3D Face Synthesis Driven by Personality Impression

Yining Lang,[1] **Wei Liang,**[1] **Yujia Wang,**[1] **Lap-Fai Yu**[2]

[1]Beijing Laboratory of Intelligent Information Technology, Beijing Institute of Technology, [2]George Mason University
langyining@bit.edu.cn, liangwei@bit.edu.cn, wangyujia@bit.edu.cn, craigyu@gmu.edu.

Abstract

Synthesizing 3D faces that give certain personality impressions is commonly needed in computer games, animations, and virtual world applications for producing realistic virtual characters. In this paper, we propose a novel approach to synthesize 3D faces based on personality impression for creating virtual characters. Our approach consists of two major steps. In the first step, we train classifiers using deep convolutional neural networks on a dataset of images with personality impression annotations, which are capable of predicting the personality impression of a face. In the second step, given a 3D face and a desired personality impression type as user inputs, our approach optimizes the facial details against the trained classifiers, so as to synthesize a face which gives the desired personality impression. We demonstrate our approach for synthesizing 3D faces giving desired personality impressions on a variety of 3D face models. Perceptual studies show that the perceived personality impressions of the synthesized faces agree with the target personality impressions specified for synthesizing the faces.

Introduction

A face conveys a lot of information about a person. People usually form an impression about another person in less than a second, mainly by looking at another person's face. Researchers in psychology, cognitive science, and biometrics conducted a lot of studies to explore how facial appearances may influence personality impression (Willis and Todorov 2006; Hassin and Trope 2000). Some researchers investigated the relationship between personality impressions and specific facial features (Eisenthal, Dror, and Ruppin 2006). There are also attempts in training machine learning models for predicting personality impressions based on facial features (Gray et al. 2010; Joo, Steen, and Zhu 2015).

To create realistic 3D faces for the computer games, digital entertainments, and virtual reality applications, some works have been carried on generating realistic faces (Hu et al. 2017), vivid animations (Sohre et al. 2018), natural expressions (Marsella et al. 2013), and so on. Yet, synthesizing 3D faces that give certain personality is not explored, which is one of the most important considerations during the creative process. For example, the main characters in games and animations are usually designed to look confident and smart, whereas the "bad guys" are usually designed to look hostile. There are automatic tools for synthesizing human faces of different ethnicities and genders, however, the problem of synthesizing 3D faces with respect to personality impressions is still unsolved. We propose a data-driven optimization approach to solve this problem.

As the personality impression of a face depends a lot on its subtle details, under the current practice, creating a face to give a certain personality impression is usually done through a "trial-and-error" approach: a designer creates several faces; asks for people's feedback on their impressions of the faces; and then modifies the faces accordingly. This process iterates until a satisfactory face is created. This design process involves substantial tuning efforts by a designer and is not scalable. Manual creation of faces could also be very challenging if the objectives are abstract or sophisticated. For example, while it could be relatively easy to create a face to give an impression of being friendly, it could be hard to create a face to give an impression of being friendly but silly, which could be desirable for a certain virtual character.

We propose a novel approach to automate this face creation process. Our approach leverages Convolutional Neural Networks (CNN) techniques to learn the non-trivial mapping between low-level subtle details of a face and high-level personality impressions. The trained networks can then be applied for synthesizing a 3D face to give a desired personality impression via an optimization process. We demonstrate that our approach can automatically synthesize a variety of 3D faces to give different personality impressions, hence overcoming the current scalability bottleneck. The synthesized faces could find practical uses in virtual world applications (*e.g.* , synthesizing a gang of hostile-looking guys to be used as enemies in a game).

The major contributions of our paper include:

- Introducing a novel problem of synthesizing 3D faces based on personality impressions.

- Proposing a learning-based optimization approach and a data-driven MCMC sampler for synthesizing faces with desired personality impressions.

- Demonstrating the practical uses of our approach for different novel face editing, virtual reality applications and digital entertainments.

Related Work

Faces and personality impressions. Personality impression is an active research topic in psychology and cognitive science. Researchers are interested in studying how different factors, *e.g.*, face, body, profile, motion, influence the formation of personality impression on others (Naumann et al. 2009). Recent work (Over and Cook 2018) suggests that facial appearances play an important role in giving personality impressions.

Some works focused on examining what facial features influence personality impression. Vernon et al. (2014) modeled the relationship between physical facial features extracted from images and impression of social traits. Zell et al. (2015) studied the roles of face geometry and texture in affecting the perception of computer-generated faces. Some findings were adopted to predict human-related attributes based on a face. Xu et al. (2015) proposed a cascaded fine-tuning deep learning model to predict facial attractiveness. Xie et al. (2015) proposed a benchmark dataset for analyzing facial beauty impression. Joo et al. (2015) proposed an approach to infer the personality of a person from his face.

Motivated by these findings, we use deep learning techniques to learn the relationship between facial appearances and personality impressions based on a collected face dataset with personality impression annotations, which is applied to guide the synthesis of 3D faces to give desired personality impressions by an optimization.

Face Modeling and Exaggeration. Some commercial 3D modeling software can be used by designers for creating 3D virtual characters with rich facial details, such as Character Generator, MakeHuman, Fuse, and so on. These tools provide a variety of controls of a 3D face model, including geometry and texture, *e.g.*, adjusting the shape of the nose, changing skin color. However, to create or modify a face to give a certain personality impression, a designer has to manually tune many low-level facial features, which could be very tedious and difficult.

Another line of works closely relevant to ours is face exaggeration, which refers to generating a facial caricature with exaggerated face features. Suwajanakorn et al. (2015) proposed an approach for creating a controllable 3D face model of a person from a large photo collection of that person captured in different occasions. Le et al. (2011) performed exaggeration differently by using primitive shapes to locate the face components, followed by deforming these shapes to generate an exaggerated face. They empirically found that specific combinations of primitive shapes tend to establish certain personality stereotypes. Recently, Tian and Xiao (2016) proposed an approach for face exaggeration on 2D face images based on a number of shape and texture features related to personality traits.

