Computational Visual Media
DOI 10.1007/s41095-xxx-xxxx-x Vol. x, No. x, month year, xx–xx

Article Type (Research/Review)

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]

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.

Abstract Face view is particularly important in person-to-person communication. Disparity between the camera location and the face orientation can result in undesirable facial appearances of the participants during video conferencing. This phenomenon becomes particularly notable on devices where the front-facing camera is placed at unconventional locations such as below the display or within the keyboard. In this paper, we takes the video stream from a single RGB camera as input, and generates a video stream that emulates the view from a virtual camera at a designated location. The most challenging issue of this problem is that the corrected view often needs out-of-plane head rotations. To address this challenge, we reconstruct 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 corrected face image and background, and a template-based 3D reconstruction of glasses. Our system works well for different lighting conditions and skin tones, and is able to handle users wearing glasses. Extensive experiments and user studies demonstrate that our proposed method can achieve high-quality results.

1 Introduction

Face view plays an essential role in human communication [14, 32, 33]. 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, for one-to-one video conferencing using a mobile device or a laptop, a user tends to look at the screen area where the other participant's face is shown. At the same time, 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 appearances. This issue is becoming more noticeable

1 University of Science and Technology of China, Hefei 230026, China. E-mail: gyd2011@mail.ustc.edu.cn, juyong@ustc.edu.cn, chenyihua@mail.ustc.edu.cn, hrcai@mail.ustc.edu.cn, zhuang@ustc.edu.cn.

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

Manuscript received: XXXX-XX-XX; accepted: XXXX-XX-XX.

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 "nosecam" as it can lead to undesirable exposure of the nostrils.

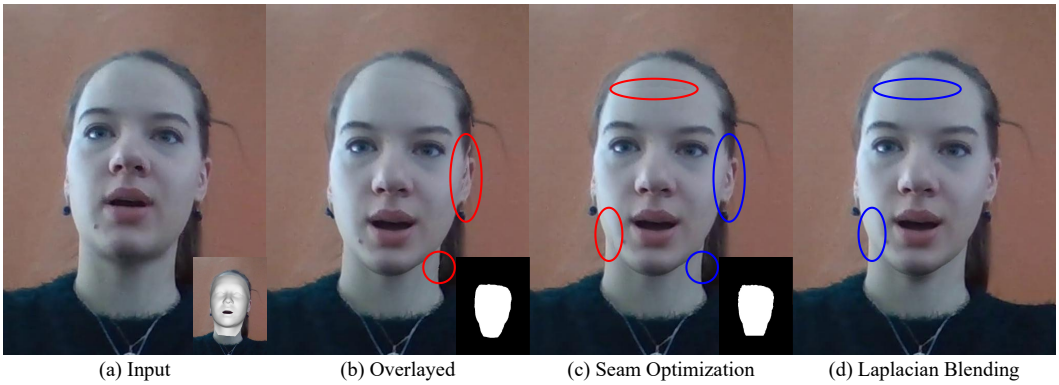
In the past, some methods based on custom-made hardware setups have been proposed to improve the facial appearance [21, 36]. However, they are often too expensive for a consumer-level system. Another possibility is to synthesize a facial image from the desired viewpoint based on the input from the real camera(s). Some approaches achieve reliable facial view synthesis using input from multiple cameras [27, 30, 31], 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. [26] used the depth information from an RGB-D camera to reconstruct 3D face geometry and generate a novel view. However, its applicability is still limited, since most existing laptops or smartphones are not equipped with such RGB-D cameras. Later, Giger et al. [13] proposed a face view correction method using a single webcam, reconstructing the 3D face shape via Laplacian deformation based on the detected face landmarks. However, it does not work well for users who wear glasses, and its robustness is affected by the accuracy of face landmarks detection. In recent years, machine learning techniques have been applied to correct or manipulate gazes in images or videos [19, 20, 25, 53], which can be used to improve eye-to-eye contacts between participants of video conferencing. 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 locations, we need a method that can automatically process the video captured by a camera to emulate the view from a virtual camera placed at a designated location. To make the solution practical, 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 be addressed here. 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 [9, 35, 43, 49], existing methods rely on parametric face models and cannot recover the shapes of accessories such as glasses. As face regions occluded by accessories in the original view may be revealed in the novel view, the accessories shapes must be considered during the novel view synthesis. Secondly, the synthesized face view needs to replace the face in the original view. Due to the view discrepancy, 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 generate the synthesized view 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 the reconstruction accuracy. Then the reconstructed face is re-rendered according to the transformation between the coordinate systems of 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 semantic segmentation mask, and 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, natural appearance can be achieved. Experimental results demonstrate that our method works well for various application scenarios.

2 Related Work

3D Face Reconstruction 3D face reconstruction from a single image and facial performance capture from a monocular video have made significant progress in recent years [56]. Most existing methods are based on parametric models such as 3D Morphable Model (3DMM) [1], FaceWarehouse [7] and FLAME [29], 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 [1, 23]. Recently, machine learning techniques have been adopted to learn a mapping from the face image to its shape parameters [8, 11, 12, 16, 22, 38, 39, 42, 45, 48, 50, 55]. Due to the lack of training data, some methods used synthetic data [16, 38, 42, 55] while others adopted



(a) Input (b) Overlayed (c) Seam Optimization (d) Laplacian Blending

Fig. 2 The pipeline of our method. (a) For an input face image, we reconstruct the 3D geometry and orientation of the face (shown at the bottom right corner). (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. Visual artifacts are highlighted with red ellipses, while the improvements are highlighted with blue ellipses.

unsupervised or weakly-supervised learning strategies [8, 12, 45, 48, 50]. To recover 3D face shapes from a monocular video, Garrido et al. [10] used a multi-layer approach and extracted a high-fidelity parameterized 3D face rig that contains a generative wrinkle formation model capturing the person-specific idiosyncrasies. Cao et al. [6] 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. [49] proposed a method to jointly fit a parametric model for identity, expression, and skin reflectance to the input color, which achieves real-time 3D face tracking and facial reenactment.

