Computational Visual Media
https://doi.org/10.1007/s41095-021-0215-y Vol. 7, No. 4, December 2021, 437–452
Research Article

Real-time face view correction for front-facing cameras

Yudong Guo[1], Juyong Zhang[1] (✉), Yihua Chen[1], Hongrui Cai[1], Zhangjin Huang[1], and Bailin Deng[2]

Abstract Face views are particularly important in person-to-person communication. Differenes between the camera location and the face orientation can result in undesirable facial appearances of the participants during video conferencing. This phenomenon is particularly noticeable when using devices where the front-facing camera is placed in unconventional locations such as below the display or within the keyboard. In this paper, we take a video stream from a single RGB camera as input, and generate a video stream that emulates the view from a virtual camera at a designated location. The most challenging issue in this problem is that the corrected view often needs out-of-plane head rotations. To address this challenge, we reconstruct the 3D face shape and re-render it into synthesized frames according to the virtual camera location. To output the corrected video stream with natural appearance in real time, we propose several novel techniques including accurate eyebrow reconstruction, high-quality blending between the corrected face image and background, and template-based 3D reconstruction of glasses. Our system works well for different lighting conditions and skin tones, and can handle users wearing glasses. Extensive experiments and user studies demonstrate that our method provides high-quality results.

Keywords face view correction; 3D face reconstruction; deep learning; online communication

1 Introduction

The face plays an essential role in human communication [1–3]. It is desirable that natural facial postures are preserved during video conferencing. However, this requirement is not always satisfied with existing consumer video conferencing systems. For example, in one-to-one video conferences using a mobile device or a laptop, the user tends to look at the screen area where the other participant's face is shown. However, the camera is placed at a location outside the screen. As a result, it will appear that the person is not facing the camera in the captured video, which can cause undesirable facial appearance. This issue has become more noticeable in recent years with the popularity of thin-bezel displays on laptops and mobile devices, as manufacturers start to place the front-facing camera in unconventional locations rather than above the display. For example, some laptops have a camera at the bottom of the display or within the keyboard, earning a nickname of "nosecam" as it can lead to undesirable exposure of the nostrils.

In the past, methods based on custom hardware setups have been proposed to improve facial appearance [4, 5]. However, they are often too expensive for a consumer-level system. Another possibility is to synthesize a facial image from the desired viewpoint based on the input from the real camera(s). Some approaches achieve reliable facial view synthesis using input from multiple cameras [6–8], but the high cost of such a system limits its application for typical consumers. Recently, some methods based on a single RGB-D or RGB camera have been proposed. For example, Kuster et al. [9] used depth information from an RGB-D camera to reconstruct 3D face geometry and generate a novel view. However, its applicability is limited, since most existing laptops and smartphones are not equipped with such RGB-D cameras. Later, Giger et al. [10] proposed a face view correction method using a single webcam, reconstructing the 3D face shape via

1 University of Science and Technology of China, Hefei 230026, China. E-mail: Y. Guo, gyd2011@mail.ustc.edu.cn; J. Zhang, juyong@ustc.edu.cn (✉); Y. Chen, chenyihua@mail.ustc.edu.cn; H. Cai, hrcai@mail.ustc.edu.cn; Z. Huang, zhuang@ustc.edu.cn.

2 School of Computer Science and Informatics, Cardiff University, Cardiff, Wales, UK. E-mail: DengB3@cardiff.ac.uk.

Manuscript received: 2021-01-29; accepted: 2021-02-24

Fig. 1 Face view correction results (bottom) with our system for input videos (top). Our method produces natural-looking results across different scenarios, including indoor and outdoor scenes, different lighting conditions, different skin tones, and users with glasses.

Laplacian deformation based on the detected facial landmarks. However, it does not work well for users who wear glasses, and its robustness is affected by the accuracy of landmark detection. In recent years, machine learning techniques have been applied to correct or manipulate gazes in images or videos [11–14], and can be used to improve eye-to-eye contact between video conferencing participants. However, they only modify the eye regions and do not correct undesirable appearance in other facial areas.

To reduce undesirable facial appearance caused by camera location, we need a method to automatically process the video captured by a camera to emulate the view from a virtual camera placed at a designated location. For practicality, we prefer to use a system with a simple hardware setup—ideally a single RGB camera—to synthesize the novel view. There are a few challenges to address. First, it is necessary to reconstruct the 3D face shape to accommodate the potentially large change between the input view and the synthesized view. Despite the recent success of monocular 3D face reconstruction [15–18], existing methods rely on parametric face models and cannot recover shapes of accessories such as glasses. As face regions occluded by accessories in the original view may be revealed in the novel view, the shapes of accessories must be considered during novel view synthesis. Second, the synthesized face view needs to replace the face in the original view. Due to the view differences, the boundaries of the two face views may not align. There may be visual artifacts around the transition region between the synthesized face and the original background. Finally, for video conferencing applications, the system must be efficient enough to synthesize views in real time.

In this paper, we propose a real-time face view correction system using a single RGB camera. Given the input video stream, our system synthesizes in real time a video stream from the view of a virtual camera with a designated location and orientation. For each input video frame, we first recover the 3D face shape and orientation using a convolutional neural network (CNN), with a novel landmark correspondence update strategy to improve reconstruction accuracy. Then the reconstructed face is re-rendered according to the coordinate transformation between the real camera and the virtual camera to derive the face view from the virtual camera, which will replace the face region from the original frame. To reduce visual artifacts between the rendered face and the background in the original frame, we perform seam optimization and Laplacian blending to achieve a natural transition between them. For users wearing glasses, we also propose a method to reconstruct the 3D shape of glasses based on the detected landmarks and a semantic segmentation mask; this is the first automatic 3D glasses reconstruction method as far as we know. By rendering the reconstructed 3D glasses shape and face shape together and handling the visible area which is invisible in the original view, a natural appearance can be achieved. Experimental results demonstrate that our method works well in various application scenarios.

2 Related work

2.1 3D face reconstruction

3D face reconstruction from a single image and facial performance capture from monocular video have made significant progress in recent years [19]. Most

existing methods are based on parametric models such as the 3D Morphable Model (3DMM) [20], FaceWarehouse [21], and FLAME [22], which learn a linear or bilinear basis from scanned 3D face data to represent general face shapes. Traditional methods reconstruct a 3D face model from an image via an analysis-by-synthesis approach, optimizing shape parameters by minimizing the difference between rendered reconstruction and the given image [20, 23]. Recently, machine learning techniques have been adopted to learn a mapping from the face image to its shape parameters [24–35]. Due to a lack of training data, some methods used synthetic data [24, 25, 28, 33] while others adopted unsupervised or weakly-supervised learning strategies [29–31, 34, 35]. To recover 3D face shapes from a monocular video, Garrido et al. [36] used a multi-layer approach and extracted a high-fidelity parameterized 3D face rig that contains a generative wrinkle formation model capturing person-specific idiosyncrasies. Cao et al. [37] presented a learning-based regression approach to fit a generic identity and expression model to an RGB face video on the fly. Thies et al. [15] proposed a method to jointly fit a parametric model for identity, expression, and skin reflectance to the input color, to provide real-time 3D face tracking and facial reenactment.

2.2 Face view correction

To improve eye contact in video conferencing, Kuster et al. [9] proposed a face view correction method based on an RGB-D camera, which directly performs a 3D transformation of the head geometry and then blends the corrected face image with the background. Later, Giger et al. [10] presented a shape deformation based method for face view correction for a single webcam. Zhai et al. [38] proposed a system that utilizes an RGB-D camera for gaze correction and face beautification. Other methods perform gaze correction only, using machine learning techniques to modify the appearance of the eye regions [11–14]. Although they can improve eye contact, these methods do not modify other face regions. They cannot correct their undesirable appearance due to camera locations.

2.3 Face normalization

Another problem related to our work is face normalization, which aims to remove perspective distortion, relight the face to emulate an evenly lit environment, and predict a frontal, neutral face given an arbitrary real world face image. Many existing works utilize 3D face geometry information to frontalize the face orientation. Hassner et al. [39] proposed a simple approach using a template 3D surface to estimate the intrinsic camera matrix, and the 2D face image is corrected based on the recovered information. Given a single portrait photo, Fried et al. [16] proposed to modify the relative pose and distance between the camera and the subject by first recovering a 3D head model and then warping the 2D image to approximate the effect of the desired change in 3D. In recent work, Zhao et al. [40] presented a learning-based approach to remove perspective distortion artifacts from unconstrained portraits by directly learning a distortion correction flow map. Ngano et al. [18] proposed a deep learning-based method that can fully normalize unconstrained face images. Yin et al. [41] presented a generative adversarial network for photo-realistic face frontalization by capturing both contextual dependencies and local consistency during training.