Compared to these works, our learning-based optimization approach provides high-level controls for 3D face modeling, by which designers can synthesize faces with respect to specified personality impressions conveniently.

Data-Driven 3D Modeling.

Data-driven techniques have been successfully applied for 3D modeling (Kalogerakis et al. 2012; Talton et al. 2011).
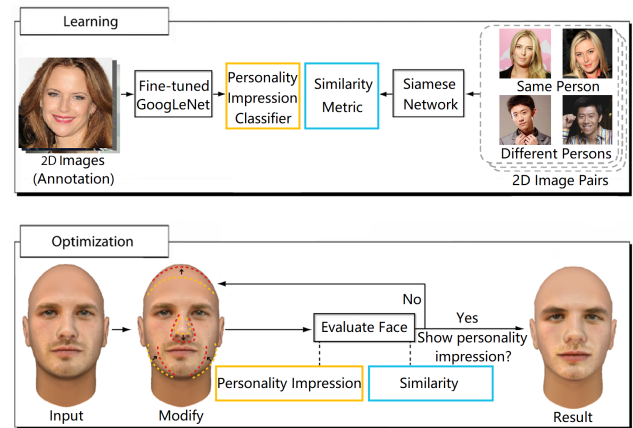

Figure 1: Overview of our approach.

Huang et al. (2017) devised deeply-learned generative models for 3D shape synthesis. Ritchie et al. (2015) used Sequential Monte Carlo to guide the procedural generation of 3D models in an efficient manner. Along the direction of face modeling, Saito et al. (2017) used deep neural networks trained with a high-resolution face database to automatically infer a high-fidelity texture map of an input face image.

Modeling the relationships between low-level facial features and high-level personality impressions is difficult. In addition, directly searching in such a complex and high-dimensional space is inefficient and unstable. In our work, we apply data-driven techniques to model the relationship between facial appearances and personality impressions. Furthermore, we speed up face synthesis by formulating a data-driven sampling approach to facilitate the optimization.

Overview

Figure 1 shows an overview of our approach. Given an input 3D face, our approach optimizes the face geometry and texture such that the optimized face gives the desired personality impression specified by the user. To achieve this goal, we present an automatic face synthesis framework driven by personality impression, which consists of two stages: **learning** and **optimization**.

In the learning stage, we define 8 types of personality impression. Then we learn a CNN personality impression classifier for each type. To train the CNN classifiers, we collected $10,000$ images from CASIA WebFace database (Yi et al. 2014) and annotated them with the corresponding personality impression. We also learn an end-to-end metric to evaluate the similarity between the synthesized face and the input one. The metric plays the role of constraining 3D face deformation.

In the optimization stage, our approach modifies the face geometry and texture iteratively. The resulting face is then evaluated by the personality impression cost function (defined by the learned personality impression classifiers), as well as the similarity cost function (defined by the learned similarity metric). To speed up the optimization, we devise a

data-driven sampling approach based on the learned priors. The optimization continues until a face giving the desired personality impression is synthesized.

Problem Formulation

Personality Impression Types. In our experiments, we use four pairs of personality impressions types: a) smart/silly; b) friendly/hostile; c) humorous/boring; and d) confident/unconfident. These personality impression types are commonly used in psychology (Mischel 2013; Asch 1946).

3D Face Representation. To model a 3D face, we use a multi-linear PCA approach to represent the face geometry and texture (Blanz and Vetter 1999), akin to the representation of (Hu et al. 2017). Our approach operates on a textured 3D face mesh model. We represent a face $(\mathbf{V}, \mathbf{T})$ by its geometry $\mathbf{V} \in \mathbf{R}^{3n}$, which is a vector containing the 3D coordinates of the $n = 6,292$ vertices of the face mesh, as well as a vector $\mathbf{T} \in \mathbf{R}^{3n}$ containing the RGB values of the n pixels of its texture image.

Each face is divided into 8 regions (eyes, jaw, nose, chin, cheeks, mouth, eyebrows and face contour). For each face region, we learn two Principal Component Analysis (PCA) models for representing its geometry and texture in low-dimensional spaces. The PCA models are learned using 3D faces from Basel Face Model database (Paysan et al. 2009).

First, we manually segment each face into the eight regions. Then, for each region, we perform a PCA on the geometry and a PCA on the texture to compute the averages and the sets of eigenvectors. In our implementation, when doing the PCAs for the r-th region, for all vertices in $\mathbf{V}$ and all pixels in $\mathbf{T}$ that do not belong to the r-th region, we just set their values to zero so that all regions have the same dimensionality and can be linearly combined to form the whole face $(\mathbf{V}, \mathbf{T})$:

$$\mathbf{V} = \sum_{r=1}^{8} (\bar{\mathbf{V}}_r + \Lambda_r \mathbf{v}_r), \ \mathbf{T} = \sum_{r=1}^{8} (\bar{\mathbf{T}}_r + \Gamma_r \mathbf{t}_r). \quad (1)$$

Here r is the index of a face region; $\bar{\mathbf{V}}_r \in \mathbf{R}^{3n}$ and $\bar{\mathbf{T}}_r \in \mathbf{R}^{3n}$ denote the average geometry and average texture for the r-th face region; $\Lambda_r \in \mathbf{R}^{3n \times m}$ and $\Gamma_r \in \mathbf{R}^{3n \times m}$ are matrices whose columns are respectively the eigenvectors of the geometry and texture. We use $m = 40$ eigenvectors in our experiments. $\mathbf{v}_r \in \mathbf{R}^m$ and $\mathbf{t}_r \in \mathbf{R}^m$ are vectors whose entries are the coefficients corresponding respectively to the eigenvectors of the geometry and texture. This representation allows our approach to manipulate an individual face region by modifying its coefficients $\mathbf{v}_r$ and $\mathbf{t}_r$. Based on the PCA models of the 8 face regions, a 3D face $(\mathbf{V}, \mathbf{T})$ is parameterized as a tuple $\theta = (\mathbf{v}_1, \mathbf{v}_2, \cdots, \mathbf{v}_8, \mathbf{t}_1, \mathbf{t}_2, \cdots, \mathbf{t}_8)$ containing the coefficients.

