STEADIFACE: REAL-TIME FACE-CENTRIC STABILIZATION ON MOBILE PHONES

Fuhao Shi, Sung-Fang Tsai, Youyou Wang, and Chia-Kai Liang

Google Inc.

ABSTRACT

We present *Steadiface*, a new real-time face-centric video stabilization method that simultaneously removes hand shake and keeps subject's head stable. We use a CNN to estimate the face landmarks and use them to optimize a stabilized head center. We then formulate an optimization problem to find a virtual camera pose that locates the face to the stabilized head center while retains smooth rotation and translation transitions across frames. We test the proposed method on fieldtest videos and show it stabilizes both the head motion and background. It is robust to large head pose, occlusion, facial appearance variations, and different kinds of camera motions. We show our method advances the state of art in selfie video stabilization by comparing against alternative methods. The whole process runs very efficiently on a modern mobile phone (8.1 ms/frame).

***Index Terms*—** video stabilization, real-time processing, mobile platforms, machine learning, CNN

1. INTRODUCTION

Stable appearance of human faces is crucial for selfie videos, live shows, or vlogging, which all are now popular features on mobile phones. However, unintentional hand shakes and head translations during recording can easily make the face unstable. Unfortunately, most existing video stabilization methods do not stabilize selfie videos well. The face motion, which is usually very different from the background, can reamian unstable after stabilization.

We present *Steadiface*, a real-time gyro-based face-centric video stabilization method that simultaneously removes hand shake and keeps head stable. Moreover, it runs real-time on the mobile phone without delaying the video stream, and provides the WYSIWYG user experience. The proposed method uses the gyroscope to obtain the camera motions and an efficient CNN to extract the face landmarks. We then formulate an optimization problem to jointly stabilize the camera and head motion. We also dynamically control the weighting in the optimization process for robustness.

Steadiface is highly efficient: it takes only 8.1 ms/frame on Google Pixel 3 (Qualcomm Snapdragon 845 CPU, Adreno 630 GPU). We tested it on many videos with challenging head poses, occlusions, and camera motions, and obtained good results in all cases [1].

This work has made the following contributions:

- The first real-time end-to-end stabilization system that simultaneously stabilizes both head and camera motion.
- A novel algorithm that combines both face and gyro stableness into a single objective function for joint optimization.
- An effective weight adjustment scheme robust to head pose variance and noisy landmark locations.
- We perform extensive comparisons between our method and the state-of-the-art ones.

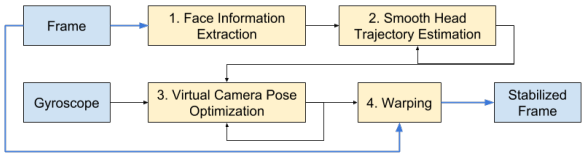


Fig. 1. *Steadiface* algorithm pipeline.

2. BACKGROUND

The electronic video stabilization systems usually consist of three components: motion estimation, motion compensation and image composition [2]. There are two popular motion estimation methods: image based and sensor based. Given the estimated motion profiles, motion compensation creates a smooth motion via filtering [3] or optimization [4], and image composition adjusts the input video into a stabilized one via shifting or warping. Modern methods also handle rolling shutter distortions during warping [5, 4, 3].

There are also non-electronic video stabilization methods, such as mechanical gimbal or the optical image stabilizer (OIS). However, they do no work for face centric videos and we skip them here for brevity.

Sensor-based stabilization methods use gyroscope, OIS or their combinations to model the camera model as rotation and translation, and stabilize the videos by smoothing the virtual pose changes [3, 6, 7]. Our work is mostly related to the fused video stabilization as used in Google Pixel 3 [7]. Similarly, we extract the gyroscope signal to integrate as the camera pose, and warp the frame by dividing the input frame into a mesh and warp each part separately to handle the rolling shutter distortion [5]. Our key novelty is to detect faces and tracks the facial landmarks from the input frame, and fuse both gyro and face information to estimate the best joint face and background stability.

Image-based stabilization methods detect/track the features across video frames, and stabilize the motion by smoothing the camera path [8, 9, 10, 11]. As the feature tracking is noisy or camera motion estimation can be difficult in degenerated cases, most methods focus on improving the robustness. However, they are not designed to handle dominant moving subjects like faces.

