Realistic Dynamic Facial Textures from a Single Image using GANs

Kyle Olszewski[*1,3,4], Zimo Li[†1], Chao Yang [‡1], Yi Zhou[§1], Ronald Yu[¶1,3], Zeng Huang[‖1], Sitao Xiang[**1], Shunsuke Saito[††1,3], Pushmeet Kohli[‡‡2], and Hao Li[§§1,3,4]

[1]University of Southern California
[2]Microsoft Research
[3]Pinscreen
[4]USC Institute for Creative Technologies

Abstract

We present a novel method to realistically puppeteer and animate a face from a single RGB image using a source video sequence. We begin by fitting a multilinear PCA model to obtain the 3D geometry and a single texture of the target face. In order for the animation to be realistic, however, we need dynamic per-frame textures that capture subtle wrinkles and deformations corresponding to the animated facial expressions. This problem is highly underconstrained, as dynamic textures cannot be obtained directly from a single image. Furthermore, if the target face has a closed mouth, it is not possible to obtain actual images of the mouth interior. To address this issue, we train a Deep Generative Network that can infer realistic per-frame texture deformations, including the mouth interior, of the target identity using the per-frame source textures and the single target texture. By retargeting the PCA expression geometry from the source, as well as using the newly inferred texture, we can both animate the face and perform video face replacement on the source video using the target appearance.

1. Introduction

Many methods have been developed in recent years to realistically render and animate faces that are then combined with real images. Such techniques have a variety of applications, ranging from special effects for film and television to the recent and controversial phenomenon of real images and videos of prominent public figures being surreptitiously modified to create "fake news." Recent efforts towards achieving such capabilities include, but are not limited to, video rewriting [2], face replacement [6], and real-time video reenactment [35]. Though the aforementioned works achieve many of their stated goals, they require a video of the target subject whose face is to be modified as input. Thus, they cannot be used for target subjects for whom high resolution video sequences do not exist and are unobtainable.

A naive approach would be to simply fit a 3D mesh to the face of the target subject in the image, and animate it using the expressions of another person (whom we call the "driver"), using the static texture captured from the original image for the entire animation. However, in this approach, subtle wrinkles that do not appear in the initial image and are too small to be represented by deformations in the 3D mesh will not appear in the resulting animations. Realistic animation requires these wrinkles to form and disappear corresponding to the appearance of the target subject and the expression that is being performed.

Our goal in this paper is to generate dynamic per-frame facial textures that can represent details such as the inner mouth and wrinkles from a single RGB image. Needless to say, this problem is highly underconstrained. In fact, it is impossible to perfectly solve for fine scale geometry, let alone temporal facial deformations, from a single image. For example, if the target image has a closed mouth, then there is no way that we can actually know what the teeth, tongue, or inner mouth look like. We must therefore infer these details. Likewise, it is impossible to directly tell the type of wrinkles that form under different expressions from

[*]olszewski.kyle@gmail.com (equal contribution)
[†]zimoli@usc.edu (equal contribution)
[‡]harryyang.hk@gmail.com (equal contribution)
[§]zhou859@usc.edu
[¶]ronaldyu@usc.edu
[‖]zenghuan@usc.edu
[**]sitaoxia@usc.edu
[††]shunsuke.saito16@gmail.com
[‡‡]pkohli@microsoft.com
[§§]hao@hao-li.com

looking at a single texture map from a neutral expression.

To address this, we employ deep learning to infer the dynamic texture deformations necessary for a realistic animation. In particular, we leverage the power of the recently popularized Generative Adversarial Framework [15] in order to infer realistic and high-resolution texture deformations transferred from a sequence of source expression textures to a target identity texture. These texture deformations include the inference of the inner mouth and teeth, which are included in the texture training data. After inferring the mouth, we refine it using optical flow from the source video's mouth region.

After learning texture deformations, we are able to re-render a face with realistic wrinkling, expressions, and teeth using the source video, as well as compositing the newly rendered appearance back onto the source video to replace the original actor using the schema of [6]. This is, to the best of our knowledge, the first time realisitic animation and video compositing has been achieved using only a single target image. Our contributions are summarized as follows:

1. We propose a novel framework to generate realistic dynamic textures using a conditional generative adversarial network to infer detailed deformations in texture space such as blinking, teeth, tongue, wrinkles, and lip motion from a single image.

2. Our system can composite the target face image onto the faces in the source videos with new realistic appearances.

3. We introduce a dataset of synchronized, high-resolution (1920x1080) videos of captured subjects with varying facial appearances saying the Harvard Sentences [16] and making a set of facial expressions based on the Facial Action Coding System (FACS) [8]. We plan to release this dataset to the public in the future.

2. Related Work

2.1. Facial Retargetting and Enactment

Recent advances in capture technology such as [3, 35, 21] have been able to capture faces with high-fidelity using only monocular input. [29] uses a head-mounted rig and a mouth-camera to regress model PCA expression coefficients for high-fidelity speech animation.

[35] reenacts a video sequence by using multiple frames of the target video sequence to construct a geometric identity of the target. Similarly, [2] adopts a multi-scale approach to replace a face from one video sequence with the appearance from a different video sequence. The limitation here is that the animated face's identity is fixed - the system cannot puppeteer an arbitrary person from an image. To address this issue, [6] switches the appearance of one person's face onto another in a realistic fashion. In [26], the authors compute a nonlinear temporal synchronization between two videos of the same actor that can be blended for a novel performance. The work of [10] can replace the face of the actor in a target video with the face of a source video. The authors of [12] are able to reconstruct detailed 3D facial geometry from a monocular video. However, all these methods require a video of the target in order to do photo-realistic facial retargetting, rather than using only a single target image as in our method.