Facial Attributes. Although different faces can be synthesized by changing the face coefficients $\mathbf{v}_i$ and $\mathbf{t}_i$, in general these coefficients do not correspond to geometry and texture facial attributes that can be intuitively controlled by a human modeler for changing a face's outlook. It would be desirable to devise a number of facial attributes in accordance with human language (*e.g.* , "changing the mouth to be wider"), to facilitate designers in interactively modifying a 3D face,

and to allow our optimizer to learn from and mimic human artists on the tasks of modifying a face with respect to personality impression.

We describe how the effect of changing a facial attribute a can be captured and subsequently applied for modifying a face. For simplicity, we assume that each facial attribute is defined only in one face region rather than across regions. Based on a set of exemplar faces $\{(\mathbf{V}_i, \mathbf{T}_i)\}$ from the Basel Face Model database with assigned facial attribute a, we compute the sums:

$$\Delta \mathbf{V}_a = \frac{1}{A} \sum_{i=1} \mu_i (\mathbf{V}_i - \bar{\mathbf{V}}), \ \Delta \mathbf{T}_a = \frac{1}{A} \sum_{i=1} \mu_i (\mathbf{T}_i - \bar{\mathbf{T}}), \quad (2)$$

where $\bar{\mathbf{V}}$ and $\bar{\mathbf{T}}$ are the average geometry and average texture computed over the whole Basel Face Model dataset. $\mu_i \in [0, 1]$ is the markedness of the attribute in face $(\mathbf{V}_i, \mathbf{T}_i)$, which is manually assigned. $A = \sum_{i=1} \mu_i$ is the normalization factor. Given a face $(\mathbf{V}, \mathbf{T})$, the result of changing facial attribute a on this face is given by $(\mathbf{V} + \beta \Delta \mathbf{V}_a, \mathbf{T} + \beta \Delta \mathbf{T}_a)$, where β is a parameter for controlling the extent of applying facial attribute a.

In total, we devise 160 facial attributes. Each attribute is modeled by 5 example faces. We demonstrate the effect of each attribute on an example face. It is worth noting that the representation of a 3D face can be replaced by other 3D face representations that provide controls of a face.

Optimization Objectives. We synthesize a 3D face to give a desired personality impression by an optimization process, which considers two factors: (1) **Personality Impression**: how likely the synthesized face gives the desired personality impression. (2) **Similarity Metric**: how similar the synthesized face is with the input face.

Given an input 3D face and a desired personality impression type, our approach synthesizes a 3D face which gives the desired personality impression by minimizing a total cost function:

$$\mathbf{C}(\theta) = (1 - \lambda) \mathbf{C}_{\mathrm{p}}(I_\theta, P) + \lambda \mathbf{C}_{\mathrm{s}}(I_\theta, I_{\mathrm{i}}), \quad (3)$$

where $\theta = (\mathbf{v}_1, \mathbf{v}_2, \cdots, \mathbf{v}_8, \mathbf{t}_1, \mathbf{t}_2, \cdots, \mathbf{t}_8)$ contains the face coefficients for synthesizing a 3D face. $\mathbf{C}_{\mathrm{p}}(\cdot)$ is the personality impression cost term for evaluating image I_θ of the face synthesized from θ with regard to the desired personality impression type P. The face image is rendered using the frontal view of the face. Lambertian surface reflectance is assumed and the illumination is approximated by second-order spherical harmonics (Ramamoorthi and Hanrahan 2001). $\mathbf{C}_{\mathrm{s}}(\cdot)$ is the similarity cost term, which measures the similarity between the image I_θ of the synthesized face and the image I_{i} of the input face, constraining the deformation of the input face during the optimization. λ is a trade-off parameter to balance the costs of personality impression and similarity.

Personlaity Impression Classification

To compute the personality impression cost $\mathbf{C}_{\mathrm{p}}$ for a synthesized face in each iteration of the optimization, we leverage modern deep CNN with high-end performances and train a classifier for each personality impression type, which provides a score for the synthesized face with regard to

the personality impression type. To achieve this, we create a face image dataset annotated with personality impression labels based on CASIA WebFace database (Yi et al. 2014), which consists of $10,000$ face images covering both genders and different ethnicities. Then, we fine-tune GoogLeNet (Szegedy et al. 2015) with a personality impression classification task on the dataset.

Learning. We construct our network based on the original GoogLeNet with pre-trained parameters. The network is 22 layers deep with 5 average pooling layers. It has a fully connected layer with $1,024$ units and rectified linear activation. The images with the corresponding labels in the personality impression dataset are fed to the network and an average classification loss is applied. Thus, the original GoogLeNet model is fine-tuned to adapt to a personality impression classification task.

We use a GPU-based engine and implement asynchronous stochastic gradient descent with 0.9 momentum in the fine-tuning stage and a fixed learning rate schedule (with learning rate decreased by 4% every 8 epochs). The mini-batch size is 128.

After fine-tuning, each face image of our dataset is propagated to the fine-tuned network, and a $1,024$-dimension feature vector $\mathbf{g}$ is extracted from the 22-nd layer. Based on those feature vectors, our approach learns a linear SVM classifier for each personality impression type.

Face Similarity Metric

To constrain the synthesized face to look similar to the input face, we evaluate the similarity between the image I_θ of the synthesized face and image I_i of the original input face in the optimization. To achieve this, we train a Siamese network (Chopra, Hadsell, and LeCun 2005), an end-to-end network, to evaluate whether a pair of face images correspond to the same face. The network learns a feature extractor which takes face images and outputs feature vectors, such that the feature vectors corresponding to images of the same face are close to each other, while those corresponding to images of different faces are far away from each other.