Yu and Ramamoorthi proposed an image-based method for face-centric stabilization [12]. The head motion is modeled as the 3D head center of the reconstructed head, and the background motion is tracked by dense optical flow. An optimization for homography and additional mesh adjustment is performed to obtain the best trade-off between face and background stability. However, their approach is slow and sensitive to motion estimation errors. The required 3D face reconstruction, despite the advances from 3D fitting using 2D landmarks [13, 14, 15, 16] to direct regression/CNN inference [17, 18, 19], can be either slow or power-consuming for the video recording task on a mobile phone.

Our method is different from theirs in four parts. First, we use 2D facial features to represent the head motion without expensive 3D head

reconstruction. Second, we use gyroscope to obtain the camera motion and do not require feature tracking or optical flow. Third, we use a two-step process to explicitly control the head stability and avoid error accumulations. We first estimate a stabilized head trajectory and then optimize the virtual camera pose to align the face to that trajectory. Finally, we use a novel metric for head and background stability with dynamic weight adjustment to handle pose variations and noisy landmarks.

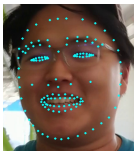
3. REAL-TIME VIDEO STABILIZATION

Steadiface takes the input video frame as well as gyroscope readout as inputs, and outputs a warping mesh that warps the original video frame to the stabilized result. The overall pipeline is shown in Fig. 1. First, face information is extracted from the input video frame, including face bounding box and 2D facial landmarks. We then estimate a smooth target head trajectory from the landmark locations. A joint optimization then takes the gyroscope, face information, and head trajectory to find the optimal vitrual camera pose. Finally the warping is aplied to transform the input frame to the stabilized one.

All processings are performed at each frame sequentially, and in the following discussions, we will drop the frame index or time value for brevity. We will describe the first two steps in this section.

3.1. Face Information Extraction

This step takes the video frame as input, detects the face bounding boxes, select the best face to process, and then infers its facial landmarks.



We get the face bounding box from the face detection (several popular ones work equally well in our experiments), and then feed the cropped face into a CNN to obtain the dense landmarks . The network architecture is a variant of the mobile net [20] which takes $192 \times 192 \times 3$ input image and returns 133×2 2D facial landmark coordinates (see inset). Our GPU implementation can perform at $5.4ms$ per face on Snapdragon 845.

Face of Interest Selection Due to the time budget, we only select one face to stabilize. We use a simple yet effective method: for the first frame that contains face(s), we use the largest one returned by the face detection module. We then keep tracking and stabilizing this face until it is lost.

3.2. Smooth Head Trajectory Estimation

In this section, we describe how to obtain a smooth head trajectory from the 2D landmarks, which will be the target head center we want to put the face to. This smoothing step is critical as some landmarks can have flickering or gross errors. We model this as an optimization that keeps the head as stable as possible while not causing the virtual frame to move beyond the real frame domain. The objective function is defined as

$$\begin{aligned} &\text{argmin}_{\mathbf{H}} w_1 \|\mathbf{H} - \mathbf{H}_{-1}\|_2^2 + w_2 \max(|\mathbf{H} - \mathbf{C}|_{x,y})/d_{ref})^2, \\ &s.t. |\mathbf{H} - \mathbf{C}|_{x,y} < r, \end{aligned} \tag{1}$$

where $\mathbf{H}$ is the target 2D center on the stabilized virtual frame domain, $\mathbf{H}_{-1}$ is the head center of the previous frame, $\mathbf{C} = 1/N \sum_i \mathbf{L}_i$ is the 2D landmark center over the landmarks $\{\mathbf{L}_1, ..., \mathbf{L}_N\}$, and d_{ref} is a reference deviation that we can tolerant. If the target head center is not too far away from the real center, the second term would be small. Thus, $\mathbf{H}$ will tend to be stable as its previous location. Otherwise, the second term will produce a large penalty and force $\mathbf{H}$ to follow the real landmark center $\mathbf{C}$. r is the cropping ratio at each side of the frame after stabilization.

4. VIRTUAL CAMERA POSE OPTIMIZATION

With the extracted face information $\{\mathbf{L}_i\}$ and the target head center $\mathbf{H}$, we now describe how to estimate the virtual camera pose so that it fits the face center to the target one, and meanwhile keeps the virtual camera pose changing smoothly across frames.