3 Our method

3.1 Overview

Our system takes captured video as input and in real time generates a video that shows the view from a virtual camera at a prescribed location and orientation. We assume that the virtual camera has the same intrinsic parameters as the real camera, so that the virtual camera can be considered to be the result of moving the real camera to a different location and/or orientation. We further assume that the relative orientation between the two cameras is fixed during the whole process, which is typical in real-world applications. To generate the view from the virtual camera, we first use a CNN to recover the shape, location, and orientation of the 3D face with respect to the real camera. The 3D face shape is then transformed into the camera coordinate system of the virtual camera and rendered to derive a new face image that replaces the face in the original frame. Finally, the rendered face image is blended with the original frame to generate the final output. The algorithm pipeline is shown in Fig. 2.

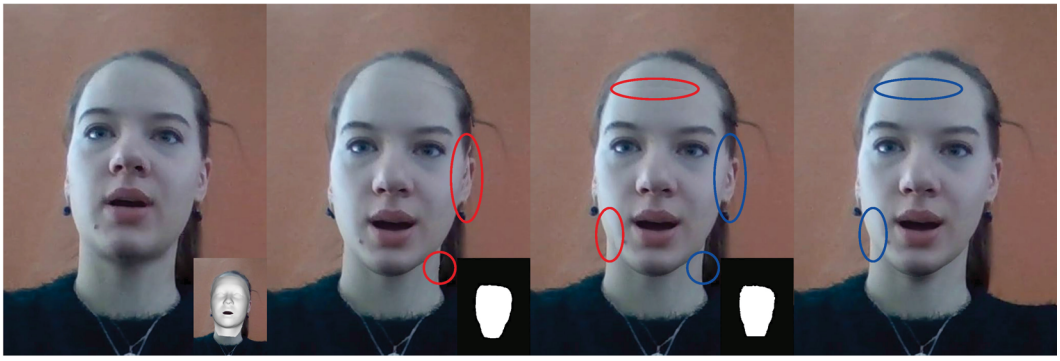


Fig. 2 Pipeline of our method. (a) Given an input face image, we reconstruct the 3D geometry and orientation of the face (bottom right). (b) The reconstructed face is rendered from the view of a virtual camera and overlayed onto the original image. (c) We optimize the seam between the rendered face and the original image to reduce visual artifacts. The bottom-right corners of (b) and (c) show the rendered face region mask before and after seam optimization, respectively. (d) We apply Laplacian blending to refine the transition and produce the final result. Red ellipses: visual artifacts. Blue ellipses: after improvement.

3.2 3D face reconstruction

3.2.1 Parametric face model

We use a bilinear face model based on FaceWarehouse [21] to encode facial identity and expression. To facilitate correction, we follow Ref. [10] and only keep the face and neck parts of the head model, as shown in Fig. 3(right). We collect vertex coordinates of all face meshes from FaceWarehouse into a third-order tensor and perform 2-mode singular value decomposition (SVD) reduction along the identity mode and the expression mode to generate a bilinear model that approximates the original dataset:

$$\boldsymbol{F} = \boldsymbol{C}_{\mathrm{r}} \times_2 \boldsymbol{\alpha}_{\mathrm{id}} \times_3 \boldsymbol{\alpha}_{\mathrm{exp}} \tag{1}$$

where $\boldsymbol{C}_{\mathrm{r}}$ is the reduced core tensor computed from the SVD reduction, $\boldsymbol{\alpha}_{\mathrm{id}}$, $\boldsymbol{\alpha}_{\mathrm{exp}}$ are identity and expression coefficients that control face shape, while $\times_2$ and $\times_3$ represent multiplication in the 2nd mode (identity) and the 3rd mode (expression), respectively. Following Ref. [20], facial albedo $\boldsymbol{b}$ is represented via principal component analysis (PCA):

$$\boldsymbol{b} = \overline{\boldsymbol{b}} + \boldsymbol{A}_{\mathrm{alb}}\boldsymbol{\alpha}_{\mathrm{alb}} \tag{2}$$

where $\overline{\boldsymbol{b}}$ is the average facial albedo, $\boldsymbol{A}_{\mathrm{alb}}$ are the principle axes extracted from a set of textured face meshes, and $\boldsymbol{\alpha}_{\mathrm{alb}}$ is the albedo coefficient vector. The albedo basis is obtained by transforming from the Basel Face Model (BFM) [42] to the FaceWarehouse model via nonrigid registration [43].

3.2.2 Camera and illumination model

We render the facial image using the weak perspective projection model:

$$\boldsymbol{p}'_v = \boldsymbol{\Pi}\boldsymbol{R}\boldsymbol{p}_v + \boldsymbol{t} \tag{3}$$

where $\boldsymbol{p}_v \in \mathbb{R}^3$ and $\boldsymbol{p}'_v \in \mathbb{R}^2$ are the locations of vertex v in the world coordinate system and in the image plane respectively, $\boldsymbol{R} \in SO(3)$ is a rotation matrix, $\boldsymbol{t} = [t_x, t_y]^{\mathrm{T}}$ is a translation vector, and $\boldsymbol{\Pi} = s\begin{bmatrix}1 & 0 & 0\\ 0 & 1 & 0\end{bmatrix}$ is the scaled projection matrix with scaling factor s.

To model the lighting condition, we approximate the global illumination using spherical harmonic (SH) [44] basis functions under the assumption that the face is a Lambertian surface. The irradiance of a vertex v is determined by its normal $\boldsymbol{n}_v$ and albedo $\boldsymbol{b}_v$ via

$$\boldsymbol{I}(\boldsymbol{n}_v, \boldsymbol{b}_v|\boldsymbol{\gamma}) = \boldsymbol{b}_v \cdot \sum_{k=1}^{(\ell+1)^2} \gamma_k \phi_k(\boldsymbol{n}_v) \tag{4}$$

where $\boldsymbol{\phi}_k$ are the SH basis functions, and $\boldsymbol{\gamma} = [\gamma_1, \cdots, \gamma_{(\ell+1)^2}]^{\mathrm{T}}$ are the SH coefficients with ℓ being the maximal order of SH basis ($\ell = 2$ in this paper).

3.2.3 Learning-based 3D face reconstruction

To recover 3D face shape, we use ResNet-18 [45] to directly regress the parameter vector $\boldsymbol{\chi} = \{\boldsymbol{\alpha}_{\mathrm{id}}, \boldsymbol{\alpha}_{\mathrm{exp}}, \boldsymbol{\alpha}_{\mathrm{alb}}, s, \boldsymbol{R}, \boldsymbol{t}, \boldsymbol{\gamma}, \boldsymbol{c}\}$ from an input image. Here $\boldsymbol{c}$ is a vector that describes the characteristics of the image, including the occlusion ratio of the face region and whether the subject wears glasses. Such characteristics are used for glasses reconstruction and validity judgement in the correction step, as explained in Sections 3.5 and 3.6. Following recent self-supervised CNNs for 3D face reconstruction [29, 34], we guide CNN training using the following loss function for each training image:

$$E(\boldsymbol{\chi}) = E_{\mathrm{pho}} + w_1 E_{\mathrm{lan}} + w_2 E_{\mathrm{reg}} + w_3 E_{\mathrm{cha}} \tag{5}$$

The photometric loss:

$$E_{\text{pho}} = \frac{1}{|\mathcal{M}|} \sum_{m \in \mathcal{M}} \|\boldsymbol{I}_{\text{syn}}(m) - \boldsymbol{I}_{\text{real}}(m)\|_2$$

measures the consistency between the input image and the face image resulting from the regressed parameters, where $\mathcal{M}$ denotes the set of pixels in the visible face region, and $\boldsymbol{I}_{\text{syn}}(m)$, $\boldsymbol{I}_{\text{real}}(m)$ are the synthetic color and the real color at pixel m, respectively. The landmark loss:

$$E_{\text{lan}} = \frac{1}{|\mathcal{L}|} \sum_{v \in \mathcal{L}} \|\boldsymbol{q}_v - (\boldsymbol{\Pi R p}_v + \boldsymbol{t})\|_2^2$$

evaluates the distance between the detected landmarks in the input image and the projections of the corresponding landmark vertices from the 3D face model, where $\mathcal{L}$ is the set of landmark vertices, and $\boldsymbol{p}_v, \boldsymbol{q}_v$ denote 3D coordinates of a landmark vertex v and the 2D coordinates of its corresponding detected landmark, respectively. The term:

$$E_{\text{reg}} = \sum_{i=1}^{50} \left(\frac{\boldsymbol{\alpha}_{\text{id},i}}{\boldsymbol{\sigma}_{\text{id},i}}\right)^2 + \sum_{i=1}^{50} \left(\frac{\boldsymbol{\alpha}_{\text{alb},i}}{\boldsymbol{\sigma}_{\text{alb},i}}\right)^2 + \sum_{i=1}^{47} \left(\frac{\boldsymbol{\alpha}_{\text{exp},i}}{\boldsymbol{\sigma}_{\text{exp},i}}\right)^2$$

regularizes the parameters for the face shape and albedo, where $\boldsymbol{\sigma}_{\text{id}}$, $\boldsymbol{\sigma}_{\text{exp}}$, and $\boldsymbol{\sigma}_{\text{alb}}$ are the corresponding singular values obtained from the 2-mode SVD reduction or PCA. E_{cha} is a loss function for the characteristics of the image; its definition will be given in Section 3.6. Scalars w_1, w_2, and w_3 are tuning weights which we set to 3, 0.01, and 0.5 respectively. To train the network, we constructed a large-scale training dataset consisting of nearly 900k face images from 500 subjects. The images were captured using RGB cameras in different consumer laptops, smartphones, and tablets. During acquisition, the subjects sat or stood in a variety of environments and performed various actions such as scratching the head, gesturing, making phone calls, and so on. To improve the robustness of reconstruction, we captured these face images from a variety of angles. We used the method from Ref. [46] to detect facial landmarks for all images.

3.2.4 Landmark correspondence update

The landmark vertices on the face mesh are labeled based on the frontal pose. For non-frontal face images, the detected 2D landmarks along the face contour may not correspond well with the landmark vertices. We update the silhouette landmark vertices according to the current rotation matrix $\boldsymbol{R}$ during training. Specifically, we pre-process the original face mesh to derive a dense set of horizontal lines covering the potential silhouette region from a rotated view (see Fig. 4). For each face model in every mini-batch during training, we choose the vertex with the smallest value of $|\boldsymbol{N} \cdot \boldsymbol{V}|$ from each horizontal line to construct the estimated silhouette, where $\boldsymbol{N}$ and $\boldsymbol{V}$ are the vertex normal and view direction, respectively. Then for each 2D contour landmark, we update its corresponding landmark vertex to the silhouette vertex whose projection computed with Eq. (3) is closest (see Fig. 4).

Unlike other facial features, the shape of the eyebrows can vary greatly between different persons (see Fig. 5). Therefore, a fixed eyebrow landmark template as shown in Fig. 3(right) may not give a good fit to the detected eyebrow landmarks in the input image, due to the limited number of parameters for the 3D face shape. Inaccurate eyebrow shape estimation will cause visual artifacts after view correction due to depth ambiguity. To solve this problem, we propose a novel strategy that can adaptively adjust the eyebrow shape according to the input face image. We first label a set of default eyebrow landmark vertices on the

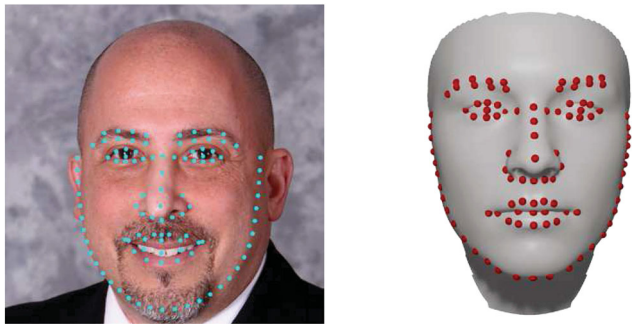


Fig. 3 Detected 2D landmarks on an input face image (left) and the corresponding 3D landmarks on the face mesh (right).

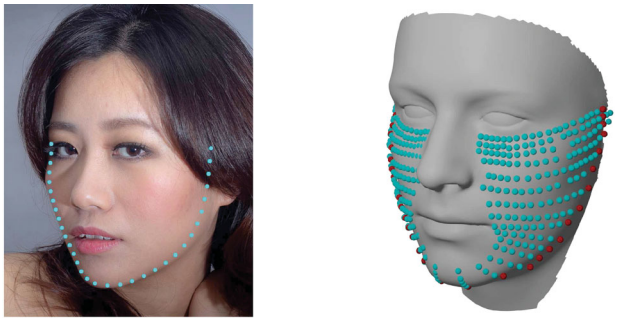


Fig. 4 For a non-frontal face image (left), we dynamically update the 3D face silhouette landmarks (red) according to the current estimated rotation, to give better correspondence with the detected 2D silhouette landmarks (left, cyan).

Fig. 5 Eyebrows with different shapes, and detected landmarks for the eyebrow and the eye.

template mesh. During training, the actual landmark vertices are dynamically updated according to the current 3D face shape. Specifically, we compute a tangential correction vector $\boldsymbol{\delta}_v$ for each default eyebrow landmark vertex v (parameterized using local coordinates in its tangent plane on the face mesh), so that the projection $\boldsymbol{p}_v + \boldsymbol{\delta}_v$ becomes closer to its corresponding 2D eyebrow landmark. Afterwards, the mesh vertex closest to the corrected position $\boldsymbol{p}_v + \boldsymbol{\delta}_v$ is chosen as the updated landmark vertex. The correction vectors are computed simultaneously via:

$$\min \sum_{v\in\mathcal{L}_{\text{eb}}} (\|\boldsymbol{\Pi R}(\boldsymbol{p}_v+\boldsymbol{\delta}_v)+\boldsymbol{t}-\boldsymbol{q}_v\|_2^2+w_{\text{eb}}\|\Delta\boldsymbol{\delta}_v\|_2^2) \quad (6)$$

where $\mathcal{L}_{\text{eb}}$ denotes the set of default eyebrow landmark vertices. The second term in the target function is a smoothness energy for the tangential corrections with a weight w_{eb}, which is set to 0.01. We first connect the default eyebrow landmark vertices to form a closed polyline that outlines the eyebrow boundary. Then $\Delta\boldsymbol{\delta}_v = \boldsymbol{\delta}_v - \frac{1}{2}(\boldsymbol{\delta}_v^+ + \boldsymbol{\delta}_v^-)$ is the discrete Laplacian operator of the tangential corrections along the polyline at vertex v, where $\boldsymbol{\delta}_v^+, \boldsymbol{\delta}_v^-$ are the tangential corrections at its preceding and succeeding landmark vertices, respectively. The Laplacian energy regularizes the tangential corrections so that the updated landmarks form a reasonable outline of the eyebrow shape.

3.3 Pose correction

With the learned parameters described above, $\boldsymbol{Rp}_v$ in Eq. (3) represents the learned position of a face vertex v (up to a common translation for all vertices) using the camera coordinate system of the real camera. Recall that the relative orientation between the real camera and the virtual camera is fixed. Therefore, we can pre-compute the rotation $\boldsymbol{R}_{\text{c}}$ between the camera coordinate systems, so the position of v (up to a common translation for all vertices) in the virtual camera's coordinate system is $\boldsymbol{R}_{\text{c}}\boldsymbol{Rp}_v$. Recall further that the two cameras have the same intrinsic parameters, so the mapping $\boldsymbol{\Pi}$ in Eq. (3) also describes the scaled projection matrix of the virtual camera. Therefore, we can derive the following image coordinates for v from the view of the virtual camera:

$$\boldsymbol{p}_v'' = \boldsymbol{\Pi R}_{\text{c}}\boldsymbol{Rp}_v + \boldsymbol{t}'' \quad (7)$$

where $\boldsymbol{t}'' \in \mathbb{R}^2$ is a common translation for all vertices; its determination will be explained shortly. Based on this relation, we render the face image from the view of the virtual camera and use it to replace the face region in the video frame from the real camera, while retaining the other parts of the frame. To determine the texture of the rendered face, we use the weak perspective projection of the real camera to assign color information from the input video frame to the texture of the face model, and reuse the texture to render the virtual camera view. Since the rendered face replaces the original face, we determine the common translation $\boldsymbol{t}''$ so that the two faces overlap. Specifically, $\boldsymbol{t}''$ is determined by minimizing the ℓ_2 distance from the projected landmark locations $\{\boldsymbol{p}_v'' \mid v \in \mathcal{L}\}$ of the rendered face to the detected landmarks $\{\boldsymbol{q}_v \mid v \in \mathcal{L}\}$ in the original frame:

$$\min \sum_{v\in\mathcal{L}} \|\boldsymbol{p}_v'' - \boldsymbol{q}_v\|_2^2 \quad (8)$$

An example is shown in Fig. 2(b).

3.4 Background blending

Directly overlaying the corrected rendered face onto the original image may result in unnatural transitions around the boundary of the face region, as the rendered face and the original face may not fully align (see Fig. 2(b)). Therefore, we apply a blending operation between the rendered face and the original image to improve the appearance. We first optimize a seam between the original image and the rendered face to reduce the visual artifact across the seam. Afterwards, we further refine the result using Laplacian blending [47].