We train the Siamese network using the LFW dataset (Huang et al. 2007). The training dataset is constructed as $\{(I_a, I_b, l)\}$, where I_a and I_b are any two images from the LFW dataset, and l is the label. If I_a and I_b are from the same face, $l = 1$, otherwise $l = 0$.

The Siamese network consists of two identical Convolutional Networks that share the same set of weights W. The training process learns the weights W by minimizing a loss function $\mathcal{L} = l\mathcal{L}_1 + (1 - l)\mathcal{L}_2$, where $\mathcal{L}_1 = \|G_W(I_a) - G_W(I_b)\|$ and $\mathcal{L}_2 = \max(0, \rho - \|G_W(I_a) - G_W(I_b)\|)$. $G_W(I)$ is the mapped features of an input face image I, which are synthesized by the learned identical Convolutional Network. By minimizing the loss function $\mathcal{L}$, the distance between the mapped features of I_a and I_b is driven by $\mathcal{L}_1$ to be small if I_a and I_b correspond to the same face, and is driven by $\mathcal{L}_2$ to be large vice versa. The constant ρ is set as 2.0. The parameters are learned by standard cross entropy loss and back-propagation of the error.
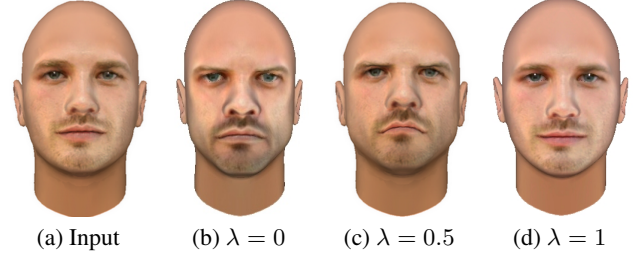

(a) Input   (b) $\lambda = 0$ (c) $\lambda = 0.5$ (d) $\lambda = 1$

Figure 2: The influence of λ on the synthesized face when optimizing an example face to give a hostile personality impression.

Cost Functions

Given a textured 3D face model and a desired personality impression type as the input, our approach employs a data-driven MCMC sampler to update the face coefficients θ iteratively so as to modify the face. In each iteration, the synthesized face represented by θ is evaluated by the total cost $\mathbf{C}(\cdot) = (1 - \lambda)\mathbf{C}_p(\cdot) + \lambda\mathbf{C}_s(\cdot)$. The optimization continues until a face giving the desired personality impression is synthesized. We discuss the personality impression cost $\mathbf{C}_p$ and the similarity cost $\mathbf{C}_s$ in the following.

Personality Impression Cost. The image I_θ of the face synthesized by face coefficients θ is evaluated with respect to the desired personality impression type P in the cost function $\mathbf{C}_p$, defined based on the fine-tuned GoogLeNet:

$$\mathbf{C}_p(I_\theta, P) = 1 - \frac{\exp(x_1)}{\exp(x_1) + \exp(x_2)}, \qquad (4)$$

where $[x_1, x_2]^T = \mathbf{w}_P^T \mathbf{g}$ is the output of the full connected layer of the fine-tuned network. x_1 and x_2 reflect the possibilities of the image I_θ belonging to the personality impression type P or not, respectively. $\mathbf{g} \in \mathbf{R}^{1,024}$ is the face feature vector of I_θ on the 22-nd layer of the network; $\mathbf{w}_P \in \mathbf{R}^{1,024 \times 2}$ contains the parameters of the full connected layer, which map the feature vector $\mathbf{g}$ to a 2D vector (our fine-tuned network is a two-category classifier).

A low cost value means the synthesized face image gives the desired type of personality impression, according to the classifier trained by face images annotated with personality impression labels.

Similarity Cost. We want to constrain the synthesized face to look similar to the input face. To achieve this, we apply the Siamese network trained for evaluating the similarity between a pair of face images to define a similarity cost as a soft constraint of the optimization:

$$\mathbf{C}_s(I_\theta, I_i) = \frac{1}{G}\|G_W(I_\theta) - G_W(I_i)\|, \qquad (5)$$

where $G_W(I_\theta)$ and $G_W(I_i)$ are the feature vectors of the image I_θ of the synthesized face and the image I_i of the input face computed by the Siamese network. $G = \max(\{\|G_W(I) - G_W(I_i)\|\})$ is a normalization factor computed over all face images I from the LFW dataset. A low

cost value means that the synthesized face image I_θ is similar to the input face image I_i.

To demonstrate how the similarity cost and personality impression cost affect the face synthesis results during the optimization, we do an ablation study of optimizing a face model with the personality impression type of hostile in Figure 2. When the trade-off parameter λ is set as 0, the face is optimized to become more hostile-looking yet it differs from the input face significantly. When λ is set as 0.5, the face is optimized to look somewhat hostile and it resembles the input face closely. When λ is set as 1, the face resembles the input one more closely, but it shows less hostile. We can find from the results that a larger λ constrains the synthesized face to resemble the input face more closely and show fewer personality impression.In our experiments, we set $\lambda = 0.5$ by default.

Face Synthesis by Optimization

We use a Markov chain Monte Carlo (MCMC) sampler to explore the space of face coefficients efficiently. As the top-down nature of MCMC sampling makes it slow due to the initial "burn-in" period, we devise a data-driven MCMC sampler for our problem. We propose two types of data-driven Markov chain dynamics: Region-Move and Prior-Move, corresponding to local refinement and global reconfiguration of the face.

Region-Move. We want to learn from how human artists modify faces to give a certain personality impression, so as to enable our sampler to mimic such modification process during an optimization. Considering that each face region's contribution to a specified personality impression is different, we devise a Region-Move which modifies a face according to "important" face regions likely to be associated with the specified personality impression in training data.