4.1. Representation

Unlike previous images-based methods that use free-form transformations [4, 11], we restrict the stabilized camera to have a valid rotation and a shifted projection. This approach greatly reduces the degree of freedom in optimization and improves the robustness. We represent the virtual camera pose as a set of 3D rotation and 2D translation:

$$\mathcal{P}_v = \{\mathbf{r}_v, \mathbf{t}\}, \tag{2}$$

where $\mathbf{r}_v$ is rotation represented by quaternion and $\mathbf{t} = [t_x, t_y]^T$ is 2D principal offset to the projection center. Note that the pose, rotation and translation are all functions of time, and we drop the time index for brevity.

The virtual camera intrinsic matrix is $\mathbf{K}_v = [f_v, 0, 0.5 + t_x; 0, f_v, 0.5 + t_y; 0, 0, 1]$, where f_v is the virtual focal length, which is manually chosen and fixed in our system.

We represent the real camera pose and intrinsic matrix in a similar way: $\mathcal{P}_r = \{\mathbf{r}_r, 0\}$ and $\mathbf{K}_r = [f_r, 0, 0.5; 0, f_r, 0.5; 0, 0, 1]$. The real camera does not have principal point shift, and focal length f_r is obtained by calibration. $\mathbf{r}_r$ is the integration of the angular velocity signal from the gyroscope [3].

Given $\mathbf{K}_r$ and $\mathbf{r}_r$ of the current frame, the projection of an image point from the real camera domain to virtual camera domain is decided by a homography transform:

$$\mathbf{\Pi}(\mathcal{P}_v, \mathcal{P}_r) = \mathbf{K}_v \mathbf{R}_v (\mathbf{K}_r \mathbf{R}_r)^{-1}. \tag{3}$$

where $\mathbf{R}$ is the matrix form of the rotation $\mathbf{r}$. The projection of a 2D facial landmark $\mathbf{L}$ from the input image to the stabilized virtual domain is

$$\texttt{proj}(\mathbf{L}, \mathcal{P}_v, \mathcal{P}_r) = \mathbf{\Pi} \cdot [\mathbf{L}_x, \mathbf{L}_y, 1]^T. \tag{4}$$

Note that if $\mathbf{t} = 0$, this transform would only work for objects sufficiently far away from the camera. The additional principal offset enables us to properly transform close subject like faces.

4.2. Objective Function

The goal of stabilization is to find the optimal virtual camera pose $\mathcal{P}_v$ at each frame. For real-time viewfinder and streaming applications, we also want to calculate these values without relying on future (non-casual) information. We cast this process as an optimization problem to minimize the following objective function:

$$\begin{aligned} \text{argmin}_{\mathcal{P}_v} w_f E_f(\mathcal{P}_v) + w_d E_d(\mathbf{r}_v) + w_o E_o(\mathbf{r}_v) + \\ w_r E_r(\mathbf{r}_v) + w_t E_t(\mathbf{t}) + w_p E_p(\mathcal{P}_v). \end{aligned} \tag{5}$$

The fixed inputs, such as $\mathcal{P}_r$, $\{\mathbf{L}_i\}$ and $\mathbf{H}$, are skipped in the argument for brevity.

The *landmark fitting term* E_f measures the fitting error of the projected landmarks to the target center:

$$E_f(\mathcal{P}_v) = \sum_i \|\texttt{proj}(\mathbf{L}_i, \mathcal{P}_v, \mathcal{P}_r) - \mathbf{H}\|_2^2. \tag{6}$$

The *distortion term* E_d measures the spherical angle Ω between $\mathbf{r}_v$ and $\mathbf{r}_r$:

$$E_d(\mathbf{r}_v) = (\texttt{logistic}(\Omega(\mathbf{r}_v, \mathbf{r}_r)) \cdot \Omega(\mathbf{r}_v, \mathbf{r}_r))^2. \quad (7)$$

A logistic regression function is applied here so that the penalty is close to zero when Ω is smaller than a threshold, and increases when Ω becomes large. In other words, this term tolerants the virtual-real camera pose difference within a threshold, and creates large penalty after the difference is further increased.