[32] learns a controllable model of a face from an image collection that can be puppeteered. Though it does not require video, it does require many different images to work. [19] uses a deep network to swap faces between two images.

2.2. Capturing and Retargeting Photorealistic Mouth Interior

Capturing and modeling the mouth region has always been a challenge. Existing works include [5, 36, 33, 9, 7], which use audio input to render the mouth region. [37] fits a 3D model to reconstruct a teeth model from a photograph as input. [11] and [34] improve the photorealism of their 3D teeth models with 3D textured teeth proxies. These works are not well suited for photo-realistic teeth retargeting because they only track the mouth of the source image and do not consider a target image at all. In their recent work, [35] fills in the inner mouth region by searching for similar mouth frames in a target video. In contrast with previous work, we do not need a target video to infer photo-realistic teeth texture. Instead, we only need a single RGB image of the target image that does not necessarily contain teeth and use a conditional GAN to infer contents in the mouth which are further upsampled and refined by flowing the pixels from the source video.

2.3. Deep Generative Model for Texture Synthesis

The rise of deep learning has brought many advances in image synthesis. Generative Adversarial Models [15] have proved especially capable of generating sharp and realistic images. More specifically, we use a conditional GAN [27] to enable end-to-end synthesis of high-resolution facial textures conditioned on the target identity and source facial expressions. While conditional GANs have been used for a wide range of applications such as image-to-image translation [17], face image generation [13], image inpainting [30], and style transfer [22], to our knowledge conditional generative adversarial models have never been used for the purpose of generating high-detailed textures.

Previous facial synthesis works that do not use conditional GANs include [23], which hallucinates high-resolution faces from low-resolution images using a local patch-based Markov network, and [28], which generates

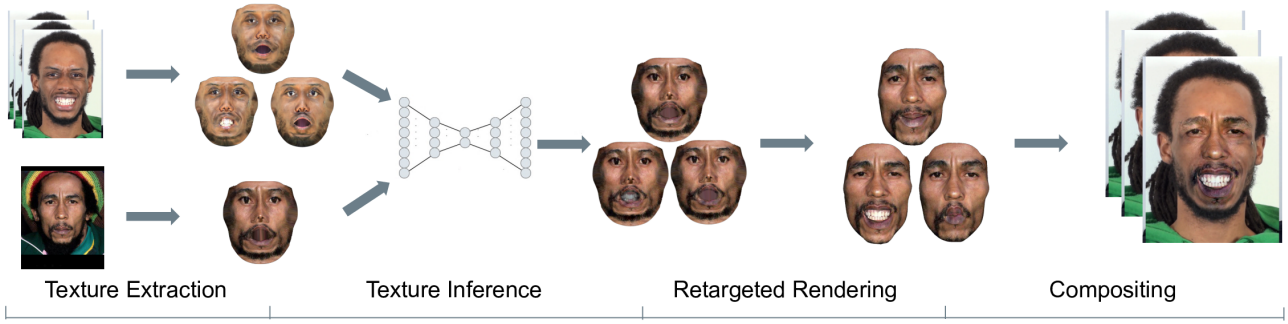


Figure 1. Overview of the pipeline.

novel face images by learning a probabilistic model from a set of training faces. However, neither of these results are realistic and artifact-free in high-resolution. [14] uses statistical models to generate wrinkles and pores on facial geometry, but cannot do so for textures. [31] uses a deep learning framework to generate high resolution textures from a single image, but they cannot infer details like wrinkles and the inner mouth region on their textures for different expressions as we can.

3. Overview

Our pipeline consists of the following steps (illustrated in Fig. 1):

1. Fit a 3D model to extract static albedo textures from each frame in the source video sequence and the single RGB target image (Section 4).

2. Infer dynamic textures and retarget the per-frame texture expressions from the source video frames onto the target image texture using a generative adversarial framework (Section 5).

3. Composite the target mesh with the generated dynamic textures into each frame in the source video (Section 6).

4. Fitting the Face Model

We model the face shape S and albedo as a multilinear PCA model with $n = 53k$ vertices and $106k$ faces:

$$S(\beta_{id}, \beta_{exp}) = \hat{S} + B_{id}\beta_{id} + B_{exp}\beta_{exp} \quad (1)$$

$$I(\alpha_{alb}) = \hat{I} + A_{alb} * \alpha_{alb} \quad (2)$$

The identity and expression are represented as a multivariate normal distribution with $B_{id} \in \mathbf{R}^{3n\times80}$, $B_{exp} \in \mathbf{R}^{3n\times29}$ and $A_{alb} \in \mathbf{R}^{3n\times80}$. The dimensions of the mean shape are $\hat{S} = \hat{S}_{id} + \hat{S}_{exp} \in \mathbf{R}^{3n}$, and the mean albedo is given by $\hat{I} \in \mathbf{R}^{3n}$. The standard deviations are given by: $\sigma_{id} \in \mathbf{R}^{80}$, $\sigma_{exp} \in \mathbf{R}^{29}$ and $\sigma_{alb} \in \mathbf{R}^{80}$.

We use the Basel Face Model [1] for B_{id}, A_{alb}, $\hat{S}$ and $\hat{I}$ as well as FaceWarehouse [4] for B_{exp}. We model the illumination using second order Spherical Harmonics and assume Lambertian surface reflectance. We denote the illumination as $L \in \mathbf{R}^{27}$.

