Real-time Facial Performance Capture and Manipulation

by
Luming Ma

A dissertation submitted to the Department of Computer Science,

College of Natural Sciences and Mathematics

in partial fulfillment of the requirements for the degree of

Doctor of Philosophy
in Computer Science

Chair of Committee: Zhigang Deng

Committee Member: David Mayerich

Committee Member: Guoning Chen

Committee Member: Shishir Shah

University of Houston
May 2020

ACKNOWLEDGMENTS

I would like to thank my advisor Dr. Zhigang Deng, for all the support and valuable advice during the past five years. I sincerely appreciate his guidance in my critical thinking, and research activities and abilities. Without his lasting motivation, I would not accomplish my Ph.D study.

I would like to thank Dr. Fuhao Shi and Dr. Yizhou Yu for their constructive ideas and selfless help at the early stage of my Ph.D. Particularly, I'm truly grateful to Dr. Fuhao Shi for sharing with me his knowledge in facial reconstruction. His insightful thoughts enlightened my research path.

I would like to thank my committee members: Dr. David Mayerich, Dr. Guoning Chen, and Dr. Shishir Shah, for their constructive comments.

I would also like to thank my colleagues and friends: Yuting Zhang, Mingxuan Luo, Huikun Bi, Xiaoya Wang, Xin Zhou, Yu Ding, Lieyu Shi, Kaoji Xu, Fang Yang, Aobo Jin, Lei Shi, Xiantian Zhou, Qixin Deng, Kunpeng Zhang, Shuguang Chen and Qiang Fu. I hope you all have a great future.

Lastly, my special thanks go to my parents and Rui Yi. Your understanding and support is my courage and power in this memorable journey.

ABSTRACT

Acquisition and editing of facial performance is an essential and challenging task in computer graphics, with broad applications in films, cartoons, VR systems, and electronic games. The creation of high-resolution, realistic facial animations often involves controlled lighting setups, multiple cameras, active markers, depth sensors, and substantial post-editing from experienced artists. This dissertation focuses on the capture and manipulation of facial performance from regular RGB video.

First, a novel method is proposed to reconstruct high-resolution facial geometry and appearance in real-time by capturing an individual-specific face model with fine-scale details, based on monocular RGB video input. Specifically, after the coarse facial model is reconstructed from the input video, it is subsequently refined using shape-from-shading techniques, where illumination, albedo texture, and displacements are recovered by minimizing the difference between the synthesized face and the input RGB video. To recover wrinkle level details, a hierarchical face pyramid is built through adaptive subdivisions and progressive refinements of the mesh from a coarse level to a fine level. The proposed approach can produce results close to off-line methods and better than previous real-time methods.

On top of the reconstruction method, two manipulation approaches upon facial expressions and facial appearance are proposed, namely facial expression transformation and face swapping. In facial expression transformation, desired and photo-realistic facial expressions are directly generated on top of input monocular RGB video without the need of any driving source actor. An unpaired learning framework is developed to learn the mapping between any two facial expressions in the facial blendshape space. The proposed method automatically transforms the source expression in an input video clip to a specified target expression through the combination of the 3D face reconstruction, the learned bi-directional expression mapping, and automatic lip correction. It can be applied to new users with different identities, ages, speeches, and expressions, and without additional training.

In face swapping, a high-fidelity method is presented to replace the face in a target video clip by the face from a single source portrait image. First, the face reconstruction method is run on both the source image and the target video. Then, the albedo of the source face is modified by a

novel harmonization method to match the target face. The face geometry is predicted as the source identity performing the target expression with person-specific wrinkle style. Finally, the source face is re-rendered and blended into the target video using the lighting and camera parameters from the target video. The proposed method runs fully automatically and at a real-time rate on any target face captured by cameras or from legacy videos. More importantly, unlike existing deep-learning-based methods, it does not need to pre-train any models, i.e., pre-collecting a large image/video dataset of the source or target face for model training is not required.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

1 Introduction

The face technology has great research value in computer graphics and computer vision community. There are lots of face applications on the market, e.g., Face ID, Animoji, and face retouching applications. All these applications rely on a fast and accurate face capture algorithm. High-end movies and games also use face capture, and they are pushing the limit of the face related technology. In some movies, animations are captured from actor performance and transferred to the 3D character for rendering vivid physics-based facial animations to cheat human eyes. In computer games, realistic characters can give players an immersive experience. As shown in Figure 1.1, face reconstruction plays an essential role in the family of face related techniques, as many other techniques depend on it. This dissertation focus on real-time face reconstruction from video and two types of face manipulations. Expression transformation edits the emotion of a face video, and face swapping changes the face identity.

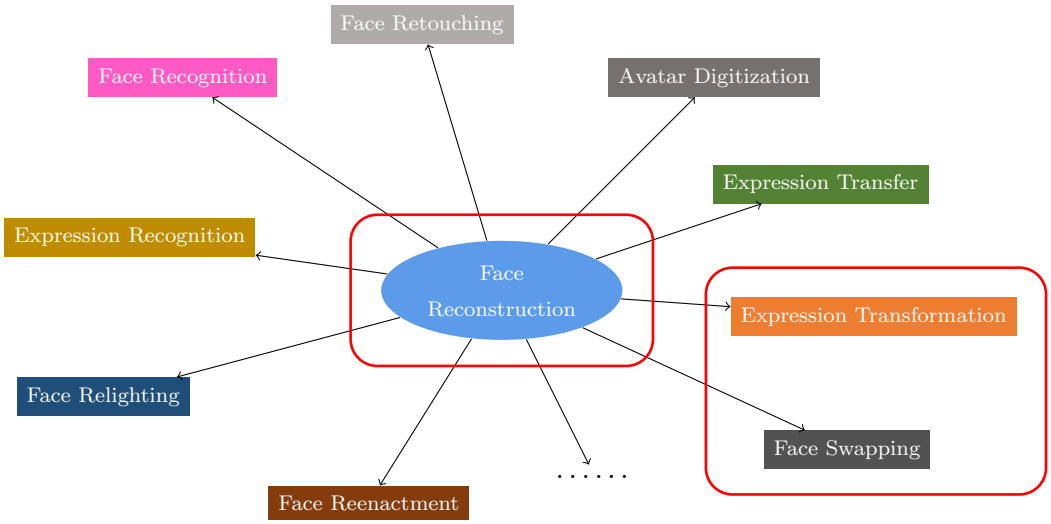


Figure 1.1: A diagram of the face technology family.

1.1 Face Reconstruction

Capturing human facial performance has been a long-standing problem in computer graphics with a wide range of applications, such as films, VR systems, and digital games [169]. Since the human

eyes are particularly sensitive to facial performance and appearance, the creation of high-resolution 3D face models and realistic facial animations often involves controlled lighting setups, multiple cameras, active markers, and substantial post-editing from experienced artists [151]. In recent years, social media and mobile applications have brought increasing demands for the light-weight acquisition of facial performance on consumer devices. To this end, passive methods leveraging stereo cameras [143] or depth information [148, 90] have been proposed to capture detailed facial performance using binocular or RGB-D cameras. These methods achieve impressive results but are limited to the requirement of binocular footage or depth, data which are often unavailable for legacy video footage.

Recently, researchers sought to only rely on monocular RGB video for facial reconstruction [47, 125, 50]. These methods utilize shading information for shape refinements and achieve quality on par with the methods with stereo and depth cameras. In the meantime, these methods require substantial computational time as well as information from forward and backward frames of the video; therefore, they are unsuitable for real-time acquisition applications. On the other hand, real-time facial tracking and animation systems have also been developed by Cao et al. [24, 23] with a single RGB camera. Their data-driven methods, however, rely on strong face priors and could not capture individual-specific or transient details, such as wrinkles on the forehead and around the eyes. Recent development from Cao et al. [21] incorporates medium-scale frequencies into the coarse face model by training regressors from high-resolution facial capture data. Even though plausible face wrinkles are produced, in a nutshell, this learning-based (or data-driven) model is yet an approximation of the true facial performance and is also limited by the data used for training. As shown in Figure 1.2, despite considerable advances in facial performance tracking, a method that accurately captures facial details, with the low-cost acquisition (e.g., a monocular camera) and at a real-time rate, is still demanding.

In this dissertation, a geometry-based method [100] is proposed to reconstruct high-resolution facial geometry and appearance in real-time (Figure 1.3). It takes in RGB video from a single RGB camera and captures an individual-specific face model of the subject with wrinkle-level details. The

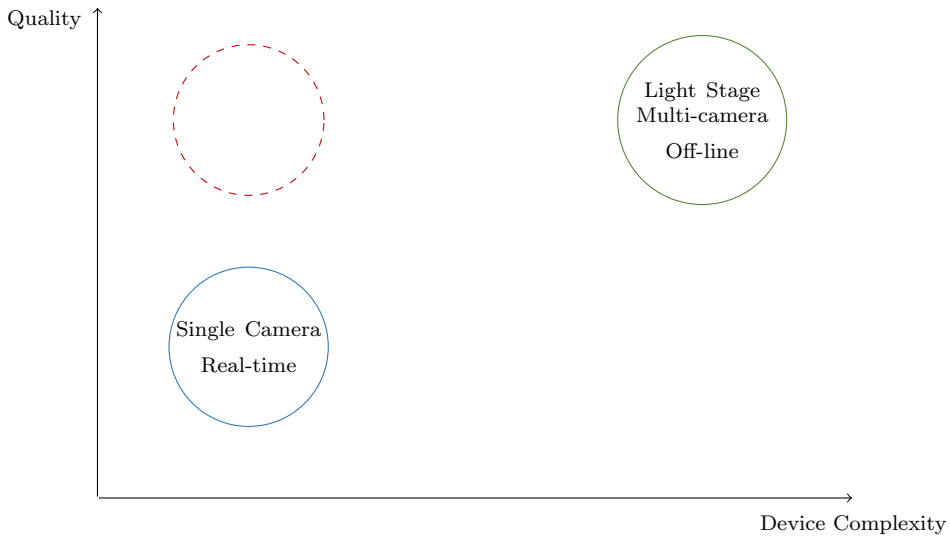


Figure 1.2: Face reconstruction methods according to device complexity and result quality. Capture rich, robust, and accurate facial performance using simple devices is still demanding (red circle).

system runs fully automatic and does not require off-line preprocessing for a novel user. It is shown that the results are close to existing off-line methods. I believe this opens up more possibilities in consumer-level interactive applications, such as facial performance avatar retargeting, on-line preview, immersive VR games, and photo-realistic facial makeups. In the following, two kinds of applications are mainly described: expression transformation and face swapping.

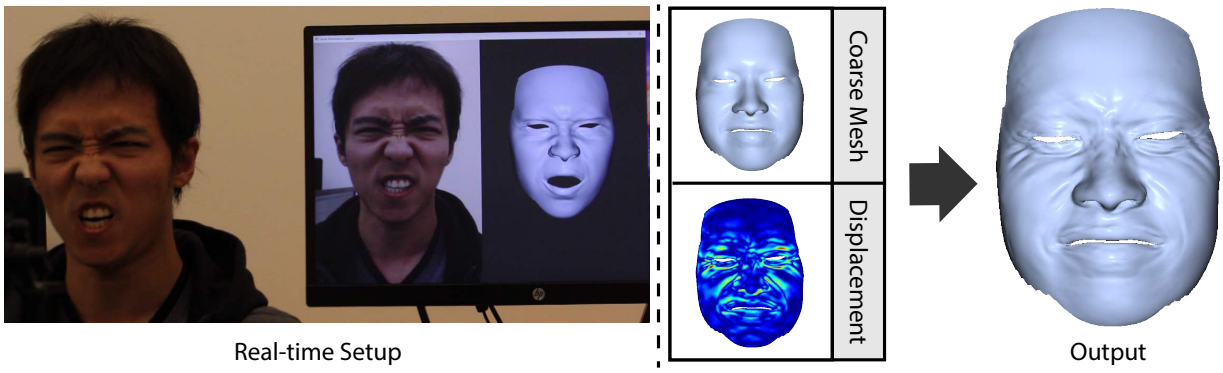


Figure 1.3: The proposed face reconstruction method captures fine-scale facial performance from a monocular RGB camera in real-time (left). The large-scale facial performance is augmented by per-vertex displacements via shape-from-shading to capture wrinkle-level details (right).

1.2 Expression Transformation

Realistic facial expression creation and transformation aim to generate or modify face videos with the desired expression. Thus far, popular approaches usually require a driving source or the combination of multiple ones, such as capturing a subject's performance and then transferring it to virtual faces [37, 124, 139, 3, 145], and speech-driven facial animation [19, 42, 39, 38, 133, 135, 74, 166]. However, these methods only provide a way to drive the face to follow the performed expressions. They do not provide the flexibility to synthesize new facial expressions on top of the original, such as being happier or being angry instead of neutral while speaking. Besides, the transferring approaches usually break the synchronization between the face reenactment and audio from the source video and thus are unsuitable for speech video.

An ideal solution to the above problem is to generate desired and photo-realistic facial expressions on top of the source expression of an input monocular video clip, without the need for any driving sources. One straightforward way is to per-frame edit the source expression. Clearly, this is quite tedious and time-consuming; furthermore, it is non-trivial to ensure the temporal dynamics of the edited facial expression. Another technical path explored previously is to learn a mapping $\mathbb{M} : X \Rightarrow Y$ between two sequences X and Y that are semantically aligned. For example, image-to-image translation approaches [70, 73] are used to address this problem by transferring image style or content between image pairs. However, they often require a large number of aligned face images of various identities, expressions, and environment lighting as the training data. The problem becomes even more difficult when dealing with speech video, because lip synchronization needs to be preserved in the transformed and re-rendered video, besides the intrinsic complexity and subtlety of facial expressions.

To tackle the above problem, a complete pipeline is proposed to real-time transform the source expression of the subject in an input (source) monocular RGB video clip to a user-specified target expression and then photo-realistically re-render the same performance but with the target expression [99] (Figure 1.4). The generated facial expression sequence is temporally dynamic, coherent, and lip-synchronized to the source audio.

Figure 1.4: The proposed facial expression transformation method can real-time transform the neutral facial expression in input video (top) to happy expression (bottom).

1.3 Face Swapping

A typical face swapping scenarios can be described as follows: given a target video/image, the appearance of the inner face is swapped by the face from a source video/image, while the facial expression, skin color, hair, illumination, and background of the target video/image are preserved [48, 35]. To date, a number of off-the-shelf applications have been designed to achieve this goal, including Deepfakes [36] and Face Swap[1].

Although potential legal and ethical concerns have emerged in the society in recent years, the face swapping technique itself has rich research values and numerous useful application scenarios in film making, video editing, and identity protection. For instance, the face of a stunt actor who performs in a dangerous environment can be replaced by a star actor's face captured in a safe studio. It is also applicable to revive the dead actors in legacy films by replacing them with the face of the substitute. For video amateurs, an automatic tool that can put the faces of themselves or friends into movies or video clips to create fun content with minimal manual involvement is in high demand. Furthermore, replacing the face with another identity or virtual avatar in real-time video streaming or conference could be practically needed to protect identity privacy.

[1]https://faceswap.ms/

Even though noticeable progress has been made on face swapping over the past several years, *video-realistic* face swapping is still challenging. The differences of face shapes, expressions, head poses, and illuminations between the source and the target faces have posed significant difficulties on the problem. In addition, the human eyes are particularly sensitive to the imperfection in synthesized facial performance and appearance. Previously, researchers sought to tackle the problem by searching for the most similar images/frames from an image database [11] or video frames [48] and replacing faces through image warping. This line of methods highly relies on the similarity of head poses, expressions, and illuminations between the source and the target images. Another line of approaches resorted to reconstructing 3D face models from both the source and the target images and then re-render the source face into the target background photo-realistically. Although promising results have been presented [13, 35], these methods typically involve manual interventions (e.g., face alignment) from users. More recently, deep learning approaches [36] have been proposed to automatically swap the faces of two identities. However, they require a large image dataset of the face identities and expensive training of the model before running, which undermines the broad applicability, accessibility, and generality of these methods.

In this dissertation, a novel, automatic, real-time method is proposed to swap the face in the target video by the face from a *single* source portrait image [101] (Figure 1.5). Just imagine a selfie image of yourself and an actor interview video clip are given, the proposed method can create a new video clip in which you were taking the interview. In the method, 3D face models with wrinkle level details, appearances, head poses, and illuminations are first reconstructed from the source image and the target video, respectively. Then, a novel face image is rendered using the identity, predicted wrinkles, and adapted albedo of the source face and the head pose, expression, and illumination of the target face.

Compared to the state-of-the-art methods, the main advantages of the proposed method include: (i) little dependency of the source face data (i.e., only need a single still portrait image), (ii) fully automatic and real-time processing, and (iii) swapping both face shape and appearance. More importantly, unlike existing deep learning based methods, the proposed method does not require

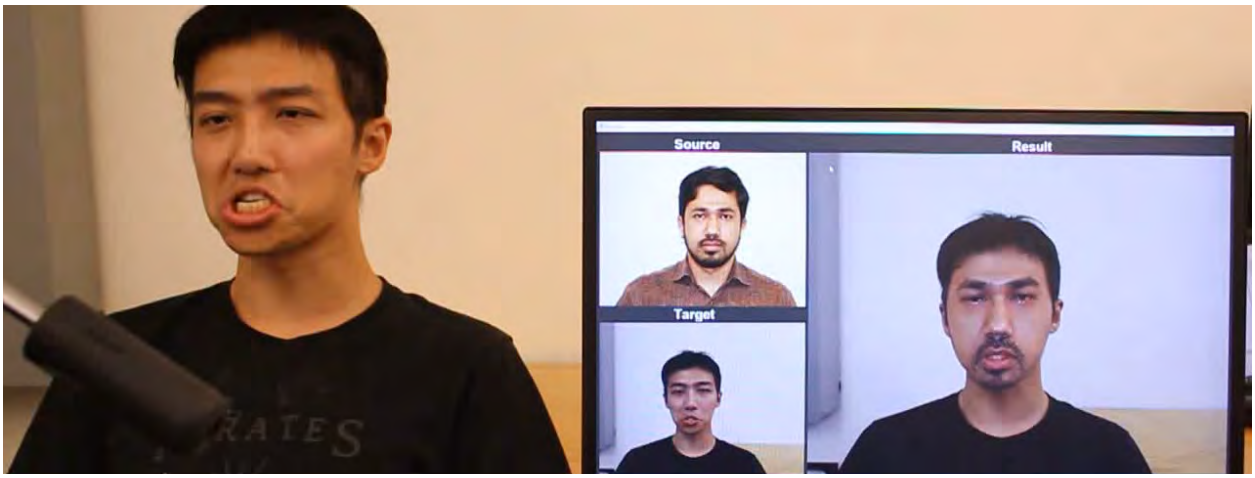


Figure 1.5: The proposed face swapping method swaps the face from a single source portrait image into an RGB live video stream. The result video retains the facial performance of the target actor while with the identity of the source.

any assumption of the input face nor require any training data. Therefore, the method does not need to collect a large number of face images for expensive and time-consuming model training, which can bring significant convenience and efficiency to users. As a result, the proposed method can also generalize well to unseen faces.

2 Related Work

The proposed system reconstructs 3D facial performance of the subject from an input video clip/image, transforms his/her facial expression to a user-specified target expression, or swaps the face from a source portrait to target video, and finally photo-realistically re-renders the video. Thus, the literature review in this section specifically focuses on the recent, most related efforts on facial reconstruction, video-based face reenactment, expression manipulation, image transformation, and face swapping.

2.1 Face Reconstruction

Off-line Face Reconstruction Many previous works have focused on building 3D face models in controlled environments [82]. The works of [163, 102] employ structured light and photometric stereo for face scanning. Some methods attach markers on the face [150, 10, 66] to acquire dynamic deformations of 3D facial performance. Despite high-quality face models reconstructed, the above methods typically involve complex intrusive setups to the subjects. Passive solutions were also developed using stereo images [6, 8, 17, 56, 15] or lightweight binocular cameras [143]. A significant portion is data-driven methods stemming from the seminal morphable face model [14, 12], where a statistical model is employed to reconstruct facial identity and expression from images and/or video. Vlasic et al. extend this method with a multi-linear model that is constructed along the axes of vertices, identities, expressions, and visemes [145]. Similarly, the FaceWarehouse [25] employs a bilinear face model that consists of 47 FACS-based [41] blendshapes for each identity. Li et al. [94] learn a similar FLAME model using PCA for identity, expression and blend skinning for the neck, jaw, and eyeballs from a large sequence of temporal 3D scans. High-frequency details such as wrinkles are typically missing from these methods, which requires further refinements [9].

However, the aforementioned methods usually require delicate camera and lighting setup in a controlled environment, which is unfriendly to amateur users and also lacks the ability to process

online video clips. Reconstruction from monocular video [131, 47, 125, 45] in uncontrolled environments has attracted more attention in recent years due to its low-cost setup and applicability to various legacy video footage. The active appearance model (AAM) [34, 72] is proposed to estimate the parameters of a 3D PCA model using 2D features. The blendshape model [88] has also been widely used for representing expressions constrained by image feature points [112, 28]. The works of [132, 50] build controllable face rigs and appearance from video for animation by fitting a parametric blendshape and regression of medium level details. Similarly, Ichim et al. [67] create fully rigged [110] 3D personalized avatars from hand-held video input. A comprehensive survey of face reconstruction from image can be found in [129]. All the above methods, however, require intensive off-line processing and do not apply to real-time scenarios.

On the other hand, physics-based methods [87, 126, 68] build an anatomically accurate, volumetric model with facial musculature, tissue, and skeleton. Muscle activations in the physical model are capable of simulating contacts and collisions of the face with external objects. Ichima et al. [69] introduce volumetric blendshapes combining intuitive control of blendshapes and the capability of realistic physics-based simulation. Similarly, Cong et al. [32] create a blendshape system for facial muscles that drives underlying anatomical and biomechanical muscle dynamics. Wu et al. [153] present a high-quality anatomically-constrained local face model for tracking 3D faces from monocular video.

A variety of methods endeavor to capture facial appearance [82] and illumination. Lombardi et al. [98] learn a deep appearance model encoding view-dependent appearance using a variational autoencoder. It's suitable for rendering faces from novel views in virtual reality. Yamaguchi et al. [160] infer high-resolution skin surface reflectance maps and medium and high-frequency displacement maps from an unconstrained image using deep learning. Gotardo et al. [56] acquire high-quality dynamic properties of facial skin appearance, including albedo, specular, and normal maps from multi-camera setup. Meka et al. [106] learn deep reflectance fields to relight the face in a pair of images under any lighting condition.

Real-time Face Capture Real-time facial tracking systems have been developed using structured light [149], where an individual-specific face model is fitted off-line first and then tracked online for expression transferring. Another category combines depth information from a single RGB-D camera [148, 90, 29, 168, 16, 64, 139, 141, 142] to track facial expression deformations of personalized blendshape or to deform a face template mesh as-rigid-as-possible in real-time. Liu et al. [96] track facial expressions based on a Kinect sensor with video and audio input. Saragih et al. [122] propose a real-time facial puppetry system that captures the user's facial expression from a single image and transfers to an avatar. Regression-based face tracking has been proposed by Cao et al. [24, 23] to capture coarse 3D facial geometry from a single monocular camera in real-time. Their follow-up work [21] learns displacement patches from captured texture to predict medium-scale details, and in [22], they learn a dynamic rigidity prior for rigid stabilization of face tracking. Their methods run at a high frame rate and on low-end hardware. However, they focus on re-targeting facial animation to virtual avatar rather than photo-realistic reconstruction and rendering. Recently, Wang et al. [147] track facial expression and eye gaze simultaneously, and Thies et al. [140] fit parametric face models by combining the photometric consistencies from RGB input sequence. From the perspective of computer vision community, deep learning methods, such as CNN [123, 60, 136] and autoencoder [137, 4, 98, 154], have been extensively used to reconstruct facial performance from images/video. Even though noticeable progress has been made in the area of real-time face capture, fine-scale details reconstruction is still a weakness compared to off-line methods.

Shape-from-Shading Acquiring 3D shape from a single image given the known lighting condition and surface reflectance is a well-established technique, a.k.a. shape-from-shading (SfS) [63], in the area of photometric stereo. Vlasic et al. [146] capture dynamic normal maps from multiple views using SfS under designed lighting. A popular use case of SfS is coarse shape refinements under uniform lighting [7] or even uncontrolled lighting [156, 155] by approximating lighting with

spherical harmonics [5]. This technique has also been widely used in facial reconstruction for fine-scale geometry refinements since human faces are generally assumed to be Lambertian surfaces with statistical shape priors. The works of [131, 47, 125, 50, 76, 67] reconstruct fine-scale face shapes and albedo from RGB video in an off-line manner. A real-time method related to ours is [158], where lighting and refined depth map are estimated from shading cues using an RGB-D camera. Their method, however, cannot reconstruct parametric face models or albedo texture, while the proposed method builds a textured face blendshape model that is ready for animation and expression reenactment using only a monocular camera.

2.2 Face Manipulation

Expression Manipulation Some previous works manipulate the facial expression or facial components in images or video. For example, Yang et al. [162] transfer a local facial component (e.g., smiling mouth) from one image to another. Some other methods aim at manipulating eye gaze in 2D video [86, 46] or editing 3D facial animation crafted by artists at the sequence level [93, 104]. The work of [161] achieves exaggeration, attenuation, or replacement of facial expression in parts of a 2D video. In this method, the resulting video is synthesized through image warping or frames reordering, and thus it cannot effectively handle illumination changes incurred by the change of expression. It also lacks the capability of creating novel facial expressions while the proposed method can create novel target facial expressions that do not exist in the original source video. Malleson et al. [105] continuously blend the facial performance video of an actor, which may contain different facial expressions or emotional states. Kemelmacher-Shlizerman et al. [78] generate temporal coherent face animations from large image collections of the same person. Fried et al. [43] modify the relative pose and distance between camera and subject given a single portrait photo. Using a single neutral-face input image, Nagano et al. [108] are able to photo-realistically synthesize arbitrary expressions both in image space and UV texture space.

Image Transformation Some recent image transformation/style transfer methods have been developed in the computer vision community. "pix2pix" [70] employs conditional adversarial networks to learn the mapping between image pairs. The works of [73, 52] combine the style and content from two images and synthesize a novel image using CNN. Taigman et al. transfer face images into emojis by training a domain transfer network on millions of face images [134]. The proposed approach builds on the CycleGAN framework [167] that learns a mapping function between two unpaired image domains using two GAN models [55]. The works of [92, 127, 30] developed variants of GAN models for face attributes synthesis, such as gender/age modification and expression transformation. However, those imaged based GAN models are usually limited to generating low-resolution images and likely to incur artifacts on the face or background change. In addition, none of the above methods can handle photo-realistic and temporal consistent transformations for image sequences. Thus they cannot be straightforwardly extended for video-based facial expression transformation.

Face Reenactment Face Reenactment transfers the expression of a source actor to a target video. Researchers proposed to use a RGB-D camera to transfer facial expressions in real-time [159, 139, 140]. Useful scenarios of this technique include Vdub [49], which transfers a dubber's mouth motion to the actor in the target video; FaceVR [141] which transfers the facial expression of a source actor who is wearing a head-mounted display (HMD) to the target video; and portrait animation which transfers the source expression to a portrait image [3] or video [142]. A style-preserving Vdub that translates the source actor's idiosyncratic style using CycleGAN is proposed in [79]. Expression mapping [97] transfers a target expression to a neutral source face, but does not preserve the target head motion and illumination. Kemelmacher-Shlizerman et al. [77] transfer pose and expression by seeking a face image database of the subject. Li et al. [91] presented a data-driven solution to synthesize the target video from a driving actor by retrieving frames from a pre-recorded dataset of the target person. Geng et al. [53] drive a portrait animation from a video using the GAN model. Photo-realistic re-animation of facial expression, head pose, and eye gaze

in portrait videos has been learned in [80] using the generative neural network from an input RGB video. Both off-line and real-time expression transfer for actor-to-avatar [148, 45, 31, 116, 117] have also been extensively explored previously. In addition, prior works [27, 133] produce photo-realistic speech animation in which lip motion matches with input audio. Taylor et al. [135] generate natural-looking speech animation that synchronizes to input speech using neural network and AAM. Fried et al. [44] edit talking-head video based on an input transcript to produce a realistic output video while maintaining a seamless audio-visual flow.

Face Swapping Most face swapping methods can be categorized into image-based, model-based, and learning-based. *2D image-based* methods [48] select the most similar frame from the source video and warp it to the target face. Image-to-image methods [11, 75] swap the face by automatically selecting the closest face from a large face database. Even though compelling results are produced, they cannot be applied to video since the temporal consistency is not considered. *3D model-based* methods [13, 35] track the facial performance for both the source and the target faces and re-render the source face under target conditions. The proposed method is also model-based, but it does not need any manual work to help the tracking and does not search for the closest frame in the source sequence, which enables it to run in real-time. In addition, Dale et al. [35] do not render novel faces but re-time the source video using dynamic time warping and blend the source and the target images directly. Therefore, their method also highly relies on the similarity between the source video and the target video. The proposed method builds a 3D face model from the source image at initialization and then renders it into the target. It maximally reduces the dependence on source input. Recently, *learning-based* methods were proposed to use CNN [84] or autoencoder [36] to learn face representations under various poses, expressions, and lighting conditions. If enough training data can be collected, these methods can produce robust and realistic results with proper post-processing. However, collecting sufficient, often large-scale, training data for specific faces is non-trivial and time-consuming, or even infeasible for some cases (e.g., legacy face videos). Furthermore, the face images they produce are generally low resolution, while the

proposed method does not have the above issues. Recently, Nirkin et al. [109] proposed to train a generalized face segmentation network on large face datasets, so that no additional data was required for face swapping during testing. Similar to image-based methods, this method cannot guarantee the temporal smoothness of the output sequence.

3 Background

Reconstruction of facial performance from the video is a reverse engineering problem in computer graphics and computer vision. It solves the reverse rendering, which tries to revert the image formulation process. A typical solution is analysis-by-synthesis [83]. The general idea is that a face image is synthesized from the initial guess and compared with the observed input image. The error (the difference between the synthetic and observed data) is analyzed. The parameters are updated towards reducing the error. Then, the imagery is re-synthesized and compared with the observation again. The process is repeated until converge (the error is below a threshold). In real-time applications, the number of iterations is often restricted, considering that only a total of 33 milliseconds is allowed for each frame computation.

To synthesize an image of a face, a model of face geometry, albedo, and illumination is required. First, the bilinear face model used in this dissertation is described in Section 3.1. Then, how to approximate environment illumination with Spherical Harmonics is shown in Section 3.2 so that a lit face image can be synthesized. Lastly, the camera model used to project a 3D face onto the 2D image plane is explained in Section 3.3. More details on face reconstruction and parameter optimization can be found in Section 4.