The *rotation following term* E_o measures how the virtual camera follows the real camera. Unlike the distortion term above, it consistently puts a penalty if the virtual camera rotation is different from the real camera rotation. The goal is to reduce the change of hitting boundary due to the virtual camera being too stable.

$$E_o(\mathbf{r}_v) = \|\mathbf{r}_v - \mathbf{r}_r\|_2^2. \quad (8)$$

The *rotation smoothness term* E_r measures how smooth virtual rotation changes across frames. It consists of two terms, which controls the C0 and C1 smoothness.

$$\begin{aligned} E_r(\mathbf{r}_v) = w_{r,C0}\|\mathbf{r}_v - \mathbf{r}_{v,-1}\|_2^2 \\ + w_{r,C1}\|\mathbf{r}_v\mathbf{r}_{v,-1}^{-1} - \mathbf{r}_{v,-1}\mathbf{r}_{v,-2}^{-1}\|_2^2, \end{aligned} \quad (9)$$

where the subscript $_{-1}$ and $_{-2}$ denote values from the previous and previous-previous frames, respectively.

Similarly, the *translation smoothness term* E_t measures how smooth the principal offset changes across frames, which are

$$E_t(\mathbf{t}) = w_{t,C0}\|\mathbf{t} - \mathbf{t}_{-1}\|_2^2 + w_{t,C1}\|2\mathbf{t}_{-1} - (\mathbf{t} + \mathbf{t}_{-2})\|_2^2. \quad (10)$$

Finally, the *protrusion term* E_p measures how the warped frame protrudes the real image boundary:

$$E_p(\mathcal{P}_v) = \|\texttt{protrude}(\mathcal{P}_v, \mathcal{P}_r)/\alpha\|_2^2, \quad (11)$$

where $\texttt{protrude}(\mathcal{P}_v, \mathcal{P}_r)$ is the amount that the warped frame protrudes the real image boundary (we actually make it more sensitive by shrinking the real image boundary to a smaller bounding box), and α is a reference protrusion value we can tolerate (see Fig. 2). This concept was introduced for post-processing in [6], and here we combine it into the joint optimization process.

4.3. Optimization

The objective function can be effectively solved by non-linear least square solver such as Ceres [21]. For the first frame, we initialize the virtual camera pose to identify rotation and zero offset. For the following frames, we initialize $\mathbf{r}_v$ by applying the real camera rotation between current and previous frames to the previous virtual camera rotation, and $\mathbf{t}_v$ with the previous virtual principal offset:

$$\mathcal{P}_v = \{(\mathbf{r}_r\mathbf{r}_{r,-1}^{-1})\mathbf{r}_{v,-1}, \mathbf{t}_{-1}\}. \quad (12)$$

The optimization usually converges within 3 iterations and runs at 2.7ms/frame on Snapdragon 845.

4.4. Dynamic Optimization Weight Adjustments

Up to now, we are able to stabilize the face motion based on 2D landmarks and gyro. However, there are three practical issues remained. First, when camera is relatively stable, users are expecting a stable virtual camera. However, face translations will move the virtual camera around as it follows the face. Second, the virtual camera will move when user is simply rotating the head. This is because the 2D landmark center is not pose-invariant. Finally, the landmarks can be noisy especially when the face rotates away from the camera. The optimized virtual camera pose will jitter in these cases.

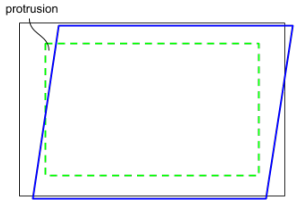


Fig. 2. Protrusion visualization. A protrusion is the amount that the warped video frame (blue) protrudes the the stabilized frame (green) which is cropped from the full frame (black).

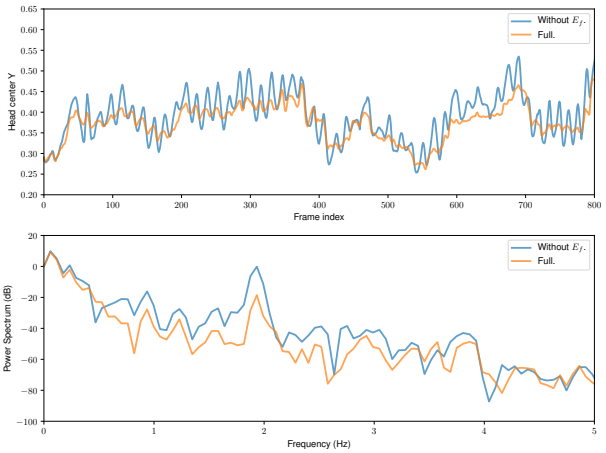


Fig. 3. 2D head center trajectory and the power spectrum with/without the fitting term E_f.