Following the optimization scheme of [35], we jointly solve for all the unknowns $\mathbf{Y} = \{S, I, R, t, P, L\}$ leveraging the Gauss-Newton method applied to iteratively reweighted least squares with three levels of image pyramid, where P are the camera parameters. One can refer to [35] for details of this optimization. In short, our objective function is:

$$E(\mathbf{Y}) = w_{col}E_{col}(\mathbf{Y}) + w_{lan}E_{lan}(\mathbf{Y}) + w_{reg}E_{reg}(\mathbf{Y}) \quad (3)$$

We use energy weights $w_{col} = 1$, $w_{lan} = 10$ and $w_{reg} = 2.5 \times 10^{-5}$. The photo-consistency term is given by

$$E_{col}(\mathbf{Y}) = \frac{1}{|M|}\sum_{p\in M} ||C_{gt}(p) - C_{render}(p)||_2 \quad (4)$$

where C_{gt} is the input image and C_{render} is the synthesized image. $p \in M$ denotes pixel visibility in the source image. The landmark term is given by:

$$E_{lan}(\mathbf{Y}) = \frac{1}{|F|}\sum_{f_i\in F} ||f_i - \Pi_P(RS_i + t)||_2^2 \quad (5)$$

where $f_i \in F$ is a 2D facial feature following the method presented in [18]. The regularization E_{reg} term ensures that faces stay close to the normal distribution. This term prevents degenerative faces when performing the fitting:

$$E_{reg}(\mathbf{Y}) = \sum_{i=1}^{80}[(\frac{\beta_{id,i}}{\sigma_{id,i}})^2 + (\frac{\alpha_{alb,i}}{\sigma_{alb,i}})^2] + \sum_{i=1}^{29}(\frac{\beta_{exp,i}}{\sigma_{exp,i}})^2 \quad (6)$$

5. Dynamic Texture Synthesis

5.1. Deep Learning Framework

The core of our dynamic texture synthesis pipeline for inferring fine details is a Conditional Generative Adversarial Network used to infer deformations from a source texture onto a target texture. Broadly speaking, a Generative

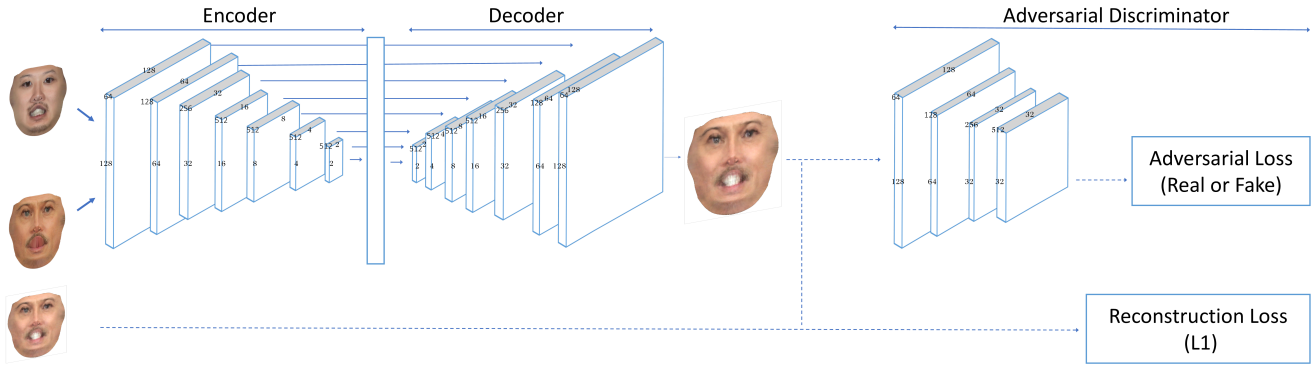


Figure 2. The network architecture. The encoder takes as input the identity texture and the expression texture, and the decoder outputs the dynamic texture. The adversarial discriminator is used during training to judge whether the output texture looks real or not.

Adversarial Network (GAN) G is a function which produces "realistic" output from a noise vector. That is, given a distribution M, a variable $z \sim M$, and a set of ground truth "real" examples X, we would like $G(z)$ to be indistinguishable between any $x \sim X$. Formally, by indistinguishable we mean to say that the generator function fools as best as possible a discriminator function, D, trained precisely to separate the generated output $G(z)$ from the real output drawn from X.

A conditional GAN, on the other hand, takes as input both the noise vector z, along with additional input y in order to produce the output x. Notice that y and x are not necessarily from the same set.

In our setting, we attempt to learn a target texture deformation given a source texture deformation. That is, given a source texture U_{source} that encodes a given expression, we would like to transfer this expression to a neutral-pose target texture N_{target}. For example, the source texture expression might contain a wrinkle on the left cheek that we would like to synthesize onto the target neutral texture. In this case, the output is U_{target}, the target texture with the wrinkle, and we are conditioning on U_{source} and N_{target}.

Note that neural networks are inclined to be heavily affected by noise and variation within the training corpus. For this reason, we work in the UV texture space of the captured facial albedos. This provides many benefits during training. First of all, it minimizes variations in the input due to factors such as lighting, image background and head pose. In UV space, the mouth, eyes, and nose are in the exact same location in each image - the only thing that changes is the content of those locations. Working in this space makes it possible to retarget mouth motion, blinking, scrunching, and various other skin deformations that would be much harder to learn otherwise.