3.4.1 Seam optimization

The goal of seam optimization is to find a seam between the rendered face image and the original image, such that the image content outside the seam (which comes from the original image) is as consistent as possible with the content inside the seam (which

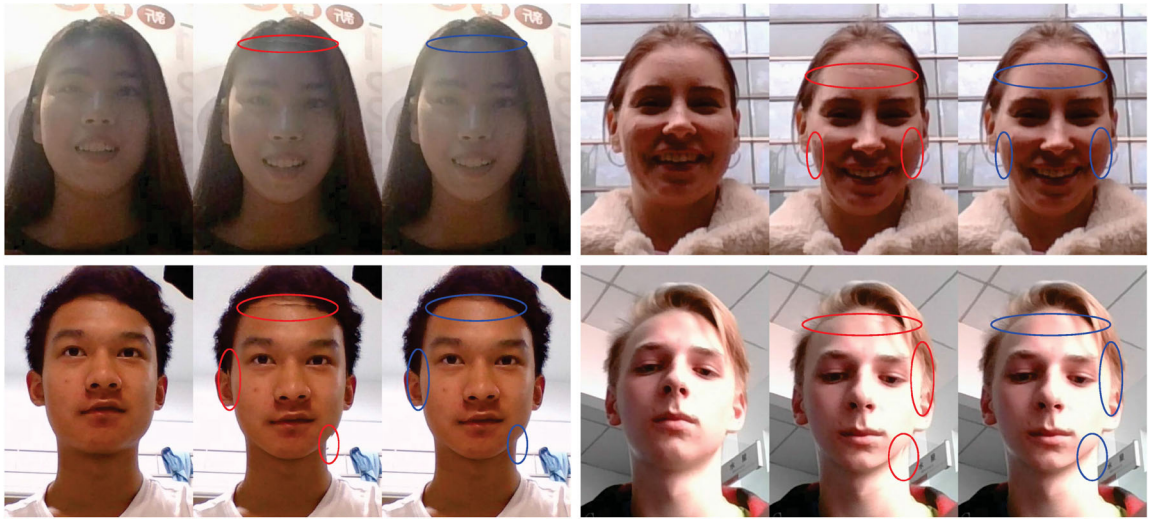


Fig. 6 Further examples of seam optimization and Laplacian blending. In each case, left to right: original image, corrected image after seam optimization, and final image after Laplacian blending. Red ellipses: visual artifacts. Blue ellipses: improvement after Laplacian blending.

comes from the rendered face). Following Ref. [48], we formulate the seam optimization as a graph cut problem over a fusion area that is a region of the rendered face around its boundary. To determine the fusion region, we first take the optimized seam from the previous frame and apply a translation that best aligns the detected landmarks in the two frames (computed by optimization similar to Eq. (8)) to derive a closed curve B. Then we perform a breadth-first search from B, and derive the fusion area as the union of any pixel location $\boldsymbol{x}$ that lies within the rendered face region and satisfies $d_B(\boldsymbol{x}) \leqslant 10$, where $d_B(\boldsymbol{x})$ is the BFS distance from $\boldsymbol{x}$ to B. Again following Ref. [48], we search for a seam that lies in the fusion area and minimizes the following target function:

$$E_{\text{seam}} = \sum_{(\boldsymbol{x},\boldsymbol{y})\in\mathcal{P}} \alpha(\boldsymbol{x},\boldsymbol{y}) \cdot (\|\boldsymbol{I}(\boldsymbol{x}) - \boldsymbol{J}(\boldsymbol{x})\|_2 + \|\boldsymbol{I}(\boldsymbol{y}) - \boldsymbol{J}(\boldsymbol{y})\|_2)$$

where $\mathcal{P}$ denotes the set of adjacent pixels across the seam, and $\boldsymbol{I}(\cdot)$, $\boldsymbol{J}(\cdot)$ denote the pixel color in the original image and the rendered face, respectively. The term $\|\boldsymbol{I}(\boldsymbol{x}) - \boldsymbol{J}(\boldsymbol{x})\|_2 + \|\boldsymbol{I}(\boldsymbol{y}) - \boldsymbol{J}(\boldsymbol{y})\|_2$ indicates consistency of color between the two images across the seam [48], with the weight $\alpha(\boldsymbol{x},\boldsymbol{y}) = \exp(\min(d_B(\boldsymbol{x}), d_B(\boldsymbol{y})))$ favoring a seam with similar shape to the optimized seam from the previous frame. Like Ref. [48], we solve the optimization problem using graph cut [49]. An example of seam optimization is shown in Fig. 2(c), showing a more natural appearance than directly overlaying the corrected rendered face (Fig. 2(b)).

3.4.2 Laplacian blending

After seam optimization, the result may still contain artifacts if there is a large difference between the face poses in the original image and the rendered image. Thus we further refine the result via Laplacian blending [47], using the rendered face within the seam as the foreground. An example is shown in Fig. 2(d). For Laplacian blending, we set the level of the pyramid to 5 in all our experiments. Figure 6 shows further examples of the effectiveness of Laplacian blending in improving appearance. A comparison between generated video sequences with and without Laplacian blending can be found in the Electronic Supplementary Material (ESM).

3.5 Faces with glasses

3.5.1 Approach

Using the characteristic vector $\boldsymbol{c}$ returned by the neural network in Section 3.2, we can determine whether the user is wearing glasses. For a face with glasses, reconstructing only the 3D face shape may produce unnatural results: since we reuse the texture information from the real camera to render the face for the virtual camera, the texture for the glasses may appear distorted due to the view discrepancy between the two cameras (see Fig. 10(middle)). For a more natural appearance, we reconstruct the 3D shape of the glasses, which is then transformed and rendered together with the face. The 3D glasses shape is reconstructed by deforming a template mesh (see

Fig. 7) to align its 2D projection with the glasses area from the input frame. We prescribe 12 landmarks (red in Fig. 7) on the template mesh to facilitate the alignment: four around the boundary of each lens, one at each hinge between the lens and the temples, and one at the end of each temple. For reconstruction, we first use neural networks to detect the landmarks and determine a segmentation mask for the glasses from the input frame. Then we deform the template model to align its 2D projection with the 2D glasses image, to obtain the 3D shape of the glasses. Finally, we rotate the face together with the glasses, and render them to the 2D plane. As glasses shape and position relative to the face are usually fixed, for efficiency we only perform this reconstruction once at the beginning. In the following, we provide algorithmic details for each step.

3.5.2 3D glasses reconstruction

From the input frame, we use U-Net [50] to segment the glasses, and ResNet-18 [45] to regress the landmark positions and determine whether the user wears glasses. The two networks are trained using 2600 images with manually labeled landmarks and segmentation masks; Fig. 8 gives some examples from the training set.

To reconstruct the shape of the glasses, we first optimize a similarity transformation of the template mesh to align the projection of 10 landmarks (eight around the boundaries of the lenses and two at the hinges) with their corresponding detected landmarks.

Fig. 7 Template mesh and the corresponding 12 landmarks (indicated with red circles) used in 3D glasses reconstruction.

Fig. 8 Manually labeled landmarks and segmentation masks for glasses.

Then we fix the lens regions of the mesh and rotate each template region around its hinge to align the projections of the two landmarks at the end of each temple with their corresponding detected landmarks. The whole mesh is further deformed non-rigidly to match the segmentation mask of the glasses. Specifically, we use the iterative solver from Ref. [51] to optimize a mesh deformation that aligns its projected boundary with the boundary of the segmentation mask, while enforcing the smoothness of the deformation using a Laplacian energy. Figure 9(b) shows an example of 3D glasses shape reconstruction.