Face View Correction To improve eye contact in video conferencing, Kuster et al. [26] 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. [13] presented a shape deformation based method for face view correction for a single webcam. Zhai et al. [52] proposed a system that utilizes an RGB-D camera for gaze correction and face beautification. Other methods aim at gaze correction only, using machine learning techniques to modify the appearance of the eye regions [19, 20, 25, 53]. Although they can improve eye contact, these methods do not modify other face regions. They cannot correct their undesirable appearance due to camera locations.

Face Normalization Another problem related to our work is face normalization, which aims to remove perspective distortions, relight the face to an evenly lit environment, and predict a frontal and neutral face for an input "in-the-wild" face image. Many existing works utilize the 3D face geometry information to frontalize the face orientation. Hassner et al. [17] 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. [9] proposed to modify the relative pose and distance between the camera and the subject by first recovering the 3D head model and then warping the 2D image to approximate the effect of the desired change in 3D. In very recent work, Zhao et al. [54] presented a learning-based approach to remove perspective distortion artifacts from unconstrained portraits by directly learning a distortion correction flow map. Ngano et al. [35] proposed a deep learning-based method that can fully normalize unconstrained face images for the above tasks. Yin et al. [51] presented a generative adversarial network for photo-realistic face frontalization by capturing both contextual dependencies and local consistency during training.

3 Our Method

Our system takes the captured video as input and generates a video that shows the view from a virtual camera with a prescribed location and orientation in real-time. We assume that the virtual camera has the same intrinsic parameters as the real camera, so that the virtual camera can be considered as 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 a common scenario in real-world applications. To generate the view from the virtual camera, we first use 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.

3.1 3D Face Reconstruction

Parametric Face Model We use a bilinear face model based on FaceWarehouse [7] to encode facial identity and expression. To facilitate the correction in the later step, we follow [13] and only keep the face and neck parts of the head model, as shown in Fig. 3 (right). We collect the 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:

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

where $\mathbf{C}_{\mathrm{r}}$ 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 the face shape, $\times_2$ and $\times_3$ represent the multiplication in the 2*nd* mode (identity) and the 3*rd* mode (expression) respectively. Following [1], the facial albedo $\mathbf{b}$ is represented with principal component analysis (PCA):

$$\mathbf{b} = \bar{\mathbf{b}} + \mathbf{A}_{\mathrm{alb}} \boldsymbol{\alpha}_{\mathrm{alb}}, \tag{2}$$

where $\bar{\mathbf{b}}$ is the average facial albedo, $\mathbf{A}_{\mathrm{alb}}$ is 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) [37] to the FaceWarehouse model via nonrigid registration [44].

Camera and Illumination Model We render the facial image using the weak perspective projection model:

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

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

To model the lighting condition, we approximate the global illumination using the spherical harmonics (SH) [34] basis functions under the assumption that the face is a Lambertian surface. The irradiance of a vertex v is determined by its normal $\mathbf{n}_v$ and albedo $\mathbf{b}_v$ via

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

where $\boldsymbol{\phi}_k$ are the SH basis functions, and $\boldsymbol{\gamma} = [\boldsymbol{\gamma}_1, \ldots, \boldsymbol{\gamma}_{(\ell+1)^2}]^T$ are the SH coefficients with ℓ being the maximal order of SH basis ($\ell = 2$ in this paper).

Learning-based 3D Face Reconstruction To recover the 3D face shape, we use ResNet-18 [18] to directly regress the parameters $\boldsymbol{\chi} = \{\boldsymbol{\alpha}_{\mathrm{id}}, \boldsymbol{\alpha}_{\mathrm{exp}}, \boldsymbol{\alpha}_{\mathrm{alb}}, s, \mathbf{R}, \mathbf{t}, \boldsymbol{\gamma}, \mathbf{c}\}$ from an input image. Here $\mathbf{c}$ 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 will be used for glasses reconstruction and validity judgment in the correction step, as explained in Sections 3.4 and 3.5. Following recent self-supervised CNNs for 3D face reconstruction [8, 48], we guide the CNN training using the following loss function for each training image:

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

Here the photometric loss

$$E_{\mathrm{pho}} = \frac{1}{|\mathscr{M}|} \sum_{m \in \mathscr{M}} \|\mathbf{I}_{\mathrm{syn}}(m) - \mathbf{I}_{\mathrm{real}}(m)\|_2$$

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

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

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 $\mathscr{L}$ is the set of landmark vertices, and $\mathbf{p}_v, \mathbf{q}_v$ denote 3D coordinates of a landmark vertex v and the 2D coordinates of its corresponding detected landmark, respectively. The term

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

regularizes the parameters for the face shape and albedo, where $\boldsymbol{\sigma}_{\mathrm{id}}$, $\boldsymbol{\sigma}_{\mathrm{exp}}$ and $\boldsymbol{\sigma}_{\mathrm{alb}}$ 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, and its definition will be explained in Section 3.5. Scalars w_1, w_2 and w_3 are tuning weights and we set them to 3, 0.01 and 0.5 respectively. To train the network, we construct a large-scale training dataset consisting of nearly 900K face images from 500 subjects. The images are captured using RGB cameras in different consumer laptops, smartphones, and tablets. During the acquisition, the subjects sit or stand in a variety of environments and perform various actions such as scratching head, gesturing, making phone calls, and so on. To improve the robustness of reconstruction, we capture the face images from a variety of angles. We use the method from [4] to detect facial landmarks for all images.

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

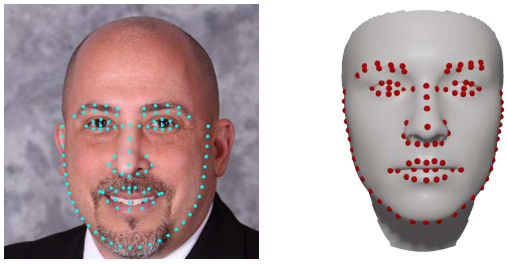


Fig. 3 The detected 2D landmarks on an input face image (left) and the corresponding 3D landmarks on the face mesh (right).