To address these challenges, we dynamically adjust the optimization weights based on gyro and landmarks. First, we examine the mean magnitude of angular velocity over a period, and adjust w_f proportionally. Next, we decrease the fitting weight and increase the smoothness weights when face pose is large. Finally, we check the variations of landmark center and scales and decrease the fitting weight if they are large. As a result, the final virtual camera motion does not move with face when camera is stable or head pose is fast changing, and robust to noisy landmark locations.

Protrusion Handling The final stabilized frame is a center crop from the warped frame (Fig. 2), and any protruded area would be undefined. In rare cases, protrusions can still occur, and we eliminate them by binary searching between $\mathcal{P}_v$ and $\mathcal{P}_r$. If the binary search failed (no valid solution), we apply the previous warping directly to the current frame.

5. RESULTS

In this section, we first demonstrate the effectiveness of our method on videos with a variety of head and hand motions. We show our method stabilizes both the head motion and the background, and is robust to large head pose variations (e.g. $v0$, $v7$ in the accompanying video [1]), illumination changes (e.g. $v0$, $v5$) and occlusions (e.g. $v1$, $v2$). We then validate the method by evaluating the importance of each term. Next, we compare our method with the state-of-the-art gyro-based stabilization on mobile phone [7]. Finally, we compare our method with the state-of-the-art selfie video stabilization method [12]. Our results show comparable or better face stability, and do not suffer from artifacts caused by unreliable optical flow/landmarks.

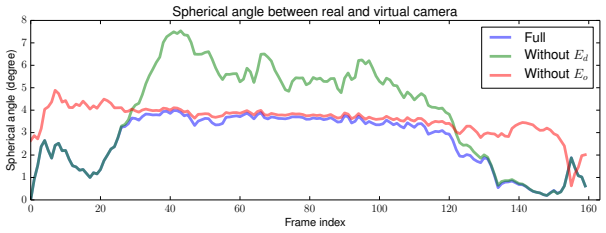


Fig. 4. Virtual camera deviation with/without the rotation following (E_o) or the distortion (E_d) terms.

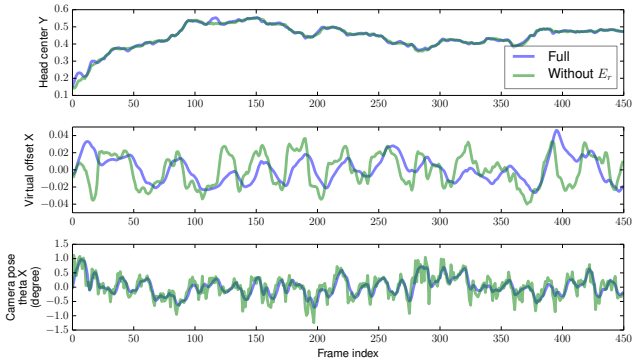


Fig. 5. Virtual camera rotation and principal point offset with/without smoothness terms. With those terms, the camera (rotation and translation) becomes smoother while the head center (top) remains stable. This shows that both the head and background are properly stabilized.

We tested our method on 43 videos with a combination of different head motions, expressions (e.g. talking to camera, looking around) and hand motions (e.g. tripod, walking, panning). All results are generated with the identical parameter set, and they are best seen in the accompanying video at [1].

5.1. Importance of Each Term

We show the importance of each term by disabling them during the optimization. Fig. 3 shows the head trajectory and power spectrum with/without the fitting term E_f. As we can see, the head remains unstable when stabilizing using only gyro, and the fitting term effectively reduces the head motion and produces a lower power density over frequencies. Note that the stabilized trajectory is not perfectly smooth as it is a trade-off between the fitting term and other terms.

Fig. 4 shows the deviation between the virtual camera pose and the real camera pose during fast panning. Clearly, the rotation following term E_o makes the virtual pose well defined when multiple solutions exist and the solution that is close to the real pose is selected. Meanwhile, the distortion term E_d further refines the solution space by adaptively imposing penaulty when the real-virtual rotation deviation tends to be large.

Finally, the Fig. 5 shows the rotation and principal offset curves with/without the smoothness terms which demonstrates their necessity for balancing the head and background stability.

