette from the camera angle, and deform each polystrip according to the as-rigid-as-possible warping function presented in [Li et al. 2009]. Then, we deform each polystrip to follow the input 2D orientation map as described in [Chai et al. 2016; Hu et al. 2015]. Possible intersections between the head and the hair model due to this deformation are resolved using simple collision handling via force repulsion [Luo et al. 2013].

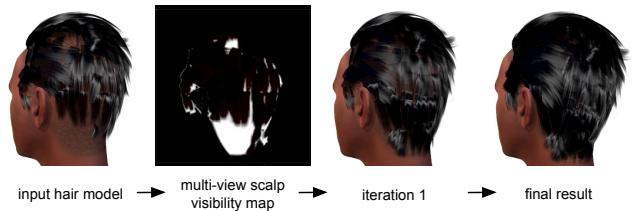


Fig. 8. Our iterative optimization algorithm for polystrip patching.

Polystrip Patching Optimization. With the benefit of having a low computational overhead, a polystrip-based rendering with a bump map and an alpha mask produces locally plausible hair appearance for a wide range of hairstyles. However, such rendering is prone to a lack of scalp coverage, especially for short hairstyles. We propose an iterative optimization method to ensure scalp coverage via patching with minimum increase in the number of triangles.

We measure the coverage by computing the absolute difference between the alpha map in a model view space with and without hair transparency from multiple view points (see Figure 8). Regions with high error expose the scalp surface and need to be covered by additional hair meshes. Without transparency, all polystrips are rendered with alpha value 1.0. When a hair alpha mask is assigned by the hair style classification, the polystrips are rendered via order-independent transparency (OIT), resulting in alpha values of range $[0, 1]$. First, we convert the error map into a binary map by thresholding if the error exceeds 0.5, and apply blob detection on the binary map. Given the blob with highest error, a new polystrip is then placed to cover the area.

We find the k -closest polystrips to the region with the highest error and resample two polystrips within this set so that their average produces a new one that covers this region. We use $k = 6$ for all our examples. The two polystrips are re-sampled so that they have consistent vertex numbers for linear blending. By averaging the polystrips, we can guarantee that the resulting strips are inside the convex hull of the hair region. Thus, our method does not violate the overall hair silhouette after new strips are added. We iterate this

process until the highest error has reached a certain threshold or when no more scalp region is visible.

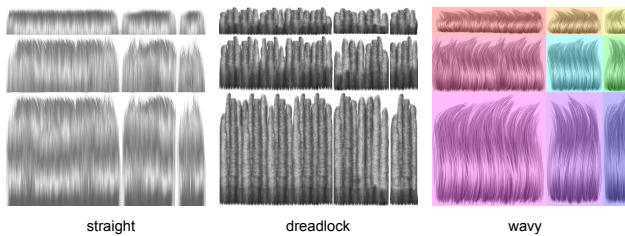


Fig. 9. Example polystrip textures for characterizing high-frequency structures of different hair types. Each texture atlas contains a 9-uv map for polystrips of different sizes.

Hair Rendering and Texturing. We render the resulting polystrips using a variant of [Sadeghi et al. 2010]. The hair tangents are directly obtained from the directions of the mesh’s UV parameterization. We use our classification network to determine the semantic shader parameters, such as the width and the intensity of the primary and secondary highlights. To approximate the multiple scattering components, we add the diffuse term from Kajiya and Kay [1989]. We perform alpha blending between the polystrips using an order-independent transparency (OIT) algorithm based on depth peeling.

Our classification network also specifies for each input image the most similar local hairstyle texture. As illustrated in Figure 9, we characterize a hairstyle’s local high-frequency structure into different categories. These textures are manually designed by an artist based on pre-categorized images that are also used for training. As demonstrated in many games, these type of hair textures can represent a wide range of hair appearances. As different hair types are associated with custom shaders, some styles may be associated with a bump map, which is also prepared by the artist.

For the texture lookups, we use a hierarchical UV atlas which depends on the world dimensions of individual polystrips after the deformation step. The polystrip textures are grouped into nine categories of sizes in a single map. Using multiple texture sizes for each hair patch reduces stretching and compression artifacts in both U and V directions, and also increases texture variations.

6 RESULTS

We created fully-rigged 3D avatars with challenging hairstyles and secondary components for a diverse set of inputs from a wide range of image sets. Even though the input resolutions are inconsistent, there is no a-priori knowledge about the scene illumination or intrinsic camera parameters, and the subjects within the inputs may have tilted or partially covered heads with different expressions, we were still able to produce automatically digitized outputs. We also processed short and long hairstyles of different local structures including straight, wavy, and dreadlock styles. As illustrated in Figure 10, our proposed framework successfully digitizes textured face models and reproduces the volumetric appearance of hair, which is shown from the front and the back. Facial details are faithfully digitized in unseen regions and fully covered hair polystrips can be

reconstructed using our iterative patching optimization algorithm. Our accompanying video shows several animations produced by a professional animator using the provided controls of our avatar. We also demonstrate an avatar animation applications using a real-time facial performance capture system, as well as the simulated hair motions of our hair polystrip models using a mass-spring system based on rigid body chains and hair-head collision (see Figure 13).