3.1 Face Model

The objective of face reconstruction from RGB video is to build a 3D face model and texture that best match the face shape and color in each frame of the video. Two constraints are imposed on the problem: (i) the topology (the number of vertices, the number of faces (triangles), and their connectivity) of the 3D face model is fixed across all frames. (ii) the face texture is fixed across all frames. Unlike depth camera and multi-view reconstruction, monocular RGB video does not contain depth information of the face. Thus, recovering the absolute depth (z) value of the 3D face model is impossible. Instead, a 3D face template is fitted to the input image. A face template is a triangle mesh with artist-designed topology. A template mesh with m vertices can be presented as

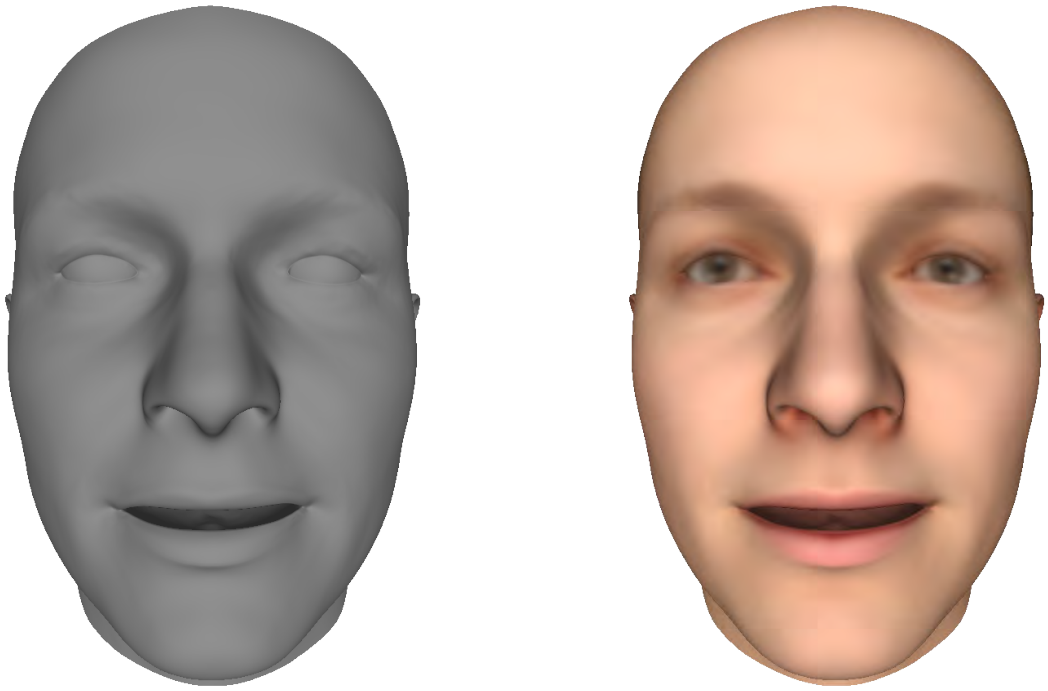
vector b:

$$b = (x_1, y_1, z_1, x_2, y_2, z_2, \ldots, x_m, y_m, z_m)^\intercal,$$

where x, y, z represents the position of a vertex in the face object space. Likewise, a template face texture is presented as

$$\rho = (r_1, g_1, b_1, a_1, r_2, g_2, b_2, a_2 \ldots, r_m, g_m, b_m, a_m)^\intercal,$$

where r, g, b represents the color of a vertex and a represents its transparency. An example of the template is shown in Figure 3.1.



(a) 3D Face Mesh Template (b) Mesh rendered with texture template

Figure 3.1: An example face template from the Basel Face Model [54].

3.1.1 Blendshape

The term "blendshape" was first introduced in the computer graphics industry and widely used for making facial animations. A blendshape is a set of mesh vectors representing individual facial expressions, such as *jaw down, eye close*, etc. The vectors are called the basis of the blendshape. The matrix form of a blendshape with n basis is shown below:

$$\mathrm{B} = \Big[b_1, b_2, \ldots, b_n\Big] = \begin{bmatrix} x_{11} & x_{12} & \ldots & x_{1n} \\ y_{11} & y_{12} & \ldots & y_{1n} \\ z_{11} & z_{12} & \ldots & z_{1n} \\ x_{21} & x_{22} & \ldots & x_{2n} \\ y_{21} & y_{22} & \ldots & y_{2n} \\ z_{21} & z_{22} & \ldots & z_{2n} \\ \vdots & \vdots & \ldots & \vdots \\ x_{m1} & x_{m2} & \ldots & x_{mn} \\ y_{m1} & y_{m2} & \ldots & y_{mn} \\ z_{m1} & z_{m2} & \ldots & z_{mn} \end{bmatrix}.$$

Note that the bases of a blendshape usually have semantic meaning and are human-interpretable. They consist of typical facial muscle movements defined in the FACS [41] system. Some examples of basis mesh are shown in Figure 3.2.

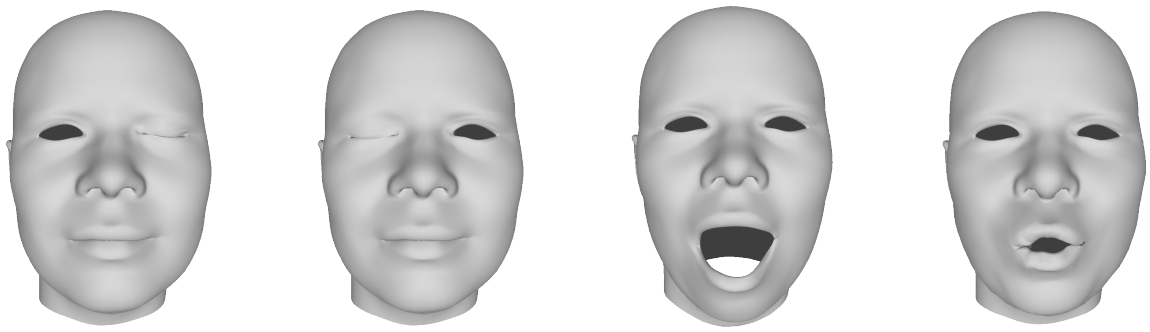


Figure 3.2: Some example basis of a blendshape from the Facewarehouse [25].

Typically, the bases of a blendshape are linearly combined to present a specific expression. Each basis is given a weight of w, and all the weighted bases are summed up. For example, by giving weight one to the "left eye close" and "right eye close" basis, a face with both eyes closed will be created. The product face model f is expressed as

$$\mathrm{f} = \sum_{k=0}^{n} w_k b_k = \mathrm{BW} = \mathrm{B}(w_1, w_2, \ldots, w_k)^\intercal.$$

Therefore, fitting a template mesh to an input image can be reformulated as estimating the weight vector W of a blendshape such that the product mesh f match the facial expression of the input image. Although the blendshape system has a strong capability in conveying various facial expressions, the bases of a particular blendshape share the same face identity, e.g., a round face. If the face shape (e.g., a square face) in the input image is far from the employed blendshape, the fitting result may not be satisfactory. To present both expression and identity in a single model, researchers proposed bilinear face model, described in the next section.

3.1.2 Bilinear Face model

Bilinear face model is a set of blendshapes that share the same topology. Its blendshape bases also represent the same set of expressions. Each blendshape represents an individual face, which is called identity. A bilinear model T that contains k identities is expressed in vector form as

$$\mathrm{T} = (\mathrm{B}_1, \mathrm{B}_2, \ldots, \mathrm{B}_k)^\intercal.$$

Note that each blendshape B is a matrix. Thus T is a rank-three tensor. The three dimensions are vertex positions, expressions, and identities, as shown in Figure 3.3. Intuitively, if we cut the tensor along the identity axis, we will get a blendshape of a specific identity. If we cut the tensor along the expression axis, we will get a set of face meshes with the same expression from all the identities.

Since the tensor is very large and contains redundant data, it is usually compressed using

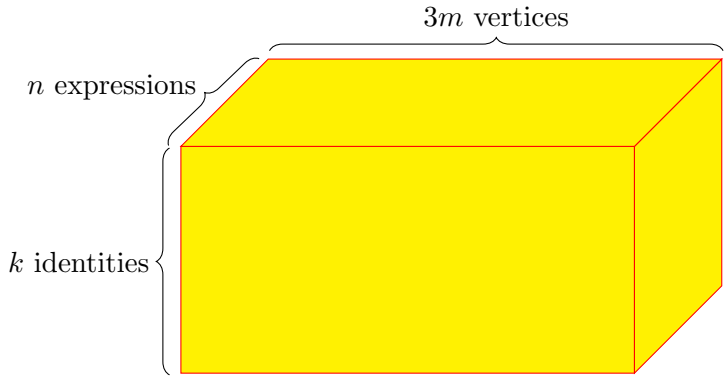


Figure 3.3: An rank-three data tensor.

singular value decomposition (SVD). As the number of vertices m should not be changed, SVD is only performed on the second mode (identity) and the third mode (expression). After truncating the insignificant components, a reduced model C_r is created to approximate the original data tensor

$$\mathrm{T} \simeq C_r \times_2 \mathrm{U}_{id} \times_3 \mathrm{U}_{exp},$$

where U_{id} and U_{exp} are the truncated orthonormal transform matrices, which contain the singular value vectors of the second mode (identity) and third mode (expression) respectively. In practice, the reduce core tensor C_r is used to fit the face to the input image. Any facial expression of any identity can be approximated by the tensor contraction

$$V = C_r \times_2 m_{id}^{\intercal} \times_3 m_{exp}^{\intercal}, \tag{3.1}$$

where m_{id} and m_{exp} are the column vectors of identity weights and expression weights, respectively. Note that each face within the core tensor C_r is no longer semantically meaningful. In some scenarios, such as expression transformation (more details in Section 5.1), it may be preferred to compute weights in the blendshape space. Given an identity vector m_{id} and a blendshape weight vector w, the same face mesh can be obtained

$$V = C_r \times_2 m_{id}^{\intercal} \times_3 (\mathrm{U}_{exp} w). \tag{3.2}$$

3.2 Spherical Harmonics

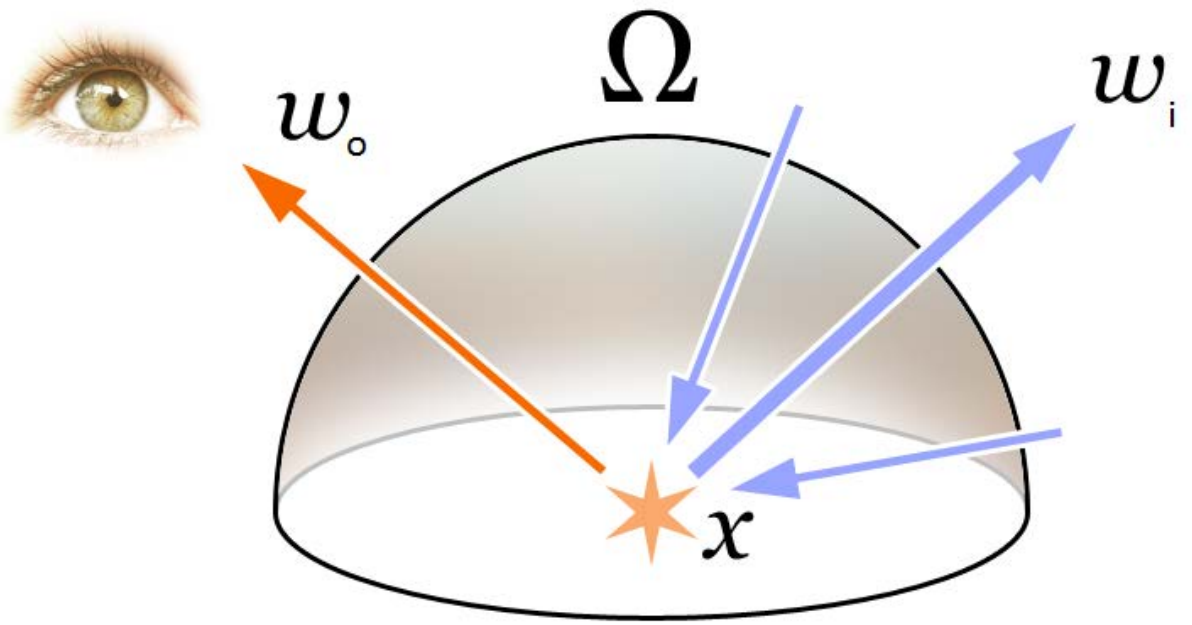


Figure 3.4: The rendering equation describes the total amount of light emitted from a point x along a particular viewing direction, given a function for incoming light and a BRDF.

To synthesize a realistic face photo, a virtual illumination model needs to be defined. In the real world, the lighting condition could be very complex. There might be multiple directional light and point light sources in various colors. In computer graphics, the rendering equation (Figure 3.4) is used to measure the mount radiance into viewing direction from a surface point

$$L_o(x, \omega_o) = L_e(x, \omega_o) + \int_\Omega f_r(x, \omega_i, \omega_o) L_i(x, \omega_i)(\omega_i \cdot n) d_{\omega_i},$$

where

- $L_o(x, \omega_o)$ is the total outward radiance along the direction ω_o from a surface point x.
- $L_e(x, \omega_o)$ is the emitted radiance along the direction ω_o from a surface point x.

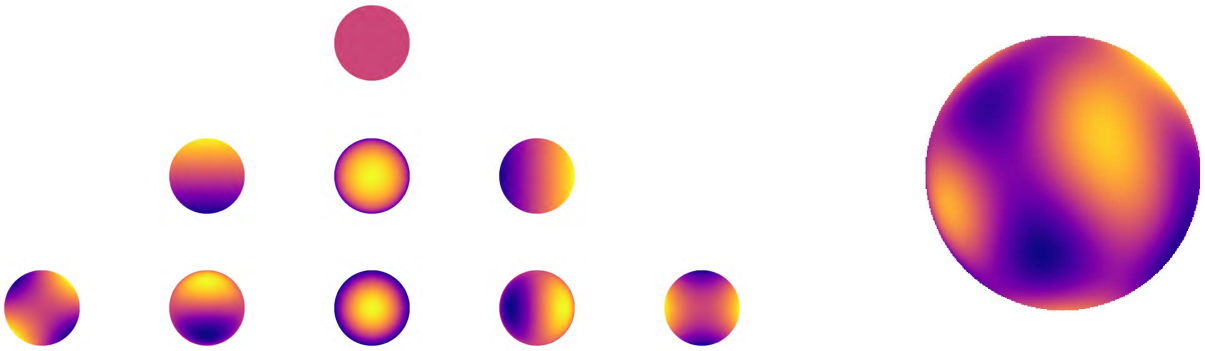
- $\int_\Omega$ is the integral over the hemisphere Ω centered around the surface point x containing all possible values for incoming light direction ω_i.
- $f_r(x, \omega_i, \omega_o)$ is the bidirectional reflectance distribution function (BRDF [58]), measuring how much incident light from direction ω_i is scattered to outgoing direction ω_o.
- $L_i(x, \omega_i)$ is the radiance coming inward toward x from direction ω_i.
- $\omega_i \cdot n$ is the dot product of incoming light direction ω_i and surface normal n at x.

To simplify the problem, faces are assumed to be Lambertian surfaces, i.e., emission and specular reflection are ignored [48, 143, 157]. The BRDF f_r becomes a constant albedo ρ. It is also assumed that light sources are distant from faces such that incoming lights equally arrive at the surface. The incident light L_i only depends on the light direction ω_i. Then, the rendering equation boils down to

$$L_o(x, \omega_o) = \rho \int_\Omega L_i(\omega_i)(\omega_i \cdot n) d_{\omega_i}.$$

A per-vertex albedo ρ has been defined in Section 3.1. The normal vector n can also be computed given the face geometry. To represent the incident light L_i, a common approach in computer graphics is using cube mapping. Cube mapping is a method of environment mapping that uses the six faces of a cube to approximate the sphere enclosing the target object. Each face of the cube is a texture storing the lighting information. To get L_i, a ray is caste from the surface point x in direction ω_i, and the hit point on the cube is sampled as the radiance L_i. Obviously, six textures are required to represent the illumination in this method with each texture n^2 unknown pixel values per color channel. Another much more compact way is to represent the irradiance environment mapping method by spherical harmonics [113].

Spherical harmonics are orthogonal functions defined on the surface of a sphere. Each function defined on the surface of a sphere can be written as a sum of these spherical harmonics. This is similar to the Fourier series, where a periodic function can be composed of a weighted sum



(a) First three degrees of SH (b) Composed illumination

Figure 3.5: First three degrees of SH (spherical harmonics) (left) and an example of weighted composition.

of sine and cosine functions. An example of spherical harmonics basis functions and composite illumination on a sphere is shown in Figure 3.5. Spherical harmonic basis functions are defined on polar coordinates. A complex exponential representation is defined as

$$Y_l^m = Ne^{im\phi}P_l^m(cos\theta).$$

Here Y_l^m is called a spherical harmonic function of degree l and order m. P_l^m is an associated Legendre polynomial in Cartesian coordinates [107], N is a normalization constant, and θ and ϕ represent colatitude and longitude, respectively. As shown in Figure 3.5, each degree contains $2l+1$ basis functions with $m \in [-l, l]$. A particular illumination function $g(\omega_i)$ that is defined on a spherical environment map can then be approximated using these basis functions as

$$L(\omega_i) \approx \sum_{l=0}^{d-1} \sum_{m=-l}^{l} g_l^m Y_l^m,$$

where d is the number of used degrees, and g_l^m are the coefficients of the corresponding basis functions. Following [113, 114], the environment map is so smooth that three SH degrees $b = 3$ are sufficient to achieve an average error below 1%. This results in only $b^2 = 9$ variables per color channel. Table 1 shows the first three degrees of the SH basis functions in Cartesian coordinates

[71]. As can be seen, higher degrees of functions represent higher frequency. Finally, the radiance of the surface point x can be approximated as

$$L_o(x, \omega_o) \approx \rho \sum_{l=0}^{2} \sum_{m=-l}^{l} g_l^m Y_l^m(n). \tag{3.3}$$

Table 1: First three degrees of the SH basis function Y_l^m.

Degree (l)	Order (m)				
0			$\frac{1}{2\sqrt{\pi}}$		
1		$\frac{\sqrt{3}}{2\sqrt{\pi}}y$	$\frac{\sqrt{3}}{2\sqrt{\pi}}z$	$\frac{\sqrt{3}}{2\sqrt{\pi}}x$	
2	$\frac{\sqrt{15}}{4\sqrt{\pi}}(x^2-y^2)$	$\frac{\sqrt{15}}{2\sqrt{\pi}}xz$	$\frac{\sqrt{5}}{4\sqrt{\pi}}(3z^2-1)$	$\frac{\sqrt{15}}{2\sqrt{\pi}}yz$	$\frac{\sqrt{15}}{2\sqrt{\pi}}xy$

3.3 Camera Model

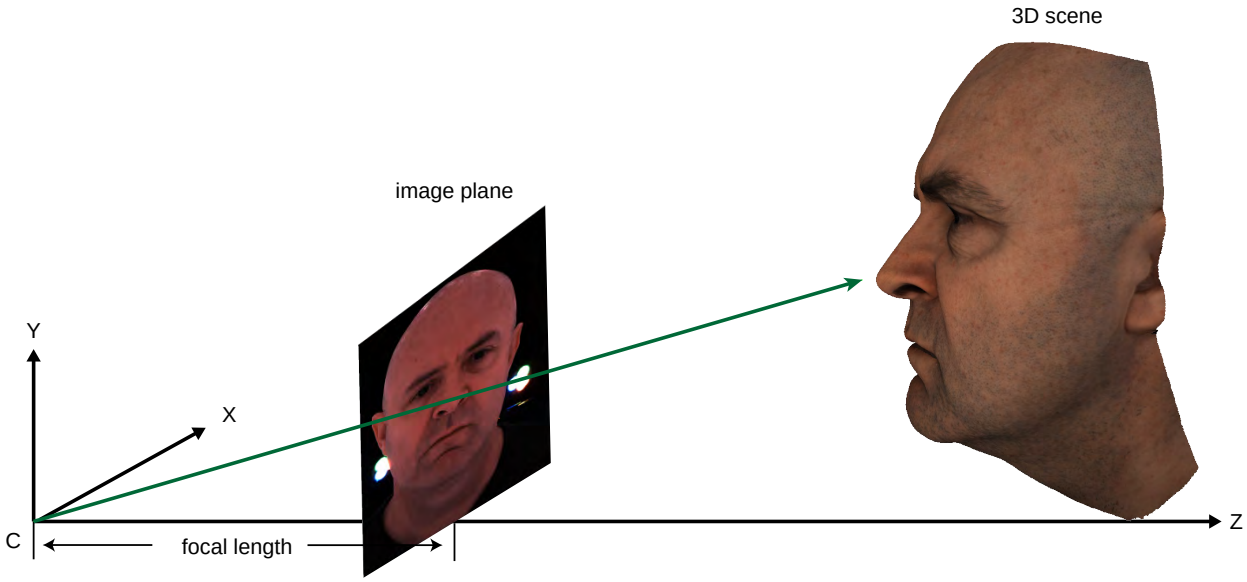


Figure 3.6: Pinhole camera model.

To render a face image with the geometry and illumination discussed previously, a camera model is required for the projection of 3D objects onto the 2D image plane. In this dissertation, commodity RGB cameras are used in the experiment setup since they are widely available on modern laptops and smartphones. Videos prevalent on the internet are almost always taken by

RGB cameras. Another advantage is that RGB cameras are passive. In contrast to some types of depth sensors, they do not influence the 3D scene that is being captured. The pinhole camera model [62] is used as shown in Figure 3.6. In this model, a scene view is formed by projecting 3D points into the image plane using a perspective transformation

$$y = \Pi(Rx + t),$$

where y is the projected location on the image plane, Π denotes the camera projection matrix, and R and t are rotation and translation of the 3D object. In homogeneous coordinate, the equation can be expanded as

$$\begin{bmatrix} u \\ v \end{bmatrix} = \Pi \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_x \\ r_{21} & r_{22} & r_{23} & t_y \\ r_{31} & r_{32} & r_{33} & t_z \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}.$$

The matrix Π is the intrinsic parameters of a camera that does not depend on the scene viewed. So, once estimated, it can be re-used as long as the focal length is fixed. The rotation-translation matrix $[R|t]$ is called extrinsic parameters. It describes the rigid motion of the captured object in the 3D scene, which needs to be estimated for each frame within a video. Note that the pinhole camera model does not take distortion into account, which is common in real lenses. In a controlled setup, the intrinsic parameters can be estimated through calibration [18, 165]. The camera takes several views of a calibration board, which has aligned features that can be easily detected with known distance in 3D space.

However, in uncontrolled setups, such as internet videos, calibration is impossible. Instead, facial landmark correspondences are used to estimate the parameters. As shown in Figure 3.7, a number of automatic facial landmark detection algorithms [115, 121] have been designed to locate 2D facial landmarks on the observed image. On the 3D face model counterpart, the index of

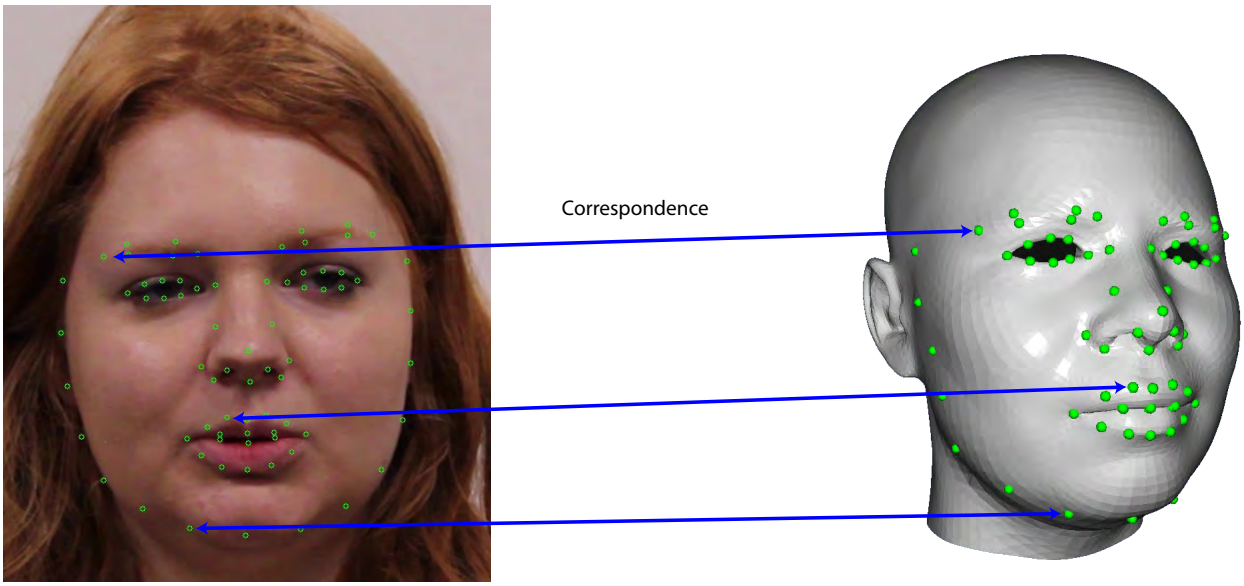


Figure 3.7: Correspondences between detected 2D landmarks and 3D vertices.

vertices that corresponds to the 2D landmarks are pre-defined. Then, the camera parameters could be estimated by minimizing the distance of detected 2D landmarks and the projected corresponding 3D vertices

$$\arg\min_{\Pi,R,t} \sum_{i=1}^{k} \|y_i - \Pi(Rx_i + t)\|_2^2 ,$$

where k is the number of landmarks used in the landmark detection algorithm. To solve the equation, the unknown face model is initialized with an average identity and neutral expression. After the estimation of camera parameters, the face model is then updated. The process is repeated until converge.

4 Face Reconstruction

This chapter introduces how to reconstruct fine-scale facial performance from RGB video in real-time. First, a low-resolution facial geometry is reconstructed from the input video. This model presents large-scale facial expressions but lacks wrinkle level details. It is subsequently refined using shape-from-shading techniques. The coefficients of incident illumination, albedo texture, and displacements are recovered by minimizing the difference between the synthesized face and the input RGB images. To endow the face model the capability to capture fine-scale details, the mesh is adaptively subdivided into a hierarchy of multiple resolutions. A corresponding pyramid for the input images is also built. The vertex displacements estimated on a coarse mesh capture low-frequency deformations and are prolonged to finer levels, where higher frequency details will be captured from images of higher resolutions. Figure 4.1 shows the schematic pipeline of the proposed system.

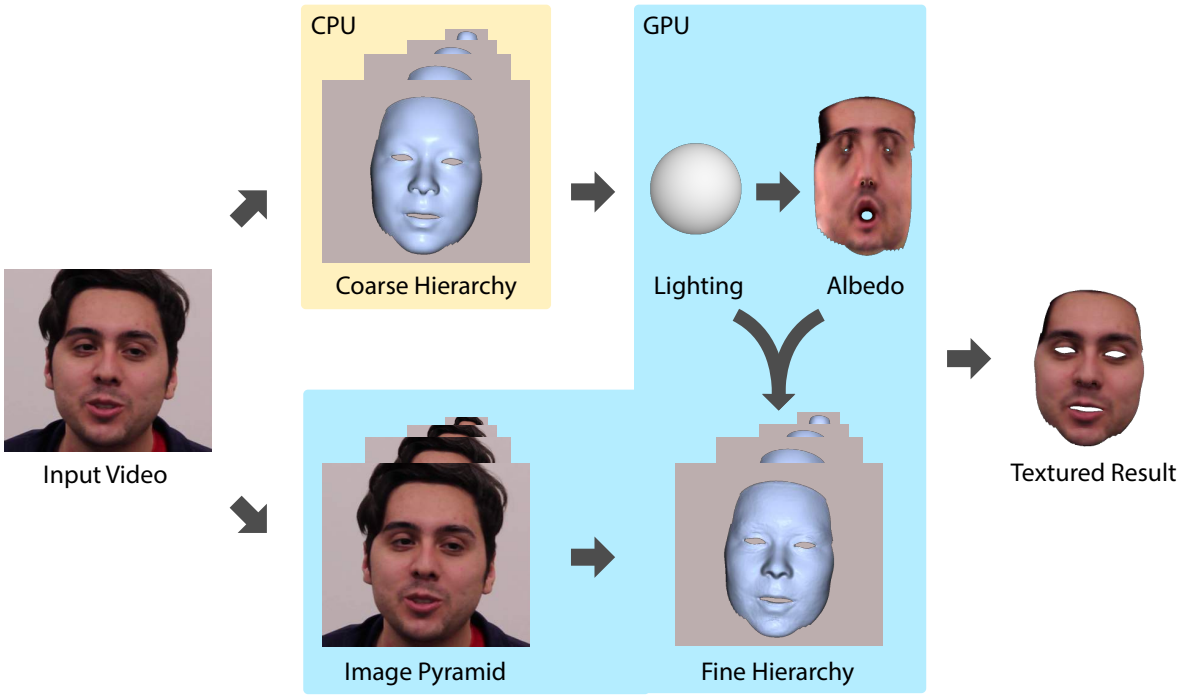


Figure 4.1: System overview. The proposed system takes RGB video as input and builds coarse mesh layers and image layers for hierarchical reconstruction. It then computes lighting, albedo, and vertex displacements using shape-from-shading techniques. The output is a high resolution mesh with wrinkle-level details.

The main contributions of the work in this chapter can be summarized as:

- a complete, fully automatic system to capture fine-scale facial performance from monocular RGB video;
- a hierarchical reconstruction method that is efficient and robust to reconstruct high resolution face models; and
- a novel vertex displacement formula to solve the shape-from-shading problem.

4.1 Large-scale Face Tracking

At first, the large-scale 3D facial performance is reconstructed frame by frame from the input video. A 3D facial models is represented using the multi-linear models [145, 24]. Specifically, a 3D face is described by using two low-dimensional vectors that control identity and expression, respectively:

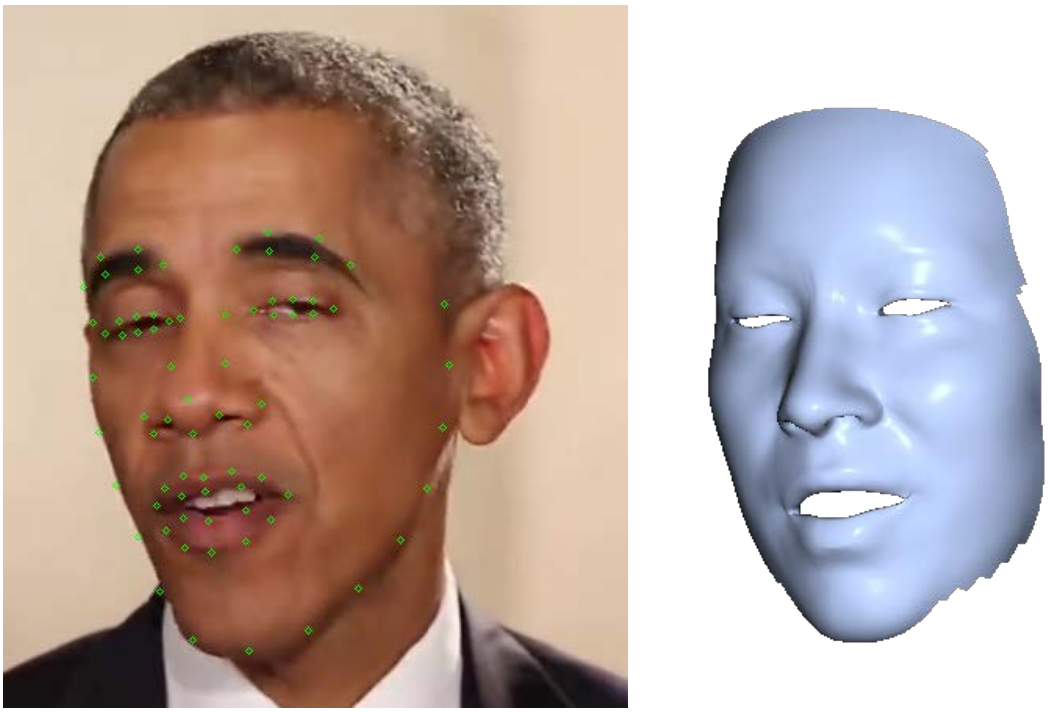
$$M = R(C_r \times_2 m_{id}^\intercal \times_3 m_{exp}^\intercal) + T, \tag{4.1}$$

where M represents the facial geometry in camera space, R and T represent the global rotation and translation of the head, C_r is the reduced core tensor, and m_{id} and m_{exp} are identity and expression parameters (more details in Section 3.1.2). The multi-linear model is constructed based on the FaceWarehouse dataset [25]. In the experiments, the dimensions of the identity vector m_{id} and the expression vector m_{exp} are set to 50 and 25, respectively.