5.2. Loss Function

The energy function we minimize is given by

$$L_{GAN}(G,D) = E_{x,y\sim p_{data}(x,y),z\sim p_z z}[\log D(t)] + E_{x,y\sim p_{data}(x),z\sim p_z(z)}[\log(1 - D(G(x,z)))] \quad (7)$$

In our formulation, x is the pair (U_{source}, N_{target}) and y is given by U_{target}. In addition to this energy, the generator G also attempts to minimize the reconstruction loss to the target y in the L1 sense. That is, $L_{recon}(G) = E[\| y - G(x,y,z) \|_1]$ [17].

G attempts to minimize this objective while D attempts to maximize it. In other words: $G_{opt} = \arg\min\max L_{cGAN}(G,D) + \lambda L_{L1}(G)$, where μ encodes the relative weighting of the different errors [17]. In our implementation of the conditional GAN, we do not input any noise during generation, but otherwise the optimization program is accurate. We set $\mu = 0.001$ in all experiments. This parameter can be viewed as a balancing between the need to reconstruct accurate wrinkles with the need to have the final texture remain "realistic".

5.3. Network Architecture

We use a similar architecture as [17]. First, we concatenate the driver expression and the neutral target identity, (U_{source}, N_{target}), along their color-channel dimension as input. The generator output is the target expression y, which is the expression transferred from the source to target texture.

Second, we use a masked prior to define the ℓ_1 and adversarial loss. The mask is applied on the ℓ_1 loss such that the the loss around the mouth and eye regions is ten times higher than in other areas - we do this because wrinkles, blinking, and most texture deformations are a lot more prevalent in these areas. For the discriminator, we adopt a Markovian Discriminator (PatchGAN) with patch size set to 70×70. We found that adding skip connections between encoder layers and decoder layers, skipping around the code

layer, greatly reduces noise in the output image. The parameters are optimized using ADAM with back-propagation. Our input and output resolution are 256×256.

5.4. Mouth Synthesis

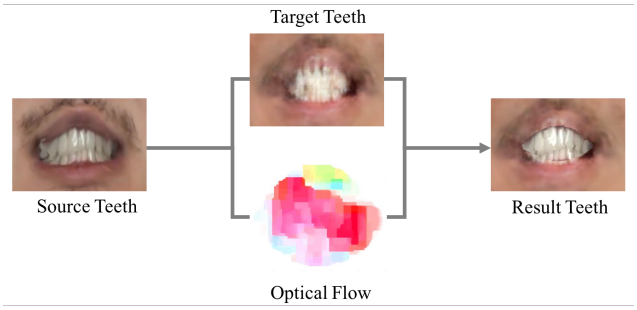


Figure 3. Visualization of flow-upsampling from source mouth to low-resolution inferred mouth. Left is source mouth, top is target mouth, bottom is flow field, right is final result.

We model the mouth interior region as a part of the UV texture. When the mouth is closed, the area between the lips get projected onto this area to make a pink color. When the mouth is open, the mouth interior, including the teeth and tongue, are projected to this region (Fig. 3).

Using our deep-learning framework, we are able to transfer open-mouth expressions onto the closed-mouth neutral target textures and infer the inner mouth region. However, due to lack of training data for the mouth interior, the inferred texture here tends to be of rather low resolution. In order to improve this area, we use SIFT-Flow [25] to redraw the target mouth using the source mouth for each frame. In particular, after computing SIFT features at the pixel level, we perform matching based on the following energy term:

$$E(w) = \sum_p \min(||s_1(p)s_2(p+w(p))||_1, t) + \sum \eta(|u(p)| + |v(p)|) + \sum_{(p,q)\in\epsilon} min(\alpha|u(p) - u(q)|, d) + min(\alpha|v(p) - v(q)|, d) \quad (8)$$

The terms in the equation are, in order, the data term, the small displacement term, and the spatial regularization term [24], where $w(p) = (u(p), v(p))$ is the flow vector at a point $p = (x, y)$, and s_1 and s_2 are the sift features compunted for either image 1 or image 2 at the point p. The second and third term regularize the matching by pushing nearby points to get mapped to each other.

Inferring the inner mouth also has the added benefit of improving the lip texture around the mouth during tracking failure of the original video: tracking of tight-lipped expressions such as kissing often fail and the lip texture gets projected to the interior region in UV space. During rendering, this causes the lips to be thinner than they should be, which gives an unnatural appearance. By inferring this inner-mouth region, we are able to synthesize realistic kissing faces on the target even when lip tracking fails on the source.

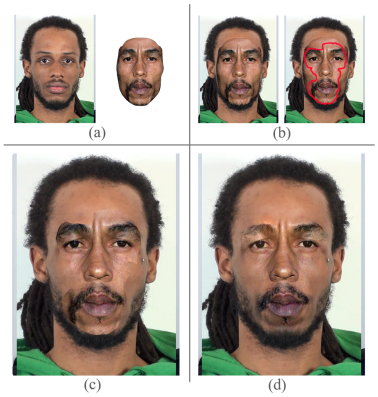


Figure 4. a) The input consists of the frames of the source video (left) and the rendered retargeted mesh (right). b) Naively projecting all of the target mesh's vertices onto the source frame results in an unrealistic and incoherent result (left) when a more optimal partition is outlined in red on the image on the right. c) The composition of the two images using the optimal scene. d) Linearly blending the pixels along the seam gives a smooth and realistic result.