Evaluation. We evaluate the robustness of our system and consistency of the reconstruction using a variety of input examples of the same subject as shown in Figure 11. Our combined facial segmentation [Saito et al. 2016], texture inference [Saito et al. 2017] and PCA-based shape, appearance, and lighting estimation [Thies et al. 2016a] framework is robust to severe lighting conditions. We can observe that the visual difference between the reconstructed albedo map of a same person, captured under contrasting illuminations, is minimal. We also demonstrate how our linear face model can discern between a person’s identity and its expression up to some degree. Our visualization shows the resulting avatar in the neutral pose. While some slightly noticeable dissimilarity in the face and hair digitization remains, both outputs are plausible. For large smiles in the input image, the optimized neutral pose can still contain an amused expression.

While traditional hair database retrieval techniques [Chai et al. 2016; Hu et al. 2015] are effective for strand-based output, our hair polystrip modeling approach relies on clean mesh structures and topologies as they are mostly preserved until the end of the pipeline. As shown in Figure 12, a deep learning-based hair attribute classification step is critical in avoiding wrong hair types being used during retrieval. Table 1 lists a few annotated hair attributes, as well as their prediction accuracies from the trained network. Although the predictions are sometimes not accurate due to the lack of training data, we can still retrieve similar hairstyles which are further optimized by subsequent steps in the pipeline.

attribute	possible values	accuracy (%)
hair_length	long/short/bald	72.5
hair_curve	straight/wavy/curly/kinky	76.5
hairline	left/right/middle	87.8
fringe	full/left/right	91.8
hair_bun	1 bun/2 buns/...	91.4
ponytail	1 tail/2 tails/...	79.2
spiky_hair	spiky/not spiky	91.2
shaved_hair	fully/partially shaved	81.4
baldness	fully bald/receded hair	79.6

Table 1. We train a network to classify the above attributes of hairstyles, achieving accuracies around 70-90%.

Comparison. We compare our method against several state-of-the-art facial modeling techniques and avatar creation systems in Figure 14. Our deep learning-based framework [Saito et al. 2017] can infer facial textures with more details comparing to linear morphable face models [Blanz and Vetter 1999; Thies et al. 2016a]. In addition to producing high-quality hair models, our generated face meshes and textures are visually comparable to the video-based



Fig. 10. Our proposed framework successfully generates high-quality and fully rigged avatars from a single input image in the wild. We demonstrate the effectiveness on a wide range of subjects with different hairstyles. We visualize the face meshes and hair polystrips, as well as their textured renderings. Original images courtesy of Getty Images.

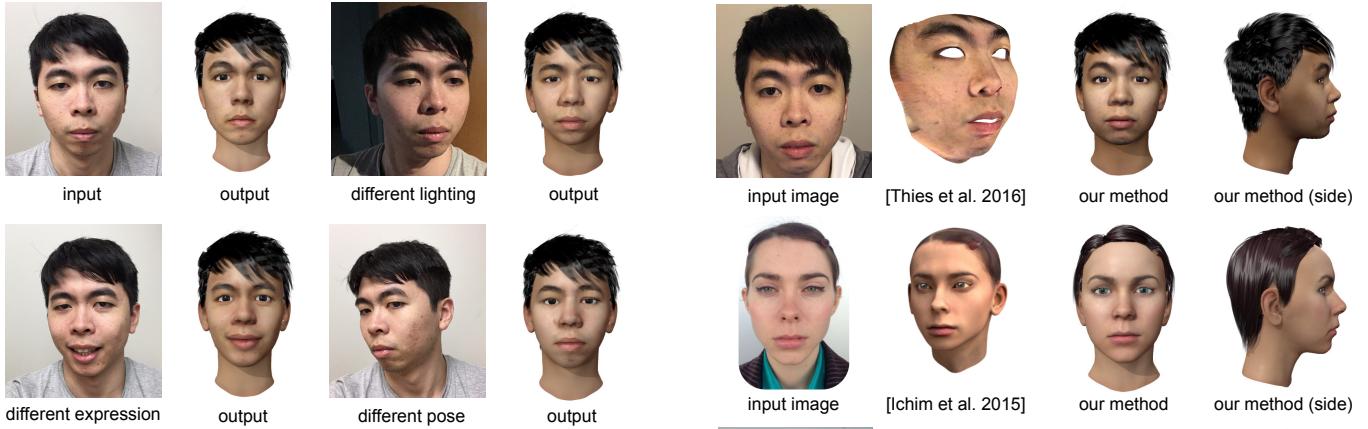


Fig. 11. We evaluate the robustness of our framework by validating the consistency of the output under different capture conditions.



Fig. 12. We assess the importance of our deep learning-based hair attribute classification. Original image courtesy of Getty Images.