It is assumed that the camera projection is a full perspective Π of $\{f_x, f_y\}$, where f_x and f_y are the focal length in x and y direction. Then, the 2D projection (i, j) in image space of a 3D vertex $(X, Y, Z)^\intercal$ in camera space is represented as:

$$\begin{pmatrix} i \\ j \end{pmatrix} = \Pi \begin{pmatrix} X \\ Y \\ Z \end{pmatrix} = \begin{pmatrix} f_x(X/Z) + c_x \\ f_y(Y/Z) + c_y \end{pmatrix}, \tag{4.2}$$

where (c_x, c_y) is the camera's principal point and is set to the image center.



(a) Landmarks (b) Large-scale Mesh

Figure 4.2: Large-Scale 3D face reconstruction from an input frame: detected 2D facial landmarks (green circles) (a), and coarse 3D face model (b). Image courtesy: The White House (public domain, via YouTube https://goo.gl/mAb6iP).

For each frame, 73 2D facial landmark locations are detected (see Figure 4.2a) using the local binary feature (LBF) based regression [147, 115]. Similar to [125], the large-scale face model (Figure 4.2b) is reconstructed by minimizing the following energy function:

$$E(m_{id}, m_{exp}, R, T, f_x, f_y) = E_{lan} + w_1 E_{prior} + E_s. \tag{4.3}$$

Feature Constraint The distance between the detected 2D landmark locations p_i and the projection of predefined corresponding 3D facial features M_i is measured as:

$$E_{lan} = \sum_{i=1}^{73} \|\Pi(M_i) - p_i\|_2^2 . \tag{4.4}$$

Regularization Constraint A Gaussian prior constraint E_{prior} is imposed for regularizing m_{id} and m_{exp} which is formulated as:

$$E_{prior} = \left\| \frac{m_{id} - \bar{m}_{id}}{\sigma_{id}} \right\|_2^2 + \left\| \frac{m_{exp} - \bar{m}_{exp}}{\sigma_{exp}} \right\|_2^2, \tag{4.5}$$

where $(\bar{m}_{id}, \sigma_{id})$ and $(\bar{m}_{exp}, \sigma_{exp})$ are the mean and covariance prior for m_{id} and m_{exp}, respectively, constructed from the FaceWarehouse dataset [25].

Temporal Constraint To encourage temporal smoothness of expression m_{exp} and head pose $\{R, T\}$, a smoothness term is employed

$$E_s = w_2 \left\| m_{exp}^i - m_{exp}^{i-1} \right\|_2^2 + w_3 \left\| R^i - R^{i-1} \right\|_2^2 + w_4 \left\| T^i - T^{i-1} \right\|_2^2. \tag{4.6}$$

$\{m_{id}, f_x, f_y\}$ are solved at the start of the video and keep fixed for the remaining frames. The expression and head pose are estimated using block coordinate descent algorithms for each frame. Considering both the dimension of unknown variables and the number of residuals are relatively small, a CPU multi-threaded trust region solver is used to solve the non-linear Equation 4.3, for parallel residual/gradient evaluation and efficient dense matrix operations. w_1,w_2,w_3 and w_4 are set to $0.00001, 100, 10$ and 1 in all the experiments.

4.2 Hierarchical Reconstruction

The captured face model contains 5.6K vertices and 33K triangle faces presenting large-scale facial deformations. Next, the coarse face model is augmented with wrinkle and fold details.

Mesh and Image Hierarchy The resolution of the mesh is too low to faithfully reproduce the subtle details of the input image through vertex displacements. Therefore, a hierarchical approach is proposed to progressively capture finer details using higher resolution meshes. Specifically, the coarse mesh is used as the control mesh and iteratively subdivided to build a mesh hierarchy

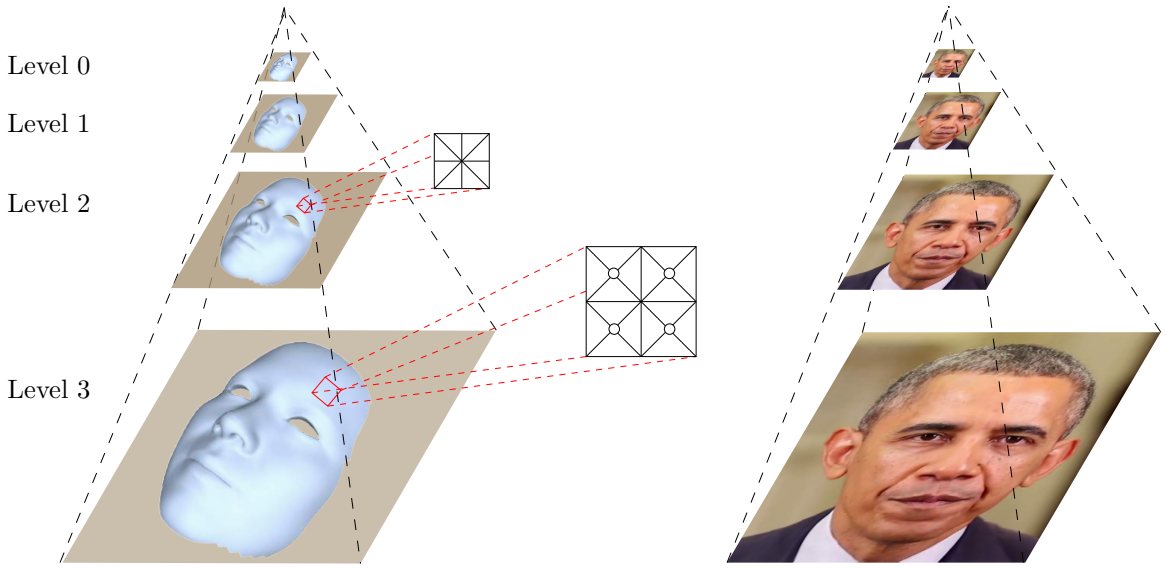


Figure 4.3: An example of 4-level hierarchy where lower levels have higher resolution meshes and images. The closeups show the topology for the same mesh blocks at two consecutive levels, with the lower level containing 4 more vertices denoted as circles. Image courtesy: The White House (public domain, via YouTube https://goo.gl/mAb6iP).

$\{M_0, M_1, \ldots, M_k\}$ with M_0 denoting the coarsest level and M_k denoting the finest level. A corresponding image pyramid $\{I_0, I_1, \ldots, I_k\}$ is also constructed from the GPU generated mipmap of the input frame. Figure 4.3 shows an example of the constructed hierarchy, where the top level has the lowest resolution mesh and image, while the bottom level has the highest resolution. The closeups show that four new vertices (circle) are inserted into a mesh block (red square) after subdivision to account for the additional pixels in the corresponding higher resolution image.

4-8 subdivision The 4-8 subdivision scheme [144] is employed to adaptively subdivide the mesh. The reason for choosing this scheme is two-fold: (i) the coarse mesh M_0 is a triangulated quadrilateral mesh which exactly meets the requirement of 4-8 subdivision. (ii) 4-8 subdivision results in conforming meshes (without edge cracks) and simple adaptive subdivisions. As shown in Figure 4.4, the initial control mesh consists of thousands of quad blocks. The two triangles $\{f_1, f_2\}$ are called the *mate* triangles of a block $\{v_1, v_2, v_3, v_4\}$. Notice that a regular triangle f_1 has two vertices v_1, v_3 with valence $= 8$ and one vertex v_2 with valence $= 4$. The edge v_1v_3 which is opposite to the

vertex with valence = 4 is denoted as an interior edge. The basic subdivision operation of a mesh block is bisection. In essence, a new vertex is inserted to bisect the interior edge, and two new edges are created to connect the new vertex to the two vertices with valence = 4. The positions of the new vertex and the old vertices are computed using the face mask and vertex mask, respectively, as shown in the middle of Figure 4.4.

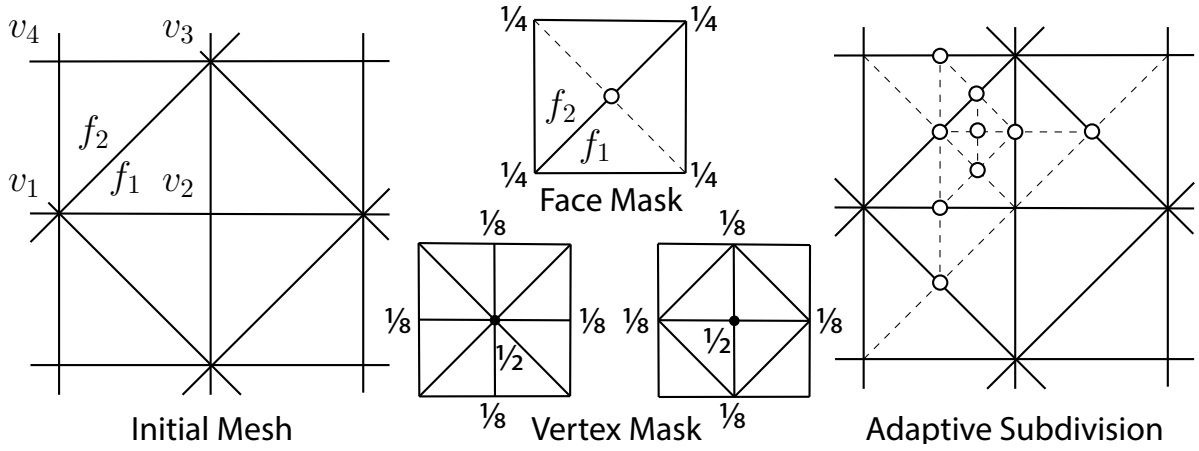


Figure 4.4: 4-8 subdivision. Left: a coarse initial mesh; Middle: the face/vertex mask for mesh block bisection; Right: the multi-level adaptive subdivision. The added vertices and edges are denoted as circles and dashed lines, respectively.

Each finer mesh M^{i+1} is built from the coarser mesh M^i by looping through all the faces $f \in F$ with the *adaptFace* function (Algorithm 1). To prevent over-refinements, i.e., multiple vertices are projected to a single pixel, the triangle faces, whose interior edge is smaller than a threshold ε when projected to the image space (line 4) using the estimated head pose and camera parameters, are skipped. The subdivision terminates when no triangle face needs to be adapted. Prior to the bisection of a face f, the *mate* face of f (denoted as $f.mate$) needs to be at the same subdivision level as f (line 5) to prevent cracks, since $f.mate$ might be skipped by previous subdivision steps. Then, the positions of the two end vertices of the interior edge of f are updated using the *adaptVertex* function. The actual bisection operation of *adaptFace* is performed by bisecting the interior edge (line 10) and the two triangles f (line 12) and $f.mate$ (line 13). Note that the mesh hierarchy is only constructed at the start of the video and then keep fixed. At run-time, the mesh is progressively

refined from coarse levels and prolonged to fine levels following the same topology and face/vertex mask. The displacements-based hierarchical mesh refinements will be discussed in more details in Section 4.3.3.

Algorithm 1: Adaptive 4-8 Subdivision

```
1  Function adaptFace(V, F, f):
2      if f has not been processed then
3          e ← getInteriorEdge(f);
4          if project(e).length > ε then
5              while f.mate.level ¡ f.level do
6                  adaptFace(V, F, f.mate);
7              end
8              adaptVertex(V, F, e.start, f.level);
9              adaptVertex(V, F, e.end, f.level);
10             v ← bisectEdge(e);
11             put v in V;
12             f1 ← bisectTriangle(f);
13             f2 ← bisectTriangle(f.mate);
14             put f1, f2 in F;
15         end
16     end
17 end
18 Function adaptVertex(V, F, v, l):
19     if v has not been processed then
20         for f ∈ F containing v do
21             while f.level ¡ l do
22                 adaptFace(V, F, f);
23             end
24         end
25         updateVertexPosition(v);
26     end
27 end
```

4.3 Shading-based Refinements

Similar to previous off-line methods [125], human faces are assumed as Lambertian surfaces. Shape-from-shading is employed to capture dynamic fine details such as wrinkles and folds. The fine surface bumps are encoded as the displacements along surface normals and recovered jointly with the unknown illumination and albedo from the input RGB images. The incident lighting is parameterized

with spherical harmonics [5], and assumed to contribute equally to each RGB channel. Therefore, the reflected irradiance R at vertex i is represented as:

$$R_i = l^\intercal \cdot SH(n_i)\rho_i, \tag{4.7}$$

where ρ_i is the face albedo at vertex i, l is the vector of spherical harmonics coefficients of incident lighting, and SH is the spherical harmonics basis functions taking a unit length surface normal n_i as the input. The 2nd order harmonic approximation is taken which captures more than 99% of the energy in a face image [76]. Then $SH(n) \in \mathbb{R}^9$ evaluates to:

$$SH(n) = (1, n_x, n_y, n_z, n_x n_y, n_x n_z, n_y n_z, n_x^2 - n_y^2, 3n_z^2 - 1)^\intercal,$$

where n_x, n_y, and n_z are the components of the normal n.

The inverse rendering equation (Equation 4.7) is solved in an analysis-by-synthesis strategy. The lighting, albedo, and displacements are iteratively optimized by minimizing the difference between the current observed image I and the rendered image R:

$$E_{data} = \sum_{i=1}^{K} \|I_i - R_i\|_2^2, \tag{4.8}$$

where I_i is the sampled image color by projecting vertex i onto the image plane according to head pose and camera parameters, and K is the number of vertices. At a frame t, the refined (target) face mesh is roughly approximated as the captured coarse mesh augmented with the displacements from the previous frame:

$$M_d^t = M^t + d^{t-1} N^t,$$

where M^t is the coarse mesh without refinements, N^t is its derived per-vertex normals, and M_d^t is the augmented mesh with the previous frame's displacements d^{t-1}. The superscript t will be

dropped in the remainder of this chapter. Then, the lighting can be estimated (Section 4.3.1) using the recomputed vertex normals from M_d and the albedo from the previous frame. The albedo is subsequently updated (Section 4.3.2) with the computed lighting and normals. Note that the albedo is computed only at the start of the video and remains fixed thereafter. Finally, the displacements are re-estimated (Section 4.3.3) for current frame by formulating normals as a function of displacements. The details of each step are described below.

4.3.1 Lighting Estimation

The illumination is assumed to vary across frames, but sudden changes between consecutive frames are penalized with a smoothing term. The total energy function for illumination becomes:

$$E(l) = E_{data} + w_1 \left\| l^t - l^{t-1} \right\|_2^2, \tag{4.9}$$

where w_1 is set to 0.005 in the experiments. Minimizing this energy in terms of l is equivalent to solving the linear system $Al = b$, where A is a $(3K + 9)$ by 9 matrix. The top $3K$ rows evaluate to the product of $SH(n)$ and each channel of ρ, and the bottom 9 rows are an identity matrix. Only those visible vertices are considered, whose normals face towards the camera ($n_z > 0$). This leads to a highly over-constrained linear system and substantial set-up and computational time if conducted on the CPU. To achieve real-time performance, a pure GPU solution is employed. The matrix A and the vector b are constructed on the GPU with each thread assembling one row. Then, $A^\intercal A$ and $A^\intercal b$ are computed using cuBLAS[2] and solve the *normal equation* $A^\intercal Ab = A^\intercal b$ using Preconditioned Conjugate Gradient (PCG) solver (Section 4.3.4) implemented on the GPU. As $A^\intercal A$ is a low dimensional matrix (9×9), the PCG solver runs on a single block with one kernel call and converges in 9 iterations. Since illumination is a global environment attribute, its accuracy will not be significantly improved on finer levels. Hence, the coarsest mesh M_0 is directly used for fast computation.

[2]https://developer.nvidia.com/cublas

4.3.2 Albedo Recovery

Given the estimated lighting and approximated normals, the albedo at each vertex can be naively computed as

$$\rho_i = \frac{I_i}{l^\intercal \cdot SH(n_i)}.$$

However, this leads to baking the residuals of E_{data} into the albedo, such that fine details, such as wrinkles, are interpreted as albedo change. Therefore, a Laplacian regularization term is incorporated to adapt the albedo to be as smooth as the prior average albedo. The energy function becomes:

$$E(\rho) = E_{data} + w_2 \left\| L\rho - L\bar{\rho} \right\|_2^2, \tag{4.10}$$

where $\bar{\rho}$ is the average albedo provided by the FaceWarehouse [25], and L represents the graph Laplacian matrix with respect to the mesh.

Minimizing Equation 4.10 is equivalent to solving a sparse linear least square problem $A\rho = b$. For a mesh with E edges, this problem has $3K$ unknowns, $(3K + 3E)$ residuals, and $(3K + 6E)$ non-zero values. Similar to the lighting estimation, it is also converted to the *normal equation* using cuSPARSER[3] and solve it using a PCG solver on the GPU. ρ is initialized as $\bar{\rho}$ at the first frame and updated for 100 PCG iterations with $w_2 = 0.5$. Note that the albedo is only updated within the first several frames and keeps fixed afterward. To avoid baking geometric detail into the albedo, it is desired for all the mesh levels in the hierarchy to have as close as possible albedo. Therefore, the albedo is computed on the coarsest mesh M_0 and prolonged to finer levels where the albedo for additional vertices are interpolated with the vertex mask in Figure 4.4. Figure 4.5 shows an example of the recovered albedo and lighting.

[3] https://developer.nvidia.com/cusparse

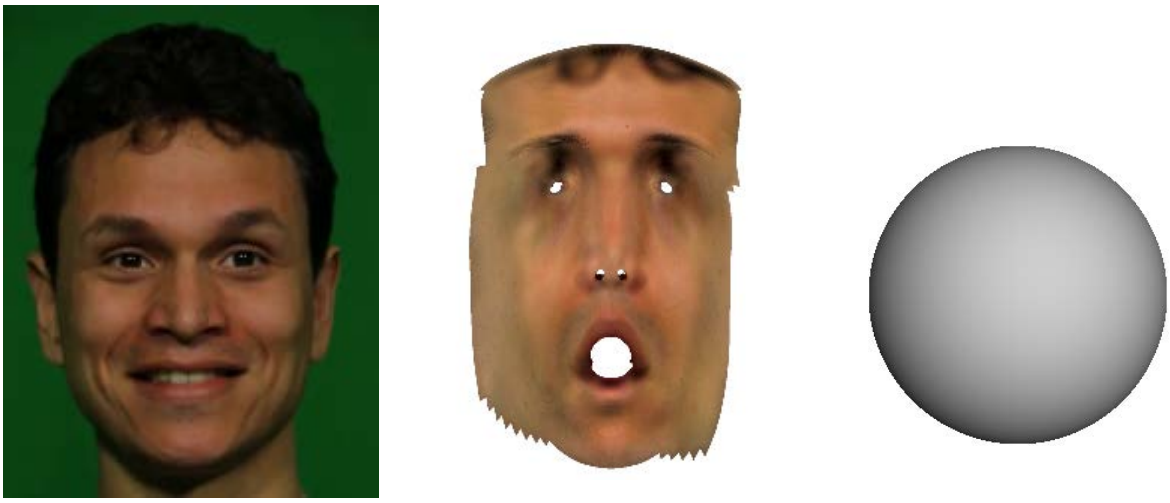


Figure 4.5: From an input RGB image (left), the albedo texture (middle) and incident lighting (right) are recovered.

4.3.3 Displacements Refinement

This section introduces how to augment the coarse mesh with geometric details via shape-from-shading. Equipped with the initialized mesh hierarchy described in Section 4.2, the corresponding image pyramid is first built on the GPU for a particular frame t. Starting from the coarsest level, the mesh is enhanced with vertex displacements to improve the consistency between surface reflectance R and the corresponding image I. The computed displacements on a coarser level are then prolonged to the next finer level as initialization for further improvement as well as a constraint to regularize the displacements on the finer level to be close to those on the coarser level. The advantages of this progressive approach over direct refinements on the finest level include improved robustness and less noise. Therefore, to obtain a smooth face surface, a smoothness term that measures local displacement smoothness is required in the energy function. The proper weight of this term, however, is very difficult to find: a large weight may result in an over-smoothed surface that does not contain fine details, while a small weight may bring the noise to the surface. In the proposed approach, coarser levels are used to capture lower frequency bands of geometric features that serve as reasonable guidance for refinements at finer levels. On finer levels, higher frequency details are added, and noise is removed using the higher resolution input image. The

difference between with and without hierarchical refinements can be seen in Figure 4.6: using a small smoothness weight, the direct reconstruction method incurs noise around the nose, while the hierarchical method gradually removes noise from coarse levels and adds details on finer levels.

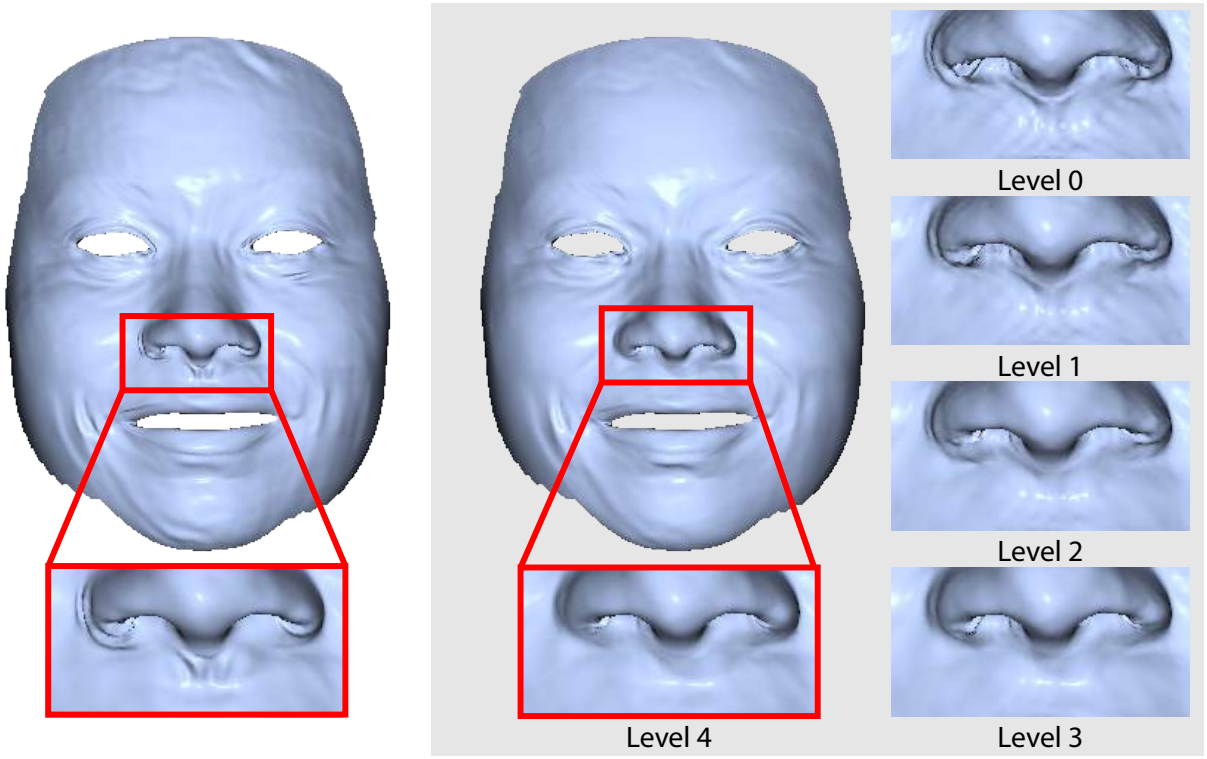


Figure 4.6: The proposed hierarchical method (right) progressively refines the coarse mesh, producing a less noisy, high resolution geometry, compared to the direct reconstruction method (left) that only refines on the highest resolution level.

To solve the displacements d, E_{data} is formulated as a function of d. For a vertex v_i, its normal n_i is calculated as a weighted sum of all the neighboring face normals. If the weight ω is set to be twice of the area of the neighbor triangle, the vertex normal calculation can be converted to:

$$n_i = \sum \omega \frac{(v_j - v_i) \times (v_k - vi)}{|(v_j - v_i) \times (v_k - v_i)|} = \sum (v_j - v_i) \times (v_k - v_i),$$

where v_j and v_k are the other two vertices of a triangle containing v_i. In this way, the vertex normals of the coarse mesh can be obtained. Recall that the refined vertex position can be presented as a

displacement along the coarse normal, thus the vertex normal on the desired refined mesh can be expressed as a function of displacements d:

$$n_i^* = \sum((v_j + d_j n_j) - (v_i + d_i n_i)) \times ((v_k + d_k n_k) - (v_i + d_i n_i)), \tag{4.11}$$

where d_i, d_j, and d_k denote the unknown displacements for vertex v_i, v_j, and v_k, respectively. By substituting Equation 4.11 into E_{data} (Equation 4.8), an energy function where d is the only unknown variable is derived. Nonetheless, the problem is still under-constrained. Instead, the following non-linear energy function is optimized with additional constraints:

$$\begin{aligned} E(d) =& E_{data} + w_3 E_{smooth} + w_4 E_{hier} + w_5 E_{temp} + w_6 E_{coarse}, \\ \text{s.t.} \quad & \partial M(d) = 0, \end{aligned} \tag{4.12}$$

where ∂M denotes the boundary of the face surface, serving as the Dirichlet boundary condition.

Smoothness Constraint For a C^2 surface, its local displacements should change smoothly. Similar to albedo, a graph Laplacian is employed to improve the displacement smoothness:

$$E_{smooth} = \|Ld\|_2^2 = \sum_{i=1}^{K} \left\| d_i - \sum_{j=1}^{r} \frac{1}{r} d_j \right\|_2^2,$$

where r is the number of 1-ring neighbors for vertex v_i, and d_j is the j-th neighbor's displacements.

Hierarchical Constraint For the mesh at a level higher than 0, its displacement is initialized as the prolongation of the displacement at the coarser level. The prolonged displacement pd is also used as a regularizer for the current level:

$$E_{hier} = \sum_{i=1}^{K} \|pd_i - d_i\|_2^2 .$$

Temporal Constraint The displacement is also enforced to be temporally coherent. Sudden changes are penalized by:

$$E_{temp} = \sum_{i=1}^{K} \left\| d_i^t - d_i^{t-1} \right\|_2^2 .$$

Coarse Constraint It is assumed that the coarse mesh already provides a good approximation of the ground truth, thus the displacements are expected to be small:

$$E_{coarse} = \sum_{i=1}^{K} \| d_i \|_2^2 .$$

In all the experiments, the weights w_3, w_4, w_5, and w_6 are set to 10, 30, 1, and 7, respectively. Figure 4.7 presents the enhanced surface details for the normal map and the geometry after refinements. The non-linear energy Equation 4.12 consists of K unknowns and $6K$ residuals. A data-parallel GPU Gauss-Newton solver is employed for real-time optimization.

Figure 4.7: The normal map and coarse mesh before (left) and after (right) the shape-from-shading refinements.

4.3.4 Energy Minimization

Gauss-Newton Solver The non-linear energy $E(d)$ can be rewritten as the sum of squared residuals:

$$E(d) = \sum_{k=1}^{6K} \| \mathrm{r}_k(d) \|_2^2 = \sum_{k=1}^{6K} \| \mathrm{y}_k - \mathrm{f}_k(d) \|_2^2 . \tag{4.13}$$

Initialized by the displacements from the previous frame, the local optimal displacements $d^* \in \mathbb{R}^K$ is going to be found, where the gradient is zero. To this end, the parameter vector d is updated through several Gauss-Newton steps. At a step n, d^n is updated as:

$$d^{n+1} = d^n + \Delta d \quad \text{and} \quad J^\intercal J \Delta d = J^\intercal(\mathrm{y} - \mathrm{f}(d)), \tag{4.14}$$

where J is the Jacobian matrix of f with respect to d:

$$J(d) = \left[\frac{\partial \mathrm{f}_i(d)}{\partial d_j}\right]_{i=1,..,6K, j=1,..,K} = \begin{bmatrix} \nabla \mathrm{f}_1(d)^\intercal \\ \nabla \mathrm{f}_2(d)^\intercal \\ \vdots \\ \nabla \mathrm{f}_{6K}(d)^\intercal \end{bmatrix}.$$

The data term of J is evaluated only at the initialization of each step and stored into the global memory, while the other terms are evaluated on-the-fly for less memory access. The optimal step length Δd is computed by the PCG solver, described below.

Preconditioned Conjugate Gradient (PCG) Solver Similar to the work of [168], the PCG solver requires 2 kernel calls at initialization and 3 kernel calls per iteration loop for synchronizations across thread blocks. The Jacobi conditioner is used to ensure a fast convergence. The inverses of the diagonal entries of $J^\intercal J$ are computed at initialization. However, to take advantage of the sparsity of J, $J^\intercal J$ is never explicitly evaluated. Instead, whenever the multiplication of matrix $J^\intercal J$ with a vector p is needed, it is converted to two sequential matrix-vector multiplications Jp and $J^\intercal(Jp)$. Algorithm 2 illustrates the solving of unknown x given Jacobian J, observed data y, and initial value x_0, in t iterations.

4.4 Results

In this section, the implementation and runtime of the method are described. Then, the method is quantitatively and qualitatively evaluated by comparing the results with input video, ground-truth

Algorithm 2: Preconditioned Conjugate Gradient Solver

Function `pcg`(J, y, x_0, t)**:**
  $\mathrm{r}_0 = \mathrm{y} - J^\intercal J x_0$;
  $M = trace(J^\intercal J)$;
  $z_0 = M^{-1}\mathrm{r}_0$;
  $p_0 = z_0$;
  $k = 0$;
  while $k < t$ **do**
    $\alpha_k = \frac{\mathrm{r}_k^\intercal z_k}{p_k^\intercal J^\intercal J p_k}$;
    $x_{k+1} = x_k + \alpha_k p_k$;
    $r_{k+1} = r_k - \alpha_k J^\intercal J p_k$;
    $z_{k+1} = M^{-1} r_{k+1}$;
    $\beta_k = \frac{\mathrm{r}_{k+1}^\intercal z_{k+1}}{\mathrm{r}_k^\intercal z_k}$;
    $p_{k+1} = z_{k+1} + \beta_k p_k$;
    $k = k + 1$;
  end
  return x_t;
end

meshes, and results by existing offline and real-time methods.