3.5.3 Rendering glasses with face

The relative position between the glasses and the face is fixed and can be pre-computed using the initial frame of the video sequence. For each subsequent frame, we directly use the learned face pose to determine the location of the glasses within the real camera's coordinate system, and render the face together with the glasses from the virtual camera view following Eq. (7). The texture information from the input frame is assigned to the visible regions of the face and the glasses and reused to render them. Due to the view discrepancy between the real camera and the virtual camera, a face region occluded by the glasses in the real camera view may

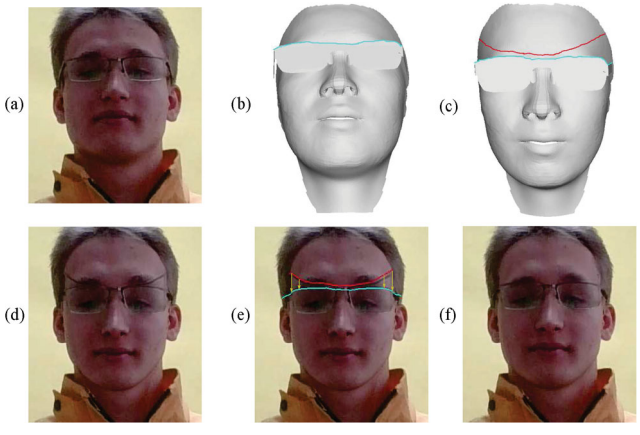


Fig. 9 Reconstruction and rendering of glasses. (a) Input face image. (b) Reconstructed face and glasses mesh from the real camera view. Cyan: top boundary of the glasses. (c) Rconstructed face and glasses from the virtual camera view. Red: bottom boundary of the face region above the glasses, visible from the real camera. (d) Directly rendering the face and glasses from the virtual camera view produces an unnatural result: the face region between the red and cyan curves in (c) is occluded in the real camera view and lacks texture information. (e) To remove the texture gap between red and cyan curves, we slide each column of pixels in the face region above the red curve downwards to meet the cyan curve. (f) Final result after merging the modified rendered image with the original image using seam optimization and Laplacian blending.

be visible in the virtual camera view, as shown in Fig. 9(c), where the virtual camera is placed above the real camera and exposes a region occluded in the real camera view. The exposed region appears in the virtual camera view as a gap without texture information, located between the top boundary of the glasses (cyan) and the face region above the glasses that is visible from the real camera (bottom boundary in red). To determine the texture of the exposed region for rendering, we could potentially use image in-painting [52], but this is computationally involved. Since the rendered face still needs to be merged with the original frame, we adopt a simple approach to handle the gap without texture. For the face region above the red curve, we take each column of its pixels and slide the whole column downwards vertically until it meets the top of the glasses region (i.e., the cyan curve). This effectively fills the gap while removing some pixels from the top of the face. Afterwards, we perform Laplacian smoothing on each horizontal row of pixels within the original gap region to create a smooth transition. This modified rendered face image is then merged with the original frame via seam optimization and Laplacian blending, as for a face without glasses. In this way, the removed top region of the rendered face is replaced by face pixels from the original frame, and the merging afterwards produces a natural appearance: see our experiments. Figure 9(f) shows an example of the final merged image. Figure 10 further compares correction results with and without reconstruction of 3D glasses shapes, clearly showing the benefit in reducing distortion of the glasses. More examples are available in the ESM.

3.6 Validity judgment

When there is slight occlusion around the face boundary in the input frame (e.g., occlusion by hair), some of the occluding object's color information may be treated as texture on the face model. In this case, seam optimization and Laplacian blending help to create a natural transition between the occluding texture and the part of the occluding object that lies outside the face region. However, when a larger part of the face is occluded by an object that lies across the face region and the background, the correction result may still look unnatural after the blending. In addition, if the input face is severely occluded, learning-based 3D face reconstruction may produce inaccurate results. Therefore, for each input frame, we check the face occlusion ratio from the characteristic vector returned by the neural network in Section 3.2, and apply face view correction only if the threshold is below a pre-defined threshold. We set the threshold to 25% in all experiments. When training the network, we manually label the occluded face region in each training image to provide a ground-truth ratio of face occlusion. We also label each image to indicate whether the subject is wearing glasses. The loss function term E_{cha} for a training image combines two quantities: the squared difference between the predicted face occlusion ratio and the ground-truth ratio, and a softmax classification loss for glasses. Figure 11 shows examples of occlusion ratios predicted by our network.

Fig. 10 Correction results for users wearing glasses. Above: input frames. Middle: results without reconstructing the 3D shapes of the glasses. The glasses region in the input is treated as texture on the face model, resulting in distortion in the virtual camera view. Below: correction results using reconstructed glasses shapes.

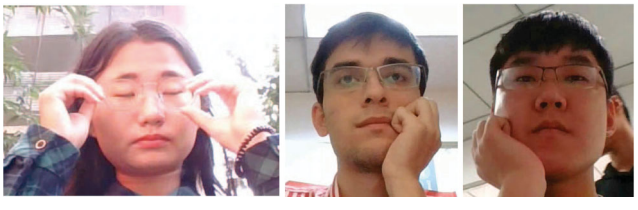


Fig. 11 Images with occlusion. The 3D face reconstruction network outputs the face occlusion ratio for validity judgement. The estimated occlusion ratios for these three face images are 8.5%, 14.3%, and 13.5%, respectively.

Furthermore, we do not apply correction if the face pose change is too large. We check the rotation matrix $\boldsymbol{R}$ returned by the neural network for face reconstruction. If the magnitude of a rotation angle exceeds a threshold, we gradually reduce the correction to identity in the following four frames to avoid abrupt changes in the output video. Specifically, we replace the pre-computed rotation $\boldsymbol{R}_{\mathrm{c}}$ in Eq. (7) by another rotation $\overline{\boldsymbol{R}}_{\mathrm{c}}$, with $\overline{\boldsymbol{R}}_{\mathrm{c}}$ transitioning from $\boldsymbol{R}_{\mathrm{c}}$ to identity. Similarly, if the captured face rotation falls back to lie within the threshold, we gradually change the rotation $\overline{\boldsymbol{R}}_{\mathrm{c}}$ back to $\boldsymbol{R}_{\mathrm{c}}$ over the next four frames. In our experiments, we set the thresholds for yaw, pitch and roll to 20°, 35°, and 14°, respectively.

4 Results

In this section, we evaluate the performance of our method, and compare it with state-of-the-art methods. Our evaluation used a laptop with an Intel Core i7-8565U, 8 GB of RAM, an Nvidia GeForce MX250, and a webcam located within the keyboard. Unless stated otherwise, input video was captured using the laptop webcam, and the virtual camera is at the center of the display with a viewing direction orthogonal to the screen.

4.1 Efficiency

Our system is fully automatic and runs in real time, our un-optimized implementation achieving 20 fps for 1280×720 input videos. For an input frame, 3D face reconstruction typically takes 26 ms, re-rendering, 8 ms, and seam optimization and blending, 13 ms. For a user with glasses, we need a pre-processing step to reconstruct the 3D glasses shape, and an additional step for each frame to handle the gap due to exposed occluded regions. These two steps typically take 63 and 2 ms, respectively. As glasses reconstruction is only performed once, it has negligible impact on the efficiency of the system.

4.2 Robustness

We tested the robustness of our system to different lighting conditions, poses, glasses, and the user motion such as head rotation and movement. Some video results are included in the ESM. Figure 1 demonstrates that our system is robust under various environments, including indoor and outdoor scenes and different lighting conditions, and works well for different skin tones. Figure 12 shows that our system can correctly handle horizontal rotations of the user's head. Figure 13 shows an example where the user quickly approached the camera; our method is robust to such fast movements. Further examples can be found in the ESM. We also evaluated our system for users wearing different types of glasses. As shown in Fig. 10 and the ESM, our system can correctly handle different glasses to produce natural-looking results.

4.3 Face reconstruction accuracy

We evaluated the accuracy of our 3D face reconstruction network by conducting quantitative comparisons with state-of-the-art learning-based methods [29, 34, 53–55]. With the same setting as Ref. [54], we compared our results on 180 meshes of 9 subjects from FaceWarehouse. Following the evaluation protocol of Ref. [34], we computed the

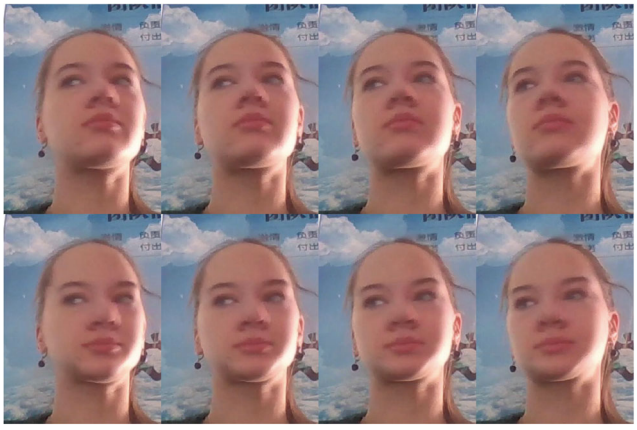


Fig. 12 Correction results for a user with horizontal head rotation. Above: input frames. Below: correction results.

Fig. 13 Above: input frames for a user quickly approaching the camera. Below: correction results

point-to-point distances between the reconstructed meshes and the ground-truth meshes after alignment by running ICP with uniform scaling. The point-to-point errors are listed in Table 1. We also show the reconstruction results and error maps in Fig. 14. It can be seen that our method outperforms the method of Deng et al. [34] in terms of shape and expression reconstruction.