Fig. 13. Real-time hair simulation using a mass-spring system.

reconstruction system of Ichim et al. [2015]. We can also reproduce similarly compelling avatars as in [Cao et al. 2016], but using only one out of many of their input images. While their approach is still associated with some manual labor, our system is fully automatic. Additionally, we provide two comparisons with two existing commercial solutions. In particular, we notice that the system of Loom.ai [2017] fails to retrieve the correct hairstyle, while itSeez3D's Avatar SDK [2017] does not automatically produce hair models, nor allows the avatar to be animated.

We further compare our polystrip-based results with the state-of-the-art single-view hair modeling technique from Chai et al. [2016] as shown in Figure 15. Their methods are constrained to strand-based hairstyles and lose effectiveness on local features compared to our polystrips method. While strand-based renderings are typically more realistic, we argue that our representation is more versatile (especially for very short hair) and suitable for efficient character rendering in highly complex virtual scenes. In particular, a single polystrip patch can approximate a large number of strands using a



Fig. 14. We compare our method with several state-of-the-art avatar creation systems. Original image (row 4) courtesy of Getty Images.

single texture with an alpha mask, which can significantly increase rendering performance.

Performance. All our experiments are performed using an Intel Core i7-5930K CPU with 3.5 GHz equipped with a GeForce GTX Titan X with 12 GB memory. 3D head model reconstruction takes 5 minutes in total, consisting of 0.5 second of face model fitting, 75 s of feature correlation extraction, 14 s of computing the convex blending weight, 172 s of the final synthesis optimization. The secondary component fitting and facial rigging are done within 1 second. Hair polystrip reconstruction takes less than 1 s to classify the hair attributes from the input image, less than 1 s to retrieve the closest exemplar, and 10 s to deform a hairstyle. 5 s are needed to handle collision. Polystrip patching optimization is done within 1 minute for 2 iterations.

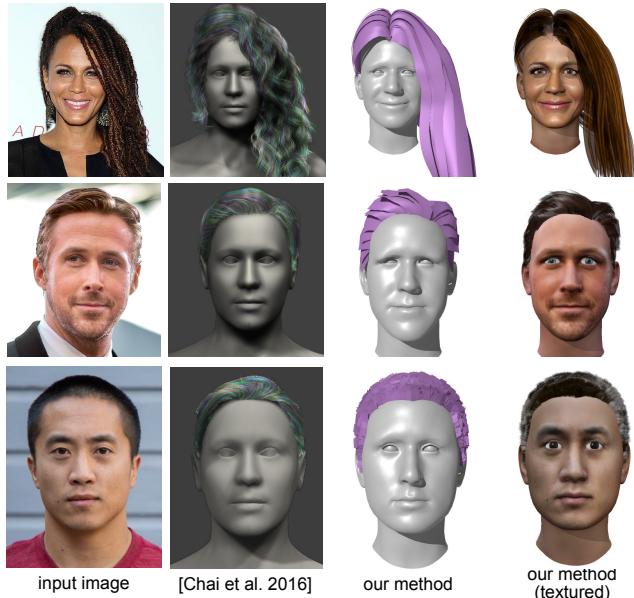


Fig. 15. We compare our method with the latest single-view hair modeling technique, AutoHair [Chai et al. 2016]. Original images (row 1, 2) courtesy of Getty Images.

7 DISCUSSION

The concept of single-view modeling of avatars with hair has been first demonstrated in [Cao et al. 2014b] as part of their “real-time performance-based facial image animation” application. The system is based on a hair reconstruction pipeline for portrait manipulation [Chai et al. 2012]. However, the technique is not fully automatic and requires manual key point corrections and hair strokes.

While the automatic digitization of faces [Blanz and Vetter 1999; Saito et al. 2017; Thies et al. 2016a] and hair [Chai et al. 2016] from single views have been introduced separately, we demonstrate an end-to-end framework that integrates the computation of both components. The ability to create complete models from a single unconstrained image is particularly suitable for consumer use, as well as for scalable content creation in virtual production. We can now easily produce animator-friendly models of a person with intuitive controls, as illustrated in our examples.

Previous single-view hair reconstruction techniques mostly focus on the digitization of strand geometry; however, we also infer hair appearance, taking into account the custom shading properties for the rendering engine. Even though the digitization of high-quality strands is possible, the rendering costs involved are significant for complex multi-character virtual environments. Our focus is to provide a unified solution for capturing a wide range of hairstyles and the ability to integrate them into existing real-time game engines such as Unity. We have shown that polystrips are versatile hair representations and suitable for the efficient rendering and animation of compelling avatars. We also note the importance of rendering capabilities such as order-independent transparency for producing convincing looking volumetric hair.

The effectiveness of our methodology is grounded on a careful integration of state-of-the-art modeling and synthesis techniques for faces and hair. Several key components, such as segmentation, semantic hair attributes extraction, and eye color recognition, are only possible due to recent advances in deep learning. Our experiments also indicate the robustness of our system, where consistent results of the same subject can be obtained when captured from different angles, under contrasting lighting conditions, and with different input expressions.