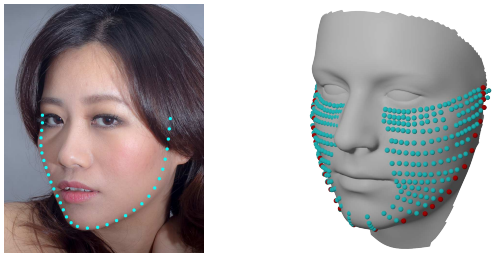


Fig. 4 For a non-frontal face image (left), we dynamically update the 3D face silhouette landmarks (shown in red) according to the current estimated rotation for better correspondence with the detected 2D silhouette landmarks (shown in cyan on the left).

along the face contour may not correspond well with the landmark vertices. We update the silhouette landmark vertices according to the current rotation matrix $\mathbf{R}$ 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 $|\mathbf{N}\cdot\mathbf{V}|$ from each horizontal line to construct the estimated silhouette, where $\mathbf{N}$ and $\mathbf{V}$ 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 the most nearby (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 fit well 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 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 $\boldsymbol{\delta}_v$

Fig. 5 Some eyebrows in different shapes, and the detected landmarks for the eyebrow and the eye.

for each default eyebrow landmark vertex v (parameterized using local coordinates in its tangent plane on the face mesh), so that the projection $\mathbf{p}_v+\boldsymbol{\delta}_v$ gets closer to its corresponding 2D eyebrow landmark. Afterward, the mesh vertex closest to the corrected position $\mathbf{p}_v+\boldsymbol{\delta}_v$ is chosen as the updated landmark vertex. The correction vectors are computed simultaneously via:

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

where $\mathscr{L}_{eb}$ 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.

3.2 Pose Correction

With the learned parameters described above, $\mathbf{R}\mathbf{p}_v$ 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 $\mathbf{R}_{\text{c}}$ between their camera coordinate systems, and derive the position of v (up to a common translation for all vertices) in the virtual camera's coordinate system as $\mathbf{R}_{\text{c}}\mathbf{R}\mathbf{p}_v$. 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:

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

where $\mathbf{t}''\in\mathbb{R}^2$ is a common translation for all vertices, and its determination will be explained later. Based on this

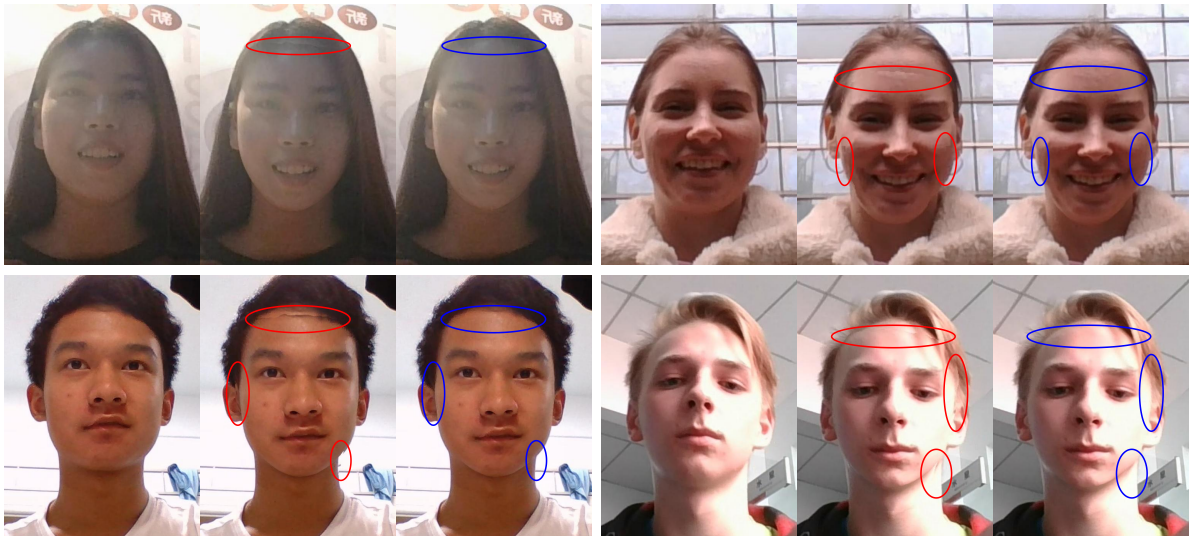


Fig. 6 More examples of seam optimization and Laplacian blending. For each example, we show from left to right the original image, the corrected image after seam optimization, and the final image after Laplacian blending. The red ellipses highlight visual artifacts, and the blue ellipses highlight the improvement with Laplacian blending.

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 for the rendering from the virtual camera view. Since we will use the rendered face to replace the original face, we determine the common translation $\mathbf{t}''$ so that the two faces overlap. Specifically, $\mathbf{t}''$ is determined by minimizing the ℓ_2 distance from the projected landmark locations $\{\mathbf{p}''_v \mid v \in \mathscr{L}\}$ of the rendered face to the detected landmarks $\{\mathbf{q}_v \mid v \in \mathscr{L}\}$ in the original frame:

$$\min \sum_{v \in \mathscr{L}} \|\mathbf{p}''_v - \mathbf{q}_v\|_2^2, \tag{8}$$

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

3.3 Background Blending

Directly overlaying the corrected rendered face onto the original image may result in unnatural transition 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. Afterward, we further refine the result using Laplacian blending [5].

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 comes from the rendered face). Similar to [28], 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 (which is 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 $\mathbf{x}$ that lies within the rendered face region and satisfies $d_B(\mathbf{x}) \leq 10$, where $d_B(\mathbf{x})$ is the BFS distance from $\mathbf{x}$ to B. Then similar with [28], we search for a seam that lies in the fusion area and minimizes the following target function

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

where $\mathscr{P}$ denotes the set of adjacent pixels across the seam, and $\mathbf{I}(\cdot)$, $\mathbf{J}(\cdot)$ denote the pixel color in the original image and the rendered face, respectively. Here the term $\|\mathbf{I}(\mathbf{x}) - \mathbf{J}(\mathbf{x})\|_2 + \|\mathbf{I}(\mathbf{y}) - \mathbf{J}(\mathbf{y})\|_2$ indicates the consistency of color between the two images across the seam [28], with the weight $\alpha(\mathbf{x},\mathbf{y}) = \exp(\min(d_B(\mathbf{x}), d_B(\mathbf{y})))$ favoring a seam that has a similar shape as the optimized seam from the previous frame. Similar to [28], we solve the optimization problem via graph cut [3]. 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)).