4.4.1 Implementation

The nonlinear optimization problem in large-scale reconstruction is solved using the Trust Region strategy on the CPU. The block coordinate descent algorithms is also used for faster convergences: the pose, identity, and expression parameters usually converge in five iterations. The hierarchical refinement is solved on the GPU using CUDA. The two components run in fully parallel. Ten outer Gauss-Newton steps and seven inner PCG iterations are run for displacements refinement. The convergences of the solver with various configurations is compared, as shown in Figure 4.8. The ten/seven configuration is found to achieve the optimal balance between the convergence rate and the computational cost.

Table 2 presents the runtime statistics of the proposed system that runs on a desktop computer with Intel Core i7 CPU @3.7 GHz and nVidia Geforce GTX 2080Ti GPU. Live facial performance of subjects was captured using a Logitech C922x Pro webcam, which is capable of recording at

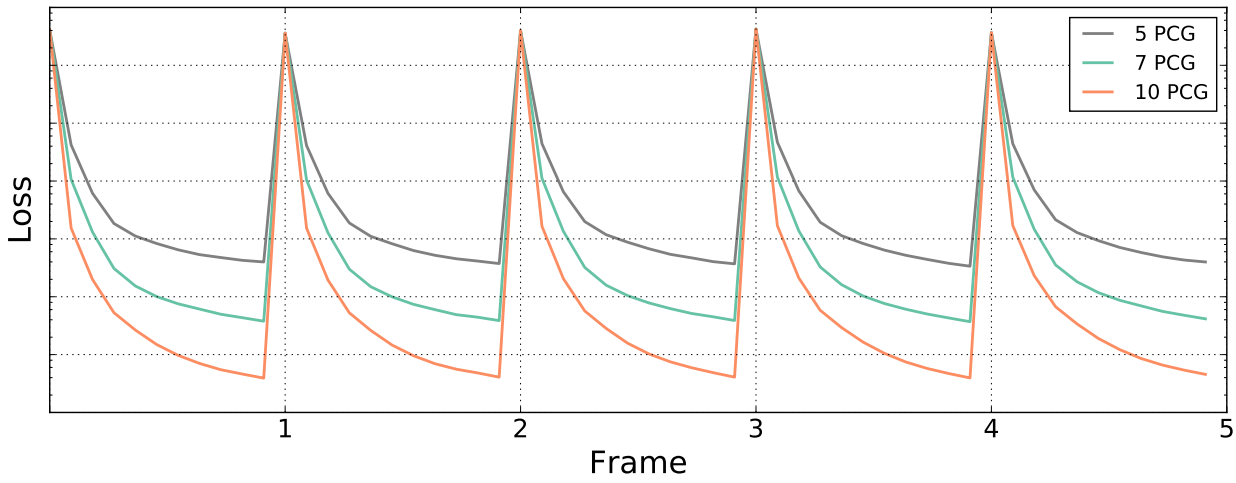


Figure 4.8: The convergences of of Gauss-Newton solver for 10 Gauss-Newton steps within 5 frames.

Table 2: Runtime statistics for the proposed method and timing comparison with [21] and [60]. Note that the albedo is not computed per frame in the proposed method.

Ours				
	Albedo	Lighting	Displacement	
1280x720	112 ms	1 ms	35 ms	28 fps
800x600	98 ms	1 ms	16 ms	50 fps
[21]				
		Global Tracker	Local Detail	
648x860		32 ms	23 ms	18 fps
[60]				
		CoarseNet	FineNet	
256x256		5 ms	15 ms	50 fps

720p resolution and 60 fps. The system ran at 28 fps for 720p video and 50 fps for 800x600 video. The speedup from the GPU solver is due to fewer levels in the hierarchy for lower resolution input video. The final frame rate is determined by the slower side: GPU in the case of 720p video and CPU in the case of 800x600 video. All the results presented were recorded at 720p for better visual quality.

4.4.2 Evaluation

The accuracy and effectiveness of the proposed method is demonstrated in Figure 4.9 and 4.10, where fine-scale facial geometries are captured for subjects with various skin colors, head poses,

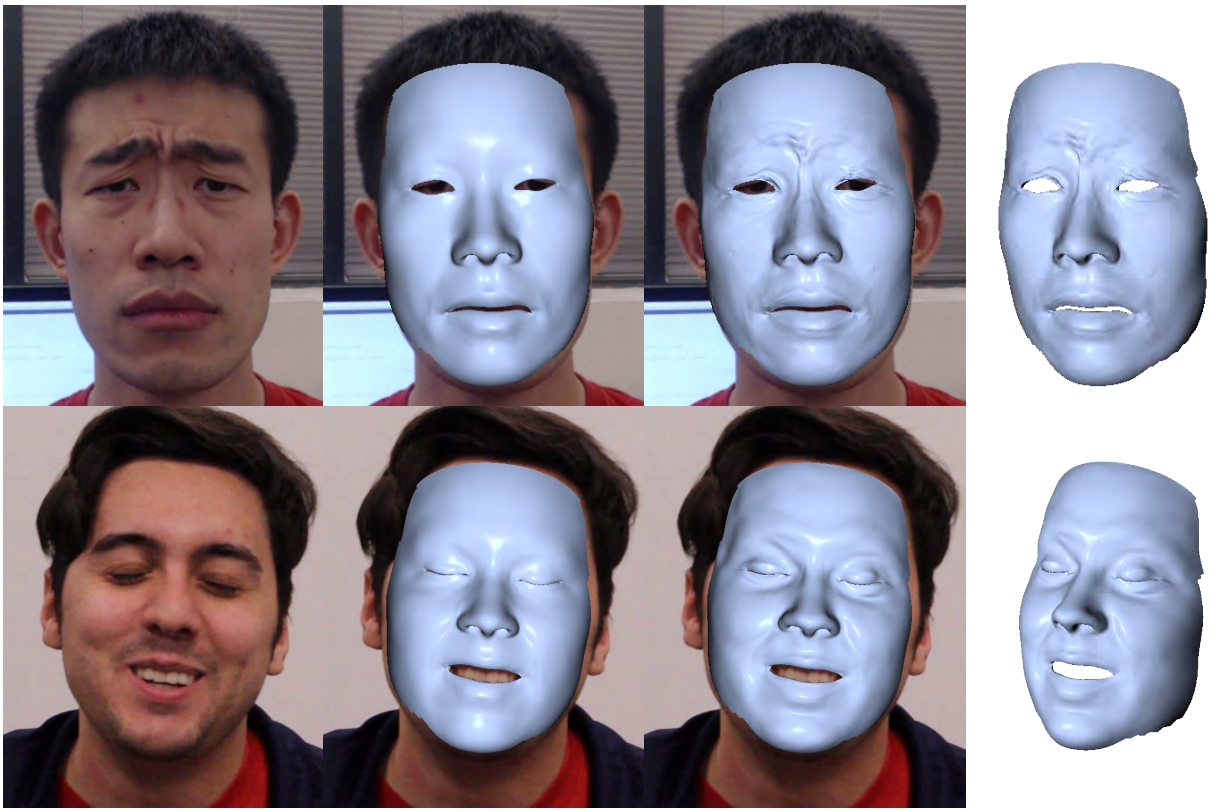


Figure 4.9: The proposed method captures coarse-scale (the second row) facial performance as well as fine-scale (the third and fourth rows) details on various identities, expressions, and head poses without any preprocessing or manual corrections.

expressions, and wrinkles. The proposed method is shown to be capable of capturing fine-scale skin details from shading changes for live camera video streams as well as legacy video footage.

Quantitative Evaluation The proposed method was first quantitatively evaluated by comparing the input video (i.e., images) with the synthesized textured face using estimated head pose, geometry, illumination, and albedo. As shown in Figure 4.11, the photometric error is relatively low except at the corner of the forehead, where the highlight breaks the Lambertian surface assumption. Then, the method was evaluated on the binocular FaceCap dataset [143], where only the image sequence from one camera was used as input. The mesh as registered with the ground-truth mesh from [143]. A point-to-mesh distance was computed. As shown in Figure 4.12, the proposed method achieved the average error = 1.96 mm and the standard deviation = 1.35 mm, which is

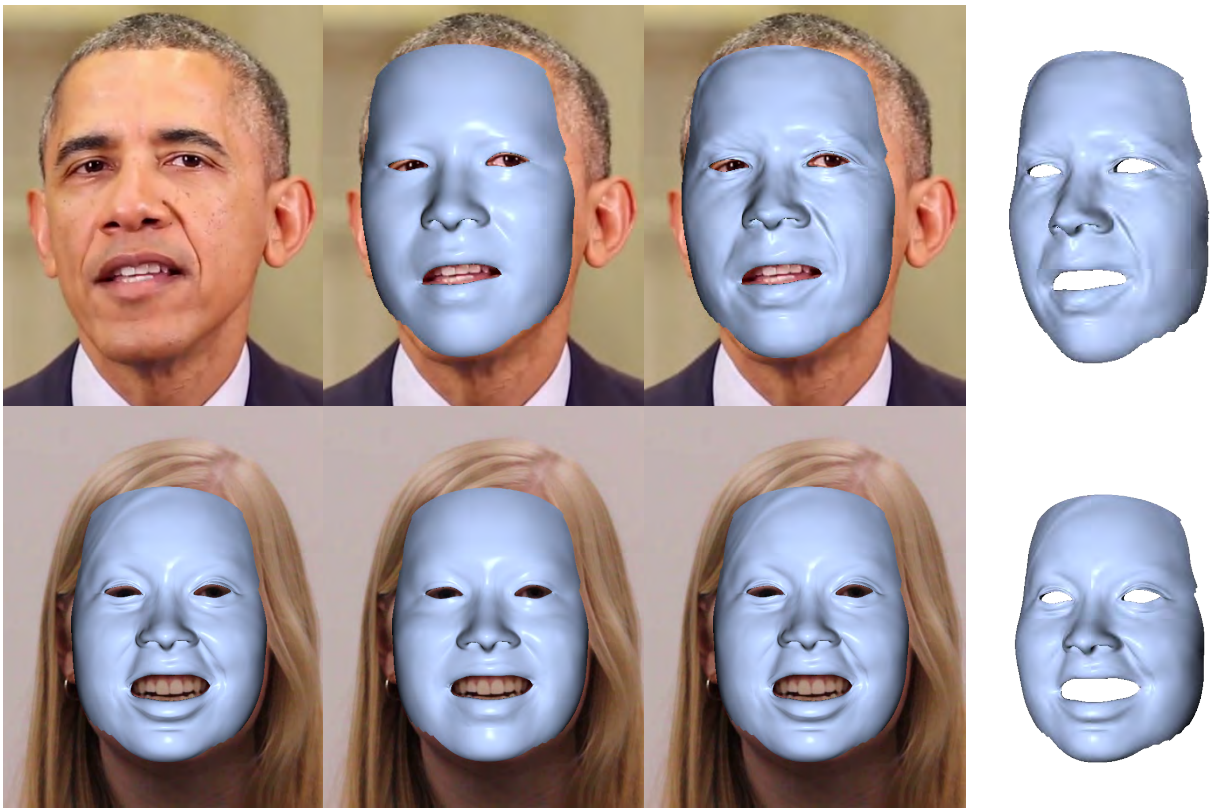


Figure 4.10: More results on the facial performance capture method. Image courtesy: The White House (public domain, via YouTube https://goo.gl/mAb6iP).

more accurate than the state-of-the-art, CNN-based, real-time method [60] that had the average error = 2.08 mm and the standard deviation = 1.63 mm.

Comparisons with Off-line Methods The proposed method was compared with the state-of-the-art off-line methods that reconstruct faces from RGB video. Beeler et al. [8] capture high-quality facial performance with multiple synchronized cameras. This hardware setup prevents their method from processing legacy video footage (e.g., Youtube video). Moreover, their method requires the manual selection of an anchor frame while the proposed method runs in a fully automatic manner. The work of Garrido et al. [47] also lacks the capability to process legacy video as it relies on a high-quality face scan of the subject for blendshape construction. Besides, offline methods typically involve iterative refinements across the whole sequence, thereby are unsuitable for on-line tracking. As shown in Figure 4.13 and 4.14, the proposed method produced the results of close

Figure 4.11: The photometric accuracy of the proposed method (from left to right): the input frame, the rendered face, and the heat map showing photometric errors. Image courtesy: The White House (public domain, via YouTube https://goo.gl/mAb6iP).

quality to them but in real-time, without any preprocessing for the subject or forward/backward information of input video.

Comparisons with Real-time Methods Most existing real-time methods https://yahoo.com [24, 23, 140] target on capturing coarse-scale facial expression and head motion from RGB video. Recently, Cao et al. [21] proposed to extend the coarse global tracker with predicted local details. Their approach trains local regressors by learning correlations between image patches and surface details from a database of high-quality face scans. While plausible displacements are combined to the coarse mesh, their augmented mesh is not geometrically correct but only a close approximation of the true surface. Furthermore, medium-scale details are only added to locations where image patches are detected as wrinkles, while other regions are left with surface skins learned from the database. As a result, large scale deformations such as sunken cheeks (top row of Figure 4.13) are often absent in their results. Similarly, the CNN-based real-time method by Guo et al. [60] produces an approximated but noisy surface (Figure 4.12). By contrast, the proposed method can produce truly reconstructed geometric details by minimizing the shading energy over the whole face. Therefore, it is able to capture more accurate facial details as presented in Figures 4.12, 4.13

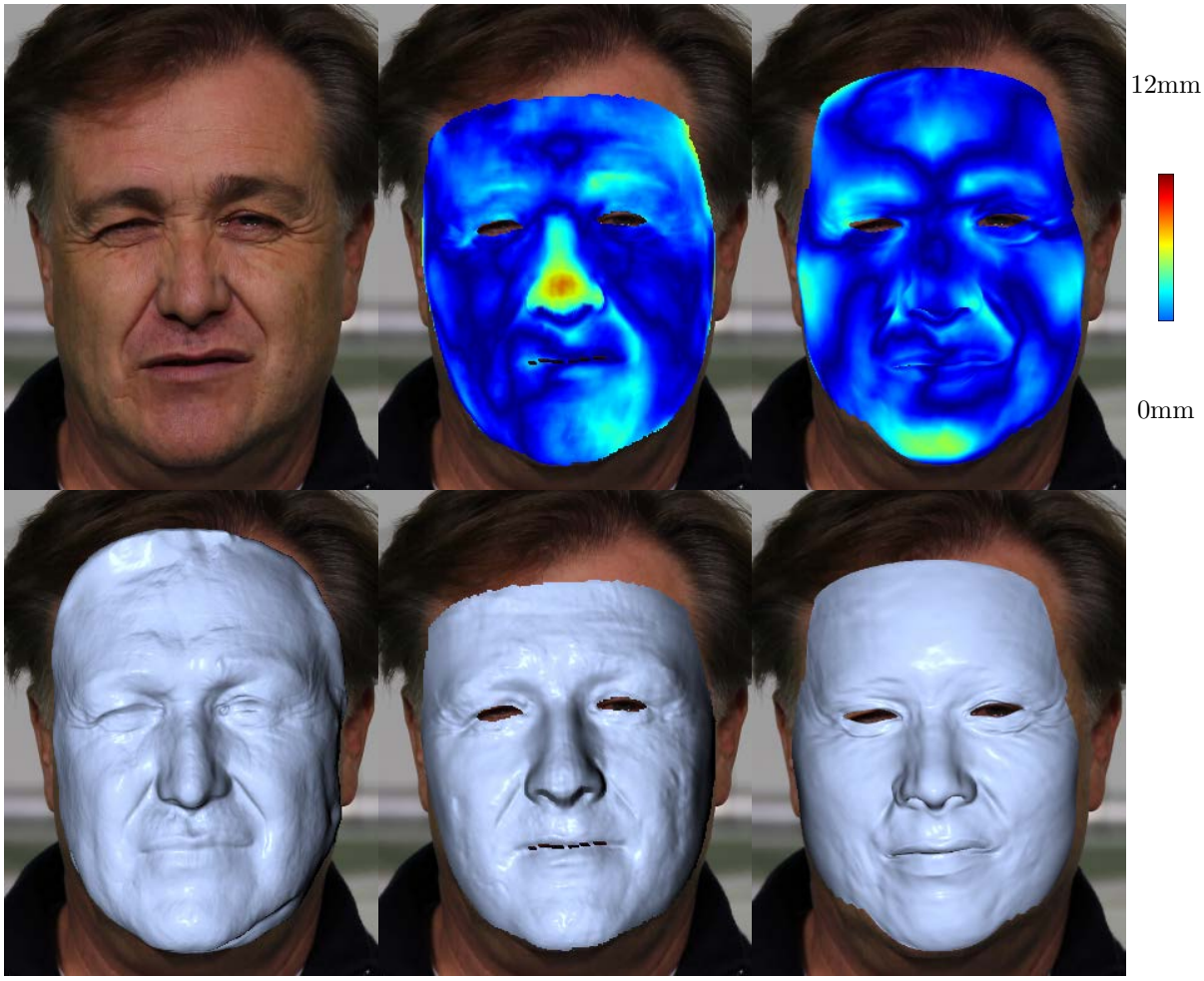
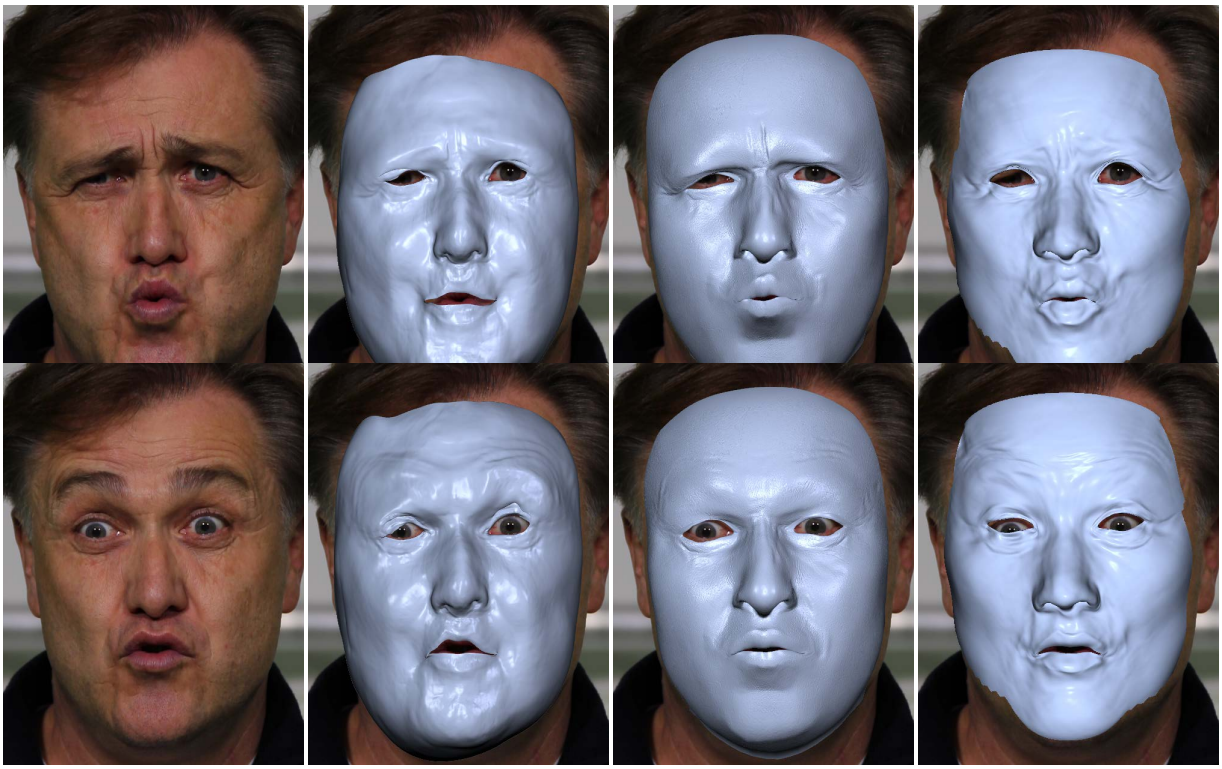


Figure 4.12: Comparison between the proposed method and the CNN-based method [60]. From left to right: the input video frame and ground truth mesh from the binocular method [143], the result by the CNN-based method, and the result by the proposed method.



Input Garrido et al. [47] Cao et al. [21] Ours

Figure 4.13: Compared to the offline monocular method [47], the proposed method produces similar results. Compared to the real-time method [21], the proposed method excels in capturing large-scale deformations, such as sunken cheeks on the top two rows.

Input Beeler et al. [8] Cao et al. [21] Ours

Figure 4.14: Compared to the multi-view based method [8], the proposed method produces similar results. Compared to the real-time method [21], the proposed method excels in capturing large-scale deformations.

and 4.14. Compared to the above two real-time methods, in terms of running time, the proposed achieves the highest frame rate while at higher resolutions (see Table 2).

4.5 Limitations

The current approach relies on the detection of a sparse set of 2D facial landmarks. Inaccurate detections could lead to incorrect head poses or subject identities. Since the face surface is assumed to have Lambertian reflectance, specular highlights, unsmooth illumination, and cast shadows could incur artifacts. Similarly, occlusions such as hair and glasses might be misinterpreted as geometric changes. The albedo texture is recovered within the first several seconds. Therefore, as much as possible face orientations are encouraged during this stage for a complete albedo texture recovery. Figure 4.15 shows some failure cases for specular highlights, occlusions, and incomplete albedo

texture on a novel head pose.

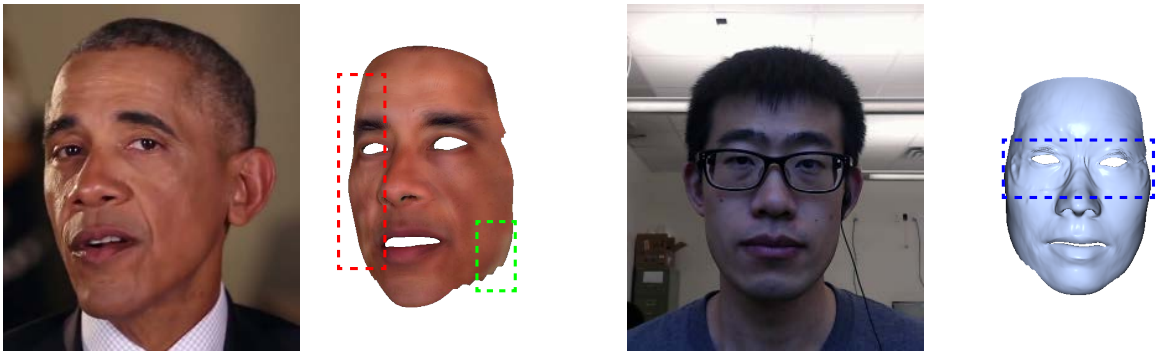


Figure 4.15: Limitations. Specular highlights (red) and occlusions (blue) cause artifacts. Insufficient head orientations during initialization leads to incomplete albedo (green). Image courtesy: The White House (public domain, via YouTube https://goo.gl/mAb6iP).

5 Expression Transformation

In this chapter, a complete pipeline for real-time facial expression transformation is proposed. The system transforms the source expression of the subject in an input (source) monocular RGB video clip to a user-specified target expression and then photo-realistically re-render the same performance but with the target expression. The generated facial expression sequence is temporally dynamic, coherent, and lip-synchronized to the source audio. Specifically, the main components of the system (Figure 5.1) are briefly described as follows:

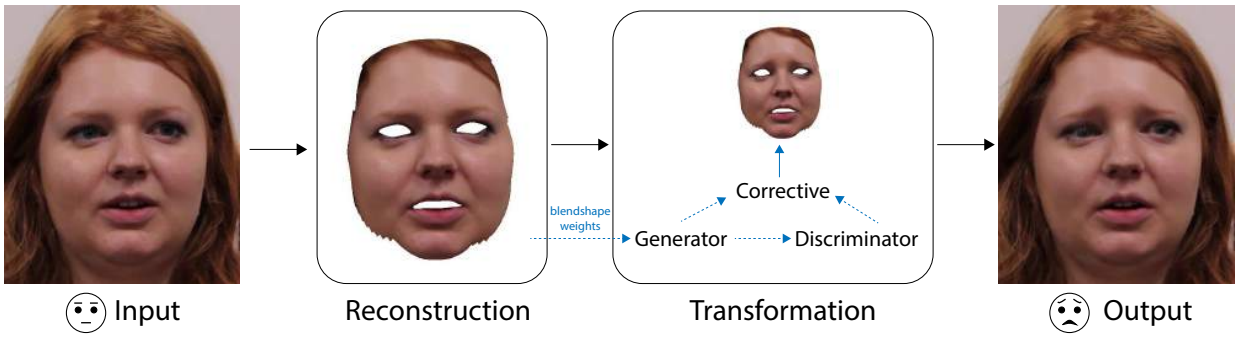


Figure 5.1: From an input face video clip, the proposed system first reconstructs the head pose, identity, expression, and albedo map of its 3D face, as well as camera parameters and environment lighting. After that, the source expression in the face is transformed to the desired (target) expression through CycleGAN-based expression mapping in the blendshape space and followed by mouth corrective and smoothing. The re-rendered face with the target expression is finally blended back to the input video.

- **3D facial performance reconstruction**. The process starts by reconstructing 3D facial performance from an input video clip. The facial performance is reconstructed by detecting 2D facial landmarks and then estimating the 3D facial deformations (i.e., identity and expression parameters) as well as 3D head poses and camera parameters. Then, the lighting conditions and facial albedo are estimated using the shading cues (see Section 5.1).

- **CycleGAN-based expression mapping**. A CycleGAN based expression mapping model is trained to learn the bi-directional mapping between two facial expressions based on an unpaired face video dataset. Different from existing image-based approaches using

Convolutional Neural Networks (CNN), the model is learned in a facial blendshape space. Its architecture and training will be discussed in Section 5.2.

- **Expression transformation and lip correction**. The trained CycleGAN-based expression mapping is applied to a small window of neighboring frames for smoother and more confident results in target expression (see Section 5.3). The lip motion is synchronized to the source audio by minimizing the distance of key lip vertices before and after transformation in Section 5.3.2.

- **Composition**. The face is re-rendered with the transformed facial expression, original head poses, lighting, and albedo map. Since the shapes of the mouth might be significantly different between the original source expression and the transformed target expression, the proposed approach warps the mouth region from the source frame into the transformed face. Finally, the composition is completed by putting the re-rendered face and the warped mouth regions in different image layers and blending back to the original input video (see Section 5.3.3).

The main contributions of this work can be summarized below:

- a complete, real-time pipeline for facial expression transformation on monocular RGB video, without the need of any driving sources;

- a data-driven approach to solving the automatic bi-directional transformation between a pair of facial expressions; and

- a novel optimization formula for temporally coherent and lip synchronized expression transformation.

5.1 Blendshape Reconstruction from Video

Firstly, the real-time facial performance capture method described in Section 4 is used to reconstruct the identity, expression, albedo, head pose, and environment lighting from the input video. Figure 5.2 shows an example of the reconstruction result. However, there is a difference in expression

(a) Landmarks (b) Geometry (c) Texture

Figure 5.2: An example of 3D face reconstruction from an input frame: The detected 2D facial landmarks (a), the resulting face model without (b) and with (c) texture.

estimation. In Section 4.1, the expression vector is computed in the truncated PCA space. The recovered expression is not semantically meaningful. In other words, a facial action in the FACS system, e.g., mouth open, is not affected by a single dimension but by multiple elements. Similarly, if one value of the PCA expression vector is changed, multiple FACS will change. This would impose extra difficulties when only some specific regions of the face is desired to be changed. To tackle this problem, the original expression weights in the blendshape space is computed. As mentioned in Section 3.1.2, the PCA expression vector m_{exp} can be obtained as the product of the truncated expression orthonormal transform matrix U and the blendshape weights vector w. Then, Equation 3.2 can be reformulated as

$$V = C_r \times_2 m_{id}^{\intercal} \times_3 (\mathrm{U}_{exp} w) = (C_r \times_2 m_{id}^{\intercal} \times_3 \mathrm{U}_{exp}) w = \mathrm{B}w.$$

where B represents the blendshape reconstructed from the identity parameter m_{id}. In the Facewarehouse data, there are 47 bases in the blendshape. The Delta blendshape formulation [88] is used, in which the first basis B_0 is the neutral expression. The expression parameter becomes a 46-dimensional vector representing the difference between neutral expression and the remaining

bases. Then, Equation 4.1 can be rewritten as

$$M = R(\mathrm{B}_0 + \sum_{i=1}^{46} v_i(\mathrm{B}_i - \mathrm{B}_0)) + T.$$

5.2 Cycle-Consistent Expression Mapping

This section describes how to transform the source expression in the input video to a specified (target) expression. Facial expressions can be quite complicated; directly mapping the facial appearance from one expression to another requires a significant amount of training examples to cover the varieties across illuminations, races, ages, genders, etc. Instead, with the above reconstructed facial deformations and texture, this expression mapping is proposed to be learned in the blendshape weights space $\{v_i\}_{i=1}^{46}$. Since this approach may need to take speech video as the input, theoretically, a large number of training blendshape pairs are needed to be collected, which are aligned at the phoneme level to learn an effective mapping between two expressions. Collecting such a training dataset is dauntingly time-consuming and error-prone due to phoneme alignments. Besides, the transformed expression needs to be continuous and smooth.

To address the above challenges, at the first step, the Cycle-Consistent Generative Adversarial Network (CycleGAN) [167] is employed to learn the mapping between a source expression and a target expression, without phoneme-phoneme alignments. Compared to the original CycleGAN model that can only take an image as input, the proposed model directly consumes blendshape weights, which are transparent to identity, pose, texture, and lighting conditions. The model training can also converge faster and requires significantly fewer training data than the original CycleGAN model.

5.2.1 Blendshape Weights Mapping

This section describes how to employ the CycleGAN model to learn the expression mapping in the blendshape weights space. From the reconstructed blendshape weights of the used training video dataset [61], training expression pairs (x_i, y_i) are sampled independently from a source domain

and a target domain. Next, given samples in two expression domains X and Y (e.g., neutral and happiness), a mapping function $\mathbb{G} : X \to Y$ is learned, with the expectation that the transformed samples $\mathbb{G}(x)$ are as close as possible to real samples in the domain Y.

The method constructs a CycleGAN-based expression mapping model by learning a backward mapping function $\mathbb{F} : Y \to X$. As illustrated in Figure 5.3, a real sample x in domain X is transformed to $\mathbb{G}(x)$ in domain Y and then mapped back to $\mathbb{F}(\mathbb{G}(x))$ in domain X. Similarly, a cycle transformation of y is expressed as $\mathbb{G}(\mathbb{F}(y))$. To reduce the space of possible mapping functions, the result of cycle transformation is enforced to be as close as possible to the corresponding real samples, i.e., $\mathbb{F}(\mathbb{G}(x)) \approx x$ and $\mathbb{G}(\mathbb{F}(y)) \approx y$.