Even in cases where the subject is only partially visible, the image is of low resolution, and the illumination conditions unknown, we can obtain high-quality textured meshes of the face and compelling hair renderings similar to those of characters in recent games. Our approach is qualitatively comparable to existing avatar creation systems, which require multiple photographs and manual input [Cao et al. 2016; Ichim et al. 2015].

While our proposed polystrip optimization algorithm is a critical component for our automatic avatar digitization framework, we believe that it can also be a useful tool during the design process of polystrip-based hair models in general. Once a rough hair mesh is created, an artist could use this patching optimization instead of manually duplicating and perturbing with additional polystrips.

Limitations. Due to the ill-posed problem of highly incomplete input and the low-dimensionality of our linear face models, our shape models may not be fully accurate and our facial texture inference technique may add details in wrong places. With the dramatic progress in deep learning research, we believe that a massive collection of high-resolution 3D faces in controlled capture settings could be used to improve the fidelity of our face models, as well as the performance of shape inference algorithms.

Since only a single input image is used, our face modeling pipeline transfers a generic FACS-based linear blendshape model to every subject. In reality these blendshapes would need to be individualized for specific subjects. While it is possible that certain expressions would correlate with the shape of the face, it is most likely that multiple input images would be necessary to form accurate facial expression models using optimization techniques as introduced by Li et al. [2010]. In addition, the accuracy of our hair classification network is not 100%; for example, ponytails can be ambiguous. Similar to previous papers, our method would fail to retrieve the correct hair model when the input hairstyle differs greatly from those in the database (Figure 16).

We use a simple mass-spring system technique to produce motion simulation. While the use of hair polystrips is highly efficient and a reasonable approximation of strand-based models [Chai et al. 2016; Hu et al. 2015], convincing strand-level simulations [Chai et al. 2014] are not yet possible with our representation.

Though the use of polystrips and textures with alpha masks can capture the volumetric look of hair as opposed to image-based alternatives [Cao et al. 2016], we cannot digitize props such as headwear or glasses. Our method would also fail for longer facial hair such as beards, since our database does not contain these objects. We believe that adding more object types as samples in our database could make such inference possible. In addition, our system currently only

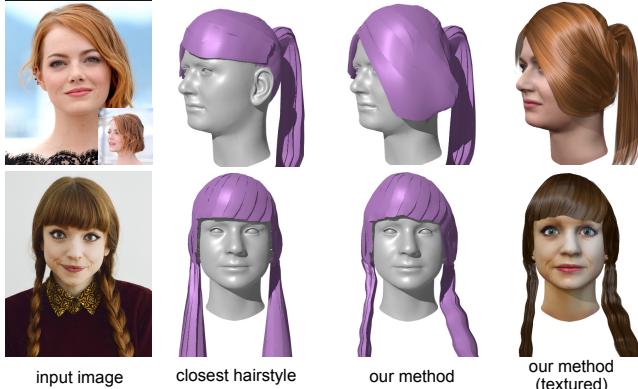


Fig. 16. Limitations. Wrong hairstyles can be retrieved due to incomplete visibility or insufficient hair samples in the database. Original images courtesy of Getty Images (row 1) and Alexandra Spence (row 2).

captures a single hair color for each subject. More powerful texture analysis and synthesis techniques would be needed to generate plausible multi-color hairstyles.

Future Work. Since our framework is designed around today's real-time rendering environments and facial animation systems, we are still using commonly used parametric models for faces and hair, and the results may still look uncanny. In the future, we plan to explore end-to-end deep learning-based inference methods to generate more realistic avatars with dynamic textures and more compelling hair rendering techniques. Research in generative adversarial networks are promising directions.

ACKNOWLEDGEMENTS

We would like to thank Jun Xing for the hair modeling technology used for augmenting our hairstyle database; Han-Wei Kung for the mass-spring system-based hair simulator in Unity; Kun Zhou and Menglei Chai for the hair digitization comparisons; David Rodriguez for the facial animations; Minjeong Shin for the facial rig; Kyle Morgenroth and Stephen Chen for the Unity-based demo application; Glenn and Robbie Derry for the headcam; as well as the anonymous reviewers. Lingyu Wei is supported by the Adobe Research Fellowship and Koki Nagano by the Google PhD Fellowship. This research is conducted by Pinscreen with support of Adobe, Oculus & Facebook, Huawei, Sony, the Google Faculty Research Award, the Okawa Foundation Research Grant, the Office of Naval Research (ONR), under award number N00014-15-1-2639, and the U.S. Army Research Laboratory (ARL) under contract W911NF-14-D-0005. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of ONR, ARL, or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purpose notwithstanding any copyright annotation thereon.

REFERENCES

- Louis Bavoil and Kevin Myers. 2008. Order independent transparency with dual depth peeling. (2008).