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 [5], using the rendered face within the seam as the foreground. One example is shown in Fig. 2 (d). For Laplacian blending, we set the level of the pyramid to 5 in all our experiments. Fig. 6 shows more examples of the effectiveness of Laplacian blending in improving the appearance. A comparison between generated video sequences with and without Laplacian blending can be found in the supplementary video.

3.4 Face with Glasses

Using the characteristic vector **c** returned by the neural network in Section 3.1, 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. 9 (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 the inset) to align its 2D projection with the glasses area from the input frame. We prescribe 12 landmarks (shown in red in the inset) 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 the 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 its position relative to the face are usually fixed, for efficiency we only perform its reconstruction once at the beginning. In the following, we provide algorithmic details for each step.

3D Glasses Reconstruction From the input frame, we use U-Net [40] to segment the glasses, and ResNet-18 [18] 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 (see Fig. 7 for some examples from the training set).

Fig. 7 Examples of manually labeled landmarks and segmentation masks for glasses.

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 lenses and two at the hinges) with their corresponding detected landmarks. 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 in a non-rigid way to match the segmentation mask of the glasses. Specifically, we use the iterative solver from [2] 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. Fig. 8 (b) shows an example of 3D glasses shape reconstruction.

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 frame afterward, 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 in the same way as in Eq. (7). The texture information from the input frame is assigned to the visible regions of the face and the glasses and reused for their rendering. 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 become visible in the virtual camera view. One example is shown in Fig. 8 (c), where the virtual camera is placed above the real camera and exposes an occluded region from the real camera view. In this case, the exposed region appears in the virtual camera view as a gap without texture information, located between the top boundary of the glasses (shown in cyan) and the face region above the glasses that is visible

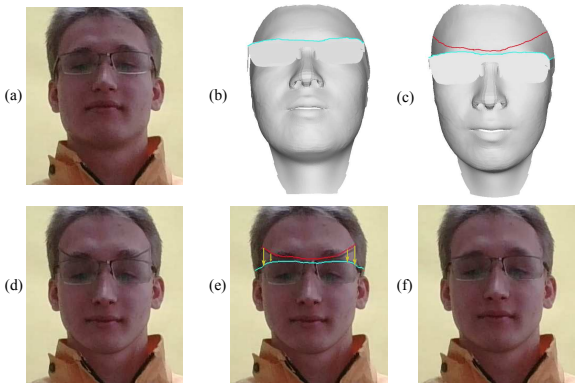


Fig. 8 One example of reconstruction and rendering of glasses. (a) The input face image. (b) The reconstructed face and glasses mesh from the real camera view. The top boundary of the glasses is shown in cyan. (c) The reconstructed face and glasses from the virtual camera view. The red curve shows the bottom boundary of the face region above the glasses that is visible from the real camera. (d) Directly rendering the face and the glasses from the virtual camera view produces an unnatural result, as the face region between the red and cyan curves in (c) is occluded in the real camera view and does not have reliable texture information. (e) To remove the texture gap between the 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.

Fig. 9 Correction results for users wearing glasses. The top row shows the input frames. The middle row shows the results without reconstructing the 3D shapes of the glasses. The glasses region in the input is assigned as the texture on the face model, resulting in a distortion from the virtual camera view. The bottom row shows the correction results using the reconstructed glasses shapes.

from the real camera (its bottom boundary is shown in red). To determine the texture of the exposed region for its rendering, we could potentially use image in-painting [15], 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 go through each column of its pixels and slide the whole column downwards vertically to until it meets the top of the glasses region (i.e., the cyan curve). This effectively fills up the gap while removing some pixels from the top of the face. Afterward, 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, similar to a face without glasses. In this way, the removed top region of the rendered face will be replaced by face pixels from the original frame, and the merging afterward produces a natural appearance in our experiments. Fig. 8 (f) shows an example of the final merged image. Fig. 9 further compares between correction results with and without the reconstruction of 3D glasses shapes, and clearly shows the benefit of such reconstruction for reducing the distortion of the glasses. More examples are available in the supplementary video.

3.5 Validity Judgment

When there is slight occlusion around the face boundary in the input frame (e.g., occlusion by hairs), some of the occluding object's color information may be assigned as the texture on the face model. In this case, seam optimization and Laplacian blending helps to create a natural transition between the occlusion texture and the part of the occluding object that lies outside the face region. On the other hand, 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 face in an input frame 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 Sec. 3.1, 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 compute its 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 is defined as a combination of two quantities: (1) the squared difference between the predicted face occlusion ratio and the ground-truth ratio; (2) a softmax classification loss for glasses. Fig. 10 shows examples of occlusion ratios predicted by our network.

Furthermore, we do not apply correction if the face pose

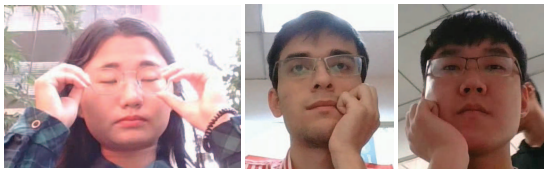


Fig. 10 Example images with occlusion. Our 3D face reconstruction network in Sec. 3.1 also outputs the face occlusion ratio for validity judgment. The predicted occlusion ratios of these three face images are 8.5%, 14.3% and 13.5%, respectively.