4.4 Visual quality

We conducted a user study on 50 participants to evaluate the visual quality of the results from our system. We collected 10 video sequences covering different scenarios, including indoor and outdoor scenes, different lighting conditions, subjects with different skin tones, with and without glasses. Each participant watched the original and corrected videos and was asked to rate their satisfaction with the results in three ways: naturalness of the results, consistency of facial appearance between the two videos, and accuracy of view correction. Each aspect was rated with a score between 1 and 5, with 1 worst and 5 best. Average scores for the three aspects were 4.35, 4.25, and 4.41, respectively, demonstrating that our method can generate natural-looking corrected views of faces.

Table 1 Mean reconstruction error on 180 meshes of 9 subjects from FaceWarehouse. Our geometric error is the lowest among all methods

Method	Ours	[34]	[55]	[54]	[29]	[53]
Mean (mm)	**1.76**	1.81	2.01	1.84	2.19	2.11
SD (mm)	**0.35**	0.50	0.41	0.38	0.54	0.46

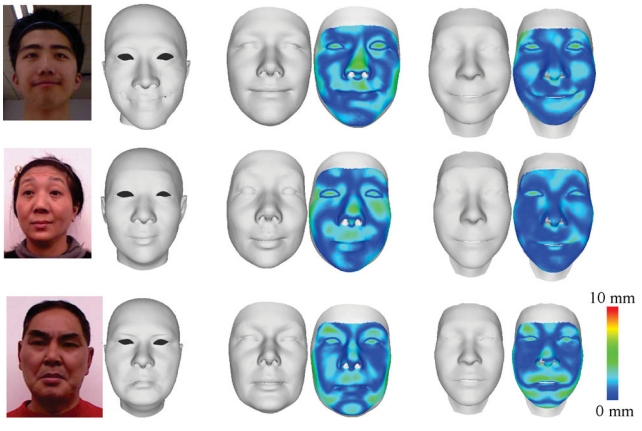


Fig. 14 Reconstruction from RGB inputs using the method of Deng et al. [34] and our method. Left to right: input images and ground-truth meshes, Deng's method's results and error maps, our results and error maps.

4.5 Comparison to a state-of-the-art method

A method particularly relevant to our work is the webcam-based approach from Ref. [10]. For a fair comparison, we only replaced our 3D face reconstruction component with the deformation based method in their paper, while all other steps remained unchanged. Examples are shown in Fig. 15 and in the ESM. Our method produces more natural results as its learning-based 3D face reconstruction utilizes face shape priors and landmark update strategies that correctly handle large discrepancies between the real and virtual cameras. In comparison,

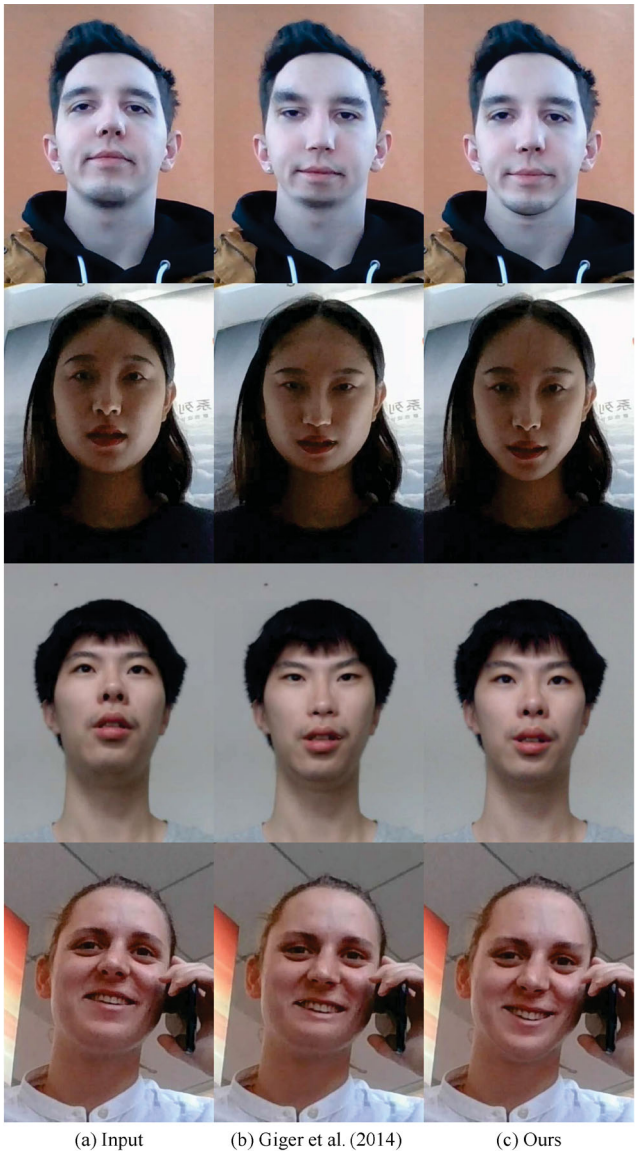


Fig. 15 Comparisons between Ref. [10] and our method. The former does not utilize shape priors of human faces and may produce unnatural results.

the method from Ref. [10] only deforms a 3D face template to match detected 2D landmarks; lack of facial shape priors can lead to distortion. Correspondences between 2D and 3D landmarks are also fixed in their method to allow pre-factorization of the deformation matrix for fast computation, which may introduce errors under large pose changes. Although it is possible to improve the accuracy for Ref. [10] by adopting a landmark update strategy similar to ours, this would result in a deformation matrix that may need to be frequently re-factorized, increasing the computational cost. Finally, unlike our method, the deformation approach [10] does not take glasses shape into account and may produce unnatural results, as noted as a limitation in their paper.

For qualitative and quantitative comparison, we also compare the results of the two methods for video captured using a camera at the same location as the virtual camera. We first placed a webcam at a position well below the face to capture video as input for the correction algorithms, with the virtual camera located in front of the user's face (see Fig. 16 for the setup using our correction method). Then we placed a webcam of the same kind at the location of the virtual camera to capture a frontal reference video simultaneously for comparison. We collected 10 pairs of input and reference video sequences using this setup and applied our method and that of Ref. [10] to correct the input video. Figure 17 shows examples from the frontal reference videos together with the correction results using the two methods. We can see that our method produces more natural results with appearance closer to the reference videos. For

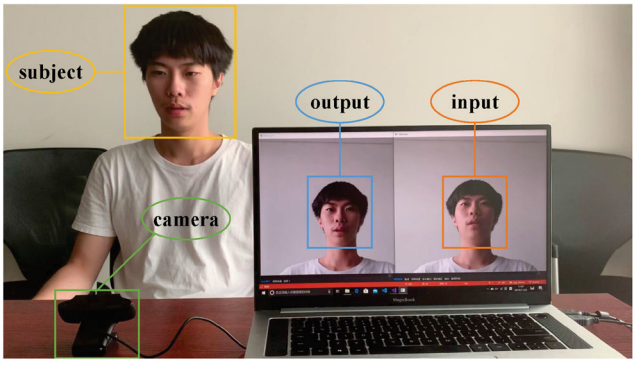


Fig. 16 Set up of our system for correcting videos captured from a webcam well below the face. Real-time results are shown in the accompanying video.

Fig. 17 Comparisons between captured real frontal videos, and corrected results from Ref. [10] and our method. Our results are closer to the real frontal videos.

further verification, we evaluated the perceptual similarity between the correction results and the reference videos using deep face recognition features. Specifically, for each frame from the input camera, we took the corresponding frame in the reference video as well as the corrected frames using the two methods and evaluated their facial recognition feature according to Ref. [56]. We then computed the cosine distance from the feature of each corrected frame to the feature of the reference frame, with a larger value indicating higher perceptual similarity. We repeated this process for all frames of all 10 input videos and computed the average cosine distance for each method. Our method achieved an average value 0.91, whereas the average value for Ref. [10] is 0.84, indicating that our results have higher perceptual similarity to the reference video. We also conducted a user study on the same 50 participants mentioned previously to compare the visual quality of results from our method and Ref. [10]. Each participant was shown the 10 pairs of corrected videos and asked to choose the better one in each pair. For a fair comparison, each participant was first shown the input video, followed by the two corrected videos in random order, without information about the correction method used. Overall, our result was considered to be the

better one in 89.6% of the pairs, showing that our method produces more visually convincing results.

We also compared our method to a face reenactment method [15]. Although the method does not target face view correction, its pipeline could be adopted for this task. We show a qualitative comparison of results in Fig. 18. It can be seen that our method produces more frontal facial images and more natural glasses correction.