6. Video Face Replacement via Blending

Once we have the per-frame textures and retargeted mesh, we are also able to transfer the target appearance back onto the source video sequence for a detailed and realistic animation (Fig. 4). We use a graph-cut approach similar to [6] in order to achieve this. In particular, for blending the retargeted faces to the source video in a photorealistic manner, we first find a graph-cut to partition each frame into two regions, which determines whether a pixel comes from the frame in the source video or the image of the rendered retargeted face model(Fig. 4a-c). We then linearly blend the target and source regions of the images along the seam to achieve a smooth and realistic result (Fig. 4d).

6.1. Graph-Cut

Similar to [6], we optimize the partition so as to ensure spatial coherence coherence between neighboring pixel values and temporal coherence between frames.

If we naively project the mesh back onto the frame of the source video, as seen in the left-hand image of Fig 4b, the result is not smooth or realistic. In order to maintain spatial coherence across large variations in pose and expression, we construct a graph-cut problem on each of the frames in the source video and their corresponding retargeted mesh as in [20]. For each frame, the graph cut algorithm labels

each vertex on the mesh as either a source vertex or target vertex in a manner that minimizes the difference in pixel values across neighboring vertices, as shown on the right-hand image of Fig. 4b. We then project the seam of the labeled mesh onto the image (Fig. 4c).

The nodes of the graph represent the vertices on the mesh for each frame in the source video. Let each vertex be denoted as $V_{t,i}$, where t refers to the tth frame, and i refers to the ith vertex on the mesh.

In order to minimize the difference between source and target pixels along the seam, for each frame, we set weights on the graph for each edge between each pair of vertices $V_{t,u}$ and $V_{t,v}$ that share an edge on the mesh as:

$$\begin{aligned} W_m(V_{t,u}, V_{t,v}) = &\|I_s(V_{t,u}) - I_s(V_{t,v})\| \\ &+ \|I_t(V_{t,u}) - I_t(V_{t,v})\| \end{aligned} \tag{9}$$

$I_s$is the source image, I_w is the target image, $I_s(V)$ is the color at the pixel where V is projected onto the source frame, and $I_t(V)$ is the color at the pixel V is projected onto in the rendered target image. For temporal coherence, we set edges between vertices $V_{t,i}$ and $V_{t+1,i}$ for all i and t, with weight as:

$$\begin{aligned} W_f(V_{t,i}, V_{t+1,i}) = &\lambda\|I_s(V_{t,i}) - I_s(V_{t+1,i}+1)\|^{-1} \\ &+ \|I_t(V_{t,i}) - I_t(V_{t+1,i}+1)\|^{-1} \end{aligned} \tag{10}$$

After computing the seam, we can composite the target appearance onto the source video and linearly blend the pixels across the seam for a realistic novel reenactment.

7. Experiments

7.1. Data Collection

In order to semantically transfer similar expressions from one texture to another, the training data must be collected in a way so that corresponding expression textures across different individuals are aligned. To do this, we hired actors to perform a series of 21 expressions based on the Facial Action Coding System (FACS) [8], and recite 20 Harvard Sentences [16]. We then hand cut, warped, and aligned the sequences using video editors. Each FACS expression was broken up and extended into two 24-frame sequences. The first of these consisted of an individual starting on neutral frame, then moving to the apex of the FACS expression. The second 24-frame sequence was a different clip of the face starting in the FACS expression and returning to neutral. The sentences were aligned using dynamic time warping on the audio sequences to align the video sequences, as was done in [29]. We plan to release this synchronized, high-resolution dataset to the public in the future.

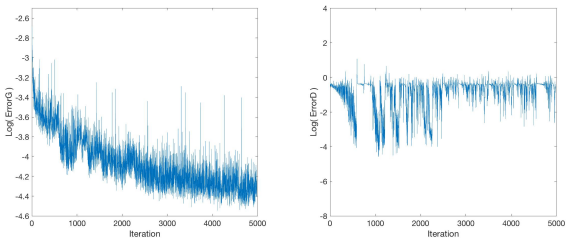
After aligning the sequences, we used a face-tracking and texture extraction process based on the formulation of [35], to compute a texture for each of the aligned video frames. Our total training dataset consists of a set of 15 identities, each with 3107 frames.

7.2. Data Augmentation

In order to increase the generalization ability of our network to subjects whose appearance varies significantly from those in this dataset, we augment it by altering the skin tones of the captured textures. Specifically, we extract lighting-independent and expression-free albedos from the texture images. We then perform PCA on the albedo images. To each albedo image, we add multiples of the found principal components with magnitudes proportional to the corresponding eigenvalues multiplied with a random variable drawn from a Gaussian with mean zero and standard deviation 0.1. This results in a new series of albedo images that differ from the original in their skin tone and local details.

These albedo textures, combined with the unmodified captured albedo textures, are used as the training data for our network. We generate 15 additonal albedo textures from each of the 15 subjects, giving us a total of 745,680 training images.

7.3. Training and Testing



Generator Error Discriminator Error

Figure 5. Generative and discriminative training loss.

We divide the dataset into a training set and a testing set. During training, we train the network in an end-to-end fashion. On each iteration, we randomly sample a pair consisting of a source expression frame and a target identity frame from the dataset as input. The corresponding ground truth texture combining the given identity and expression is also sampled. The output will be synthesized using a forward pass, and the loss is back-propagated using Adaptive Moment Estimation (ADAM). The images are scaled to 256×256 and we set each batch to contain 32 images. We trained 15 epochs on a Titan X GPU, which took roughly two days for both the generative loss and discriminative loss to converge (Fig. 5). We set the initial learning rate to $lr = 2e-4$ for the generator and $lr = 2e-5$ for the discriminator. The learning rate is lowered several times during training. During testing, we randomly sample a pair consisting of a source expression frame and a target identity frame from the test set.