Our training data is created based on 5 face models. We recruited 10 artists who are familiar with face modeling (with 5 to 10 years of experience in avatar design and 3D modeling). Each artist was asked to modify each of the 5 face models to give the 8 personality impression types by controlling the facial attributes. After the manual modifications, we project the original 5 face models and all the manually modified face models into the PCA spaces, so that each face can be represented by its face coefficients θ.

For each personality type, let $\Delta\theta = (\Delta\mathbf{v}_1, \cdots, \Delta\mathbf{v}_8, \Delta\mathbf{t}_1, \cdots, \Delta\mathbf{t}_8)$ contain the sums of face coefficients differences for the 8 face regions. $\Delta\mathbf{v}_r = \sum ||\mathbf{v}_r - \mathbf{v}'_r||$ is the sum of differences of the geometry coefficients of the r-th face region, where $\mathbf{v}_r$ and $\mathbf{v}'_r$ are the geometry coefficients of the original face model and a face model modified by an artist respectively. The sum of differences of the texture coefficients $\Delta\mathbf{t}_r$ is similarly defined. Suppose the current face is $(\mathbf{V}, \mathbf{T})$ with face coefficients θ. During sampling, a face region r is selected with probability $0.5\frac{\Delta\mathbf{v}_r}{\sum \Delta\mathbf{v}_i} + 0.5\frac{\Delta\mathbf{t}_r}{\sum \Delta\mathbf{t}_i}$. Then a facial attribute a in face region r is randomly selected and modified so as to create a new face $(\mathbf{V} + \beta\Delta\mathbf{V}_a, \mathbf{T} + \beta\Delta\mathbf{T}_a)$ with new face coefficients θ', where $\beta \sim U(-1.0, 1.0)$. The changes $\Delta\mathbf{V}_a$ and $\Delta\mathbf{T}_a$ are learned in Section Facial Attribute for

each facial attribute a.

Essentially, a face region that is more commonly modified by artists to achieve the target personality impression type is modified by our sampler with a higher probability.

Prior-Move. We also leverage the personality impression dataset to learn a prior distribution of the face coefficients θ for each personality impression, so as to guide our sampler to sample face coefficients near the prior face coefficients, which likely induce a similar personality impression.

For each personality impression type P, we estimate a prior distribution with the following steps:

(1) Select images in the personality impression dataset which are annotated with the personality impression type P; and form a subset $D_P = \{I_d\}$.

(2) Reconstruct the corresponding 3D face model for each image $I_d \in D_P$ by the implementation of (Blanz and Vetter 1999; Blanz and Vetter 2003). These 3D face models are projected onto the PCA spaces and are represented using face coefficients. Thus, we form a face coefficients set $\Theta_P = \{\theta_d\}$.

(3) Fit a normal distribution for each of the geometry and texture coefficients ($\mathbf{v}_r$ and $\mathbf{t}_r$) of each face region r based on Θ_P.

Given the prior distribution, our sampler draws a value from the normal distribution of each of the geometry and texture coefficients, to generate new face coefficients θ'.

Optimization. We apply simulated annealing with a Metropolis-Hastings state-searching step to search for face coefficients θ that minimize the total cost function C. In each iteration of the optimization, one type of moves is selected and applied to propose new face coefficients θ', which is evaluated by the total cost function $\mathbf{C}$. The Region-Move and Prior-Move are selected with probabilities α and $1 - \alpha$ respectively. In our experiments, we set $\alpha = 0.8$ by default. The proposed face coefficients θ' generated by the move are accepted according to the Metropolis criterion:

$$Pr(\theta'|\theta) = \min\left\{1, \frac{f(\theta')}{f(\theta)}\right\}, \qquad (6)$$

where $f(\theta) = \exp^{-\frac{1}{t}C(\theta)}$ is a Boltzmann-like objective function and t is the temperature parameter of the annealing process. By default, we empirically set t to 1.0 and decrease it by 0.05 every 10 iterations until it reaches zero. We terminate the optimization if the absolute change in the total cost value is less than 5% over the past 20 iterations. In our experiments, a full optimization takes about $100 - 150$ iterations (about 15 seconds) to finish.

Experiments

We conducted experiments on a Linux machine equipped with an Intel i7-5930K CPU, 32GB of RAM and a Nvidia GTX 1080 graphics card. The optimization and learning components of our approach were implemented in C++.

Results and Discussion

We test our approach to synthesizing different faces to give different personality impressions. Figure 3 shows the input
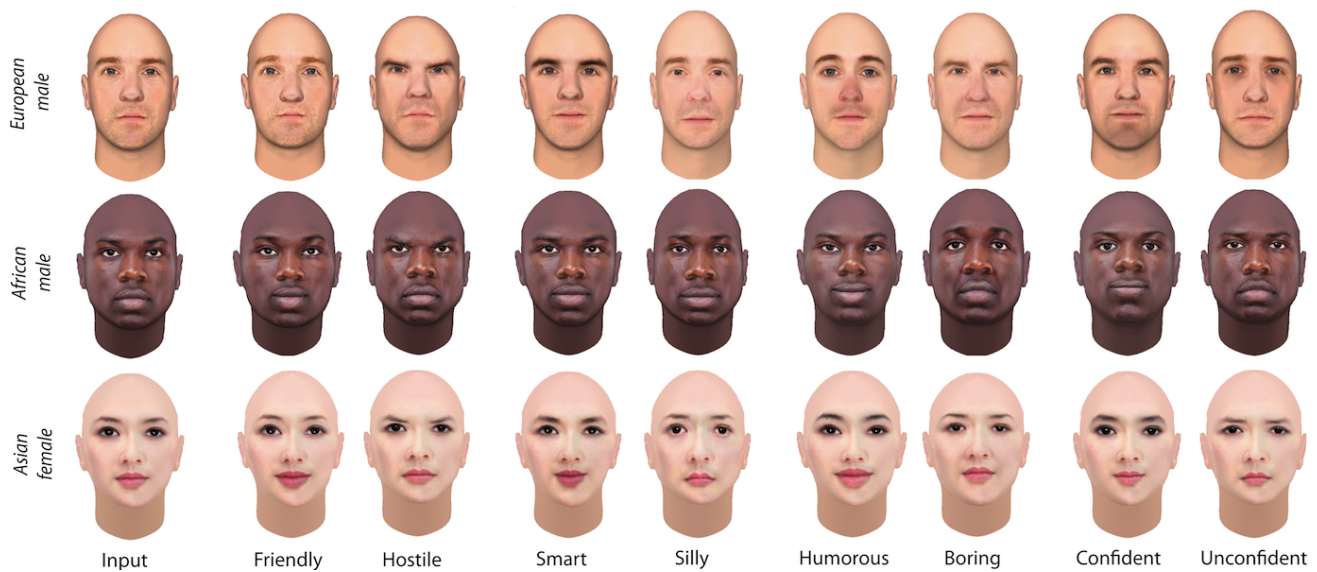
Figure 3: Results of synthesizing faces with different personality impression types.