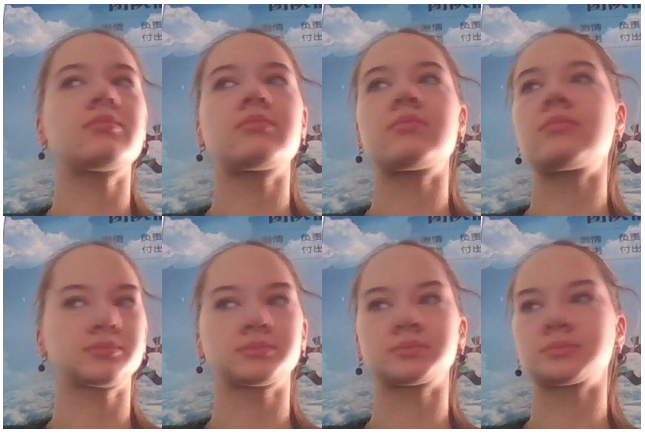


Fig. 11 Correction results for a user with horizontal head rotation. Top: the input frames. Bottom: the correction results.

captured by the real camera is too large. We check the rotation matrix $\mathbf{R}$ returned by the neural network for face reconstruction. If the magnitude of a rotation angle exceeds its 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 $\mathbf{R}_c$ in Eq. (7) by another rotation $\overline{\mathbf{R}}_c$, with $\overline{\mathbf{R}}_c$ transitioning from $\mathbf{R}_c$ to identity. Similarly, if the captured face rotation falls back to within the threshold, we gradually change the rotation $\overline{\mathbf{R}}_c$ back to $\mathbf{R}_c$ in the next four frames. In all our experiments, we set the thresholds for the yaw, the pitch and the roll to 20 degrees, 35 degrees and 14 degrees, respectively.

4 Results

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

Fig. 12 Correction results (bottom row) for a user quickly approaching the camera. The top row shows the input frames.

Efficiency Our system is fully automatic and runs in real-time, achieving 20fps for 1280×720 input videos with our un-optimized implementation. For an input frame, it typically takes 26 ms for 3D face reconstruction, 8 ms for re-rendering, and 13 ms for seam optimization and blending. 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 ms and 2 ms, respectively. As the glasses reconstruction is only performed once, it has little impact on the efficiency of the system.

Robustness We test the robustness of our system to different lighting conditions, poses, glasses, and the user's motion such as head rotation and movement. Some video results are included in the supplementary video. Fig. 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. Fig. 11 shows that our system can correctly handle horizontal rotations of the user's head. Fig. 12 shows an example where the user quickly approaches the camera, and our method is robust to such fast movements. Further examples can be found in and the supplementary video. We also evaluate our system on users wearing different types of glasses. As shown in Fig. 9 and the supplementary video, our system can correctly handle different glasses and produce natural-looking results.

Face Reconstruction Accuracy We evaluate the accuracy of our 3D face reconstruction network by conducting quantitative comparisons with state-of-the-art learning-based methods [8, 24, 46–48]. With the same setting

	Ours	[8]	[46]	[47]	[48]	[24]
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

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

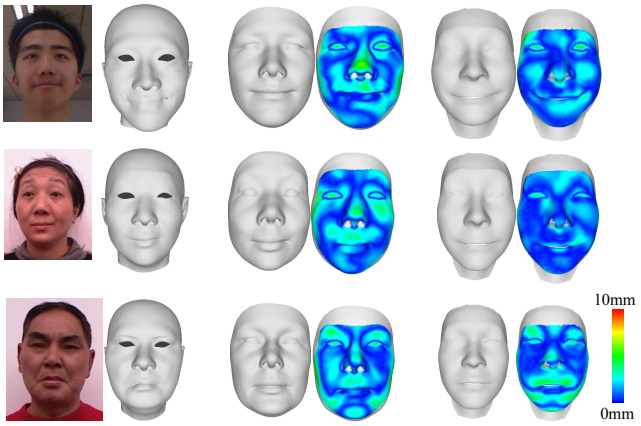
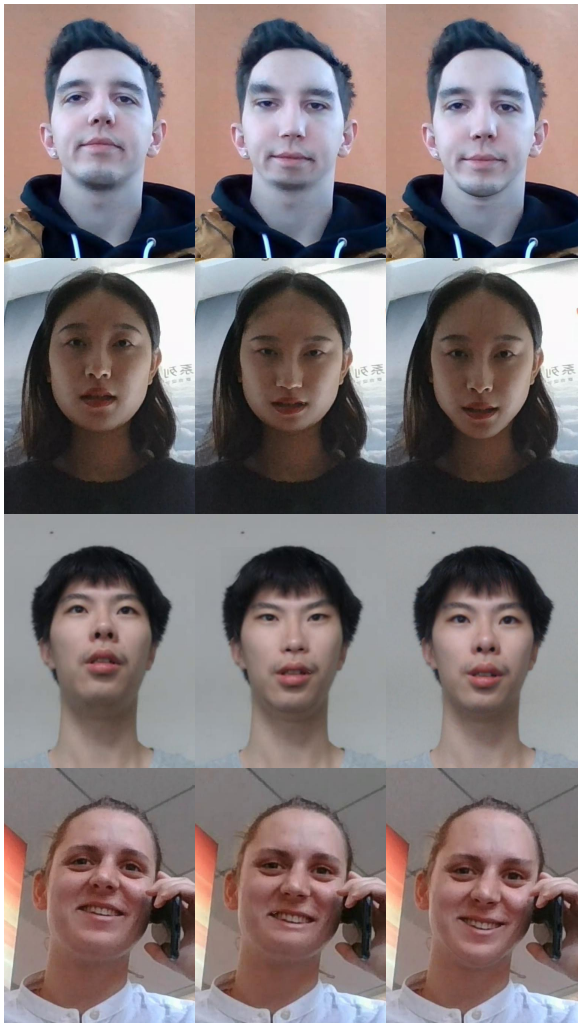


Fig. 13 Reconstruction results with RGB inputs (top row) on FaceWarehouse of the method of Deng et al. [8] and ours. From left to right: input images and ground-truth meshes, results of [8], results of our method.

as [47], we compare our results on 180 meshes of 9 subjects from FaceWarehouse. Following the evaluation protocol of [8], we compute the 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 Tab. 1. We also show the reconstruction results and error maps in Fig. 13. It can be seen that our method outperforms the method of Deng et al. [8] in terms of shape and expression reconstruction.

Visual Quality We conduct a user study on 50 participants to evaluate the visual quality of the results from our system. We collect 10 video sequences covering different scenarios, including indoor and outdoor scenes, different light conditions, for subjects of different skin tones, with and without glasses. Each participant watches the original videos and the corrected videos and is asked to rate their satisfaction to the results in three aspects: how natural the results look, consistency of the facial appearances between the two videos, and the accuracy of view correction. Each aspect is rated with a score between 1 and 5, with 1 being the worst and 5 being the best. The average scores for the three aspects are 4.35, 4.25 and 4.41, respectively, demonstrating that our method can generate a natural-looking corrected view of faces.