- Thabo Beeler, Bernd Bickel, Paul Beardsley, Bob Sumner, and Markus Gross. 2010. High-Quality Single-Shot Capture of Facial Geometry. *ACM Trans. on Graphics (Proc. SIGGRAPH)* 29, 3 (2010).
- Thabo Beeler, Bernd Bickel, Gioacchino Noris, Steve Marschner, Paul Beardsley, Robert W. Sumner, and Markus Gross. 2012. Coupled 3D Reconstruction of Sparse Facial Hair and Skin. *ACM Trans. Graph.* 31, Article 117 (2012). Issue 4.
- Thabo Beeler, Fabian Hahn, Derek Bradley, Bernd Bickel, Paul Beardsley, Craig Gotsman, Robert W. Sumner, and Markus Gross. 2011. High-quality passive facial performance capture using anchor frames. *ACM Trans. Graph.* 30, Article 75 (2011). Issue 4.
- Pascal Bérard, Derek Bradley, Markus Gross, and Thabo Beeler. 2016. Lightweight Eye Capture Using a Parametric Model. *ACM Trans. Graph.* 35, 4, Article 117 (2016).
- Andrew Blake, Sami Romdhani, Thomas Vetter, Brian Amberg, and Andrew Fitzgibbon. 2007. Reconstructing High Quality Face-Surfaces using Model Based Stereo. *2007 11th IEEE International Conference on Computer Vision* 00, undefined (2007).
- Volker Blanz, Curzio Bassi, Tomaso Poggio, and Thomas Vetter. 2003. Reanimating faces in images and video. In *Computer graphics forum*, Vol. 22. Wiley Online Library.
- Volker Blanz and Thomas Vetter. 1999. A morphable model for the synthesis of 3D faces. In *Proceedings of the 26th annual conference on Computer graphics and interactive techniques*.
- James Booth, Anastasios Zafeiriou, Allan Ponniah, and David Dunaway. 2016. A 3D Morphable Model Learnt From 10,000 Faces. In *IEEE CVPR*.
- Sofien Bouaziz, Yangang Wang, and Mark Pauly. 2013. Online Modeling for Realtime Facial Animation. *ACM Trans. Graph.* 32, 4, Article 40 (2013).
- Derek Bradley, Wolfgang Heidrich, Tiberiu Popa, and Alla Sheffer. 2010. High resolution passive facial performance capture. *ACM Trans. Graph.* 29, Article 41 (2010). Issue 4.
- Chen Cao, Qiming Hou, and Kun Zhou. 2014a. Displaced Dynamic Expression Regression for Real-time Facial Tracking and Animation. *ACM Trans. Graph.* 33, 4, Article 43 (2014).
- Chen Cao, Yanlin Weng, Shun Zhou, Yiyi Tong, and Kun Zhou. 2014b. Facewarehouse: A 3d facial expression database for visual computing. *IEEE TVCG* 20, 3 (2014).
- Chen Cao, Hongzhi Wu, Yanlin Weng, Tianjia Shao, and Kun Zhou. 2016. Real-time facial animation with image-based dynamic avatars. *ACM Trans. Graph.* 35, 4 (2016).
- Xudong Cao, Yichen Wei, Fang Wen, and Jian Sun. 2013. Face Alignment by Explicit Shape Regression. *International Journal of Computer Vision* (2013).
- 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, Article 204 (2015).
- 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, Article 116 (2016).
- Menglei Chai, Lvdi Wang, Yanlin Weng, Xiaogang Jin, and Kun Zhou. 2013. Dynamic Hair Manipulation in Images and Videos. *ACM Trans. Graph.* 32, 4, Article 75 (2013).
- 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, Article 116 (2012).
- Menglei Chai, Changxi Zheng, and Kun Zhou. 2014. A Reduced Model for Interactive Hairs. *ACM Trans. Graph.* 33, 4, Article 124 (2014).
- Byoungwon Choe and Hyeyoung-Seok Ko. 2005. A Statistical Wisp Model and Pseudo-physical Approaches for Interactive Hairstyle Generation. *IEEE Trans. Vis. Comput. Graph.* 11, 2 (2005).
- Timothy F Cootes, Gareth J Edwards, and Christopher J Taylor. 2001. Active appearance models. *IEEE TPAMI* 6 (2001).
- David Cristinacce and Tim Cootes. 2008. Automatic feature localisation with constrained local models. *Pattern Recogn.* 41, 10 (2008).
- Paul Debevec, Tim Hawkins, Chris Tchou, Haarm-Pieter Duiker, Westley Sarokin, and Mark Sagar. 2000. Acquiring the Reflectance Field of a Human Face (*SIGGRAPH '00*).
- J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. 2009. ImageNet: A Large-Scale Hierarchical Image Database. In *CVPR09*.
- Jose I. Echevarria, Derek Bradley, Diego Gutierrez, and Thabo Beeler. 2014. Capturing and Stylizing Hair for 3D Fabrication. *ACM Trans. Graph.* 33, 4, Article 125 (2014).
- FaceUnity. 2017. (2017). <http://www.faceunity.com/p2a-demo.mp4>.
- Pablo Garrido, Levi Valgaerts, Chenglei Wu, and Christian Theobalt. 2013. Reconstructing Detailed Dynamic Face Geometry from Monocular Video. In *ACM Trans. Graph. (Proceedings of SIGGRAPH Asia 2013)*, Vol. 32.
- Pablo Garrido, Michael Zollhöfer, Dan Casas, Levi Valgaerts, Kiran Varanasi, Patrick Pérez, and Christian Theobalt. 2016a. Reconstruction of Personalized 3D Face Rigs from Monocular Video. *ACM Trans. Graph. (Presented at SIGGRAPH 2016)* 35, 3 (2016).
- Pablo Garrido, Michael Zollhöfer, Chenglei Wu, Derek Bradley, Patrick Pérez, Thabo Beeler, and Christian Theobalt. 2016b. Corrective 3D Reconstruction of Lips from Monocular Video. *ACM Trans. Graph.* 35, 6, Article 219 (2016).
- Leon A Gatys, Alexander S Ecker, and Matthias Bethge. 2016. Image style transfer using convolutional neural networks. In *IEEE CVPR*.
- Abhijeet Ghosh, Graham Fyffe, Borom Tunwattanapong, Jay Busch, Xueming Yu, and Paul Debevec. 2011. Multiview Face Capture Using Polarized Spherical Gradient Illumination. *ACM Trans. Graph.* 30, 6, Article 129 (2011).

- Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *IEEE CVPR*.
- 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, Article 146 (2012).
- Pei-Lun Hsieh, Chongyang Ma, Jihun Yu, and Hao Li. 2015. Unconstrained Realtime Facial Performance Capture. In *IEEE CVPR*.
- Liwen Hu, Chongyang Ma, Linjie Luo, and Hao Li. 2014a. Robust Hair Capture Using Simulated Examples. *ACM Trans. Graph. (Proceedings SIGGRAPH 2014)* 33, 4 (2014).
- Liwen Hu, Chongyang Ma, Linjie Luo, and Hao Li. 2015. Single-View Hair Modeling Using A Hairstyle Database. *ACM Trans. Graph. (Proceedings SIGGRAPH 2015)* 34, 4 (2015).
- Liwen Hu, Chongyang Ma, Linjie Luo, Li-Yi Wei, and Hao Li. 2014b. Capturing Braided Hairstyles. *ACM Trans. Graph. (Proceedings SIGGRAPH Asia 2014)* 33, 6 (2014).
- Gary B. Huang, Manu Ramesh, Tamara Berg, and Erik Learned-Miller. 2007. *Labeled Faces in the Wild: A Database for Studying Face Recognition in Unconstrained Environments*. Technical Report 07-49. University of Massachusetts, Amherst.
- Alexandre Eugen Ichim, Sofien Bouaziz, and Mark Pauly. 2015. Dynamic 3D Avatar Creation from Hand-held Video Input. *ACM Trans. Graph.* 34, 4, Article 45 (2015).
- itSee3D: Avatar SDK. 2017. (2017). <https://avatarsdk.com>.
- Alec Jacobson, Zhigang Deng, Ladislav Kavan, and JP Lewis. 2014. Skinning: Real-time Shape Deformation. In *ACM SIGGRAPH 2014 Courses*.
- Wenzel Jakob, Jonathan T. Moon, and Steve Marschner. 2009. Capturing Hair Assemblies Fiber by Fiber. *ACM Trans. Graph.* 28, 5, Article 164 (2009).
- J. T. Kajiya and T. L. Kay. 1989. Rendering Fur with Three Dimensional Textures. In *Proceedings of the 16th Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH '89)*. ACM.
- Vahid Kazemi and Josephine Sullivan. 2014. One millisecond face alignment with an ensemble of regression trees. In *IEEE CVPR*.
- Ira Kemelmacher-Shlizerman. 2013. Internet-based Morphable Model. *IEEE ICCV* (2013).
- Ira Kemelmacher-Shlizerman and Ronen Basri. 2011. 3d face reconstruction from a single image using a single reference face shape. *IEEE TPAMI* 33, 2 (2011).
- Hyeyoungwoo Kim, Michael Zollöfer, Ayush Tewari, Justus Thies, Christian Richardt, and Theobalt Christian. 2017. InverseFaceNet: Deep Single-Shot Inverse Face Rendering From A Single Image. *arXiv preprint arXiv:1703.10956* (2017).
- Tae-Yong Kim and Ulrich Neumann. 2002. Interactive Multiresolution Hair Modeling and Editing. *ACM Trans. Graph.* 21, 3 (2002).
- Philipp Krähnert and Vladlen Koltun. 2011. Efficient Inference in Fully Connected CRFs with Gaussian Edge Potentials. In *Advances in Neural Information Processing Systems*.
- Hao Li, Bart Adams, Leonidas J. Guibas, and Mark Pauly. 2009. Robust Single-View Geometry And Motion Reconstruction. *ACM Trans. Graph. (Proceedings SIGGRAPH Asia 2009)* 28, 5 (2009).
- Hao Li, Shunsuke Saito, Lingyu Wei, Iman Sadeghi, Liwen Hu, Jaewoo Seo, Koki Nagano, Jens Fursund, Yen-Chun Chen, and Stephen Chen. 2017. Pinscreen: Creating Performance-driven Avatars in Seconds. In *ACM SIGGRAPH 2017 Real Time Live! (SIGGRAPH '17)*. ACM.
- 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. (Proceedings SIGGRAPH 2015)* 34, 4 (2015).
- Hao Li, Thibaut Weise, and Mark Pauly. 2010. Example-Based Facial Rigging. *ACM Trans. Graph. (Proceedings SIGGRAPH 2010)* 29, 3 (2010).