8. Results

As input our system takes a source video and a single target image of a face. It generates a dynamic texture of the target image for a 3D model and retargets the source mesh onto the single input image, inferring details such as wrinkles and the inner mouth region. Our results can be seen in Fig. 8. More results can be seen in the supplementary material.

Reenactment: Comparison To Previous Works When given only a single image as a target input, [35] can only generate static textures and does not capture any of the wrinkles or inner mouth details that our system captures as seen in Fig. 8.

An example of methods like [35] that do not use our inference model, thereby producing less detailed results when there is only a single target image as input, can be seen in Fig. 6.

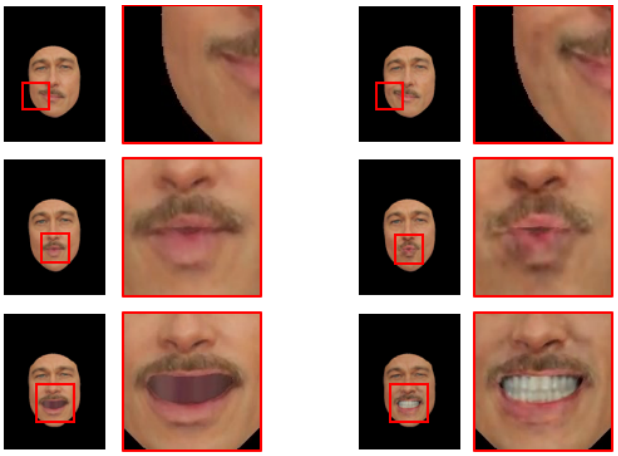


Figure 6. Wrinkle animations from a single neutral target input image. In each row, the pair of images on the left shows the facial animation achieved using a static texture generated by a classical correspondence methods such as Face2Face. The pair of images on the right show the facial animation achieved using the dynamic texture generated by our inference model. Note that the images on the right are capable of generating detailed wrinkles and filling the inner mouth cavity.

Composition: Comparison To Previous Works [6] is currently the state-of-the-art with respect to face composition, and their results are shown in Fig. 8. Our results for composition in Fig. 8 are comparable to theirs, but we only need a single target image as input while they need a video sequence.

9. Discussion

We present a method of generating realistic video sequences of faces from a single photograph which can then be used to replace the face of a source/driver video sequence. To the best of our knowledge, our method is the first to leverage GANs to produce realistic, high-resolution dynamic textures from a single target input image to be used for rendering.

Figure 7. [6] takes two video inputs and can do face replacement as shown in this figure. Our results are comparable to the results in this figure, but we are only using a single target image as input. Furthermore, [6] cannot handle only a single target image as input.

Figure 8. Non-frontal face reenactment. From left to right: the source image, the target image, the static texture and the dynamic texture. We can see that the dynamic texture contains more details like the wrinkles.

Method	Mean L1 Loss	Mean L2 Loss	SSIM
transf	1360	152%	0.8730 dB
diff	1790	211%	0.8150 dB

Table 1. Numerical comparison.

Limitations and Future Work Though we are able to infer dynamic textures, the input target face is assumed to be without extreme specular lighting and/or pronounced shadowing. Otherwise, the texture extraction phase following [35] will produce artifacts. As fitting the facial geometry precisely from a single viewpoint is a highly underconstrained problem, the extracted texture of the target subject may be improperly registered in extreme cases in which this fitting is insufficiently accurate. We believe that it is possible to address these issues with a more robust multilinear fitting, which takes greater account of specular lighting, shadowing, shading, and variation in the subject's appearance.

The resolution of the single image of the target must be sufficiently high resolution in order to generate appropriate details for the corresponding expressions. Furthermore, as