Comparisons with State-of-the-Art Method A method particularly relevant to our work is the webcam-based



(a) Input (b) [Giger et al. 2014] (c) Ours

Fig. 14 Comparisons between [13] and our method. The method from [13] does not utilize the shape priors of human faces and may produce unnatural results, as shown in the first row.

approach from [13]. For a fair comparison, we only replace our 3D face reconstruction component with the deformation based method in their paper, while all other steps remain unchanged. Some comparison examples are shown in Fig. 14 and in the accompanying video. We can observe that our method produces more natural results due to our learning-based 3D face reconstruction that utilizes the face shape priors and our landmark update strategies that correctly handles large discrepancy between the real camera and the virtual camera. In comparison, the method from [13] only deforms a 3D face template to match the detected 2D landmarks, and the lack of facial shape priors can lead to distortions. Besides, the correspondences between 2D and 3D landmarks are fixed in their method to allow pre-factorization of the deformation matrix for fast computation, which may introduce errors due to large pose changes.

Although it is possible to improve the accuracy for [13] by adopting a landmark update strategy similar to ours, this would result in a deformation matrix that may need to be frequently re-factorized and increase the computational cost. Finally, unlike our method, the deformation approach [13] does not take the glasses' shape into account and may produce unnatural results, as stated in the limitation part of their paper.

For qualitative and quantitative comparison, we also compare the results of the two methods with the video captured using a camera at the same location as the virtual camera. We first place a webcam at a position well below the face to capture videos as input for the correction algorithms, with the virtual camera located in front of the user's face (see Fig. 15 for the setup using our correction method). Then we place a webcam of the same model at the location of the virtual camera to capture a frontal reference video simultaneously for comparison. We collect 10 pairs of input and reference video sequences using this setup and apply our method and [13] to correct the input video. Fig. 16 shows some examples of the frontal reference videos together with the correction results using the two methods. We can see that our method produces more natural results with appearances closer to the reference videos. For further verification, we evaluate 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 take the corresponding frame in the reference video as well as the corrected frames using the two methods and evaluate their facial recognition feature according to [41]. We then compute 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 repeat this process for all frames of all 10 input videos and compute the average cosine distance for each method. Our method achieves an average value 0.91, whereas the average value for [13] is 0.84, indicating that our results have higher perceptual similarity with the reference video. We also conduct a user study on the same 50 participants mentioned previously to compare the visual quality of results from our method and [13]. Each participant is shown the 10 pairs of corrected videos and is asked to choose the better one among each pair. For a fair comparison, each participant is first shown the input video, followed by the two corrected videos in random order, without information about the correction method used for each video. Overall, our result is chosen to be the better one in 89.6% of the pairs, showing that our method produces more visually convincing results.

We also compare our method with one face reenactment method [49]. Although the method is not directed targeted for face view correction, the pipeline could be adopted for this task. We show the qualitative comparison results in Fig. 17. It can be seen that our method produces more frontal facial images and more natural glasses correction.

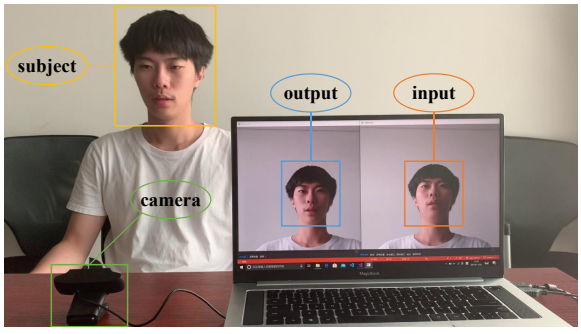


Fig. 15 A setup of our system for correcting videos captured from a webcam well below the face. The real-time results are shown in the accompanying video.

(a) Input (b) Frontal (c) [Giger et al. 2014] (d) Ours

Fig. 16 Comparisons between captured real frontal videos, [13] and our method. Our results are closer to the real frontal videos.

4.1 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 might 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

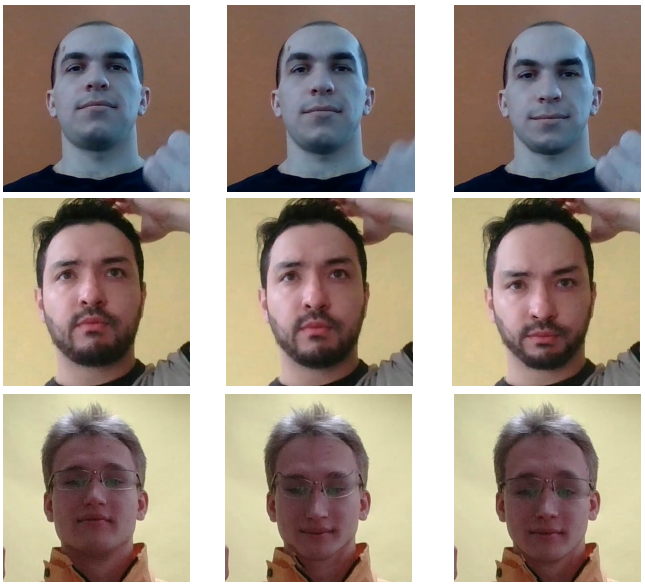


Fig. 17 Comparisons between our method and [49]. From left to right: input images, correction results of [49], correction results of our method.

from the real camera to render glasses for the virtual camera view, which does not correctly capture the optical effects of the lenses such as refraction. One possible solution is to further refine the corrected results by generative adversarial network.

5 Conclusion

We 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 the 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 of video conferencing applications.