4.6 Limitations and future work

Our system has several limitations that need to be addressed as future work. First, we reuse the facial texture from the real camera view to render a facial image for the virtual camera view. If the view discrepancy between the two cameras is too large, there may be face areas that are visible from the virtual camera but invisible from the real camera, and our method cannot handle such cases. This issue can potentially be resolved by running a short pre-calibration session to capture the full facial texture from different views. Second, we directly use the glasses texture from the real camera to render glasses for the virtual camera view, which does not correctly capture optical effects of the lenses such as refraction. One possible solution is to further refine the corrected results using a generative adversarial network.

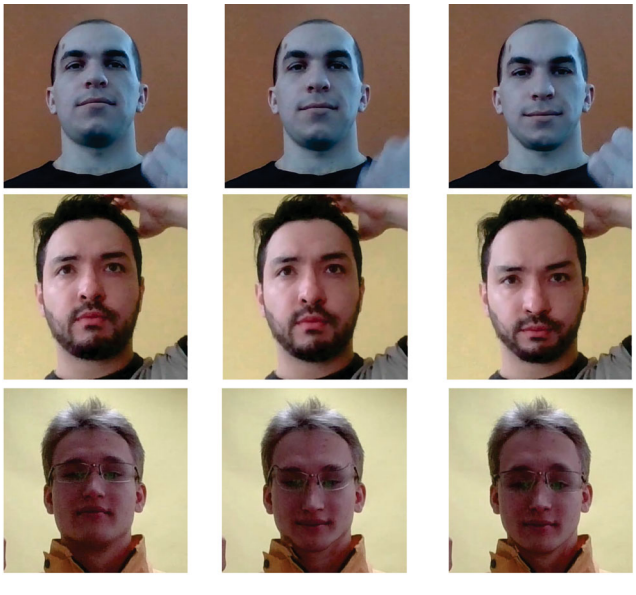


Fig. 18 Comparison between our method and Ref. [15]. Left to right: input images, correction results of Ref. [15], correction results of our method.

5 Conclusions

We have proposed a fully automatic face view correction system based on a single RGB camera. We trained a neural network to reconstruct 3D face shape using the video input from the camera and generate a video that imitates a novel view from a virtual camera. Our method can also correct face videos where the users wear glasses by reconstructing the 3D shape of the glasses. Our system is robust to different conditions, including lighting conditions, skin tones, and glasses. It operates in real time on a consumer laptop and produces visually appealing and convincing results. With its robustness and efficiency, our system can potentially be applied to various devices to improve the user experience in video conferencing applications.

Electronic Supplementary Material Supplementary material is available in the online version of this article at `https://doi.org/10.1007/s41095-021-0215-y`.

References

[1] Monk, A. F.; Gale, C. A look is worth a thousand words: Full gaze awareness in video-mediated conversation. *Discourse Processes* Vol. 33, No. 3, 257–278, 2002.

[2] Grayson, D. M.; Monk, A. F. Are You looking at me? Eye contact and desktop video conferencing. *ACM Transactions on Computer-Human Interaction* Vol. 10, No. 3, 221–243, 2003.

[3] Mukawa, N.; Oka, T.; Arai, K.; Yuasa, M. What is connected by mutual gaze: User's behavior in video-mediated communication. In: Proceedings of the Extended Abstracts on Human Factors in Computing Systems, 1677–1680, 2005.

[4] Ishii, H.; Kobayashi, M. ClearBoard: A seamless medium for shared drawing and conversation with eye contact. In: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, 525–532, 1992.

[5] Okada, K. I.; Maeda, F.; Ichikawaa, Y.; Matsushita, Y. Multiparty videoconferencing at virtual social distance: MAJIC design. In: Proceedings of the ACM Conference on Computer Supported Cooperative Work, 385–393, 1994.

[6] Matusik, W.; Buehler, C.; Raskar, R.; Gortler, S. J.; McMillan, L. Image-based visual hulls. In: Proceedings of the 27th Annual Conference on Computer Graphics and Interactive Techniques, 369–374, 2000.

[7] Matusik, W.; Pfister, H. 3D TV: A scalable system for real-time acquisition, transmission, and autostereoscopic display of dynamic scenes. *ACM Transactions on Graphics* Vol. 23, No. 3, 814–824, 2004.

[8] Kuster, C.; Popa, T.; Zach, C.; Gotsman, C.; Gross, M. H. FreeCam: A hybrid camera system for interactive free-viewpoint video. In: *Vision, Modeling,and Visualization*. Eisert, P.; Polthier, K.; Hornegger J. Eds. The Eurographics Association, 17–24, 2011.

[9] Kuster, C.; Popa, T.; Bazin, J. C.; Gotsman, C.; Gross, M. Gaze correction for home video conferencing. *ACM Transactions on Graphics* Vol. 31, No. 6, Article No. 174, 2012.

[10] Giger, D.; Bazin, J. C.; Kuster, C.; Popa, T.; Gross, M. Gaze correction with a single webcam. In: Proceedings of the IEEE International Conference on Multimedia and Expo, 1–6, 2014.

[11] Kononenko, D.; Lempitsky, V. Learning to look up: Realtime monocular gaze correction using machine learning. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 4667–4675, 2015.

[12] Hsu, C. F.; Wang, Y. S.; Lei, C. L.; Chen, K. T. Look at me! Correcting eye gaze in live video communication. *ACM Transactions on Multimedia Computing, Communications, and Applications* Vol. 15, No. 2, Article No. 38, 2019.

[13] He, Z.; Spurr, A.; Zhang, X. C.; Hilliges, O. Photo-realistic monocular gaze redirection using generative adversarial networks. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, 6931–6940, 2019.

[14] Zhang, J. C.; Chen, J. J.; Tang, H.; Wang, W.; Yan, Y.; Sangineto, E., Sebe, N. Dual in-painting model for unsupervised gaze correction and animation in the wild. In: Proceedings of the 28th ACM International Conference on Multimedia, 1588–1596, 2020.

[15] Thies, J.; Zollhöfer, M.; Stamminger, M.; Theobalt, C.; Nießner, M. Face2Face: Real-time face capture and reenactment of RGB videos. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2387–2395, 2016.

[16] Fried, O.; Shechtman, E.; Goldman, D. B.; Finkelstein, A. Perspective-aware manipulation of portrait photos. *ACM Transactions on Graphics* Vol. 35, No. 4, Article No. 128, 2016.

[17] Shu, Z. X.; Hadap, S.; Shechtman, E.; Sunkavalli, K.; Paris, S.; Samaras, D. Portrait lighting transfer using a mass transport approach. *ACM Transactions on Graphics* Vol. 36, No. 4, Article No. 2, 2017.

[18] Nagano, K.; Luo, H. W.; Wang, Z. J.; Seo, J.; Xing, J.; Hu, L. W.; Wei, L.; Li, H. Deep face normalization. *ACM Transactions on Graphics* Vol. 38, No. 6, Article No. 183, 2019.

[19] Zollhöfer, M.; Thies, J.; Garrido, P.; Bradley, D.; Beeler, T.; Pérez, P.; Stamminger, M.; Nießner, M.; Theobalt, C. State of the art on monocular 3D face reconstruction, tracking, and applications. *Computer Graphics Forum* Vol. 37, No. 2, 523–550, 2018.

[20] Blanz, V.; Vetter, T. A morphable model for the synthesis of 3D faces. In: Proceedings of the 26th Annual Conference on Computer Graphics and Interactive Techniques, 187–194, 1999.

[21] Cao, C.; Weng, Y. L.; Zhou, S.; Tong, Y. Y.; Zhou, K. FaceWarehouse: A 3D facial expression database for visual computing. *IEEE Transactions on Visualization and Computer Graphics* Vol. 20, No. 3, 413–425, 2014.

[22] Li, T. Y.; Bolkart, T.; Black, M. J.; Li, H.; Romero, J. Learning a model of facial shape and expression from 4D scans. *ACM Transactions on Graphics* Vol. 36, No. 6, Article No. 194, 2017.

[23] Jiang, L.; Zhang, J. Y.; Deng, B. L.; Li, H.; Liu, L. G. 3D face reconstruction with geometry details from a single image. *IEEE Transactions on Image Processing* Vol. 27, No. 10, 4756–4770, 2018.

[24] Richardson, E.; Sela, M. T.; Kimmel, R. 3D face reconstruction by learning from synthetic data. In: Proceedings of the 4th International Conference on 3D Vision, 460–469, 2016.

[25] Zhu, X. Y.; Lei, Z.; Liu, X. M.; Shi, H. L.; Li, S. Z. Face alignment across large poses: A 3D solution. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 146–155, 2016.