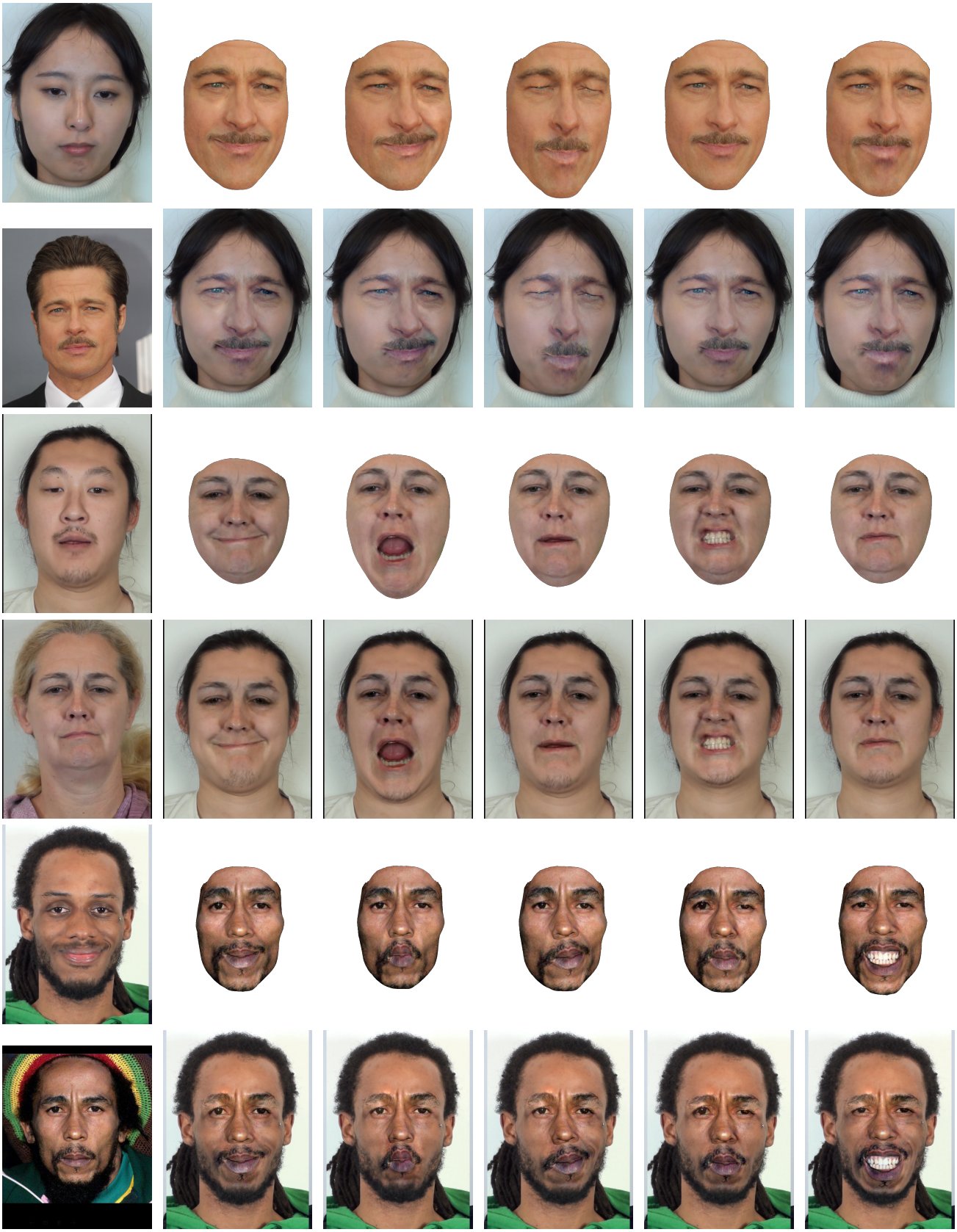


Figure 9. Three sample results of our method. Top row of each example shows the source video sequence actor along with an animated retargeted face. Bottom row shows the target single-frame input and the compositing result. Top and bottom examples are test target images. Middle example target image is a training example for comparison.

we rely on [35] to extract the target subject's facial texture and geometry, an image in which the face is largely non-frontal or partially occluded will result in incomplete or incorrect textures being extracted and, thereby causing artifacts in the generated textures. We believe it is possible to improve upon these aspects as well, as we have seen in [31] that it is possible to infer high resolution textures from partially occluded lower resolution images.

Limited variation of appearance in the training corpus is also an issue. Though the data augmentation mitigates this, the generated wrinkles and deformations will not be as sharp or as strong when the target's appearance varies greatly from those in our dataset. However, we believe that having a larger dataset with even greater appearance variations would resolve this issue.

References

[1] V. Blanz and T. Vetter. A morphable model for the synthesis of 3d faces. In *Proceedings of the 26th annual conference on Computer graphics and interactive techniques*, pages 187–194. ACM Press/Addison-Wesley Publishing Co., 1999. 3

[2] C. Bregler, M. Covell, and M. Slaney. Video rewrite: Driving visual speech with audio. In *Proceedings of the 24th annual conference on Computer graphics and interactive techniques*, pages 353–360. ACM Press/Addison-Wesley Publishing Co., 1997. 1, 2

[3] C. Cao, D. Bradley, K. Zhou, and T. Beeler. Real-time high-fidelity facial performance capture. *ACM Transactions on Graphics (TOG)*, 34(4):46, 2015. 2

[4] C. Cao, Y. Weng, S. Zhou, Y. Tong, and K. Zhou. Facewarehouse: A 3d facial expression database for visual computing. *IEEE Transactions on Visualization and Computer Graphics*, 20(3):413–425, 2014. 3

[5] E. Chuang and C. Bregler. Mood swings: Expressive speech animation. *ACM Trans. Graph.*, 24(2):331–347, 2005. 2

[6] K. Dale, K. Sunkavalli, M. K. Johnson, D. Vlasic, W. Matusik, and H. Pfister. Video face replacement. *ACM Transactions on Graphics (TOG)*, 30(6):130, 2011. 1, 2, 5, 7

[7] P. Edwards, C. Landreth, E. Fiume, and K. Singh. Jali: An animator-centric viseme model for expressive lip synchronization. 2

[8] P. Ekman and W. Friesen. *Facial Action Coding System: A Technique for the Measurement of Facial Movement.* Consulting Psychologists Press, Palo Alto, 1978. 2, 6

[9] B. Fan, L. Wang, F. K. Soong, and L. Xie. Photo-real talking head with deep bidirectional LSTM. In *2015 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2015, South Brisbane, Queensland, Australia, April 19-24, 2015*, pages 4884–4888, 2015. 2

[10] P. Garrido, L. Valgaerts, O. Rehmsen, T. Thormahlen, P. Perez, and C. Theobalt. Automatic face reenactment. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4217–4224, 2014. 2