- Hao Li, Jihun Yu, Yuting Ye, and Chris Bregler. 2013. Realtime Facial Animation with On-the-fly Correctives. *ACM Trans. Graph. (Proceedings SIGGRAPH 2013)* 32, 4 (2013).
- Shu Liang, Linda G Shapiro, and Ira Kemelmacher-Shlizerman. 2016. Head reconstruction from internet photos. In *European Conference on Computer Vision*. Springer.
- Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. 2015. Deep Learning Face Attributes in the Wild. In *Proceedings of International Conference on Computer Vision (ICCV)*.
- Loom.ai. 2017. (2017). <http://www.loom.ai>.
- Linjie Luo, Hao Li, and Szymon Rusinkiewicz. 2013. Structure-Aware Hair Capture. *ACM Trans. Graph. (Proceedings SIGGRAPH 2013)* 32, 4 (2013).
- Debbie S. Ma, Joshua Correll, and Bernd Wittenbrink. 2015. The Chicago face database: A free stimulus set of faces and norming data. *Behavior Research Methods* 47, 4 (2015).
- Wan-Chun Ma, Tim Hawkins, Pieter Peers, Charles-Felix Chabert, Malte Weiss, and Paul Debevec. 2007. Rapid Acquisition of Specular and Diffuse Normal Maps from Polarized Spherical Gradient Illumination. In *Eurographics Symposium on Rendering*.
- Erick Miller and Dmitry Pinskiy. 2009. Realistic Eye Motion Using Procedural Geometric Methods. In *SIGGRAPH 2009: Talks (SIGGRAPH '09)*. ACM, Article 75.
- Kyle Olszewski, Joseph J. Lim, Shunsuke Saito, and Hao Li. 2016. High-Fidelity Facial and Speech Animation for VR HMDs. *ACM Trans. Graph. (Proceedings SIGGRAPH Asia 2016)* 35, 6 (2016).
- 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, Article 30 (2008).
- Frederic I. Parke and Keith Waters. 2008. *Computer Facial Animation* (second ed.). AK Peters Ltd.
- Pascal Paysan, Reinhard Knothe, Brian Amberg, Sami Romdhani, and Thomas Vetter. 2009. A 3D face model for pose and illumination invariant face recognition. In *Advanced video and signal based surveillance, 2009. AVSS'09. Sixth IEEE International Conference on*. IEEE.
- Patrick Pérez, Michel Gangnet, and Andrew Blake. 2003. Poisson image editing. In *ACM Trans. Graph.*, Vol. 22. ACM.
- Pinscreen. 2017. (2017). <http://www.pinscreen.com>.
- Shaoqing Ren, Xudong Cao, Yichen Wei, and Jian Sun. 2014. Face alignment at 3000 fps via regressing local binary features. In *IEEE CVPR, 2014 IEEE Conference on*. IEEE.
- Elad Richardson, Matan Sela, Roy Or-El, and Ron Kimmel. 2016. Learning Detailed Face Reconstruction from a Single Image. *arXiv preprint arXiv:1611.05053* (2016).
- Iman Sadeghi, Heather Pritchett, Henrik Wann Jensen, and Rasmus Tamstorf. 2010. An Artist Friendly Hair Shading System. In *ACM SIGGRAPH 2010 Papers (SIGGRAPH '10)*, Vol. 29. ACM.
- Shunsuke Saito, Tianye Li, and Hao Li. 2016. Real-Time Facial Segmentation and Performance Capture from RGB Input. In *ECCV*.
- Shunsuke Saito, Lingyu Wei, Liwen Hu, Koki Nagano, and Hao Li. 2017. Photorealistic Facial Texture Inference Using Deep Neural Networks. In *IEEE CVPR*.
- Jason M. Saragih, Simon Lucey, and Jeffrey F. Cohn. 2011. Deformable Model Fitting by Regularized Landmark Mean-Shift. *Int. J. Comput. Vision* 91, 2 (2011).
- Andrew Selle, Michael Lentine, and Ronald Fedkiw. 2008. A Mass Spring Model for Hair Simulation. *ACM Trans. Graph.* 27, 3, Article 64 (2008).
- Fuhao Shi, Hsiang-Tao Wu, Xin Tong, and Jinxiang Chai. 2014. Automatic Acquisition of High-fidelity Facial Performances Using Monocular Videos. *ACM Trans. Graph.* 33, 6, Article 222 (2014).