faces and the synthesized faces. The input faces consist of an *European male* face, an *African male* face, and an *Asian female* face. For each input face, a face is synthesized using each of the 8 impression types.

We observe some interesting features that may result in the corresponding personality impressions. For instance, comparing the results of confident and unconfident faces, we observe that the confident faces usually have a higher nose bridge and bigger eyes. In addition, the eyebrows also look sharp and slightly slanted, which make a person look like in a state of concentration. The mouth corners lift slightly and the mouths show a subtle smile. As for the unconfident faces, the eyebrows are generally dropping or furrowed, showing a subtle sign of nervousness. The eye corners are also dropping, and the eyes look tired. The cheeks generally look more bonier. The mouths are also drooping, which could be perceived as signs of frustration.

We observe that usually a combination of facial features accounts for the personality impression of a face. As there are as many as 160 facial attributes, it is rather hard to manually tune these attributes to model a face. The CNN classifiers effectively learn the relationships between facial features and a personality impression type, such that they can drive face synthesis by personality impression automatically.

Perceptual Studies

We conducted perceptual studies to evaluate the quality of our results. The major goal is to verify if the perceived personality impressions of the synthesized faces match with the personality impression types. We recruited 160 participants from different countries via Amazon Turk. They are evenly distributed by gender and are aged 18 to 50. Each participant was shown some synthesized faces and was asked about the personality impression they perceived. Definitions of the personality types from a dictionary were shown as reference.

Recognizing Face Personality Impression. In this study, we want to verify if the personality impression types of the synthesized faces agree with human impressions. We used the faces from Figure 3. Each of these faces was synthesized using a single personality impression type and was voted by 40 human participants. In voting for the personality impression type of a face, a participant needed to choose 1 out of the 8 personality impression types used in our approach. In total, we obtained $1,600$ votes for 40 faces.

Figure 4 shows the results as a confusion matrix. The average accuracy is about 38.0%, compared to the chance-level accuracy of 12.5% (since we have 8 personality impression types). For each personality impression type, the matching type gets the highest number of votes as shown by diagonal.

"Friendly" and "hostile" receive a relatively high accuracy (about $45\%-50\%$), probably because the facial features leading to such personality impressions are more prominent and easily recognizable. For example, participants usually perceive a face as hostile-looking when they see dense moustache, slanted eyebrows and a drooping mouth. For other personality impressions such as humorous and boring, the accuracy is relatively lower (about 33%), probably because the facial features leading to such personality impressions are less apparent, or because the participants do not have a strong association between facial features and such personality impressions. The facial features of some personalities are overlapped, which makes some people have several different, but similar personalities. For instance, a smart person may also look confident. Thus, participants may choose a similar which reduces the total accuracy.

We also perform a t-tests on the results of perceptual study. Our null hypothesis H_0 was that participants can not recognize the personality impression type of the synthesized faces and the recognition rate is at chance level. All tests have p-values less than 0.00001. Therefore, we reject the null hypothesis H_0 in this experiment, which concludes that
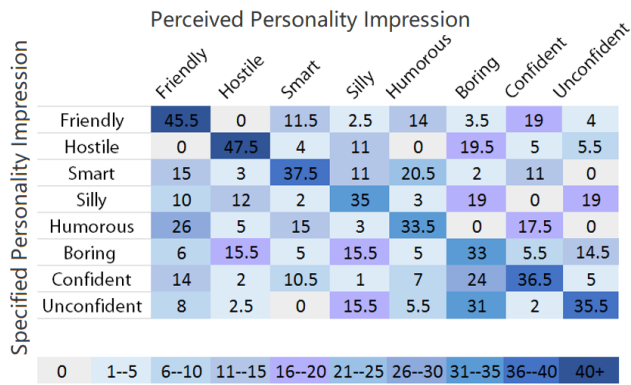
Figure 4: Accuracy of determining a single personality impression type of faces synthesized in the perceptual study. Percentages of votes are shown.

Specified \ Perceived	Friendly	Hostile	Smart	Silly	Humorous	Boring	Confident	Unconfident
Friendly	45.5	0	11.5	2.5	14	3.5	19	4
Hostile	0	47.5	4	11	0	19.5	5	5.5
Smart	15	3	37.5	11	20.5	2	11	0
Silly	10	12	2	35	3	19	0	19
Humorous	26	5	15	3	33.5	0	17.5	0
Boring	6	15.5	5	15.5	5	33	5.5	14.5
Confident	14	2	10.5	1	7	24	36.5	5
Unconfident	8	2.5	0	15.5	5.5	31	2	35.5

Legend: 0 | 1--5 | 6--10 | 11--15 | 16--20 | 21--25 | 26--30 | 31--35 | 36--40 | 40+

Figure 6: Accuracy of determining the personality impression types of synthesized faces with expressions in the perceptual study. Percentages of votes for the answers are shown.