[11] P. Garrido, L. Valgaerts, H. Sarmadi, I. Steiner, K. Varanasi, P. Perez, and C. Theobalt. Vdub: Modifying face video of actors for plausible visual alignment to a dubbed audio track. In *Computer Graphics Forum*, volume 34, pages 193–204. Wiley Online Library, 2015. 2

[12] P. Garrido, L. Valgaerts, C. Wu, and C. Theobalt. Reconstructing detailed dynamic face geometry from monocular video. 2

[13] J. Gauthier. Conditional generative adversarial nets for convolutional face generation. 2

[14] A. Golovinskiy, W. Matusik, H. Pfister, S. Rusinkiewicz, and T. Funkhouser. A statistical model for synthesis of detailed facial geometry. In *ACM Transactions on Graphics (TOG)*, volume 25, pages 1025–1034. ACM, 2006. 3

[15] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. In *Advances in Neural Information Processing Systems*, pages 2672–2680, 2014. 2

[16] Harvard. Harvard sentences, 1969. http://www.cs.columbia.edu/~hgs/audio/harvard.html. 2, 6

[17] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros. Image-to-image translation with conditional adversarial networks. *arxiv*, 2016. 2, 4

[18] V. Kazemi and J. Sullivan. One millisecond face alignment with an ensemble of regression trees. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1867–1874, 2014. 3

[19] I. Korshunova, W. Shi, J. Dambre, and L. Theis. Fast face-swap using convolutional neural networks. *CoRR*, abs/1611.09577, 2016. 2

[20] V. Kwatra, A. Schdl, I. Essa, G. Turk, and A. Bobick. Graphcut textures: Image and video synthesis using graph cuts. *ACM Transactions on Graphics, SIGGRAPH 2003*, 22(3):277–286, July 2003. 5

[21] S. Laine, T. Karras, T. Aila, A. Herva, S. Saito, R. Yu, H. Li, and J. Lehtinen. Facial performance capture with deep neural networks. *arXiv preprint arXiv:1609.06536*, 2016. 2

[22] C. Li and M. Wand. Precomputed real-time texture synthesis with markovian generative adversarial networks. *arXiv preprint arXiv:1604.04382*, 2016. 2

[23] C. Liu, H.-Y. Shum, and W. T. Freeman. Face hallucination: Theory and practice. *International Journal of Computer Vision*, 75(1):115–134, 2007. 2

[24] C. Liu, J. Yuen, and A. Torralba. Sift flow: Dense correspondence across scenes and its applications. *IEEE Trans. Pattern Anal. Mach. Intell.*, 33(5):978–994, May 2011. 5

[25] C. Liu, J. Yuen, A. Torralba, J. Sivic, and W. T. Freeman. *SIFT Flow: Dense Correspondence across Different Scenes*, pages 28–42. Springer Berlin Heidelberg, Berlin, Heidelberg, 2008. 5

[26] C. Malleson, J.-C. Bazin, O. Wang, D. Bradley, T. Beeler, A. Hilton, and A. Sorkine-Hornung. Facedirector: continuous control of facial performance in video. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 3979–3987, 2015. 2

[27] M. Mirza and S. Osindero. Conditional generative adversarial nets. *CoRR*, abs/1411.1784, 2014. 2

[28] U. Mohammed, S. J. Prince, and J. Kautz. Visio-lization: generating novel facial images. In *ACM Transactions on Graphics (TOG)*, volume 28, page 57. ACM, 2009. 2

[29] K. Olszewski, J. J. Lim, S. Saito, and H. Li. High-fidelity facial and speech animation for vr hmds. *ACM Transactions on Graphics (TOG)*, 35(6):221, 2016. 2, 6

[30] D. Pathak, P. Krahenbuhl, J. Donahue, T. Darrell, and A. A. Efros. Context encoders: Feature learning by inpainting. *arXiv preprint arXiv:1604.07379*, 2016. 2

[31] S. Saito, L. Wei, L. Hu, K. Nagano, and H. Li. Photorealistic facial texture inference using deep neural networks, 2016. 3, 9

[32] S. Suwajanakorn, I. Kemelmacher-Shlizerman, and S. M. Seitz. What makes kevin spacey look like kevin spacey. *CoRR*, abs/1506.00752, 2015. 2

[33] S. L. Taylor, M. Mahler, B.-J. Theobald, and I. Matthews. Dynamic units of visual speech. In *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, SCA '12, pages 275–284, Aire-la-Ville, Switzerland, Switzerland, 2012. Eurographics Association. 2

[34] J. Thies, M. Zollhöfer, M. Nießner, L. Valgaerts, M. Stamminger, and C. Theobalt. Real-time expression transfer for facial reenactment. *ACM Transactions on Graphics (TOG)*, 34(6):183, 2015. 2

[35] J. Thies, M. Zollhöfer, M. Stamminger, C. Theobalt, and M. Nießner. Face2face: Real-time face capture and reenactment of rgb videos. In *Proc. Computer Vision and Pattern Recognition (CVPR), IEEE*, 2016. 1, 2, 3, 6, 7, 9

[36] K. Wampler, D. Sasaki, L. Zhang, and Z. Popović. Dynamic, expressive speech animation from a single mesh. In *Proceedings of the 2007 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, SCA '07, pages 53–62, Aire-la-Ville, Switzerland, Switzerland, 2007. Eurographics Association. 2

[37] C. Wu, D. Bradley, P. Garrido, M. Zollhöfer, C. Theobalt, M. Gross, and T. Beeler. Model-based teeth reconstruction. *ACM Transactions on Graphics (TOG)*, 35(6):220, 2016. 2