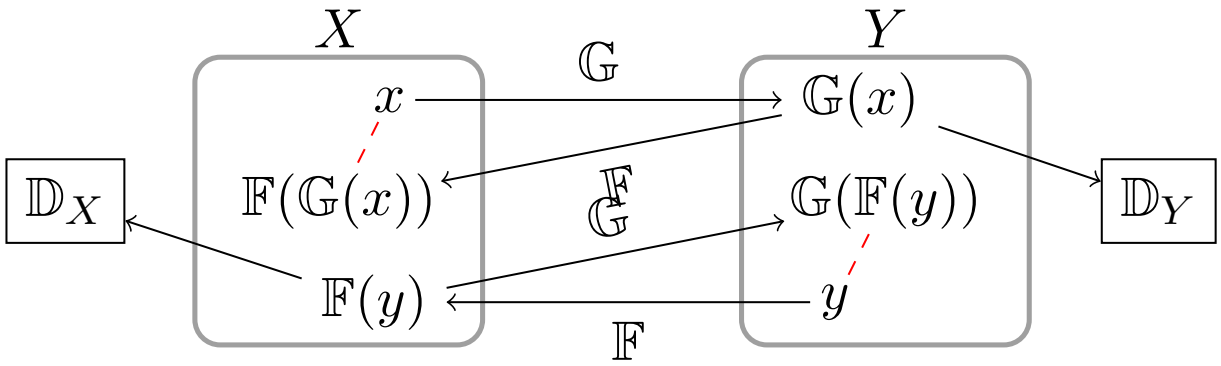


Figure 5.3: Two mapping functions $\mathbb{G}$ and $\mathbb{F}$ between two expressions X and Y are learned, where $\mathbb{G} : X \to Y$ and $\mathbb{F} : Y \to X$. $\mathbb{D}_X$ and $\mathbb{D}_Y$ are discriminators for X and Y, respectively. The loss of mapping cycle is denoted as red dashed lines.

This cycle-consistent loss is measured as:

$$E_{cyc}(\mathbb{G}, \mathbb{F}) = \|\mathbb{F}(\mathbb{G}(x)) - x\|_1 + \|\mathbb{G}(\mathbb{F}(y)) - y\|_1 . \tag{5.1}$$

The proposed model also includes two discriminators $\mathbb{D}_X$ and $\mathbb{D}_Y$ in order to distinguish between the transformed samples $\mathbb{G}(x), \mathbb{F}(y)$ and corresponding real samples y, x. Specifically, $\mathbb{D}_Y$ aims to differentiate the transformed sample $\mathbb{G}(x)$ from the real sample y and $\mathbb{D}_X$ aims to differentiate $\mathbb{F}(y)$ from x. The objective function can be expressed as:

$$
\begin{aligned}
E_{gan}(\mathbb{G}, \mathbb{D}_Y) &= E_{y \sim p_Y}[\log \mathbb{D}_Y(y)] + E_{x \sim p_X}[\log(1 - \mathbb{D}_Y(\mathbb{G}(x)))] \\
E_{gan}(\mathbb{F}, \mathbb{D}_X) &= E_{x \sim p_X}[\log \mathbb{D}_X(x)] + E_{y \sim p_Y}[\log(1 - \mathbb{D}_X(\mathbb{F}(y)))].
\end{aligned}
\tag{5.2}
$$

The full energy (Equation 5.3) is a summation of Equation 5.1 and Equation 5.2, with λ (set to 10 in the experiments) controlling the weight for the cycle-consistent loss:

$$
E(\mathbb{G}, \mathbb{F}, \mathbb{D}_X, \mathbb{D}_Y) = E_{gan}(\mathbb{G}, \mathbb{D}_Y) + E_{gan}(\mathbb{F}, \mathbb{D}_X) + \lambda E_{cyc}(\mathbb{G}, \mathbb{F}). \tag{5.3}
$$

The learned generators $\mathbb{G}$ and $\mathbb{F}$ are used for the mapping between the source expression and the target expression. Given a target expression v^g from the generator, a 3D face can be created using Equation 4.1.

5.2.2 Architecture

Two generators and two discriminators were developed for the CycleGAN based expression mapping model, as presented in Figure 5.3. A generator is a fully connected neural network (Figure 5.4) containing an input layer, 3 ReLU hidden layers, and a sigmoid output layer. Each hidden layer has 100 units, while both the input and the output layers have 46 units corresponding to the expression vector. The sigmoid function in the output layer helps to regularize each element of the target expression vector to reside in the valid range $[0, 1]$. A discriminator has a similar structure except that the output layer has only one unit producing a probability p. This probability p indicates the chance that the input comes from real data samples (i.e., 1-p chance is from a generator). For instance, 1.0 means the 100 percent chance that the input comes from real data samples, while 0.0 means the 100 percent chance that the input comes from a generator. Weights are initialized with a normal distribution $\mathcal{N}(0, 0.01)$.

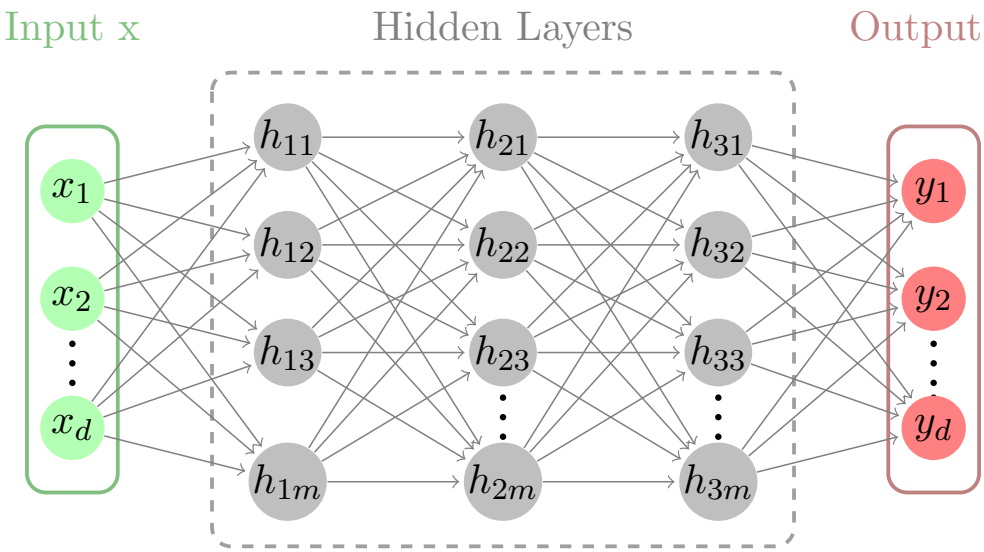


Figure 5.4: The generator $\mathbb{G}$ maps a source expression x to the target y through a deep neural network. The network consists of input and output layers with dimension $d = 46$, and 3 hidden layers with dimension $m = 100$.

5.2.3 Training

The Surrey Audio-Visual Expressed Emotion (SAVEE) dataset [61] was used for model training. The dataset contains video clips recorded from 4 male actors with multiple expressions (neutral, anger, disgust, fear, happiness, sadness, and surprise), uttering 120 sentences in English. Each video clip was recorded at 60 FPS, resulting in 110K images in total. The 3D performance reconstruction method described in Section 4 was run on each video clip. The results with inaccurate appearance were manually removed, leading to around 12K blendshape weights vectors for each expression category. The neutral was fixed as the source domain and the above CycleGAN-based expression mapping models were trained to map neutral to other expressions, using the Adam solver [81] with a learning rate of 0.0002 and momentum of 0.5. Each model was trained for 200 epochs with a batch size of 1. The networks were implemented in Tensorflow [1] and took around 8 hours for training on an nVidia Geforce GTX 1060 GPU. For any expression mapping that does not involve the neutral expression, two successive mappings were concatenated using neutral as the relay node. As shown in Figure 5.5, an expression mapping *sadness* $\rightarrow$ *anger* is translated as *sadness* $\rightarrow$ *neutral* $\rightarrow$ *anger*. The model training is offline done only once.

Figure 5.5: A mapping from sadness to anger can be translated as two consecutive mappings: sadness $\rightarrow$ neutral and neutral $\rightarrow$ anger.

5.3 Expression Transformation

Now the trained CycleGAN model is used for mapping blendshape weights w to the target expression w^*. For each frame in the input video, the 46 elements of the transformed expression vector w^g in the target expression domain Y are independently predicted by feeding the estimated weights w in the source domain X to the trained generator $\mathbb{G} : X \rightarrow Y$. The resulting facial deformation sequence presents the performance of the input video but with the target expression (Figure 5.6).

5.3.1 Window-based Smoothing

Simply forward feeding w to the trained generator $\mathbb{G}$, however, cannot guarantee to obtain a dynamically coherent, lip-synced sequence. The reason is that the training data are blendshape weight vectors from unpaired images in two domains. Therefore, smooth transitions of an input blendshape deformation sequence from the source domain cannot guarantee smoothness when mapped to the target domain. Meanwhile, the unpaired sampling mechanism cannot preserve lip-sync, since the randomly sampled images do not necessarily have the same utterance in training video clips. Figure 5.7 shows a failure case: the mouth in the source expression (Figure 5.7a) turns to be more opened after CycleGAN mapping (Figure 5.7b). To address the above issue, the following quadratic

Figure 5.6: Expression transformation from neutral to happiness by the proposed cycle-consistent expression mapping model. A higher confidence value from the discriminator indicates that the expression mapping model has more confidence in the transformed result of this particular frame, and vice versa. Note that after this step, the mouth shapes in the transformed expression may not match those in the source expression.

energy function is proposed to be minimized to solve the target expression w^* per frame:

$$\begin{aligned} \arg\min_{w^*} & \left\| w^* - \frac{\sum_{j=-k}^{k} c_j w_{g,j}}{\sum_{j=-k}^{k} c_j} \right\|_2^2 + \alpha \left\| \bar{\mathrm{B}} w^* - m \right\|_2^2 \\ \text{s.t.} \quad & 0 \leq w_i^* \leq 1, i = 1, ..., 46 \end{aligned} \tag{5.4}$$

In Equation 5.4, the first term encourages the optimal w^* to be close to the mapped expression w_g after convolution operation by a window of size $2k + 1$. j is the index of a frame within the window containing a small number of neighbors around the current frame, and c_j, and $w_{g,j}$ represent the weight and generated expression vector from the generator $\mathbb{G}$ by feeding the source expression w for the j^{th} frame, respectively. To balance the contribution of each $w_{g,j}$ within the window, the weight c_j is set to be the confidence value $\mathbb{D}(w_{g,j})$ by feeding $w_{g,j}$ to the discriminator $\mathbb{D}$. Recall that a higher value of $\mathbb{D}(y)$ indicates it is more likely y is sampled from real data rather than from a generator. The sliding window effectively helps to smooth out jitters in the synthesized target animation. k is set to one in the experiments; hence the resulting sequence has one frame delay from the source video stream. The second term in Equation 5.4 is the lip-correction term, detailed in the follow-up Section 5.3.2.

5.3.2 Lip Correction

The lip motion is synchronized with the source audio. The basic idea is to constrain the lip region with the tracked lip motion while maximally preserving the target expression characteristics. Specifically, a lip correction term (i.e., the second term) is integrated into the energy function Equation 5.4, which measures the 3D distance of the selected key lip vertices between the tracked and transformed face models. In Equation 5.4, $\bar{\mathrm{B}}$ is a matrix consisting of the corresponding rows of blendshape B for the key lip vertices, and m is a vector containing the 3D positions of the key lip vertices in the source expression. As shown in Figure 5.7, the key lip vertices after correction (Figure 5.7c) are moved toward their counterpart in Figure 5.7a. α is a weight to balance the two terms and is set to 1000 in the experiments.

(a) Source:Neutral (b) Target:Happiness (c) Corrected Lip

Figure 5.7: The lip motion is enforced to match the source audio by minimizing the 3D distance of key lip vertices between before (green dots) and after (red dots) transformation.

As shown in Figure 5.7, the key lip vertices (or called control points) are pre-defined as the four vertices bounding the upper and lower lips, respectively. The main reason why only the four key vertices in the middle of the lips are chosen, instead of more vertices on the lips, is as follows: for certain target expressions (e.g., happiness), the activation of certain mouth-relevant blendshape bases are required. Therefore, if more lip control points are imposed as a strong constraint for Equation 5.4 to ensure the transformed lip shape being as close as possible to the original one, the resulting target facial expression may be less desired. In the experiments, it is found that the selection of the four key vertices is a good trade-off to balance the overall expression realism and lip-sync. Figure 5.8 also illustrates the weight changes of a specific blendshape basis (primarily relevant to mouth movement) by different modules. As shown in this figure, a naive per-frame CycleGAN (cg) generates incoherent dynamics compared to the source animation. Per-frame CycleGAN with lip correction (lip+cg) creates synchronous but noisy movements, and finally, the proposed window-based smoothed CycleGAN with lip correction (lip+smooth) creates dynamically consistent and smooth mouth movements.

Although these pre-defined vertices only guide the opening and closing of the mouth, the proposed method is not limited to generating such simple mouth animation. The method can retain

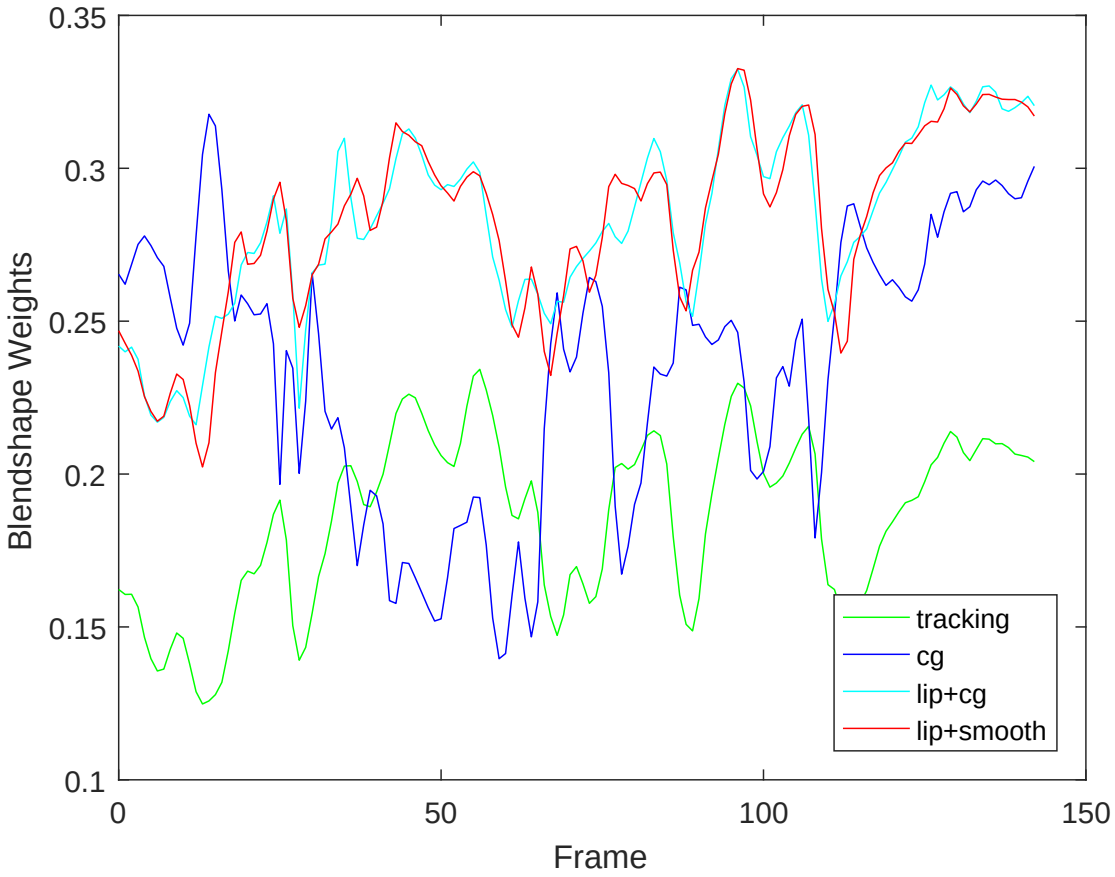


Figure 5.8: Weight curves of a specific blendshape basis (related to mouth movement) directly tracked from the source video (green, *tracking*), by the CycleGAN-based expression transformation alone (blue, *cg*), by the CycleGAN-based expression transformation + lip correction (cyan, *lip+cg*), and by the CycleGAN-based expression transformation + lip correction + smoothing (red, *lip+smooth*).

complex mouth deformations tracked from the source video when necessary. The main reason is that the learned CycleGAN-based expression mapping model mainly changes the weights of the blendshape bases related to the target expression. For example, the learned neutral-to-anger model would change eyebrows and frown but not much of the mouth. In such a case, the lip correction term is very small (the opening of the mouth is barely changed), and w^* converges at the smoothed w_g. Hence, the mouth deformation will be retained.

5.3.3 Composition

To synthesize a photo-realistic frame with the target expression, the face model is finally re-rendered with the target expression w^*, together with the head pose, camera parameters, lighting, and albedo texture estimated from the source frame. As mentioned previously, the mouth shape may be changed after transformation, so the mouth interior of the source frame needs to be accordingly warped to fill in the mouth region of the re-rendered face. A sparse set of vertices around the lip contour are pre-defined in the 3D model space, as well as their triangulation in the projected 2D image space to cover the mouth region. The mouth region of the source frame is then warped to the new projected positions of the lip contour vertices. To the end, the re-rendered face, the warped mouth, the untouched eyes region, and the background from the source frame are put onto different layers, and blended together using the Laplacian pyramid blending algorithm [20] (Figure 5.9).

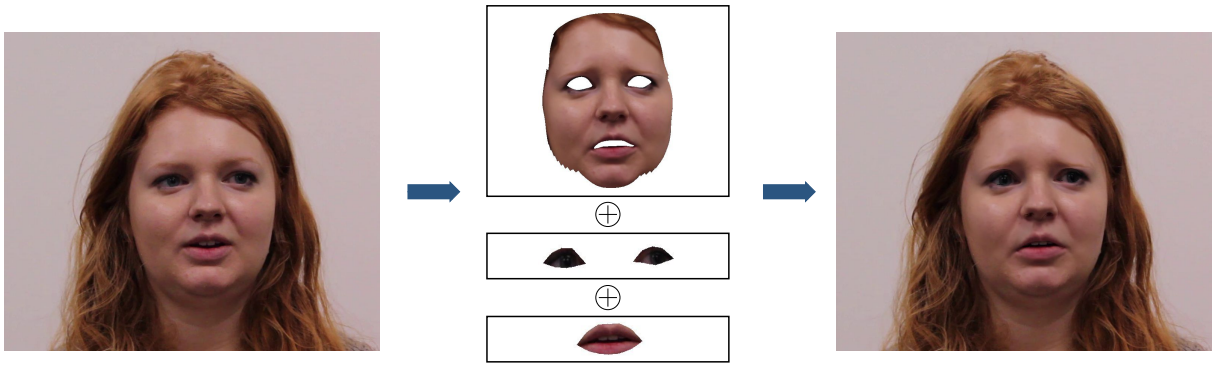


Figure 5.9: The re-rendered face with the target expression is blended back to the original video along with the warped mouth and eye regions. Note that the composite face (right) has a different expression from the input face (left).

5.4 Results

This section describes the system implementation, runtime statistics, experimental results, and comparisons with prior related works.

5.4.1 Implementation

Live facial performance of several volunteers was captured using a commodity Microsoft LifeCam HD 5000 running at 30Hz in 640×480 resolution. The method was also applied to some YouTube video clips at resolution 1280×720 to show its generality. The transformed results are shown in Figure 5.10 and 5.11.

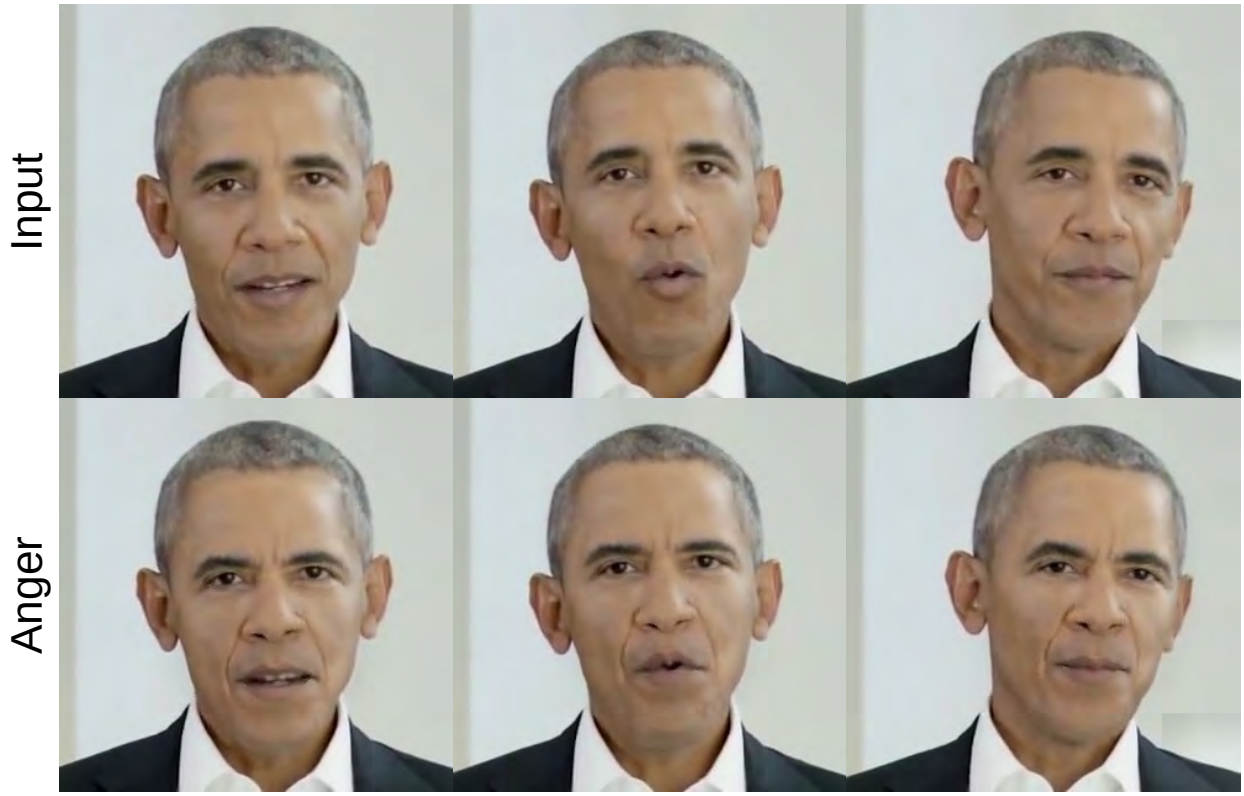


Figure 5.10: Results of the proposed system on selected Internet video clips. The input expressions are neutral.

The proposed system was implemented in C++ using Eigen [59] for linear algebra and OpenCV for image processing. The nonlinear optimization problem in face tracking is solved using the Trust Region strategy with box constraints for the expression parameters. The block coordinate descent algorithm is used for a faster convergence: the pose and identity parameters usually converge in 5 iterations, and the expression parameters terminate in 10 iterations. The CycleGAN inference

Figure 5.11: More results of the proposed system on selected Internet video clips. The input expressions are neutral.

is implemented in Tensorflow running on GPU. The constrained quadratic programming in transformation is solved by the interior point method. Similar to Face2face [140], composition is run in fragment shaders on GPU with the hardware-generated mipmaps for building image pyramids. Runtime statistics of the method is shown in Table 3. The system runs on a desktop computer with two Intel Xeon E5620 CPUs @2.4 GHz and nVidia Geforce GTX 1060 GPU.

Table 3: Runtime statistics for video clips with three different resolutions. From top to bottom: 640×480, 1280×720 and 1920×1080. The CPU and GPU computations run in parallel.

CPU		GPU		FPS
Tracking	Transform	CycleGAN	Composite	
24.89ms	7.62ms	1.79ms	1.52ms	30.76Hz
26.29ms	7.99ms	1.76ms	2.94ms	29.17Hz
29.21ms	7.46ms	1.75ms	5.31ms	27.27Hz

The effectiveness of the approach was qualitatively evaluated on real-time captured facial performance, on publicly available face datasets, and on Internet video clips. Figures 1.4, 5.10, 5.11, 5.12, 5.13, and 5.14 show some results by the approach to transform one expression to another, with various skin colors, ethnicities, and ages.

(a) Input (b) Linear Mapping + Lip Correction (c) CycleGAN [167]

(d) Yang et al. [161] (e) CycleGAN-based Expression Mapping (f) CycleGAN-based Expression Mapping + Lip Correction

Figure 5.12: Comparisons among (b) the linear mapping + lip correction method, (c) the original image-based CycleGAN model [167], (d) Yang et al. [161], and the proposed CycleGAN-based expression mapping model without (e) and with (f) lip correction and smoothing.

5.4.2 Comparisons

Since we cannot find any previous approaches that are specifically designed to achieve the *same* goal as ours, to the best of our knowledge, we compared our approach with some previous approaches that aim to achieve *similar* goals, including linear mapping, Yang et al. [161], the original CycleGAN [167], and ground truth, described below.

Comparison with linear mapping + lip correction The proposed approach as compared with the straightforward linear mapping method. Given an input blendshape coefficients vector x from the source expression domain, the target blendshape vector y can be obtained from the following linear function:

$$y = x - \bar{X} + \bar{Y}, \tag{5.5}$$

where $\bar{X}$ and $\bar{Y}$ are the average blendshape coefficients vectors for the source and the target expressions, respectively. $\bar{X}$ and $\bar{Y}$ were computed from the same dataset that was used for CycleGAN training and kept fixed in the experiments. If the above Equation 5.5 is directly used, obviously the resulting lip movements could be out-of-sync with audio. Therefore, the lip correction module proposed in this chapter was added onto the naive linear mapping method. Figure 5.12b shows a result of linear mapping + lip correction.

Compared to the proposed method (Figure 5.12f), the linear mapping + lip correction could produce acceptable visual results but with less computation/training time and less implementation effort. However, it has been observed that when computing the average delta blendshape vector $(\bar{Y} - \bar{X})$ from a large enough dataset, asymmetric expressions may neutralize each other, resulting in balanced delta values for those blendshape bases controlling the left half face and the right half face. This means that if the input expression is symmetric, the generated expression will also be symmetric. By contrast, the proposed model provides more flexibility by learning the non-linear characteristic of the expression mapping problem. Figure 5.13 shows such an example: given a neutral input with mouth corners slightly down, the linear mapping method generates flat or slightly up mouth corners, which is less consistent with the relatively strong cheek raise on the

resulting face. This is because cheeks and mouth corners are controlled by separate blendshape bases which are uncorrelated in the linear mapping method. The proposed CycleGAN mapping overcomes this issue by learning the correlation between blendshape bases, and thus generates more holistically consistent target expressions.

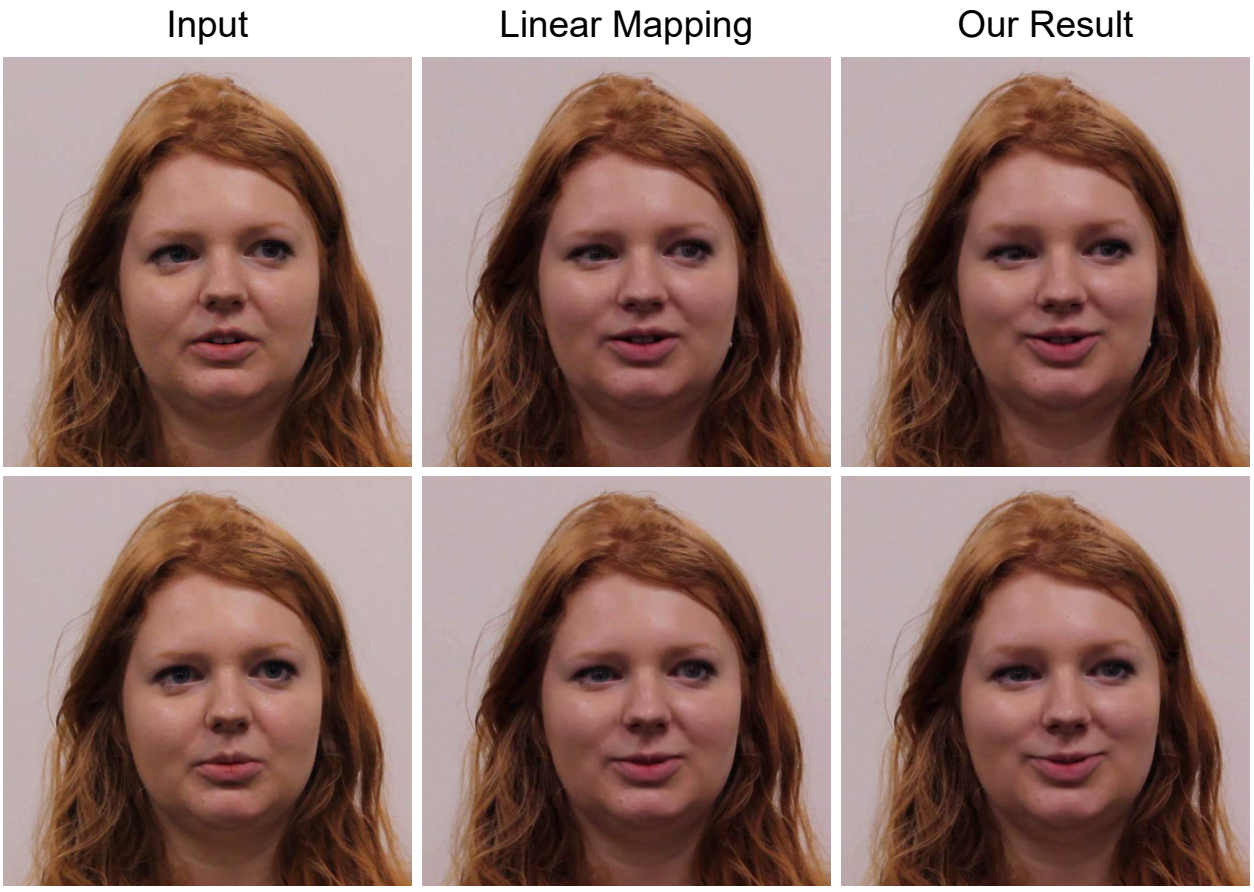


Figure 5.13: Visual comparisons between Linear Mapping + Lip Correction and the proposed method.

Comparison with Yang et al. [161] The proposed approach was compared with Yang et al. [161] on the same sequence "Talking Face Video" [33] that was used in the original work of [161]. Figure 5.12d and Figure 5.12f present the transformation to happiness from the same neutral input (Figure 5.12a) by Yang et al. [161] and the proposed approach, respectively. Note that the method by Yang et al. [161] actually replaces the neutral sub-sequence by manually choosing a happy

frame from the input source sequence as the center of gap and interpolates expression towards the boundaries of the gap. Therefore, their method cannot transform an expression to a *novel* expression that does *not* exist in the input source sequence. By contrast, the proposed method can transform the expression in the input source sequence to a novel expression for the whole sequence, not limited to the *existing* expressions in the input source sequence.

Comparison with CycleGAN [167] The proposed method was also compared with the original image-based CycleGAN model [167]. In this comparison, the latter was trained using the images from the same SAVEE dataset and the same weight parameters for the loss function. The trained network contains three stride-2 convolutions and nine residual blocks. From Figure 5.12c it could be seen that the original CycleGAN model trained on a relatively small amount of images cannot generalize well to unseen images. The generated image is not photo-realistic and the face appears blurry, since (i) the identity of the input image is far from the ones in the training data, and (ii) the background color is also changed because the background color in the training data is plain black. In addition, the image-based CycleGAN model fails to generate temporal smooth facial animation, which can be clearly seen in the result sequences. In contrast, Figure 5.12e shows that the proposed CycleGAN-based expression mapping model works more effectively on a small set of training data, generalizes well to various identities, and is more suitable for hallucinating face generation. Figure 5.12f shows the lip-corrected and smoothed result.