References

[1] V. Blanz and T. Vetter. A morphable model for the synthesis of 3D faces. In *Proceedings of SIGGRAPH*, pages 187–194, 1999. 2, 4

[2] S. Bouaziz, M. Deuss, Y. Schwartzburg, T. Weise, and M. Pauly. Shape-up: Shaping discrete geometry with projections. *Comput. Graph. Forum*, 31(5):1657–1667, 2012. 7

[3] Y. Boykov and V. Kolmogorov. An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision. *IEEE TPAMI*, 26:1124–1137, 2004. 6

[4] A. Bulat and G. Tzimiropoulos. How far are we from solving the 2d & 3d face alignment problem?(and a dataset of 230,000 3d facial landmarks). In *ICCV*, pages 1021–1030, 2017. 4

[5] P. J. Burt and E. H. Adelson. A multiresolution spline with application to image mosaics. *ACM Transactions on Graphics (TOG)*, 2(4):217–236, 1983. 6, 7

[6] C. Cao, Q. Hou, and K. Zhou. Displaced dynamic expression regression for real-time facial tracking and animation. *ACM Trans. Graph.*, 33(4):43, 2014. 3

[7] C. Cao, Y. Weng, S. Zhou, Y. Tong, and K. Zhou. Facewarehouse: A 3d facial expression database for visual computing. *IEEE TVCG*, 20(3):413–425, 2013. 2, 4

[8] Y. Deng, J. Yang, S. Xu, D. Chen, Y. Jia, and X. Tong. Accurate 3d face reconstruction with weakly-supervised learning: From single image to image set. In *IEEE CVPR Workshops*, 2019. 2, 3, 4, 9, 10

[9] O. Fried, E. Shechtman, D. B. Goldman, and A. Finkelstein. Perspective-aware manipulation of portrait photos. *ACM Trans. Graph.*, 35(4):128:1–128:10, 2016. 2, 3

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

[11] B. Gecer, S. Ploumpis, I. Kotsia, and S. Zafeiriou. GANFIT: generative adversarial network fitting for high fidelity 3D face reconstruction. In *IEEE CVPR*, pages 1155–1164, 2019. 2

[12] K. Genova, F. Cole, A. Maschinot, A. Sarna, D. Vlasic, and W. T. Freeman. Unsupervised training for 3D morphable model regression. In *IEEE CVPR*, pages 8377–8386, 2018. 2, 3

[13] D. Giger, J. C. Bazin, C. Kuster, T. Popa, and M. H. Gross. Gaze correction with a single webcam. In *IEEE ICME*, pages 1–6, 2014. 2, 3, 4, 10, 11

[14] D. M. Grayson and A. F. Monk. Are you looking at me? eye contact and desktop video conferencing. *ACM Trans. Comput.-Hum. Interact.*, 10(3):221–243, 2003. 1

[15] C. Guillemot and O. Le Meur. Image inpainting : Overview and recent advances. *IEEE Signal Processing Magazine*, 31(1):127–144, 2014. 8

[16] Y. Guo, J. Zhang, J. Cai, B. Jiang, and J. Zheng. CNN-based real-time dense face reconstruction with inverse-rendered photo-realistic face images. *IEEE TPAMI*, 41(6):1294–1307, 2019. 2

[17] T. Hassner, S. Harel, E. Paz, and R. Enbar. Effective face frontalization in unconstrained images. In *IEEE CVPR*, pages 4295–4304, 2015. 3

[18] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *IEEE CVPR*, pages 770–778, 2016. 4, 7

[19] Z. He, A. Spurr, X. Zhang, and O. Hilliges. Photo-realistic monocular gaze redirection using generative adversarial networks. In *ICCV*, October 2019. 2, 3

[20] C.-F. Hsu, Y.-S. Wang, C.-L. Lei, and K.-T. Chen. Look at me! correcting eye gaze in live video communication. *ACM Trans. Multimedia Comput. Commun. Appl.*, 15(2), 2019. 2, 3

[21] H. Ishii and M. Kobayashi. Clearboard: a seamless medium

for shared drawing and conversation with eye contact. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 525–532. ACM, 1992. 2

[22] A. S. Jackson, A. Bulat, V. Argyriou, and G. Tzimiropoulos. Large pose 3D face reconstruction from a single image via direct volumetric CNN regression. In *ICCV*, pages 1031–1039, 2017. 2

[23] L. Jiang, J. Zhang, B. Deng, H. Li, and L. Liu. 3d face reconstruction with geometry details from a single image. *IEEE TIP*, 27(10):4756–4770, 2018. 2

[24] H. Kim, M. Zollhöfer, A. Tewari, J. Thies, C. Richardt, and C. Theobalt. InverseFaceNet: Deep monocular inverse face rendering. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2018. 9, 10

[25] D. Kononenko and V. Lempitsky. Learning to look up: Realtime monocular gaze correction using machine learning. In *IEEE CVPR*, pages 4667–4675, 2015. 2, 3

[26] C. Kuster, T. Popa, J. C. Bazin, C. Gotsman, and M. H. Gross. Gaze correction for home video conferencing. *ACM Trans. Graph.*, 31(6):174:1–174:6, 2012. 2, 3

[27] C. Kuster, T. Popa, C. Zach, C. Gotsman, and M. H. Gross. Freecam: A hybrid camera system for interactive free-viewpoint video. In *Vision, Modeling, and Visualization (2011)*, pages 17–24, 2011. 2

[28] V. Kwatra, A. Schödl, I. A. Essa, G. Turk, and A. F. Bobick. Graphcut textures: image and video synthesis using graph cuts. *ACM Trans. Graph.*, 22(3):277–286, 2003. 6

[29] T. Li, T. Bolkart, M. J. Black, H. Li, and J. Romero. Learning a model of facial shape and expression from 4D scans. *ACM Trans. Graph.*, 36(6), 2017. 2

[30] W. Matusik, C. Buehler, R. Raskar, S. J. Gortler, and L. McMillan. Image-based visual hulls. In *Proceedings of SIGGRAPH*, pages 369–374, 2000. 2

[31] W. Matusik and H. Pfister. 3d TV: a scalable system for real-time acquisition, transmission, and autostereoscopic display of dynamic scenes. *ACM Trans. Graph.*, 23(3):814–824, 2004. 2

[32] A. F. Monk and C. Gale. A look is worth a thousand words: Full gaze awareness in video-mediated conversation. *Discourse Processes*, 33(3):257–278, 2002. 1

[33] N. Mukawa, T. Oka, K. Arai, and M. Yuasa. What is connected by mutual gaze?: User's behavior in video-mediated communication. In *CHI '05 Extended Abstracts on Human Factors in Computing Systems*, pages 1677–1680, 2005. 1