Specified \ Perceived	Friendly	Hostile	Smart	Silly	Humorous	Boring	Confident	Unconfident
Friendly (Surprise)	30	5	5	30	5	15	10	0
Friendly (Angry)	30	15	30	0	0	5	15	5
Hostile (Happy)	25	35	10	5	0	15	5	5
Hostile (Sad)	0	35	5	20	12.5	12.5	0	15
Smart (Disgust)	10	25	35	0	12.5	7.5	10	0
Smart (Surprise)	10	20	40	0	10	5	15	0
Silly (Angry)	5	25	0	40	10	5	15	0
Silly (Fear)	20	2.5	5	35	0	22.5	0	15
Humorous (Happy)	20	0	40	5	20	0	10	5
Humorous (Fear)	10	0	15	10	40	5	15	5
Boring (Happy)	10	5	15	10	15	25	15	5
Boring (Surprise)	15	10	0	30	0	35	10	0
Confident (Surprise)	5	15	10	0	27.5	5	37.5	0
Confident (Angry)	10	32.5	5	0	10	10	32.5	0
Unconfident (Sad)	0	2.5	0	42.5	0	15	5	35
Unconfident (Fear)	0	25	0	20	10	5	10	30

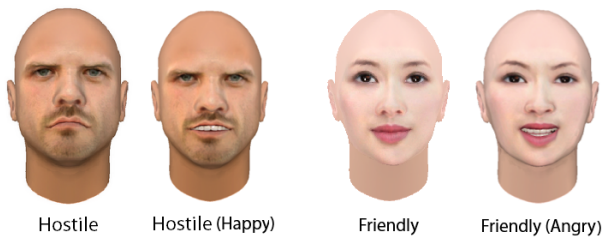Legend: 0 | 1--5 | 6--10 | 11--15 | 16--20 | 21--25 | 26--30 | 31--35 | 36--40 | 40+

Figure 5: Example faces with expressions used in the perceptual study.

the participants can recognize the personality impression types of the synthesized faces out of the eight used types.

Influence of Expression. Next we want to investigate whether facial expression changes will affect the personality impression of the synthesized faces. For example, does a face optimized to be hostile-looking still look hostile with a happy smile? Such findings could bring interesting insights for designing virtual character faces.

We conducted an empirical study to investigate the effects of expressions on the synthesized faces. In our study, 16 faces optimized with respect to each of the 8 personality impression types were used. Each face was tuned to show a facial expression chosen from happy, sad, angry, surprise, disgust or fear with Maya. Figure 5 shows some of the examples. Each face was voted by 40 human participants about its personality impression types out of the eight types used in our approach. In total 640 votes were collected for the 16 faces used.

Figure 6 shows the accuracy of determining the personality impression types on the synthesized faces with an expression. The average accuracy is about 33.4% (a drop from 38% on synthesized faces without any expression). Facial expressions do have an impact on some personality impressions. For example, with an angry expression, a face optimized to be friendly-looking may appear hostile. The accuracy of the friendly (angry) face is 30.0%; compared to the accuracy of 45.5% on the friendly face without any expression (Figure 4), the accuracy drops by 15.5%. However, the personality impression on confident-looking faces seems to
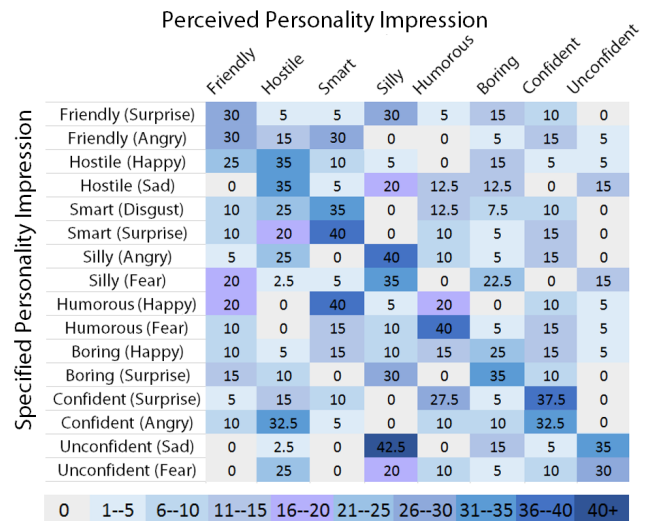
be relatively unaffected by facial expressions. For instance, even with an angry expression, a face optimized to look confident still has 32.5% votes for confident. This is probably because people have strong associations between certain facial features and "confident", and those facial features are still apparent under facial expression changes.

Though this study is not comprehensive, it gives some good insights about the effects of expressions on personality impression. We believe that a more comprehensive perceptual study will be an interesting avenue for future research.

Summary

Limitations. To stay focused on face's geometry and texture, we do not consider the influence of hair, accessories or clothing on personality impression. Besides, speech and facial movements, as well as head and body poses, can also influence the impression of one's personality, just as experienced actors can change the personality impressions they make by controlling speech, facial expression and body movements. While we only focus on static facial features in this work, we refer the reader to recent interesting efforts on adding personality to human motion (Durupinar et al. 2017).

Future Work. Our face synthesis approach could be extended to consider more personality impression types and other high-level perceptual factors. Synthesizing faces of cartoon characters to give certain personality impressions is also an interesting problem to explore, though this could be more challenging due to the lack of abundant cartoon character faces training data and the fact that different cartoon character faces may look drastically different. Our approach follows the discriminative criteria to train the personality impression classifier. For future work, it would be interesting to investigate applying a deep generative network for synthesizing 3D faces with personality impression, as the adversarial training approach (GAN) (2014) has witnessed good

successes in 2D image generation.

Acknowledgements

References

Asch, S. E. 1946. Forming impressions of personality. *The Journal of Abnormal and Social Psychology* 41(3):258.

Blanz, V., and Vetter, T. 1999. A morphable model for the synthesis of 3d faces. In *SIGGRAPH*, 187–194.

Blanz, V., and Vetter, T. 2003. Face recognition based on fitting a 3d morphable model. *IEEE PAMI* 25(9):1063–1074.