Comparison with Face2face and Bringing Portraits to Life The works of Face2face [140] and Bringing Portraits to Life [3] are two state-of-art methods for expression transfer. However, both of them require a driving source video clip from which the expression is transferred to a target video clip. Recording driving video clips with different expressions and with per frame lip-sync to the audio of the target video is practically infeasible, so a fair comparison between the proposed method and the two works cannot be performed. Nonetheless, it is noteworthy that the proposed method can be complementary to them in order to synthesize emotional expressions with little effort in modification.

Comparison with ground truth Volunteer subjects were captured speaking the same sentences with two different expressions. After that, the proposed method was applied to transform one expression to the other and then the transformed results were compared with their corresponding ground truth performance. As shown in Figure 5.14, the proposed method produced similar happiness and stronger sadness compared to the recorded ground truth of the subjects. It is noteworthy that it is impossible for the subjects to make perfectly audio-aligned performances with two different expressions; therefore, ground truth frames were manually chosen where the subjects are uttering the same phonemes as the comparison frames.

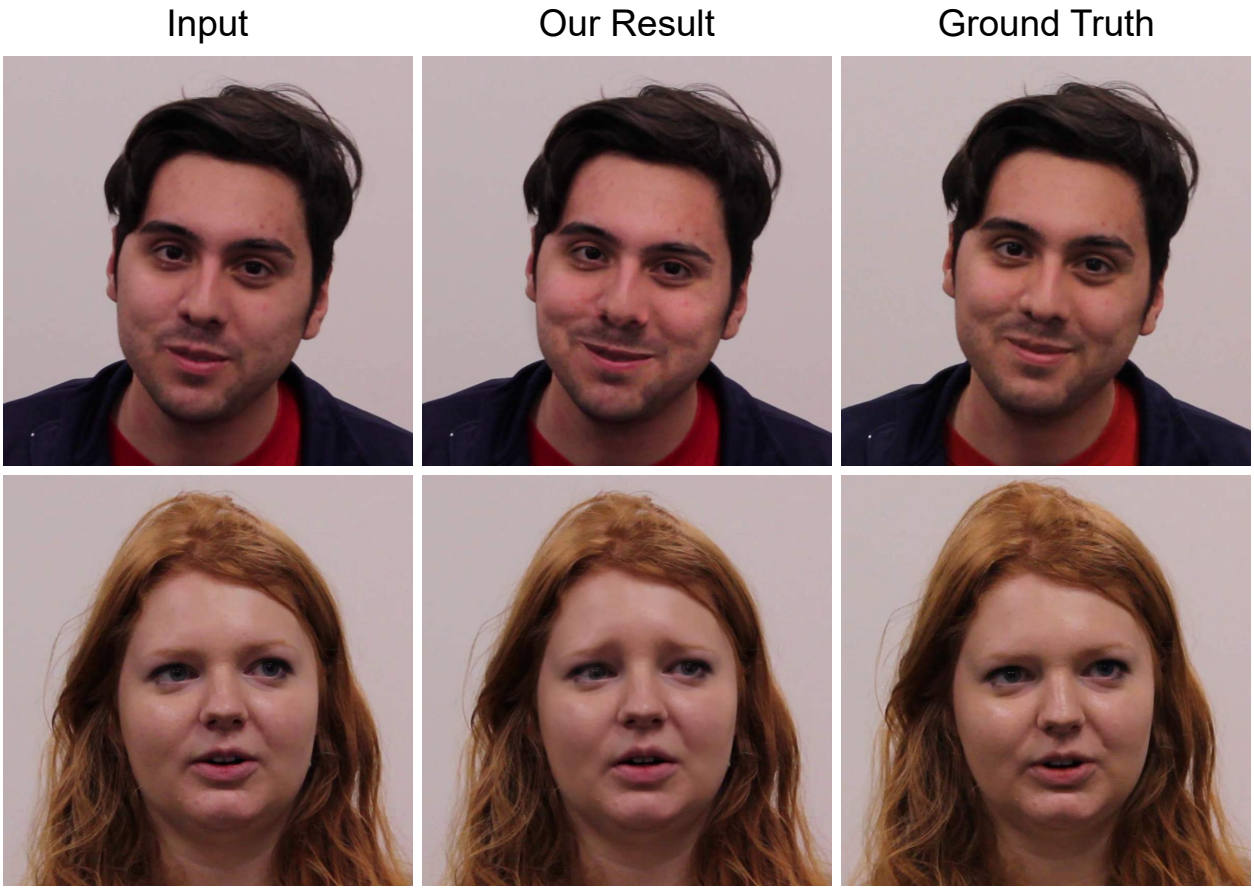
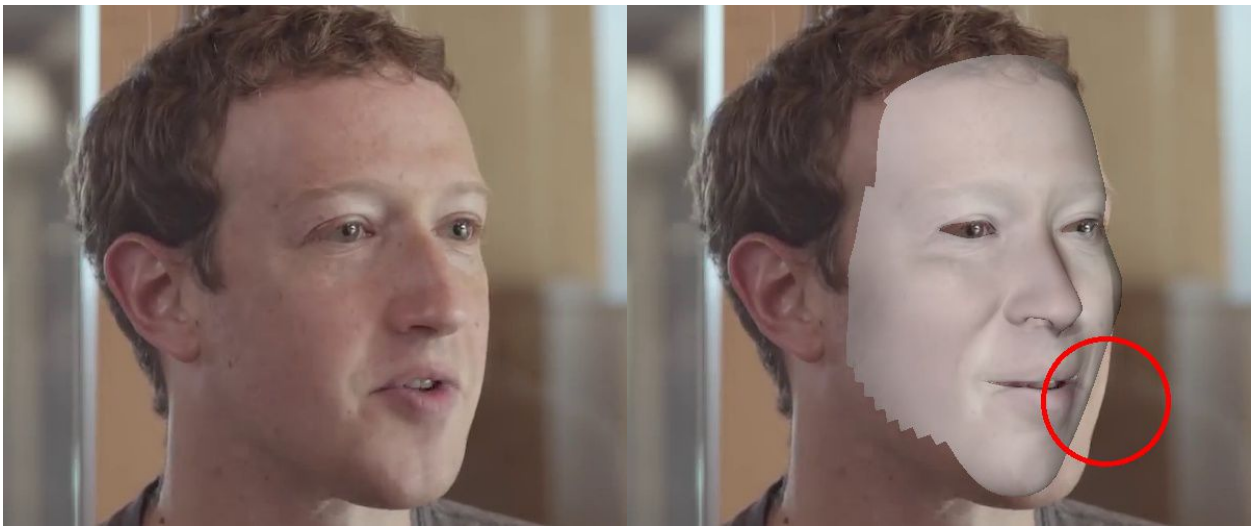


Figure 5.14: Comparison with ground truth. Both of the input expressions are neutral, and outputs are happiness and sadness from top to bottom. The ground truth frames are manually chosen where the subjects are uttering the same phonemes.

5.5 Limitations



(a) Input (b) Mesh Overlay

Figure 5.15: The proposed method cannot handle extreme head poses in input video. (b) shows a failure case in which the face contour is changed in the new expression (red circle).

Despite its demonstrated effectiveness, the current work has several limitations described below.

- The current approach cannot handle extreme head poses in the input video. As shown in Figure 5.15, the target happy expression in a side view produced a concave face contour compared to the source image.

- The current method lacks the capability to generate personalized expressions. In reality, different people may have their own different ways to express the same emotion even when speaking the same sentence multiple times. Moreover, people exhibit various facial dynamic details such as creases and wrinkles around the eyes or on the forehead. The above individual-specific expression characteristics are ignored in the current work.

- Similar to many prior related works, the current method relies on an accurate facial landmark tracker. Inaccurately detected landmarks can lead to smoothing effect at the texture estimation step (Section 4) as different pixels from input images could be mapped to the same UV position in texture space.

- The current system can produce a consistent sequence with the target expression. However, the transitions between different expressions are also important, which are not yet taken into consideration in the current method.

- Due to the forced synchronization of lip motion in real-time, the method cannot produce highly exaggerated expressions, such as surprise with mouth largely deformed compared to the source.

6 Face Swapping

In this chapter, a new, automatic, real-time method is proposed to swap the face in the target video by the face from a *single* source portrait image. The method takes a single source portrait image and a target video clip as inputs, and outputs a video-realistic video clip with the swapped source face. The approach consists of several steps as shown in Figure 6.1. The pipeline is briefly introduced in this section and the details of each step are described in the following sections.

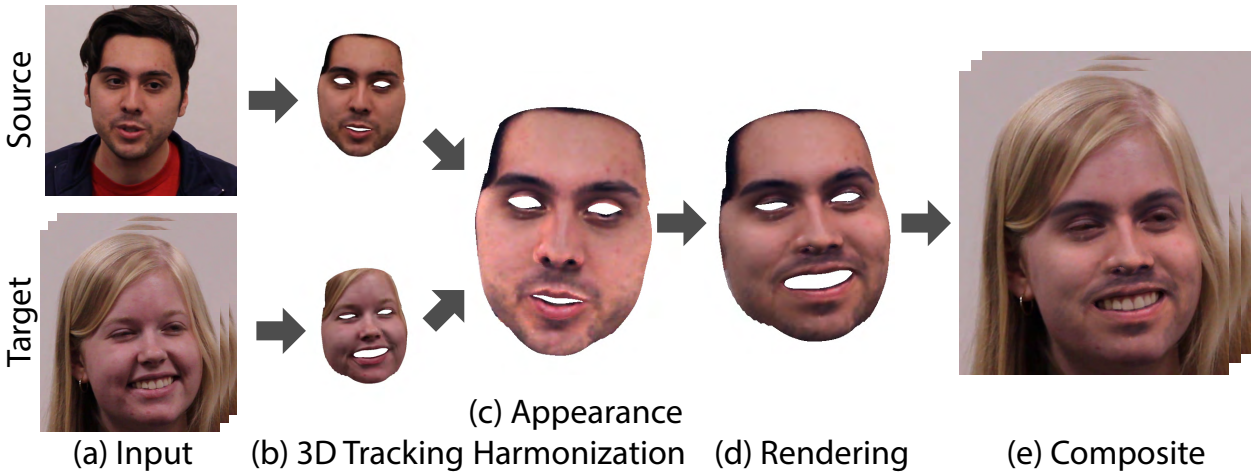


Figure 6.1: From the input source image and target video (a), the proposed system captures fine-scale 3D facial performance (b). The appearance of the source face is harmonized to match the target video (c). A novel face is rendered with the source identity, harmonized appearance under the target conditions (d). The rendered face is blended into the warped target frame (e).

The 3D face models, albedos, illuminations, and head poses are first reconstructed from the source image and each frame of the target video (Section 4). Each face model is further decomposed into a coarse model representing facial expressions (Section 4.1) and vertex displacements representing skin wrinkles (Section 4.3). Then, a novel source face mesh with target expression is synthesized and wrinkle dynamics are predicted (Section 6.1). The albedo of the source face is adapted to that of the target face through solving a Poisson equation (Section 6.2.1). The appearance is further harmonized by injecting matched noise to compensate for the different shot conditions (Section 6.2.2). A novel face image can be rendered by combining the novel mesh and harmonized appearance of the source face, with the illumination and head pose of the face in each

target frame. Finally, the target frame is warped according to the key points of the rendered face and blend them seamlessly (Section 6.3).

In sum, the contributions of this work include:

- an automatic real-time system to swap the face in a monocular RGB video by the face from a single portrait image;
- a method to predict wrinkle dynamics of the source face in target expressions; and
- an appearance harmonization method to video-realistically blend the synthesized face into the target video.

6.1 Mesh Swapping

The task of face swapping is defined as replacing the face in the target video with the face from the source image while retaining the facial performance of the target actor. The hair, body, and background in the target video are intact. Unlike recent deep learning based face swapping methods that learn face features in 2D images, the proposed method also takes care of 3D face mesh swapping. The face geometry is separated into large-scale expression and fine-scale wrinkles, which are transferred separately from the target to the source.

6.1.1 Coarse Mesh Swapping

The coarse mesh of the swapped face is represented as the combination of the identity of the source face and the expression of the target face. The mesh is generated by multiplying the FaceWarehouse core tensor by the identity parameter α^S of the source image and the expression parameter β^T of each frame in the target video:

$$\tilde{M}_t^S = C_r \times_2 \alpha^S \times_3 \beta_t^T.$$

Top right of Figure 6.2 shows the swapped coarse mesh for one frame of the target video. The whole mesh sequence is temporally smooth since α^S is constant and β^T changes smoothly in the

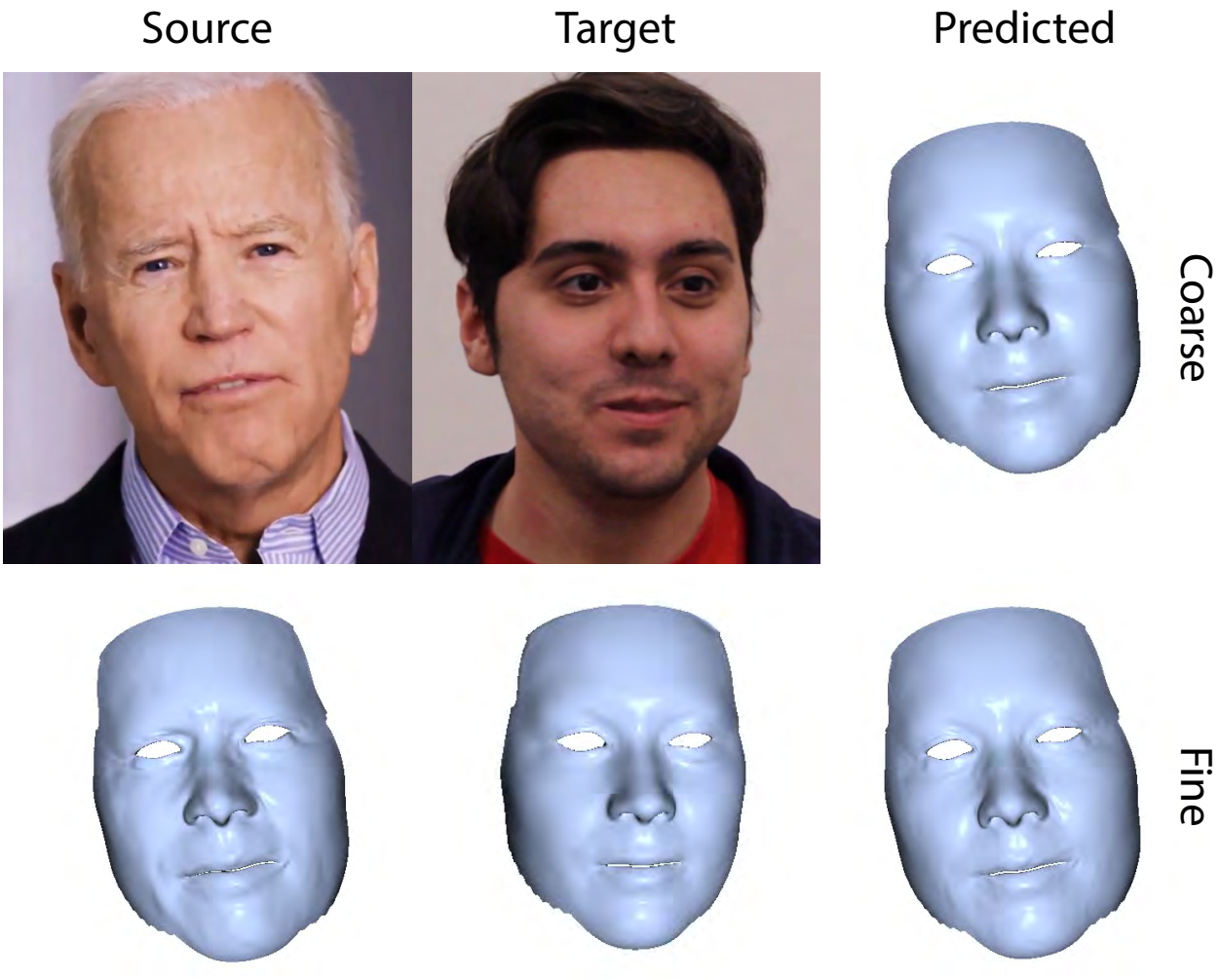


Figure 6.2: Face mesh swapping from source (left) to target (middle). Top right shows the coarse mesh of the source identity performing target expression, and bottom right is the mesh with wrinkle details. Image courtesy: Joe Biden (public domain, https://youtu.be/wUUXu93imi4).

expression PCA space.

6.1.2 Wrinkle Prediction

The coarse mesh is further augmented with wrinkle details. The objective is to predict the most plausible person-specific wrinkle motions of the source actor under the target expression. This is tackled using the Laplacian Coating Transfer technique [128]. For the source face, the Laplacian coordinates of the coarse mesh $\tilde{M}_0^S$ and the fine mesh M_0^S are are computed respectively for the

initial source face reconstructed from the image. The coating of the source mesh is defined as

$$\xi_0^S = L(M_0^S) - L(\tilde{M}_0^S),$$

where L is the Laplacian operator. Suppose that there exists a frame in the target video that has the same expression as of the source image. Then the corresponding coating of the target can be defined similarly as

$$\xi_0^T = L(M_0^T) - L(\tilde{M}_0^T).$$

For any frame t in the target video, the fine-scale motion

$$D_t^T = \xi_t^T - \xi_0^T$$

is transferred to the source mesh with a local rotation R at each vertex which is the rotation of the tangent space between the coarse meshes $\tilde{M}_t^S$ and $\tilde{M}_t^T$. The coating of the source face at frame t is then predicted as

$$\xi_t^S = \xi_0^S + R(D_t^T).$$

Finally, the fine-scale source mesh is reconstructed by solving the following inverse Laplacian:

$$M_t^S = L^{-1}(L(\tilde{M}_t^S) + \xi_t^S) = L^{-1}(L(\tilde{M}_t^S) + \xi_0^S + R(D_t^T)). \tag{6.1}$$

In practice, a frame with the most similar expression in the target video clip to that of the source image is found by measuring the squared Mahalanobis distance:

$$Sim(\beta_t^T, \beta_0^S) = (\beta_t^T - \beta_0^S)^T C_{exp}^{-1} (\beta_t^T - \beta_0^S), \tag{6.2}$$

where C_{exp} is the covariance matrix of expression constructed from the FaceWarehouse dataset. Then the mesh from the found frame is set to M_0^T and used to compute the coating transfer. For live applications, M_0^T is set as the mesh from the first frame and keeps updating whenever a closer expression is found using Equation 6.2. The bottom right of Figure 6.2 shows the result of coating transfer where the static aging wrinkles of the source actor are retained when performing the target expression.

6.2 Appearance Harmonization

Synthesis of a photo-realistic novel face video that combines the source face and the target background is challenging. The colors of the source face and the target face may be quite different, which leads to obvious seams along the face boundary. In addition, the source face image and the target video are usually shot under different lighting conditions. Even alpha blending or gradient-domain composition may produce unrealistic results. A viable solution is to apply image harmonization to the rendered face and the target frame [130]. However, this method is not suitable for the proposed real-time system, since it leads to the solving of huge sparse linear systems, and cannot be ported to GPU trivially. Experiments also show that it cannot guarantee the temporal consistency of the resulting sequence. Therefore, the facial appearance is harmonized in the texture space. The adapted albedo and the matched noise are computed for each vertex. The values are computed only for the first several frames of the video when the albedo of the target face is also being updated simultaneously. Then, they will remain fixed during the rendering of the remaining video frames so that the rendered facial appearance can be guaranteed to be temporally consistent.

6.2.1 Albedo Adaptation

An adapted albedo color at each vertex is computed for face rendering. The adapted albedo is supposed to have a similar global color to the target face while having the same local gradient to the source face. This is equivalent to the solving of a Poisson equation [111] in the texture space: the albedo values of the boundary vertices are set to the target face albedo, and the albedo gradients

of the inner face are set to the source face albedo gradients. The finite difference discretization of the Poisson equation yields the following discrete optimization problem:

$$\begin{aligned} E_{albedo} &= \sum_{i,j \in E} \left\| (\rho_i^S - \rho_j^S) - (\hat{\rho}_i - \hat{\rho}_j) \right\|_2^2, \\ \text{s.t.} \quad & \hat{\rho}_i = \rho_i^T, \quad \text{for } i \in \partial M \end{aligned} \tag{6.3}$$

where E denotes the edges of the face mesh, and ∂M denotes the boundary vertices of the face mesh. $\rho^S, \rho^T, and \hat{\rho}$ represent the albedo of the source face, the albedo of the target face, and the adapted albedo, respectively. Figure 6.3 (second column) shows the effect of the albedo adaptation.

6.2.2 Noise Matching

The rendered face using the adapted albedo color could be less noisy compared to the background of the target video, because of the imposed smoothing term for albedo estimation (see Section 4.3). A noise color is injected for each vertex to match the noise pattern in the target background. The noise γ in the source face and the target face are computed as the difference between the input image and the rendered face:

$$\gamma_i = I_i - l \cdot SH(n_i). \tag{6.4}$$

Then, histogram matching is applied to obtain the matched noise $\hat{\gamma}_i$:

$$\hat{\gamma}_i = \text{histmatch}(\gamma_i^S, \gamma_i^T), \tag{6.5}$$

where histmatch() denotes the transfer function that matches the histogram of the source noise γ_i^S with that of the target noise γ_i^T. Figure 6.3 (third column) shows the effect of noise matching.

6.3 Video Rendering and Composition

Now a new image of the source face can be rendered under the target condition and blended to the target background. The fine-scale model M_t^S of the novel face is computed using Equation 6.1.

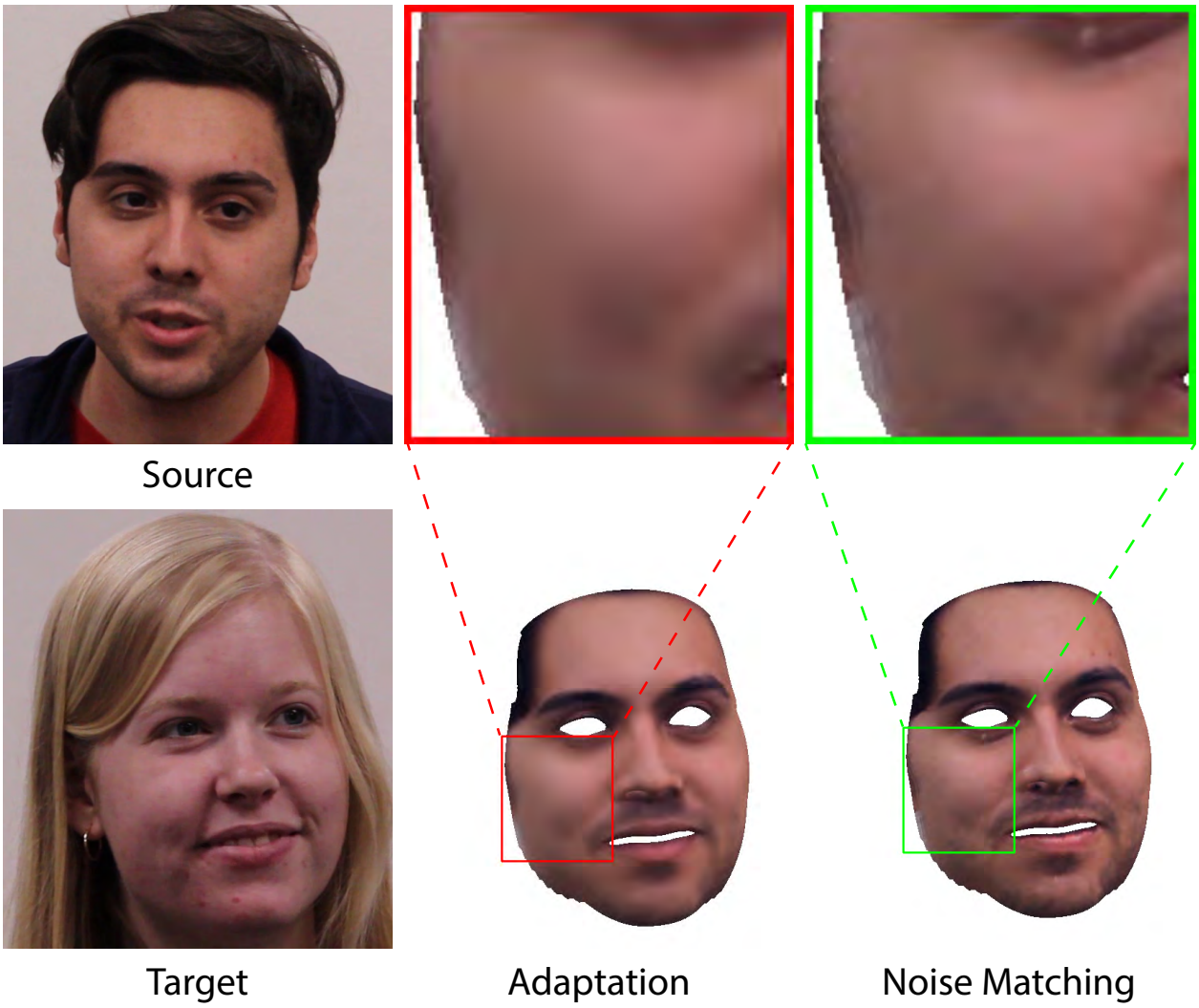


Figure 6.3: The appearance of the source face is harmonized to match that of the target face through albedo adaptation (middle) and noise matching (right).

The head pose is expected to be identical to that of the target face so that the rendered face is exactly overlaid on the target face region. The final vertex position $\hat{M}$ at frame t is computed as:

$$\hat{M} = R^T M^S + t^T, \tag{6.6}$$

where R^T, t^T represent the rotation and translation of the target face at frame t, respectively (see Section 4).

Next, the vertex normal $\hat{n}$ for the novel face model is computed and rendered with the harmonized appearance under the target illumination l^T:

$$\hat{I}_i = [l^T \cdot SH(\hat{n}_i)](\hat{\rho}_i + \hat{\gamma}_i). \tag{6.7}$$

A potential issue of face swapping is that the face shapes of the source and the target could significantly differ. For example, when an oblong face (Figure 6.4b) is swapped by a square face (Figure 6.4a), the eyes/ears in the background might be covered by the rendered face, which can lead to certain artifacts (Figure 6.4d). To alleviate the issue, the background image is warped according to the boundary and key points of the face. Specifically, the boundary and landmark vertices of the target face are projected onto the image plane and used to subdivide the image with Delaunay triangulation (Figure 6.4b). Since the face topology is fixed, the triangulation is applied only once and cached. Then, the same vertices of the novel face model $\hat{M}$ are projected to the target frame and used to warp the background (Figure 6.4c). The above operations can be efficiently done in an image quad drawing shader, where the vertex projections of the novel mesh are used as positions, and those of the target mesh are used as texture coordinates. Finally, the rendered face is blended into the warped background with GPU alpha blending. Again, an alpha map is pre-built in the texture space where boundary vertices have smaller alpha values. The final composition result is shown in Figure 6.4e.

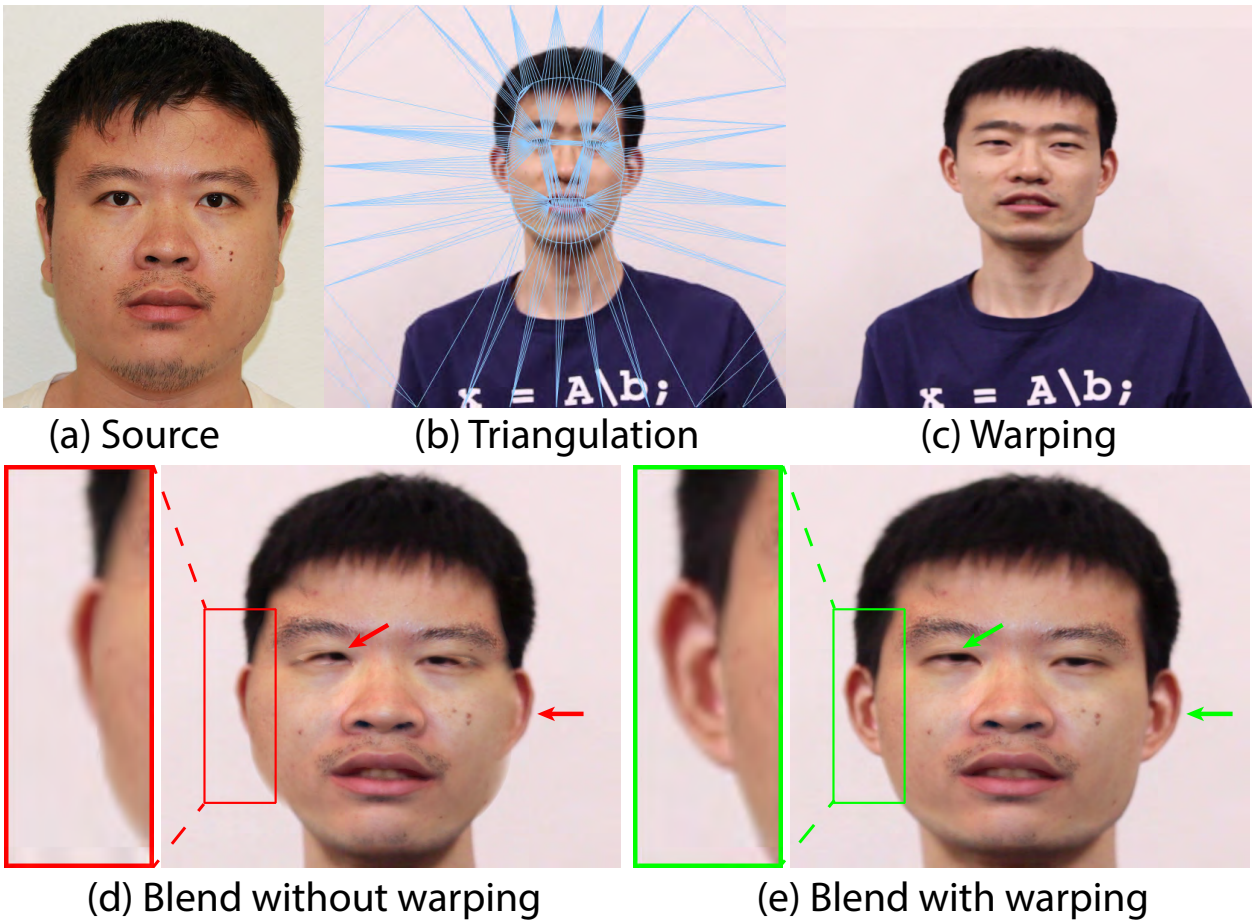


Figure 6.4: A target frame is triangulated using facial landmarks and boundary vertices (b). The frame is warped according to the positions of those vertices on the rendered face (c). (d) and (e) show the final blending results without and with warping.

6.4 Results

6.4.1 Implementation

The proposed approach was implemented in C++ and CUDA. The coarse mesh reconstruction stage runs on the CPU, and the other stages run on the GPU concurrently. The linear equation in illumination estimation is solved using the cuSolver. The non-linear least square problem in displacement recovery is solved by a Gauss-Newton solver in CUDA kernels. Each frame runs 10 Gauss-Newton steps, and the optimal step length is computed by a Preconditioned Conjugate Gradient (PCG) solver in 10 iterations. For albedo recovery and appearance harmonization, the

large sparse linear system is solved by running a similar PCG solver for 100 iterations and match the noise histogram of 256 bins in CUDA kernels.