[26] Richardson, E.; Sela, M. T.; Or-El, R.; Kimmel, R. Learning detailed face reconstruction from a single image. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 5553–5562, 2017.

[27] Jackson, A. S.; Bulat, A.; Argyriou, V.; Tzimiropoulos, G. Large pose 3D face reconstruction from a single image via direct volumetric CNN regression. In: Proceedings of the IEEE International Conference on Computer Vision, 1031–1039, 2017.

[28] Sela, M. T.; Richardson, E.; Kimmel, R. Unrestricted facial geometry reconstruction using image-to-image translation. In: Proceedings of the IEEE International Conference on Computer Vision, 1585–1594, 2017.

[29] Tewari, A.; Zollhöfer, M.; Kim, H.; Garrido, P.; Bernard, F.; Pérez, P.; Theobalt, C. MoFA: Model-based deep convolutional face autoencoder for unsupervised

monocular reconstruction. In: Proceedings of the IEEE International Conference on Computer Vision, 3735–3744, 2017.

[30] Tran, L.; Liu, X. Nonlinear 3D face morphable model. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 7346–7355, 2018.

[31] Genova, K.; Cole, F.; Maschinot, A.; Sarna, A.; Vlasic, D.; Freeman, W. T. Unsupervised training for 3D morphable model regression. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 8377–8386, 2018.

[32] Gecer, B.; Ploumpis, S.; Kotsia, I.; Zafeiriou, S. GANFIT: Generative adversarial network fitting for high fidelity 3D face reconstruction. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 1155–1164, 2019.

[33] Guo, Y. D.; Zhang, J. Y.; Cai, J. F.; Jiang, B. Y.; Zheng, J. M. CNN-based real-time dense face reconstruction with inverse-rendered photo-realistic face images. *IEEE Transactions on Pattern Analysis and Machine Intelligence* Vol. 41, No. 6, 1294–1307, 2019.

[34] Deng, Y.; Yang, J. L.; Xu, S. C.; Chen, D.; Jia, Y. D.; Tong, X. Accurate 3D face reconstruction with weakly-supervised learning: From single image to image set. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops, 285–295, 2019.

[35] Tewari, A.; Bernard, F.; Garrido, P.; Bharaj, G.; Elgharib, M.; Seidel, H. P.; Pérez, P.; Zollhöfer, M.; Theobalt, C. FML: Face model learning from videos. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 10804–10814, 2019.

[36] Garrido, P.; Zollhöfer, M.; Casas, D.; Valgaerts, L.; Varanasi, K.; Pérez, P.; Theobalt, C. Reconstruction of personalized 3D face rigs from monocular video. *ACM Transactions on Graphics* Vol. 35, No. 3, Article No. 28, 2016.

[37] Cao, C.; Hou, Q. M.; Zhou, K. Displaced dynamic expression regression for real-time facial tracking and animation. *ACM Transactions on Graphics* Vol. 33, No. 4, Article No. 43, 2014.

[38] Zhai, D. M.; Liu, X. M.; Ji, X. Y.; Zhao, D. B.; Gao, W. Joint gaze correction and face beautification for conference video using dual sparsity prior. *IEEE Transactions on Industrial Electronics* Vol. 66, No. 12, 9601–9611, 2019.

[39] Hassner, T.; Harel, S.; Paz, E.; Enbar, R. Effective face frontalization in unconstrained images. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 4295–4304, 2015.

[40] Zhao, Y. J.; Huang, Z.; Li, T. Y.; Chen, W. K.; Legendre, C., Ren, X. L.; Shapiro, A.; Li, H. Learning perspective undistortion of portraits. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, 7848–7858, 2019.

[41] Yin, Y.; Jiang, S. Y.; Robinson, J. P.; Fu, Y. Dual-attention GAN for large-pose face frontalization. In: Proceedings of the 15th IEEE International Conference on Automatic Face and Gesture Recognition, 249–256, 2020.

[42] Paysan, P.; Knothe, R.; Amberg, B.; Romdhani, S.; Vetter, T. A 3D face model for pose and illumination invariant face recognition. In: Proceedings of the 6th IEEE International Conference on Advanced Video and Signal Based Surveillance, 296–301, 2009.

[43] Sumner, R. W.; Popović, J. Deformation transfer for triangle meshes. *ACM Transactions on Graphics* Vol. 23, No. 3, 399–405, 2004.

[44] Müller, C. *Spherical Harmonics*. Springer Berlin Heidelberg, 1966.

[45] He, K. M.; Zhang, X. Y.; Ren, S. Q.; Sun, J. Deep residual learning for image recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 770–778, 2016.

[46] Bulat, A.; Tzimiropoulos, G. How far are we from solving the 2D & 3D face alignment problem? (and a dataset of 230,000 3D facial landmarks). In: Proceedings of the IEEE International Conference on Computer Vision, 1021–1030, 2017.

[47] Burt, P. J.; Adelson, E. H. A multiresolution spline with application to image mosaics. *ACM Transactions on Graphics* Vol. 2, No. 4, 217–236, 1983.

[48] Kwatra, V.; Schödl, A.; Essa, I.; Turk, G.; Bobick, A. Graphcut textures. *ACM Transactions on Graphics* Vol. 22, No. 3, 277–286, 2003.

[49] Boykov, Y.; Kolmogorov, V. An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision. *IEEE Transactions on Pattern Analysis and Machine Intelligence* Vol. 26, No. 9, 1124–1137, 2004.

[50] Ronneberger, O.; Fischer, P.; Brox, T. U-net: Convolutional networks for biomedical image segmentation. In: *Medical Image Computing and Computer-Assisted Intervention. Lecture Notes in Computer Science, Vol. 9351*. Navab, N.; Hornegger, J.; Wells, W.; Frangi, A. Eds. Springer Cham, 234–241, 2015.

[51] Bouaziz, S.; Deuss, M.; Schwartzburg, Y.; Weise, T.; Pauly, M. Shape-up: Shaping discrete geometry with projections. *Computer Graphics Forum* Vol. 31, No. 5, 1657–1667, 2012.

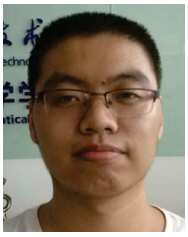
[52] Guillemot, C.; Le Meur, O. Image inpainting: Overview and recent advances. *IEEE Signal Processing Magazine* Vol. 31, No. 1, 127–144, 2014.

[53] Kim, H.; Zollhöfer, M.; Tewari, A.; Thies, J.; Richardt, C.; Theobalt, C. InverseFaceNet: Deep monocular inverse face rendering. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 4625–4634, 2018.

[54] Tewari, A.; Zollhöfer, M.; Garrido, P.; Bernard, F.; Kim, H.; Pérez, P.; Theobalt, C. Self-supervised multi-level face model learning for monocular reconstruction at over 250 Hz. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2549–2559, 2018.

[55] Tewari, A.; Bernard, F.; Garrido, P.; Bharaj, G.; Elgharib, M.; Seidel, H. P.; Pérez, P.; Zollhöfer, M.; Theobalt, C. FML: Face model learning from videos. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 10804–10814, 2019.

[56] Schroff, F.; Kalenichenko, D.; Philbin, J. FaceNet: A unified embedding for face recognition and clustering. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 815–823, 2015.



Yudong Guo is a Ph.D. student in the School of Mathematical Sciences, University of Science and Technology of China (USTC). He obtained his bachelor degree from the same university in 2015. His research interests include computer vision and computer graphics.

Juyong Zhang is an associate professor in the School of Mathematical Sciences at USTC. He received his B.S. degree from USTC in 2006, and his Ph.D. degree from Nanyang Technological University, Singapore. His research interests include computer graphics, computer vision, and numerical optimization. He is an associate editor of *The Visual Computer*.

Yihua Chen is a master student at the School of Mathematical Sciences, USTC. He received his bachelor degree in mathematical sciences from USTC in 2020. His research interests include computer vision and computer graphics.

Hongrui Cai is a master student at the School of Data Science, USTC. He got his bachelor degree from South China University of Technology in 2019. His research interests include computer vision and computer graphics.

Zhangjin Huang received his B.S. and Ph.D. degrees in computational mathematics from USTC in 1999 and 2005, respectively. He is currently an associate professor in the School of Computer Science and Technology, USTC. His current research interests include computer graphics and image processing.

Bailin Deng is a lecturer in the School of Computer Science & Informatics at Cardiff University. He received his B.Eng. degree in computer software (2005) and his M.Sc. degree in computer science (2008) from Tsinghua University, China, and his Ph.D. degree in technical mathematics (2011) from Vienna University of Technology, Austria. His research interests include geometry processing, numerical optimization, computational design, and digital fabrication. He is a member of the IEEE.