- Zhixin Shu, Ersin Yumer, Sunil Hadap, Kalyan Sunkavalli, Eli Shechtman, and Dimitris Samaras. 2017. Neural Face Editing with Intrinsic Image Disentangling. (2017). *arXiv:arXiv:1704.04131*
- Eftychios Sifakis, Igor Neverov, and Ronald Fedkiw. 2005. Automatic Determination of Facial Muscle Activations from Sparse Motion Capture Marker Data. *ACM Trans. Graph.* 24, 3 (2005).
- K. Simonyan and A. Zisserman. 2014. Very Deep Convolutional Networks for Large-Scale Image Recognition. *CoRR abs/1409.1556* (2014).
- Demetris Terzopoulos and Keith Waters. 1990. Physically-based facial modelling, analysis, and animation. *The journal of visualization and computer animation* 1, 2 (1990).
- Ayush Tewari, Michael Zollöfer, Hyeyoungwoo Kim, Pablo Garrido, Florian Bernard, Patrick Perez, and Theobalt Christian. 2017. MoFA: Model-based Deep Convolutional Face Autoencoder for Unsupervised Monocular Reconstruction. *arXiv preprint arXiv:1703.10580* (2017).
- J. Thies, M. Zollhöfer, M. Stamminger, C. Theobalt, and M. Nießner. 2016a. Face2Face: Real-time Face Capture and Reenactment of RGB Videos. In *IEEE CVPR*.
- Justus Thies, Michael Zollöfer, Marc Stamminger, Christian Theobalt, and Matthias Nießner. 2016b. FaceVR: Real-Time Facial Reenactment and Eye Gaze Control in Virtual Reality. *arXiv preprint arXiv:1610.03151* (2016).
- Paul Viola and Michael Jones. 2001. Rapid object detection using a boosted cascade of simple features. In *IEEE CVPR*, Vol. 1. IEEE.
- Daniel Vlasic, Matthew Brand, Hanspeter Pfister, and Jovan Popović. 2005. Face Transfer with Multilinear Models. *ACM Trans. Graph.* 24, 3 (2005).
- Javier von der Pahlen, Jorge Jimenez, Etienne Danvoye, Paul Debevec, Graham Fyffe, and Oleg Alexander. 2014. Digital Ira and Beyond: Creating Real-time Photorealistic Digital Actors. In *ACM SIGGRAPH 2014 Courses (SIGGRAPH '14)*. ACM, Article 1.
- Lvdi Wang, Yizhou Yu, Kun Zhou, and Baining Guo. 2009. Example-based Hair Geometry Synthesis. *ACM Trans. Graph.* 28, 3 (2009).
- Kelly Ward, Florence Bertails, Tae yong Kim, Stephen R. Marschner, Marie paule Cani, and Ming C. Lin. 2006. A survey on hair modeling: styling, simulation, and rendering. In *IEEE TVCG*.
- Thibaut Weise, Sofien Bouaziz, Hao Li, and Mark Pauly. 2011. Realtime Performance-Based Facial Animation. *ACM Trans. Graph. (Proceedings SIGGRAPH 2011)* 30, 4 (July 2011).
- Thibaut Weise, Hao Li, Luc Van Gool, and Mark Pauly. 2009. Face/Off: Live Facial Puppetry. In *Proceedings of the 2009 ACM SIGGRAPH/Eurographics Symposium on Computer animation (Proc. SCA'09)*. Eurographics Association, ETH Zurich.
- Yanlin Weng, Lvdi Wang, Xiao Li, Menglei Chai, and Kun Zhou. 2013. Hair Interpolation for Portrait Morphing. *Computer Graphics Forum* (2013).
- Chenglei Wu, Derek Bradley, Pablo Garrido, Michael Zollhöfer, Christian Theobalt, Markus Gross, and Thabo Beeler. 2016. Model-based Teeth Reconstruction. *ACM Trans. Graph.* 35, 6, Article 220 (2016).
- Xuehan Xiong and Fernando De la Torre. 2013. Supervised descent method and its applications to face alignment. In *IEEE CVPR*. IEEE.
- Cem Yuksel, Scott Schaefer, and John Keyser. 2009. Hair Meshes. *ACM Trans. Graph.* 28, 5, Article 166 (2009).
- Yongning Zhu and Robert Bridson. 2005. Animating Sand As a Fluid. *ACM Trans. Graph.* 24, 3 (2005).
- C. Lawrence Zitnick. 2010. Binary Coherent Edge Descriptors. In *Proceedings of the 11th European Conference on Computer Vision: Part II (ECCV'10)*.