Some results of the proposed method are demonstrated in Figure 6.5 and Figure 6.6. In Figure 6.5, the same source face is swapped into multiply different target face video clips. Even though the skin color is changed to be in harmony with the target face, the face shape, eyebrows, nose, and mustache of the source face remain in the result. In Figure 6.6, the target face is swapped by multiple different source faces. Note that face shapes and facial features such as eyebrows, nose, and nevus in the results are swapped by those of the source faces, while skin color, expression, and lighting are inherited from the target face. Figure 6.8 and 6.9 show more results for faces with various ethnicities, genders, and in different skin colors, head poses, and expressions.

All the experiments in this chapter ran on a desktop computer with Intel Core i7 CPU @3.7 GHz and nVidia Geforce GTX 2080Ti GPU. The input images and video clips were captured using a Logitech C922x Pro webcam. The source images were shot at 1080p resolution, and the target videos were shot at 720p resolution and 60 FPS. The coarse mesh reconstruction takes 1 ms CPU time; the mesh refinement, swapping, rendering, and composition take 8 ms CPU and 18 ms GPU time. The albedo estimation and adaptation take 3 ms in total and only run at the first several frames of the video. Overall, the system ran at 55 FPS on the experimental computer.

6.4.2 Evaluation

To quantitatively evaluate the proposed method, a self swapping experiment is present in Figure 6.7. The first frame of the video is taken as the source image and the faces in the remaining frames of the video are swapped. The heat map of the photometric error in Figure 6.7 visualizes the difference between the ground truth frame and the synthesized frame in RGB color space. Most regions have a good approximation with an error lower than 3 in the range $[0, 255]$. Large errors occur at high-frequency areas, such as eyebrows or at the background which are caused by background warping.

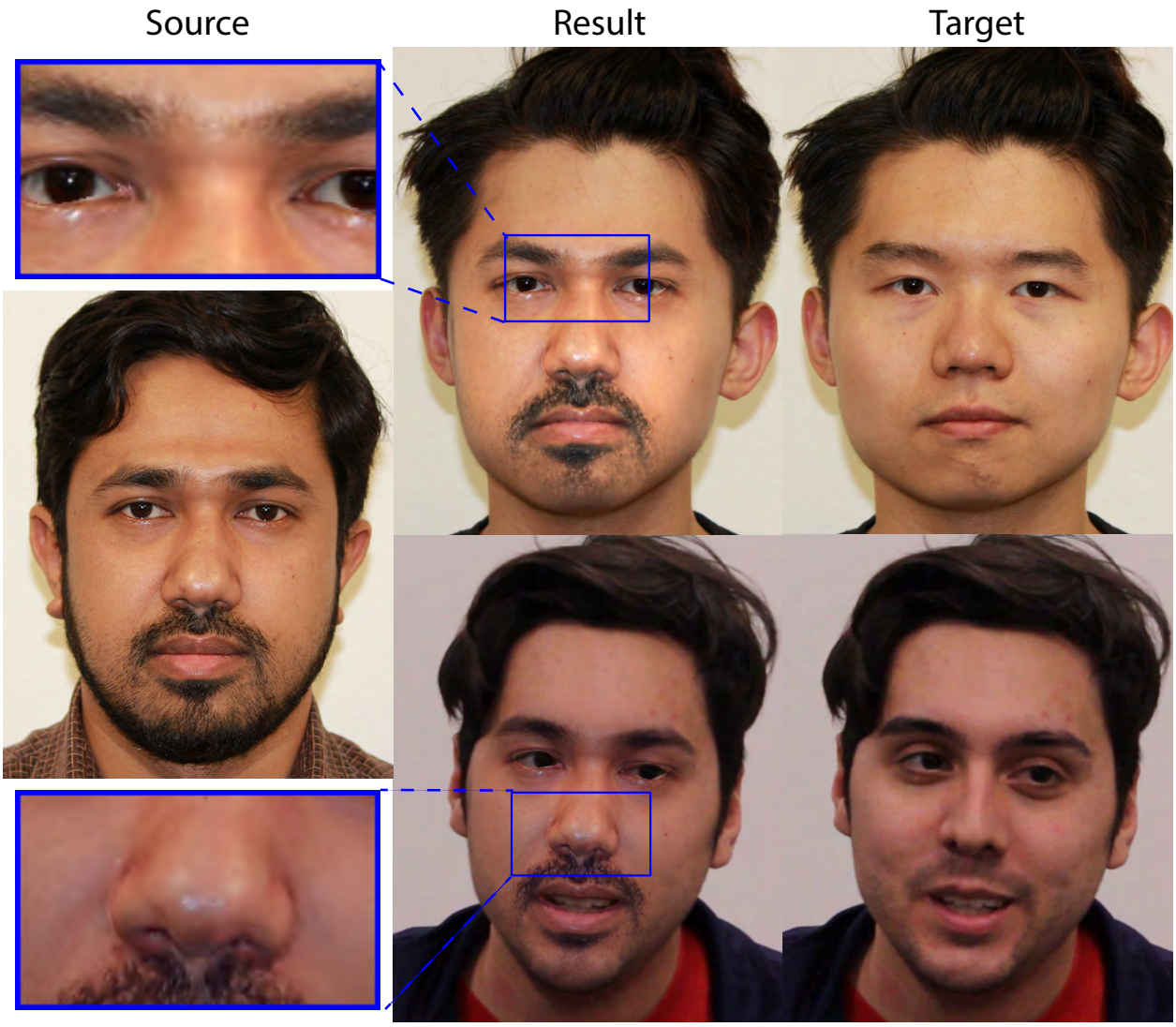


Figure 6.5: Face swapping results (second column) from the same source face (first column) to multiple target faces (third column). Rectangles show some examples of facial features (eyebrows, nose shape, mustache, etc.) are transferred from the source, while the expressions are extracted from the targets (eyebrow raising, mouth opening).

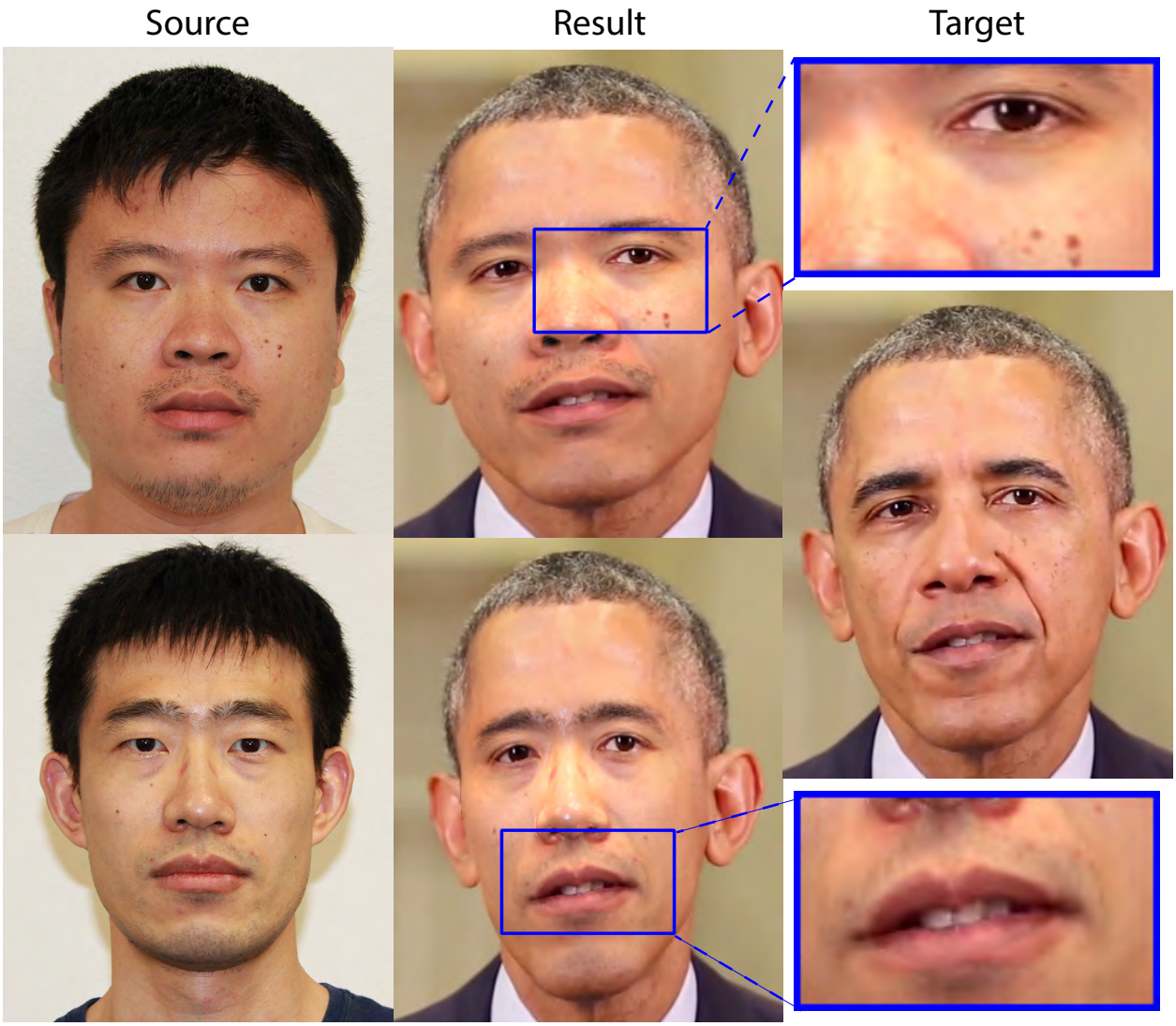


Figure 6.6: Face swapping results (second column) from multiple source images (first column) to the same target video (third column). Note face shapes are altered after swapping. Rectangles show some examples of facial features (lip shape, acne, freckles, etc.) that are transferred in high resolution. Image courtesy: The White House (public domain, via YouTube https://goo.gl/mAb6iP).

Figure 6.7: The photometric error of a self-swapping in which the first frame is used as the source image. Image courtesy: The White House (public domain, via YouTube https://goo.gl/mAb6iP).

6.4.3 Comparisons

Previous 2D image-based methods [11, 75] automatically select the most similar face from a large image database for face swapping. However, they cannot handle videos since different faces can be selected when the expression is changed. Thus, the results cannot be temporally consistent. Garrido et al. [48] use videos as input for both the source and the target faces and select the most similar frame from the source video. If the source video does not contain enough variations, artifacts may occur in the results. In the following, the proposed method is mainly compared with state-of-the-art model-based methods and learning-based methods.

Comparison with model-based methods The proposed method was compared with a 3D model-based method FaceSwap [85] in Figure 6.8. *FaceSwap* also fits a 3D face model for both the source and the target frames by minimizing the difference between the projected shape and the localized landmarks. However, the 3D model is only used as a proxy to warp the source face image to the target; the lighting difference between the two images are not taken into account. The rendered face may look unrealistic when the lighting conditions of the source and the target frames highly differ. This method does not take identity consistency into consideration either. Therefore, the synthesized facial motion is not temporally smooth.

Comparison with learning-based methods The proposed method was also compared with the state-of-the-art deep learning methods *Deepfakes* [36] and Nirkin et al. [109] in Figure 6.9.

Figure 6.8: Compared with Faceswap [85], the proposed method does not unexpectedly change the eye gaze of the target. It is also more temporal coherent as a constant face identity and texture are kept.

Deepfakes can swap faces between two subjects. However, this method needs to train one encoder and two decoders using two large face image datasets of the two specific subjects. At runtime, the source image is passed to the encoder and then decoded by the decoder of the target subject. For comparison, two high-resolution video clips (1080p) were recorded for the two subjects. The faces were cropped in a 512×512 region, yielding about 32K training images for each subject. The model was trained at batch size 64 for 100K iterations. It is obvious that the results by *Deepfakes* are much more blurred than ours, and the method of *Deepfakes* cannot change the face shape. In contrast, the proposed method produces high-resolution results with clear details and correct face shapes. Moreover, the proposed is more applicable and friendly to users than *Deepfakes*, since it does not require the collecting of large-size training data of the specific faces nor expensive and

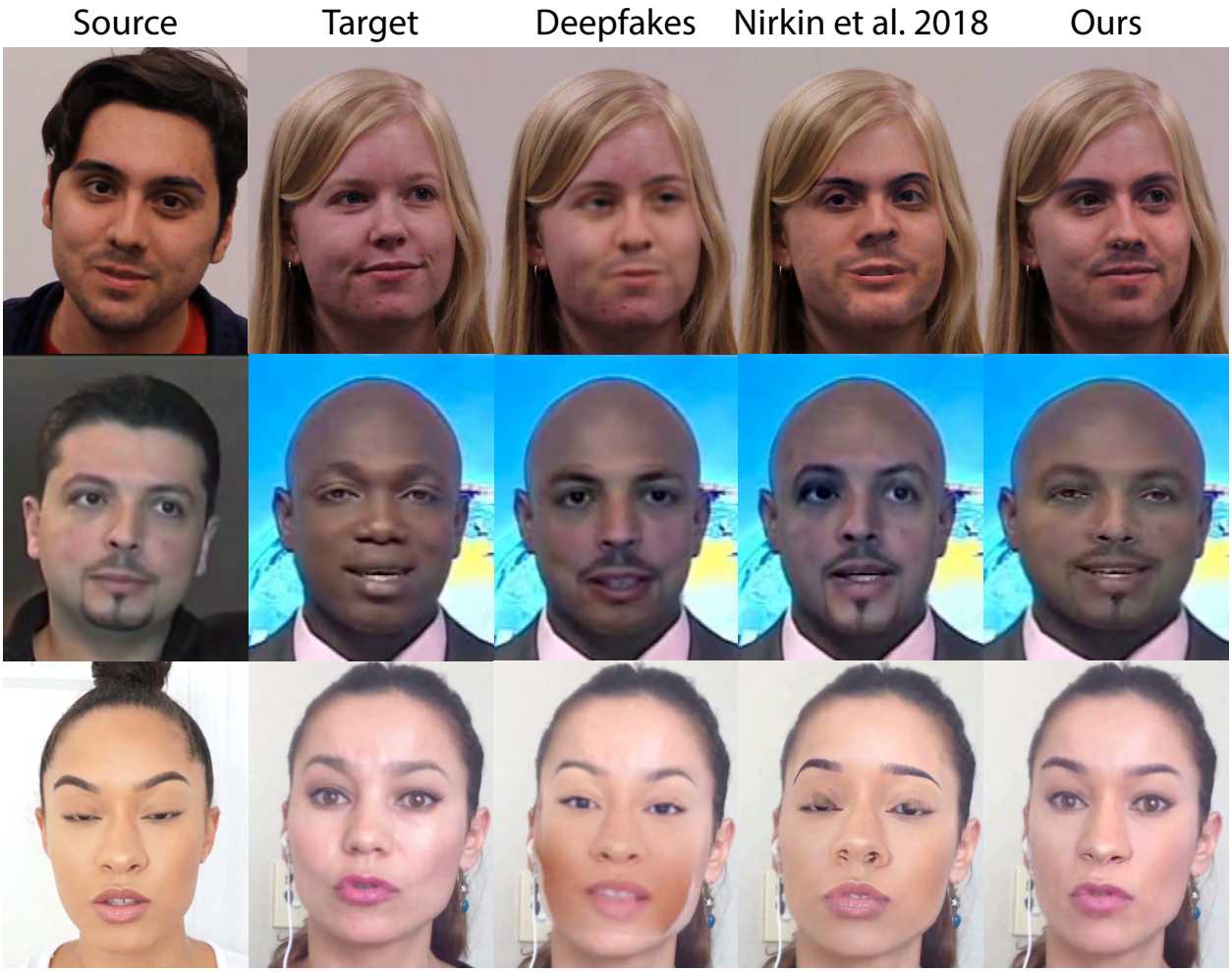


Figure 6.9: Compared to Deepfakes [36] and Nirkin et al. [109], the proposed method can change the face shape, and the result contains many more facial details without the need for any training data.

time-consuming model training. Nirkin et al. [109] is another learning-based method that trained a deep segmentation network to guide the face swapping area. The code provided by the authors was run on the same input. As shown in Figure 6.9, their result is less video-realistic compared with *Deepfakes* and the proposed method. Their result sequence is also much jumpier than the other methods.

6.5 Limitations

Although the proposed method has many advantages over previous methods, it comes at a price. Since only a single image is used to capture the source face, it is inherently ambiguous to attribute a facial feature to the identity or expression. For example, if the source image shows oblique eyebrows, two possibilities exist: the subject has oblique eyebrows in the neutral expression, or the subject has flat eyebrows but in the oblique eyebrows expression. It is very difficult to distinguish between the two possibilities, even for humans. If the method reconstructs the face with oblique eyebrows in the neutral expression, the oblique eyebrows identity feature will persist for the whole rendered video. Otherwise, the oblique eyebrows expression will be replaced by the target expression such that the rendered video will not have this facial feature anymore. To overcome this limitation, a viable solution is to use a collection of images of the source face, from which a more accurate identity can be reconstructed [118].

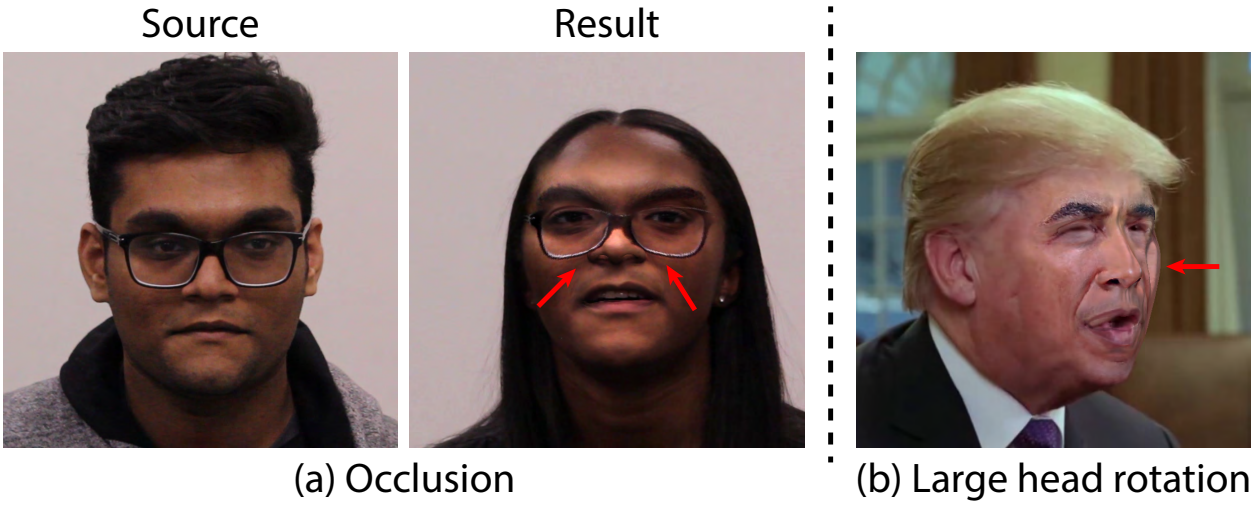


Figure 6.10: The proposed method cannot effectively handle occlusions (a) or large head rotations (b).

Furthermore, the proposed method does not swap the eyes and inner mouth of the target video. People have diverse iris and pupils in size and color. Eyes are crucial for humans to recognize faces. With the eyes untouched, the result quality is highly affected. I would like to extend the method to swap eyes and mouth regions in future work. Facial occlusions also cannot be effectively

handled by the current method. Figure 6.10(a) shows the glasses frame is warped after swapping due to expression change. In addition, large head poses such as side-view may produce artifacts, as shown in Figure 6.10(b), since the facial landmark detection algorithm is not accurate for extreme poses. Artifacts may also occur when the source and the target faces have different styles of face boundaries, e.g., one with hair at forehead and one without. After solving the Poisson equation, the hair color will bleed into the inner face such that the adapted albedo seems problematic, which hampers the realism of the synthesized face. This could be tackled by employing a face segmentation method; only the inner face without hair occlusion is swapped. In future work, I also would like to explore the possibility of hair swapping. Hair colors and styles are important features to recognize people. I believe the result quality will be highly improved if the hair can be swapped simultaneously along with the faces.

7 Conclusion and Outlook

In this dissertation, a real-time, automatic, geometry-based method is presented for capturing the fine-scale facial performance from monocular RGB video. The method reconstructs large-scale head poses and expressions as well as fine-scale wrinkles, lighting, and albedo in parallel and in real-time. A novel hierarchical reconstruction method is also introduced to robustly solve the shape-from-shading surface refinements of human faces. It is demonstrated that the proposed approach can produce results close to off-line methods and better than previous real-time methods.

Based on the reconstruction, a complete pipeline is presented to photo-realistically transform the facial expression for monocular video in real-time. A CycleGAN model is trained in blendshape weights space using fewer data and training time. A real-time smooth transformation algorithm is presented to retain lip-sync with the source audio. Moreover, an automatic, real-time method is presented to swap the facial identity and appearance in RGB videos from a single portrait image, while preserving the facial performance in terms of poses, expressions, and wrinkle motions. The method runs fully automatic, without requiring pre-collected large-size training data of both the source and the target faces. It shows that both face manipulation methods can create video-realistic results for faces of various skin colors, genders, ages, and expressions.

For future work, I would like to enhance the coarse face tracking algorithm by combining the dense correspondences of RGB values in the image space by introducing a face texture prior. However, this will bring extra computation cost on the GPU and breaks the current CPU-GPU parallel structure, since the photometric error will be transferred back to CPU for tracking improvements. Therefore, a more sophisticated solver is required for this purpose. I am also interested in updating the albedo texture with unseen pixels in novel head poses or inferring a complete albedo texture using deep learning techniques at initialization. Besides, the reconstruction of hair, eyes, mouth is also crucial in capturing the whole facial performance. The tracking of highly deformable hair and tongue, and subtle eye gaze movement is even more challenging.

I also plan to improve the expression transformation method to automatically identify the exhibited facial expression from the source video [120], instead of identified by users in the current work. The current system generates a target video with untouched audio and head motion. However, for certain target expressions such as anger, keeping the original audio and/or head motion from the input source video may seriously affect the perception of the transformed emotion, because the emotions conveyed by the facial expression, head motion, or audio channels could be substantially different or even conflict with each other. I plan to incorporate emotional transformation for speech (e.g., [57]) and head motion (e.g., [40]) into the current framework in the future.

For future applications of 3D face tracking, I believe virtual reality (VR) and augmented reality (AR) will provide a broad stage. VR has already been used in gaming and training for risky tasks, such as surgery, military, aviation, and safety-critical construction fields. In the future, visual interaction between users will be highly demanding. For example, people would like to see the faces of other people when playing multi-player online games or having a meeting using VR. A lot of use cases of VR will come out relying on realistic avatars as more and more tasks would be done remotely by people wearing VR gear. Teachers giving lectures remotely in VR would like to see real-time facial and body reactions from the students. Multiple people cooperatively working on a task, e.g., building a 3D scene in a modeling system supporting VR, would like to see other people's actions. All of these would require (i) building a 3D avatar of the user and (ii) transferring the performance of the user to the avatar in real-time. The creation of a 3D avatar has been explored in many previous works [2, 103, 67, 26, 65]. Apart from facial appearance, the hair [119, 164, 95], lips [51], teeth [152], and eye gaze [147] are also crucial in a holistic avatar system. For performance transferring, the most challenging problem is that human faces are under strong occlusion when wearing a headset. The upper region covered by the headset could be captured by cameras inside the HMD [141] or inferred by strain sensors attached to the foam liner of the HMD [89]. The lower region could be captured by cameras rigidly attached to the HMD or distant static cameras. Then, the two parts are combined to transfer the whole expression to the avatar. However, in some scenarios, an avatar representation may not be sufficient for communication. As

in teleconferencing, people may prefer seeing photo-realistic faces [138] without an HMD while in the VR environment. Appearance inference will be involved to estimate the appearance occluded by the HMD. Since a large part of the face is unseen and because of possible cast shadow caused by the HMD, illumination estimation is very challenging in such cases. Current methods are also limited in the quality of face tracking in VR. Only large-scale blendshape parameters can be captured or inferred. Wrinkle-level details are usually ignored due to the occlusion. Another constrain of face tracking in VR is the computation capability and energy capacity of the HMD. A modern VR headset, such as Oculus Quset[4] has the all-in-one design which has all the hardware embedded in a small headset. Solving optimization problems in real-time may impact the rendering and other tasks such as hand tracking, and quickly drain the battery. Machine learning-based methods with carefully designed models could be a faster and more energy-efficient choice over all-in-one devices. Similarly, AR glasses and smartphones equipped with depth sensors or multi-cameras will also bring up emerging applications using face capturing. I believe the techniques in face capture and manipulation will go one step further in quality and efficiency on those platforms.

[4]https://www.oculus.com/quest/

Bibliography

[1] Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G. S., Davis, A., Dean, J., Devin, M., et al. Tensorflow: Large-scale machine learning on heterogeneous distributed systems. *arXiv preprint arXiv:1603.04467* (2016).

[2] Alexander, O., Rogers, M., Lambeth, W., Chiang, M., and Debevec, P. The digital Emily project: Photoreal facial modeling and animation. In *ACM SIGGRAPH 2009 Courses* (New Orleans, Louisiana, 2009), SIGGRAPH '09, Association for Computing Machinery.

[3] Averbuch-Elor, H., Cohen-Or, D., Kopf, J., and Cohen, M. F. Bringing portraits to life. *ACM Trans. Graph. 36*, 6 (Nov. 2017), 196:1–196:13.

[4] Bagautdinov, T., Wu, C., Saragih, J., Fua, P., and Sheikh, Y. Modeling facial geometry using compositional vaes. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (2018), pp. 3877–3886.

[5] Basri, R., and Jacobs, D. W. Lambertian reflectance and linear subspaces. *IEEE Transactions on Pattern Analysis and Machine Intelligence 25*, 2 (2003), 218–233.

[6] Beeler, T., Bickel, B., Beardsley, P., Sumner, B., and Gross, M. High-quality single-shot capture of facial geometry. *ACM Trans. Graph. 29*, 4 (July 2010), 40:1–40:9.

[7] Beeler, T., Bradley, D., Zimmer, H., and Gross, M. Improved reconstruction of deforming surfaces by cancelling ambient occlusion. In *European Conference on Computer Vision* (Berlin, Heidelberg, 2012), Springer, Springer Berlin Heidelberg, pp. 30–43.

[8] Beeler, T., Hahn, F., Bradley, D., Bickel, B., Beardsley, P., Gotsman, C., Sumner, R. W., and Gross, M. High-quality passive facial performance capture using anchor frames. *ACM Trans. Graph. 30*, 4 (July 2011), 75:1–75:10.

[9] Bermano, A. H., Bradley, D., Beeler, T., Zund, F., Nowrouzezahrai, D., Baran, I., Sorkine-Hornung, O., Pfister, H., Sumner, R. W., Bickel, B., and Gross, M. Facial performance enhancement using dynamic shape space analysis. *ACM Trans. Graph. 33*, 2 (Apr. 2014), 13:1–13:12.

[10] Bickel, B., Botsch, M., Angst, R., Matusik, W., Otaduy, M., Pfister, H., and Gross, M. Multi-scale capture of facial geometry and motion. In *ACM SIGGRAPH 2007 Papers* (San Diego, California, 2007), SIGGRAPH '07, Association for Computing Machinery, p. 33–es.

[11] Bitouk, D., Kumar, N., Dhillon, S., Belhumeur, P., and Nayar, S. K. Face swapping: Automatically replacing faces in photographs. *ACM Trans. Graph. 27*, 3 (Aug. 2008), 39:1–39:8.

[12] Blanz, V., Basso, C., Poggio, T., and Vetter, T. Reanimating faces in images and video. *Computer Graphics Forum 22*, 3 (2003), 641–650.

[13] Blanz, V., Scherbaum, K., Vetter, T., and Seidel, H.-P. Exchanging faces in images. *Computer Graphics Forum 23*, 3 (2004), 669–676.

[14] Blanz, V., and Vetter, T. A morphable model for the synthesis of 3d faces. In *Proceedings of the 26th Annual Conference on Computer Graphics and Interactive Techniques* (New York, NY, USA, 1999), SIGGRAPH '99, ACM, pp. 187–194.

[15] Borshukov, G., Piponi, D., Larsen, O., Lewis, J. P., and Tempelaar-Lietz, C. Universal capture - image-based facial animation for "the matrix reloaded". In *ACM SIGGRAPH 2005 Courses* (Los Angeles, California, 2005), SIGGRAPH '05, Association for Computing Machinery, p. 16–es.

[16] Bouaziz, S., Wang, Y., and Pauly, M. Online modeling for realtime facial animation. *ACM Trans. Graph. 32*, 4 (July 2013), 40:1–40:10.

[17] Bradley, D., Heidrich, W., Popa, T., and Sheffer, A. High resolution passive facial performance capture. *ACM Trans. Graph. 29*, 4 (July 2010), 41:1–41:10.

[18] Bradski, G., and Kaehler, A. *Learning OpenCV: Computer Vision in C++ with the OpenCV Library*, 2nd ed. O'Reilly Media, Inc., 2013.

[19] Brand, M. Voice puppetry. In *Proceedings of the 26th Annual Conference on Computer Graphics and Interactive Techniques* (1999), SIGGRAPH '99, ACM Press/Addison-Wesley Publishing Co., pp. 21–28.

[20] Burt, P., and Adelson, E. The Laplacian pyramid as a compact image code. *IEEE Transactions on Communications 31*, 4 (1983), 532–540.

[21] Cao, C., Bradley, D., Zhou, K., and Beeler, T. Real-time high-fidelity facial performance capture. *ACM Trans. Graph. 34*, 4 (July 2015), 46:1–46:9.

[22] Cao, C., Chai, M., Woodford, O., and Luo, L. Stabilized real-time face tracking via a learned dynamic rigidity prior. *ACM Trans. Graph. 37*, 6 (Dec. 2018).

[23] Cao, C., Hou, Q., and Zhou, K. Displaced dynamic expression regression for real-time facial tracking and animation. *ACM Trans. Graph. 33*, 4 (July 2014), 43:1–43:10.

[24] Cao, C., Weng, Y., Lin, S., and Zhou, K. 3d shape regression for real-time facial animation. *ACM Trans. Graph. 32*, 4 (July 2013), 41:1–41:10.

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

[26] Cao, C., Wu, H., Weng, Y., Shao, T., and Zhou, K. Real-time facial animation with image-based dynamic avatars. *ACM Trans. Graph. 35*, 4 (July 2016).

[27] Cao, Y., Tien, W. C., Faloutsos, P., and Pighin, F. Expressive speech-driven facial animation. *ACM Trans. Graph. 24*, 4 (Oct. 2005), 1283–1302.

[28] Chai, J.-x., Xiao, J., and Hodgins, J. Vision-based control of 3d facial animation. In *Proceedings of the 2003 ACM SIGGRAPH/Eurographics Symposium on Computer Animation* (San Diego, California, 2003), SCA '03, Eurographics Association, p. 193–206.

[29] Chen, Y.-L., Wu, H.-T., Shi, F., Tong, X., and Chai, J. Accurate and robust 3d facial capture using a single rgbd camera. In *2013 IEEE International Conference on Computer Vision* (Dec 2013), pp. 3615–3622.