Chopra, S.; Hadsell, R.; and LeCun, Y. 2005. Learning a similarity metric discriminatively, with application to face verification. In *CVPR*.

Durupinar, F.; Kapadia, M.; Deutsch, S.; Neff, M.; and Badler, N. I. 2017. Perform: Perceptual approach for adding ocean personality to human motion using laban movement analysis. *TOG* 36(1):6.

Eisenthal, Y.; Dror, G.; and Ruppin, E. 2006. Facial attractiveness: Beauty and the machine. *Neural Computation* 18(1):119–142.

Goodfellow, I.; Pouget-Abadie, J.; Mirza, M.; Xu, B.; Warde-Farley, D.; Ozair, S.; Courville, A.; and Bengio, Y. 2014. Generative adversarial nets. In *NIPS*. Curran Associates, Inc. 2672–2680.

Gray, D.; Yu, K.; Xu, W.; and Gong, Y. 2010. Predicting facial beauty without landmarks. In *ECCV*, 434–447.

Hassin, R., and Trope, Y. 2000. Facing faces: studies on the cognitive aspects of physiognomy. *Journal of Personality and Social Psychology* 78(5):837.

Hu, L.; Saito, S.; Wei, L.; Nagano, K.; Seo, J.; Fursund, J.; Sadeghi, I.; Sun, C.; Chen, Y.-C.; and Li, H. 2017. Avatar digitization from a single image for real-time rendering. *TOG* 36(6):195.

Huang, G. B.; Ramesh, M.; Berg, T.; and Learned-Miller, E. 2007. Labeled faces in the wild: A database for studying face recognition in unconstrained environments. *Technical Report 07-49*.

Huang, H.; Kalogerakis, E.; Yumer, E.; and Mech, R. 2017. Shape synthesis from sketches via procedural models and convolutional networks. *TVCG* 23(8):2003–2013.

Joo, J.; Steen, F. F.; and Zhu, S.-C. 2015. Automated facial trait judgment and election outcome prediction: Social dimensions of face. In *ICCV*.

Kalogerakis, E.; Chaudhuri, S.; Koller, D.; and Koltun, V. 2012. A probabilistic model for component-based shape synthesis. *TOG* 31(4):55.

Le, N.; Why, Y.; and Ashraf, G. 2011. Shape stylized face caricatures. *Advances in Multimedia Modeling* 536–547.

Marsella, S.; Xu, Y.; Lhommet, M.; Feng, A.; Scherer, S.; and Shapiro, A. 2013. Virtual character performance from speech. In *Proceedings of Eurographics Symposium on Computer Animation*, 25–35.

Mischel, W. 2013. *Personality and assessment*. Psychology Press.

Naumann, L. P.; Vazire, S.; Rentfrow, P. J.; and Gosling, S. D. 2009. Personality judgments based on physical appearance. *Personality and Social Psychology Bulletin* 35(12):1661–1671.

Over, H., and Cook, R. 2018. Where do spontaneous first impressions of faces come from? *Cognition* 170:190–200.

Paysan, P.; Knothe, R.; Amberg, B.; Romdhani, S.; and Vetter, T. 2009. A 3d face model for pose and illumination invariant face recognition. In *Advanced Video and Signal Based Surveillance*, 296–301.

Ramamoorthi, R., and Hanrahan, P. 2001. An efficient representation for irradiance environment maps. In *SIGGRAPH*, 497–500.

Ritchie, D.; Mildenhall, B.; Goodman, N. D.; and Hanrahan, P. 2015. Controlling procedural modeling programs with stochastically-ordered sequential monte carlo. *TOG* 34(4):105.

Saito, S.; Wei, L.; Hu, L.; Nagano, K.; and Li, H. 2017. Photorealistic facial texture inference using deep neural networks. In *CVPR*.

Sohre, N.; Adeagbo, M.; Helwig, N.; Lyford-Pike, S.; and Guy, S. 2018. Pvl: A framework for navigating the precision-variety trade-off in automated animation of smiles. In *AAAI*.

Suwajanakorn, S.; Seitz, S. M.; and Kemelmacher-Shlizerman, I. 2015. What makes tom hanks look like tom hanks. In *ICCV*.

Szegedy, C.; Liu, W.; Jia, Y.; Sermanet, P.; Reed, S.; Anguelov, D.; Erhan, D.; Vanhoucke, V.; and Rabinovich, A. 2015. Going deeper with convolutions. In *CVPR*, 1–9.

Talton, J. O.; Lou, Y.; Lesser, S.; Duke, J.; Měch, R.; and Koltun, V. 2011. Metropolis procedural modeling. *TOG* 30(2):11.

Tian, L., and Xiao, S. 2016. Facial feature exaggeration according to social psychology of face perception. *Computer Graphics Forum* 35(7):391–399.

Vernon, R. J.; Sutherland, C. A.; Young, A. W.; and Hartley, T. 2014. Modeling first impressions from highly variable facial images. *Proceedings of the National Academy of Sciences* 111(32):E3353–E3361.

Willis, J., and Todorov, A. 2006. First impressions making up your mind after a 100-ms exposure to a face. *Psychological Science* 17(7):592–598.

Xie, D.; Liang, L.; Jin, L.; Xu, J.; and Li, M. 2015. Scut-fbp: A benchmark dataset for facial beauty perception. In *SMC*, 1821–1826.

Xu, J.; Jin, L.; Liang, L.; Feng, Z.; and Xie, D. 2015. A new humanlike facial attractiveness predictor with cascaded fine-tuning deep learning model. *arXiv:1511.02465*.

Yi, D.; Lei, Z.; Liao, S.; and Li, S. Z. 2014. Learning face representation from scratch. *arXiv:1411.7923*.

Zell, E.; Aliaga, C.; Jarabo, A.; Zibrek, K.; Gutierrez, D.; McDonnell, R.; and Botsch, M. 2015. To stylize or not to

stylize?: the effect of shape and material stylization on the perception of computer-generated faces. *TOG* 34(6):184.