[34] C. Müller. Spherical harmonics. In *Lecture Notes in Mathematics*, volume 17, 1966. 4

[35] K. Nagano, H. Luo, Z. Wang, J. Seo, J. Xing, L. Hu, L. Wei, and H. Li. Deep face normalization. *ACM Trans. Graph.*, 38(6):183:1–183:16, 2019. 2, 3

[36] K. Okada, F. Maeda, Y. Ichikawa, and Y. Matsushita. Multiparty videoconferencing at virtual social distance: MAJIC design. In *Proceedings of the 1994 ACM Conference on Computer Supported Cooperative Work*, pages 385–393, 1994. 2

[37] P. Paysan, R. Knothe, B. Amberg, S. Romdhani, and T. Vetter. A 3d face model for pose and illumination invariant face recognition. In *IEEE International Conference on Advanced video and signal based surveillance*, pages 296–301, 2009. 4

[38] E. Richardson, M. Sela, and R. Kimmel. 3D face reconstruction by learning from synthetic data. In *Fourth International Conference on 3D Vision (3DV 2016)*, pages 460–469, 2016. 2

[39] E. Richardson, M. Sela, R. Or-El, and R. Kimmel. Learning detailed face reconstruction from a single image. In *IEEE CVPR*, pages 5553–5562, 2017. 2

[40] O. Ronneberger, P. Fischer, and T. Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241, 2015. 7

[41] F. Schroff, D. Kalenichenko, and J. Philbin. Facenet: A unified embedding for face recognition and clustering. In *IEEE CVPR*, pages 815–823, 2015. 11

[42] M. Sela, E. Richardson, and R. Kimmel. Unrestricted facial geometry reconstruction using image-to-image translation. In *ICCV*, pages 1585–1594, 2017. 2

[43] Z. Shu, S. Hadap, E. Shechtman, K. Sunkavalli, S. Paris, and D. Samaras. Portrait lighting transfer using a mass transport approach. *ACM Trans. Graph.*, 37(2), 2017. 2

[44] R. W. Sumner and J. Popović. Deformation transfer for triangle meshes. *ACM Trans. Graph.*, 23(3):399–405, 2004. 4

[45] A. Tewari, F. Bernard, P. Garrido, G. Bharaj, M. Elgharib, H. Seidel, P. Pérez, M. Zollhöfer, and C. Theobalt. FML: face model learning from videos. In *IEEE CVPR*, pages 10812–10822, 2019. 2, 3

[46] A. Tewari, F. Bernard, P. Garrido, G. Bharaj, M. Elgharib, H.-P. Seidel, P. Pérez, M. Zöllhofer, and C. Theobalt. Fml: Face model learning from videos. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019. 9, 10

[47] A. Tewari, M. Zollhöfer, P. Garrido, F. Bernard, H. Kim, P. Pérez, and C. Theobalt. Self-supervised multi-level face model learning for monocular reconstruction at over 250 hz. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2018. 10

[48] A. Tewari, M. Zollhöfer, H. Kim, P. Garrido, F. Bernard, P. Pérez, and C. Theobalt. Mofa: Model-based deep convolutional face autoencoder for unsupervised monocular reconstruction. In *ICCV*, pages 3735–3744, 2017. 2, 3, 4, 9, 10

[49] J. Thies, M. Zollhofer, M. Stamminger, C. Theobalt, and M. Nießner. Face2Face: Real-time face capture and reenactment of RGB videos. In *IEEE CVPR*, pages 2387–2395, 2016. 2, 3, 11, 12

[50] L. Tran and X. Liu. Nonlinear 3D face morphable model. In *IEEE CVPR*, pages 7346–7355, 2018. 2, 3

[51] Y. Yin, S. Jiang, J. P. Robinson, and Y. Fu. Dual-attention gan for large-pose face frontalization. In *IEEE International Conference on Automatic Face and Gesture Recognition*, 2020. 3

[52] D. Zhai, X. Liu, X. Ji, D. Zhao, and W. Gao. Joint gaze correction and face beautification for conference video using dual sparsity prior. *IEEE Transactions on Industrial Electronics*, 66(12):9601–9611, 2019. 3

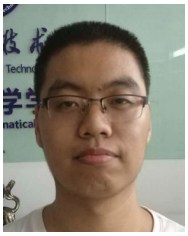
[53] J. Zhang, J. Chen, H. Tang, W. Wang, Y. Yan, E. Sangineto, and N. Sebe. Dual in-painting model for unsupervised gaze

correction and animation in the wild. In *ACM Multimedia*, 2020. 2, 3

[54] Y. Zhao, Z. Huang, T. Li, W. Chen, C. LeGendre, X. Ren, J. Xing, A. Shapiro, and H. Li. Learning perspective undistortion of portraits. In *ICCV*, 2019. 3

[55] X. Zhu, Z. Lei, X. Liu, H. Shi, and S. Z. Li. Face alignment across large poses: A 3D solution. In *IEEE CVPR*, pages 146–155, 2016. 2

[56] M. Zollhöfer, J. Thies, P. Garrido, D. Bradley, T. Beeler, P. Pérez, M. Stamminger, M. Nießner, and C. Theobalt. State of the art on monocular 3D face reconstruction, tracking, and applications. In *Computer Graphics Forum*, volume 37, pages 523–550. Wiley Online Library, 2018. 2



Yudong Guo is a PhD student at the School of Mathematical Sciences, University of Science and Technology of China. 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 University of Science and Technology of China. He received the BS degree from the University of Science and Technology of China in 2006, and the PhD 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 first-year Master student at the School of Mathematical Sciences, University of Science and Technology of China. He received the bachelor's degree in Mathematical Sciences from the University of Science and Technology of China, in 2020. His research interests include Computer Vision and Computer Graphics.

Hongrui Cai is a second-year Master student at the School of Data Science, University of Science and Technology of China. 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 the B.S. and Ph.D. degrees in computational mathematics from University of Science and Technology of China (USTC), Hefei, China, 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 and Informatics at Cardiff University. He received the BEng degree in computer software (2005) and the MSc degree in computer science (2008) from Tsinghua University (China), and the PhD 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.