[30] Choi, Y., Choi, M., Kim, M., Ha, J.-W., Kim, S., and Choo, J. Stargan: Unified generative adversarial networks for multi-domain image-to-image translation. *arXiv preprint arXiv:1711.09020* (2017).

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

[32] Cong, M., Bhat, K. S., and Fedkiw, R. Art-directed muscle simulation for high-end facial animation. In *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation* (Zurich, Switzerland, 2016), SCA '16, Eurographics Association, pp. 119–127.

[33] Cootes, T. Talking face video. http://www-prima.inrialpes.fr/FGnet/data/01-TalkingFace/talking_face.html, 2017.

[34] Cootes, T. F., Edwards, G. J., and Taylor, C. J. Active appearance models. *IEEE Transactions on Pattern Analysis and Machine Intelligence 23*, 6 (2001), 681–685.

[35] Dale, K., Sunkavalli, K., Johnson, M. K., Vlasic, D., Matusik, W., and Pfister, H. Video face replacement. *ACM Trans. Graph. 30*, 6 (Dec. 2011), 130:1–130:10.

[36] Deepfakes. https://github.com/deepfakes/faceswap, may 2019.

[37] Deng, Z., Chiang, P.-Y., Fox, P., and Neumann, U. Animating blendshape faces by cross-mapping motion capture data. In *Proceedings of the 2006 Symposium on Interactive 3D Graphics and Games* (Redwood City, California, 2006), I3D '06, ACM, pp. 43–48.

[38] Deng, Z., and Neumann, U. Expressive speech animation synthesis with phoneme-level controls. *Computer Graphics Forum 27*, 8 (2008), 2096–2113.

[39] Deng, Z., Neumann, U., Lewis, J. P., Kim, T.-Y., Bulut, M., and Narayanan, S. Expressive facial animation synthesis by learning speech coarticulation and expression spaces. *IEEE Transactions on Visualization and Computer Graphics 12*, 6 (2006), 1523–1534.

[40] Ding, Y., Shi, L., and Deng, Z. Perceptual enhancement of emotional mocap head motion: An experimental study. In *2017 Seventh International Conference on Affective Computing and Intelligent Interaction (ACII)* (2017), pp. 242–247.

[41] Ekman, P., and Friesen, W. Facial action coding system: a technique for the measurement of facial movement. *Palo Alto: Consulting Psychologists* (1978).

[42] Ezzat, T., Geiger, G., and Poggio, T. Trainable videorealistic speech animation. *ACM Trans. Graph. 21*, 3 (July 2002), 388–398.

[43] Fried, O., Shechtman, E., Goldman, D. B., and Finkelstein, A. Perspective-aware manipulation of portrait photos. *ACM Trans. Graph. 35*, 4 (July 2016).

[44] Fried, O., Tewari, A., Zollhöfer, M., Finkelstein, A., Shechtman, E., Goldman, D. B., Genova, K., Jin, Z., Theobalt, C., and Agrawala, M. Text-based editing of talking-head video. *ACM Trans. Graph. 38*, 4 (July 2019).

[45] Fyffe, G., Jones, A., Alexander, O., Ichikari, R., and Debevec, P. Driving high-resolution facial scans with video performance capture. *ACM Trans. Graph. 34*, 1 (Dec. 2014), 8:1–8:14.

[46] Ganin, Y., Kononenko, D., Sungatullina, D., and Lempitsky, V. Deepwarp: Photorealistic image resynthesis for gaze manipulation. In *Computer Vision – ECCV 2016* (Cham, 2016), Springer International Publishing, pp. 311–326.

[47] Garrido, P., Valgaert, L., Wu, C., and Theobalt, C. Reconstructing detailed dynamic face geometry from monocular video. *ACM Trans. Graph. 32*, 6 (Nov. 2013), 158:1–158:10.

[48] Garrido, P., Valgaerts, L., Rehmsen, O., Thormaehlen, T., Perez, P., and Theobalt, C. Automatic face reenactment. In *Proceedings of 2014 IEEE Conference on Computer Vision and Pattern Recognition* (June 2014), pp. 4217–4224.

[49] Garrido, P., Valgaerts, L., Sarmadi, H., Steiner, I., Varanasi, K., Pérez, P., and Theobalt, C. Vdub: Modifying face video of actors for plausible visual alignment to a dubbed audio track. *Computer Graphics Forum 34*, 2 (2015), 193–204.

[50] Garrido, P., Zollhöfer, M., Casas, D., Valgaerts, L., Varanasi, K., Pérez, P., and Theobalt, C. Reconstruction of personalized 3d face rigs from monocular video. *ACM Trans. Graph. 35*, 3 (May 2016), 28:1–28:15.

[51] Garrido, P., Zollhöfer, M., Wu, C., Bradley, D., Pérez, P., Beeler, T., and Theobalt, C. Corrective 3d reconstruction of lips from monocular video. *ACM Trans. Graph. 35*, 6 (Nov. 2016).

[52] Gatys, L. A., Ecker, A. S., and Bethge, M. Image style transfer using convolutional neural networks. In *Proceedings of 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (June 2016), pp. 2414–2423.

[53] Geng, J., Shao, T., Zheng, Y., Weng, Y., and Zhou, K. Warp-guided gans for single-photo facial animation. *ACM Trans. Graph. 37*, 6 (Dec. 2018).

[54] Gerig, T., Morel-Forster, A., Blumer, C., Egger, B., Luthi, M., Schoenborn, S., and Vetter, T. Morphable face models - an open framework. In *2018 13th IEEE International Conference on Automatic Face Gesture Recognition (FG 2018)* (May 2018), pp. 75–82.

[55] Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. Generative adversarial nets. In *Advances in Neural Information Processing Systems 27* (2014), Curran Associates, Inc., pp. 2672–2680.

[56] Gotardo, P., Riviere, J., Bradley, D., Ghosh, A., and Beeler, T. Practical dynamic facial appearance modeling and acquisition. *ACM Trans. Graph. 37*, 6 (Dec. 2018), 232:1–232:13.

[57] Grinstein, E., Duong, N., Ozerov, A., and Perez, P. Audio style transfer. *arXiv preprint arXiv:1710.11385* (2017).

[58] Guarnera, D., Guarnera, G., Ghosh, A., Denk, C., and Glencross, M. Brdf representation and acquisition. *Computer Graphics Forum 35*, 2 (2016), 625–650.

[59] Guennebaud, G., Jacob, B., et al. Eigen v3. http://eigen.tuxfamily.org, 2010.

[60] Guo, Y., j. zhang, Cai, J., Jiang, B., and Zheng, J. CNN-based real-time dense face reconstruction with inverse-rendered photo-realistic face images. *IEEE Transactions on Pattern Analysis and Machine Intelligence 41*, 6 (2019), 1294–1307.

[61] Haq, S., and Jackson, P. J. Multimodal emotion recognition. *Machine Audition: Principles, Algorithms and Systems*, 17 (2010), 398–423.

[62] Hartley, R. I., and Zisserman, A. *Multiple View Geometry in Computer Vision*, second ed. Cambridge University Press, ISBN: 0521540518, 2004.

[63] Horn, B. K. Obtaining shape from shading information. *The psychology of computer vision* (1975), 115–155.

[64] Hsieh, P.-L., Ma, C., Yu, J., and Li, H. Unconstrained realtime facial performance capture. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (June 2015), pp. 1675–1683.

[65] Hu, L., Saito, S., Wei, L., Nagano, K., Seo, J., Fursund, J., Sadeghi, I., Sun, C., Chen, Y.-C., and Li, H. Avatar digitization from a single image for real-time rendering. *ACM Trans. Graph. 36*, 6 (Nov. 2017).

[66] Huang, H., Chai, J., Tong, X., and Wu, H.-T. Leveraging motion capture and 3d scanning for high-fidelity facial performance acquisition. *ACM Trans. Graph. 30*, 4 (July 2011), 74:1–74:10.

[67] Ichim, A. E., Bouaziz, S., and Pauly, M. Dynamic 3d avatar creation from hand-held video input. *ACM Trans. Graph. 34*, 4 (July 2015), 45:1–45:14.

[68] Ichim, A.-E., Kadleček, P., Kavan, L., and Pauly, M. Phace: Physics-based face modeling and animation. *ACM Trans. Graph. 36*, 4 (July 2017), 153:1–153:14.

[69] Ichim, A.-E., Kavan, L., Nimier-David, M., and Pauly, M. Building and animating user-specific volumetric face rigs. In *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation* (Zurich, Switzerland, 2016), SCA '16, Eurographics Association, pp. 107–117.

[70] Isola, P., Zhu, J.-Y., Zhou, T., and Efros, A. A. Image-to-image translation with conditional adversarial networks. *arXiv preprint arXiv:1611.07004* (2016).

[71] Jarosz, W. *Efficient Monte Carlo Methods for Light Transport in Scattering Media*. PhD thesis, UC San Diego, September 2008.

[72] Jing Xiao, Baker, S., Matthews, I., and Kanade, T. Real-time combined 2d+3d active appearance models. In *Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2004. CVPR 2004.* (2004), vol. 2, pp. II–II.

[73] Johnson, J., Alahi, A., and Fei-Fei, L. Perceptual losses for real-time style transfer and super-resolution. In *Proceedings of European Conference on Computer Vision* (2016), pp. 694–711.

[74] Karras, T., Aila, T., Laine, S., Herva, A., and Lehtinen, J. Audio-driven facial animation by joint end-to-end learning of pose and emotion. *ACM Trans. Graph. 36*, 4 (July 2017), 94:1–94:12.

[75] Kemelmacher-Shlizerman, I. Transfiguring portraits. *ACM Trans. Graph. 35*, 4 (July 2016), 94:1–94:8.

[76] Kemelmacher-Shlizerman, I., and Basri, R. 3d face reconstruction from a single image using a single reference face shape. *IEEE Transactions on Pattern Analysis and Machine Intelligence 33*, 2 (2011), 394–405.

[77] Kemelmacher-Shlizerman, I., Sankar, A., Shechtman, E., and Seitz, S. M. Being John Malkovich. In *Proceedings of the 11th European Conference on Computer Vision: Part I* (Heraklion, Crete, Greece, 2010), ECCV'10, Springer-Verlag, p. 341–353.

[78] Kemelmacher-Shlizerman, I., Shechtman, E., Garg, R., and Seitz, S. M. Exploring photobios. *ACM Trans. Graph. 30*, 4 (July 2011).

[79] Kim, H., Elgharib, M., Zollhöfer, M., Seidel, H.-P., Beeler, T., Richardt, C., and Theobalt, C. Neural style-preserving visual dubbing. *ACM Trans. Graph. 38*, 6 (Nov. 2019).

[80] Kim, H., Garrido, P., Tewari, A., Xu, W., Thies, J., Niessner, M., Pérez, P., Richardt, C., Zollhöfer, M., and Theobalt, C. Deep video portraits. *ACM Trans. Graph. 37*, 4 (July 2018).

[81] Kingma, D., and Ba, J. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).

[82] Klehm, O., Rousselle, F., Papas, M., Bradley, D., Hery, C., Bickel, B., Jarosz, W., and Beeler, T. Recent advances in facial appearance capture. *Computer Graphics Forum 34*, 2 (2015), 709–733.

[83] Koch, R. Dynamic 3-d scene analysis through synthesis feedback control. *IEEE Transactions on Pattern Analysis and Machine Intelligence 15*, 6 (1993), 556–568.

[84] Korshunova, I., Shi, W., Dambre, J., and Theis, L. Fast face-swap using convolutional neural networks. In *Proceedings of the IEEE International Conference on Computer Vision* (2017), pp. 3677–3685.

[85] Kowalski, M. Faceswap. https://github.com/MarekKowalski/FaceSwap, dec 2019.

[86] Kuster, C., Popa, T., Bazin, J.-C., Gotsman, C., and Gross, M. Gaze correction for home video conferencing. *ACM Trans. Graph. 31*, 6 (Nov. 2012), 174:1–174:6.

[87] Lee, Y., Terzopoulos, D., and Waters, K. Realistic modeling for facial animation. In *Proceedings of the 22nd Annual Conference on Computer Graphics and Interactive Techniques* (New York, NY, USA, 1995), SIGGRAPH '95, Association for Computing Machinery, p. 55–62.

[88] Lewis, J. P., Anjyo, K., Rhee, T., Zhang, M., Pighin, F. H., and Deng, Z. Practice and theory of blendshape facial models. *Eurographics (State of the Art Reports) 1*, 8 (2014).

[89] Li, H., Trutoiu, L., Olszewski, K., Wei, L., Trutna, T., Hsieh, P.-L., Nicholls, A., and Ma, C. Facial performance sensing head-mounted display. *ACM Trans. Graph. 34*, 4 (July 2015).

[90] Li, H., Yu, J., Ye, Y., and Bregler, C. Realtime facial animation with on-the-fly correctives. *ACM Trans. Graph. 32*, 4 (July 2013), 42:1–42:10.

[91] Li, K., Xu, F., Wang, J., Dai, Q., and Liu, Y. A data-driven approach for facial expression synthesis in video. In *Proceedings of 2012 IEEE Conference on Computer Vision and Pattern Recognition* (June 2012), IEEE, pp. 57–64.

[92] Li, M., Zuo, W., and Zhang, D. Deep identity-aware transfer of facial attributes. *arXiv preprint arXiv:1610.05586* (2016).

[93] Li, Q., and Deng, Z. Orthogonal-blendshape-based editing system for facial motion capture data. *IEEE Computer Graphics and Applications 28*, 6 (2008).

[94] Li, T., Bolkart, T., Black, M. J., Li, H., and Romero, J. Learning a model of facial shape and expression from 4d scans. *ACM Trans. Graph. 36*, 6 (Nov. 2017).

[95] Liang, S., Huang, X., Meng, X., Chen, K., Shapiro, L. G., and Kemelmacher-Shlizerman, I. Video to fully automatic 3d hair model. *ACM Trans. Graph. 37*, 6 (Dec. 2018).

[96] Liu, Y., Xu, F., Chai, J., Tong, X., Wang, L., and Huo, Q. Video-audio driven real-time facial animation. *ACM Trans. Graph. 34*, 6 (Oct. 2015).

[97] Liu, Z., Shan, Y., and Zhang, Z. Expressive expression mapping with ratio images. In *Proceedings of the 28th Annual Conference on Computer Graphics and Interactive Techniques* (New York, NY, USA, 2001), SIGGRAPH '01, Association for Computing Machinery, p. 271–276.

[98] Lombardi, S., Saragih, J., Simon, T., and Sheikh, Y. Deep appearance models for face rendering. *ACM Trans. Graph. 37*, 4 (July 2018), 68:1–68:13.

[99] Ma, L., and Deng, Z. Real-time facial expression transformation for monocular rgb video. *Computer Graphics Forum 38*, 1 (2019), 470–481.

[100] Ma, L., and Deng, Z. Real-time hierarchical facial performance capture. In *Proceeding of ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games (I3D)* (Montreal, QC, Canada, May 21–23 2019), ACM, p. 10 pages.

[101] Ma, L., and Deng, Z. Real-time face video swapping from a single portrait. In *Proceeding of ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games (I3D)* (San Francisco, CA, May 5–7 2020), ACM, p. 10 pages.

[102] Ma, W.-C., Jones, A., Chiang, J.-Y., Hawkins, T., Frederiksen, S., Peers, P., Vukovic, M., Ouhyoung, M., and Debevec, P. Facial performance synthesis using deformation-driven polynomial displacement maps. *ACM Trans. Graph. 27*, 5 (Dec. 2008), 121:1–121:10.

[103] Ma, W.-C. A., Rhee, T., and Yoshiyasu, Y. Making digital characters: Creation, deformation, and animation. In *SIGGRAPH Asia 2015 Courses* (Kobe, Japan, 2015), SA '15, Association for Computing Machinery.

[104] Ma, X., Le, B. H., and Deng, Z. Style learning and transferring for facial animation editing. In *Proceedings of the 2009 ACM SIGGRAPH/Eurographics Symposium on Computer Animation* (New Orleans, Louisiana, 2009), SCA '09, ACM, pp. 123–132.

[105] Malleson, C., Bazin, J., Wang, O., Bradley, D., Beeler, T., Hilton, A., and Sorkine-Hornung, A. Facedirector: Continuous control of facial performance in video. In *Proceedings of 2015 IEEE International Conference on Computer Vision (ICCV)* (Dec 2015), IEEE, pp. 3979–3987.

[106] Meka, A., Häne, C., Pandey, R., Zollhöfer, M., Fanello, S., Fyffe, G., Kowdle, A., Yu, X., Busch, J., Dourgarian, J., Denny, P., Bouaziz, S., Lincoln, P., Whalen, M., Harvey, G., Taylor, J., Izadi, S., Tagliasacchi, A., Debevec, P., Theobalt, C., Valentin, J., and Rhemann, C. Deep reflectance fields: High-quality facial reflectance field inference from color gradient illumination. *ACM Trans. Graph. 38*, 4 (July 2019).

[107] Müller, C. Spherical harmonics. *Lecture Notes in Mathematics 17* (1966), 40–44.

[108] Nagano, K., Seo, J., Xing, J., Wei, L., Li, Z., Saito, S., Agarwal, A., Fursund, J., and Li, H. Pagan: Real-time avatars using dynamic textures. *ACM Trans. Graph. 37*, 6 (Dec. 2018).

[109] Nirkin, Y., Masi, I., Tran, A. T., Hassner, T., and Medioni, G. On face segmentation, face swapping, and face perception. In *IEEE Conference on Automatic Face and Gesture Recognition* (2018).

[110] Orvalho, V., Bastos, P., Parke, F., Oliveira, B., and Alvarez, X. A Facial Rigging Survey. In *Eurographics 2012 - State of the Art Reports* (2012), M.-P. Cani and F. Ganovelli, Eds., The Eurographics Association.

[111] Pérez, P., Gangnet, M., and Blake, A. Poisson image editing. *ACM Trans. Graph. 22*, 3 (July 2003), 313–318.

[112] Pighin, F., Hecker, J., Lischinski, D., Szeliski, R., and Salesin, D. H. Synthesizing realistic facial expressions from photographs. In *Proceedings of the 25th Annual Conference on Computer Graphics and Interactive Techniques* (New York, NY, USA, 1998), SIGGRAPH '98, Association for Computing Machinery, p. 75–84.

[113] Ramamoorthi, R., and Hanrahan, P. An efficient representation for irradiance environment maps. In *Proceedings of the 28th Annual Conference on Computer Graphics and Interactive Techniques* (New York, NY, USA, 2001), SIGGRAPH '01, Association for Computing Machinery, p. 497–500.

[114] Ramamoorthi, R., and Hanrahan, P. A signal-processing framework for inverse rendering. In *Proceedings of the 28th Annual Conference on Computer Graphics and Interactive Techniques* (New York, NY, USA, 2001), SIGGRAPH '01, Association for Computing Machinery, p. 117–128.

[115] Ren, S., Cao, X., Wei, Y., and Sun, J. Face alignment at 3000 fps via regressing local binary features. In *2014 IEEE Conference on Computer Vision and Pattern Recognition* (2014), pp. 1685–1692.

[116] Rhee, T., Hwang, Y., Kim, J. D., and Kim, C. Real-time facial animation from live video tracking. In *Proceedings of the 2011 ACM SIGGRAPH/Eurographics Symposium on Computer Animation* (Vancouver, British Columbia, Canada, 2011), SCA '11, Association for Computing Machinery, p. 215–224.

[117] Ribera, R. B. i., Zell, E., Lewis, J. P., Noh, J., and Botsch, M. Facial retargeting with automatic range of motion alignment. *ACM Trans. Graph. 36*, 4 (July 2017).

[118] Roth, J., Tong, Y., and Liu, X. Adaptive 3d face reconstruction from unconstrained photo collections. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (June 2016), pp. 4197–4206.

[119] Saito, S., Hu, L., Ma, C., Ibayashi, H., Luo, L., and Li, H. 3d hair synthesis using volumetric variational autoencoders. *ACM Trans. Graph. 37*, 6 (Dec. 2018).

[120] Sandbach, G., Zafeiriou, S., Pantic, M., and Yin, L. Static and dynamic 3d facial expression recognition: A comprehensive survey. *Image and Vision Computing 30*, 10 (2012), 683–697.

[121] Saragih, J. M., Lucey, S., and Cohn, J. F. Deformable model fitting by regularized landmark mean-shift. *Int. J. Comput. Vision 91*, 2 (Jan. 2011), 200–215.

[122] Saragih, J. M., Lucey, S., and Cohn, J. F. Real-time avatar animation from a single image. In *Face and Gesture 2011* (2011), pp. 213–220.

[123] Sela, M., Richardson, E., and Kimmel, R. Unrestricted facial geometry reconstruction using image-to-image translation. In *2017 IEEE International Conference on Computer Vision (ICCV)* (Oct 2017), pp. 1585–1594.

[124] Seol, Y., Lewis, J., Seo, J., Choi, B., Anjyo, K., and Noh, J. Spacetime expression cloning for blendshapes. *ACM Trans. Graph. 31*, 2 (Apr. 2012), 14:1–14:12.

[125] Shi, F., Wu, H.-T., Tong, X., and Chai, J. Automatic acquisition of high-fidelity facial performances using monocular videos. *ACM Trans. Graph. 33*, 6 (Nov. 2014), 222:1–222:13.

[126] Sifakis, E., Neverov, I., and Fedkiw, R. Automatic determination of facial muscle activations from sparse motion capture marker data. *ACM Trans. Graph. 24*, 3 (July 2005), 417–425.

[127] Song, L., Lu, Z., He, R., Sun, Z., and Tan, T. Geometry guided adversarial facial expression synthesis. *arXiv preprint arXiv:1712.03474* (2017).

[128] Sorkine, O., Cohen-Or, D., Lipman, Y., Alexa, M., Rössl, C., and Seidel, H.-P. Laplacian surface editing. In *Proceedings of the 2004 Eurographics/ACM SIGGRAPH Symposium on Geometry Processing* (Nice, France, 2004), SGP '04, ACM, pp. 175–184.

[129] Stylianou, G., and Lanitis, A. Image based 3d face reconstruction: a survey. *International Journal of Image and Graphics 9*, 02 (2009), 217–250.

[130] Sunkavalli, K., Johnson, M. K., Matusik, W., and Pfister, H. Multi-scale image harmonization. *ACM Trans. Graph. 29*, 4 (July 2010), 125:1–125:10.

[131] Suwajanakorn, S., Kemelmacher-Shlizerman, I., and Seitz, S. M. Total moving face reconstruction. In *European Conference on Computer Vision* (Cham, 2014), Springer International Publishing, pp. 796–812.

[132] Suwajanakorn, S., Seitz, S. M., and Kemelmacher-Shlizerman, I. What makes Tom Hanks look like Tom Hanks. In *2015 IEEE International Conference on Computer Vision (ICCV)* (Dec 2015), pp. 3952–3960.

[133] Suwajanakorn, S., Seitz, S. M., and Kemelmacher-Shlizerman, I. Synthesizing Obama: Learning lip sync from audio. *ACM Trans. Graph. 36*, 4 (July 2017), 95:1–95:13.

[134] Taigman, Y., Polyak, A., and Wolf, L. Unsupervised cross-domain image generation. *arXiv preprint arXiv:1611.02200* (2016).

[135] Taylor, S., Kim, T., Yue, Y., Mahler, M., Krahe, J., Rodriguez, A. G., Hodgins, J., and Matthews, I. A deep learning approach for generalized speech animation. *ACM Trans. Graph. 36*, 4 (July 2017), 93:1–93:11.

[136] Tewari, A., Zollhöfer, M., Garrido, P., Bernard, F., Kim, H., Pérez, P., and Theobalt, C. Self-supervised multi-level face model learning for monocular reconstruction at over 250 hz. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2018).

[137] Tewari, A., Zollöfer, M., Kim, H., Garrido, P., Bernard, F., Perez, P., and Christian, T. MoFA: Model-based Deep Convolutional Face Autoencoder for Unsupervised Monocular Reconstruction. In *The IEEE International Conference on Computer Vision (ICCV)* (2017).

[138] Thies, J. Face2face: Real-time facial reenactment. In *Eurographics Digital Library - Online Dissertations* (2017).

[139] Thies, J., Zollhöfer, M., Niessner, M., Valgaerts, L., Stamminger, M., and Theobalt, C. Real-time expression transfer for facial reenactment. *ACM Trans. Graph. 34*, 6 (Oct. 2015), 183:1–183:14.

[140] Thies, J., Zollhöfer, M., Stamminger, M., Theobalt, C., and Niessner, M. Face2face: Real-time face capture and reenactment of rgb videos. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2016), pp. 2387–2395.

[141] Thies, J., Zollhöfer, M., Stamminger, M., Theobalt, C., and Niessner, M. Facevr: Real-time gaze-aware facial reenactment in virtual reality. *ACM Trans. Graph. 37*, 2 (June 2018), 25:1–25:15.

[142] Thies, J., Zollhöfer, M., Theobalt, C., Stamminger, M., and Niessner, M. Headon: Real-time reenactment of human portrait videos. *ACM Trans. Graph. 37*, 4 (July 2018), 164:1–164:13.

[143] Valgaerts, L., Wu, C., Bruhn, A., Seidel, H.-P., and Theobalt, C. Lightweight binocular facial performance capture under uncontrolled lighting. *ACM Trans. Graph. 31*, 6 (Nov. 2012), 187:1–187:11.

[144] Velho, L., and Zorin, D. 4-8 subdivision. *Comput. Aided Geom. Des. 18*, 5 (June 2001), 397–427.

[145] Vlasic, D., Brand, M., Pfister, H., and Popović, J. Face transfer with multilinear models. *ACM Trans. Graph. 24*, 3 (July 2005), 426–433.

[146] Vlasic, D., Peers, P., Baran, I., Debevec, P., Popović, J., Rusinkiewicz, S., and Matusik, W. Dynamic shape capture using multi-view photometric stereo. *ACM Trans. Graph. 28*, 5 (Dec. 2009), 174:1–174:11.

[147] Wang, C., Shi, F., Xia, S., and Chai, J. Realtime 3d eye gaze animation using a single rgb camera. *ACM Trans. Graph. 35*, 4 (July 2016), 118:1–118:14.

[148] Weise, T., Bouaziz, S., Li, H., and Pauly, M. Realtime performance-based facial animation. *ACM Trans. Graph. 30*, 4 (July 2011), 77:1–77:10.

[149] Weise, T., Li, H., Van Gool, L., and Pauly, M. Face/off: Live facial puppetry. In *Proceedings of the 2009 ACM SIGGRAPH/Eurographics Symposium on Computer Animation* (New Orleans, Louisiana, 2009), SCA '09, Association for Computing Machinery, p. 7–16.

[150] Williams, L. Performance-driven facial animation. In *Proceedings of the 17th Annual Conference on Computer Graphics and Interactive Techniques* (Dallas, TX, USA, 1990), SIGGRAPH '90, Association for Computing Machinery, p. 235–242.

[151] Williams, L. Performance-driven facial animation. In *ACM SIGGRAPH 2006 Courses* (New York, NY, USA, 2006), SIGGRAPH '06, ACM.

[152] Wu, C., Bradley, D., Garrido, P., Zollhöfer, M., Theobalt, C., Gross, M., and Beeler, T. Model-based teeth reconstruction. *ACM Trans. Graph. 35*, 6 (Nov. 2016).

[153] Wu, C., Bradley, D., Gross, M., and Beeler, T. An anatomically-constrained local deformation model for monocular face capture. *ACM Trans. Graph. 35*, 4 (July 2016).

[154] Wu, C., Shiratori, T., and Sheikh, Y. Deep incremental learning for efficient high-fidelity face tracking. *ACM Trans. Graph. 37*, 6 (Dec. 2018), 234:1–234:12.

[155] Wu, C., Stoll, C., Valgaerts, L., and Theobalt, C. On-set performance capture of multiple actors with a stereo camera. *ACM Trans. Graph. 32*, 6 (Nov. 2013), 161:1–161:11.

[156] Wu, C., Varanasi, K., Liu, Y., Seidel, H., and Theobalt, C. Shading-based dynamic shape refinement from multi-view video under general illumination. In *2011 International Conference on Computer Vision* (2011), pp. 1108–1115.

[157] Wu, C., Wilburn, B., Matsushita, Y., and Theobalt, C. High-quality shape from multi-view stereo and shading under general illumination. In *CVPR 2011* (2011), pp. 969–976.

[158] Wu, C., Zollhöfer, M., Niessner, M., Stamminger, M., Izadi, S., and Theobalt, C. Real-time shading-based refinement for consumer depth cameras. *ACM Trans. Graph. 33*, 6 (Nov. 2014), 200:1–200:10.

[159] Xu, F., Chai, J., Liu, Y., and Tong, X. Controllable high-fidelity facial performance transfer. *ACM Trans. Graph. 33*, 4 (July 2014), 42:1–42:11.

[160] Yamaguchi, S., Saito, S., Nagano, K., Zhao, Y., Chen, W., Olszewski, K., Morishima, S., and Li, H. High-fidelity facial reflectance and geometry inference from an unconstrained image. *ACM Trans. Graph. 37*, 4 (July 2018).

[161] Yang, F., Bourdev, L., Shechtman, E., Wang, J., and Metaxas, D. Facial expression editing in video using a temporally-smooth factorization. In *Proceedings of 2012 IEEE Conference on Computer Vision and Pattern Recognition* (June 2012), IEEE, pp. 861–868.

[162] Yang, F., Wang, J., Shechtman, E., Bourdev, L., and Metaxas, D. Expression flow for 3d-aware face component transfer. *ACM Trans. Graph. 30*, 4 (July 2011), 60:1–60:10.

[163] Zhang, L., Snavely, N., Curless, B., and Seitz, S. M. Spacetime faces: High resolution capture for modeling and animation. *ACM Trans. Graph. 23*, 3 (Aug. 2004), 548–558.

[164] Zhang, M., Wu, P., Wu, H., Weng, Y., Zheng, Y., and Zhou, K. Modeling hair from an rgb-d camera. *ACM Trans. Graph. 37*, 6 (Dec. 2018).

[165] Zhang, Z. A flexible new technique for camera calibration. *IEEE Transactions on Pattern Analysis and Machine Intelligence 22*, 11 (2000), 1330–1334.

[166] Zhou, Y., Xu, Z., Landreth, C., Kalogerakis, E., Maji, S., and Singh, K. Visemenet: Audio-driven animator-centric speech animation. *ACM Trans. Graph. 37*, 4 (July 2018).

[167] Zhu, J., Park, T., Isola, P., and Efros, A. A. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Proceedings of 2017 IEEE International Conference on Computer Vision (ICCV)* (Oct 2017), pp. 2242–2251.

[168] Zollhöfer, M., Niessner, M., Izadi, S., Rehmann, C., Zach, C., Fisher, M., Wu, C., Fitzgibbon, A., Loop, C., Theobalt, C., and Stamminger, M. Real-time non-rigid reconstruction using an rgb-d camera. *ACM Trans. Graph. 33*, 4 (July 2014), 156:1–156:12.

[169] Zollhöfer, M., Thies, J., Garrido, P., Bradley, D., Beeler, T., Pérez, P., Stamminger, M., Niessner, M., and Theobalt, C. State of the art on monocular 3d face reconstruction, tracking, and applications. *Computer Graphics Forum 37*, 2 (2018), 523–550.