5.2. Comparison to State-of-the-art Gyro-based Method

We now compare our method with the fused video stabilization (FVS) [7] on Google Pixel 2/3. It is rated as one of best video stabilization solutions on mobile phones by many reviews. We show the stabilization effect by averaging the consecutive 15 frames (0.5s) in Fig. 6.

(a) (b) (c) (a) (b) (c)

Fig. 6. Head stabilitiy visualization. Each frame is the average of consecutive 15 frames. For each triplet: (a) input video with video id in the accompanying video [1], (b) results by the fused video stabilization [7], and (c) our results.

(a) (b) (a) (b) (a) (b)

Fig. 7. Comparison against video stabilization method in [12]. Each frame is the average of consecutive 15 frames. For each pair: (a) results by Yu and Ramamoorthi [12] with video id in the accompanying video [1], and (b) our results.

As we can see, the face and background are blurry in the inputs due to shaky motions. FVS stabilizes the background, but exaggerates the foreground head motion. In contrast, our *Steadiface* outputs stable head motion across frames while maintaining a good trade-off for background stability. Note that our method uses a smaller cropping ratio (15%) than FVS does (20%). This makes stabilization more challenging, but we can preserve more field-of-view for users.

5.3. Comparison to State-of-the-art Image-based Method

Finally, we compare *Steadiface* with the selfie video stabilization method [12] (Fig. 7). Note that their solution cannot reach real-time performance even on a desktop. The face stability of *Steadiface* is comparable or slightly better than them. Meanwhile, *Steadiface* does not suffer from quick jittering when landmark detection/optical flow fails (e.g. the shaky background at bottom left of Fig. 7). One drawback is the cropping ratio. Their method dynamically adjusts the crop and can preserve wider field-of-view sometimes.

In sum, we present *Steadiface*, a new real-time face-centric video stabilization method that simultaneously removes hand shake and keeps subject's head stable. It can work with different types of camera motions, and is robust to large head pose, occlusion and facial appearance variations. It is also highly efficient and runs at 8.1 ms/frame with a single core on Google Pixel 3.

6. REFERENCES

[1] "Supplemental videos," https://drive.google.com/open?id=1nSaPwAsrtkY3bhd1S35GoJM4Mh5uK1IQ.

[2] C. Morimoto and R. Chellappa, "Evaluation of image stabilization algorithms," in *ICASSP*, 1998, vol. 5, pp. 2789–2792.

[3] A. Karpenko, D. Jacobs, J. Baek, and M. Levoy, "Digital video stabilization and rolling shutter correction using gyroscopes," *CSTR*, 2011.

[4] M. Grundmann, V. Kwatra, and I. Essa, "Auto-directed video stabilization with robust L1 optimal camera paths," in *CVPR*, 2011.

[5] C.-K. Liang, L.-W. Chang, and H. H. Chen, "Analysis and compensation of rolling shutter effect," *IEEE TIP*, vol. 17, no. 8, 2008.

[6] S. Bell, A. Troccoli, and K. Pulli, "A non-linear filter for gyroscope-based video stabilization," in *ECCV*, 2014.

[7] "Fused video stabilization on the Pixel 2 and Pixel 2 XL," https://ai.googleblog.com/2017/11/fused-video-stabilization-on-pixel-2.html, Accessed: 2018-12-23.

[8] F. Liu, M. Gleicher, H. Jin, and A. Agarwala, "Content-preserving warps for 3D video stabilization," *ACM TOG*, vol. 28, no. 3, 2009.

[9] A. Goldstein and R. Fattal, "Video stabilization using epipolar geometry," *ACM TOG*, vol. 31, no. 5, 2012.

[10] F. Liu, M. Gleicher, J. Wang, H. Jin, and A. Agarwala, "Subspace video stabilization," *ACM TOG*, vol. 30, no. 1, 2011.

[11] S. Liu, L. Yuan, P. Tan, and J. Sun, "Bundled camera paths for video stabilization," *ACM TOG*, vol. 32, no. 4, 2013.

[12] J. Yu and R. Ramamoorthi, "Selfie video stabilization," in *ECCV*, 2018.

[13] V. Blanz and T. Vetter, "A morphable model for the synthesis of 3D faces," in *SIGGRAPH*, 1999, pp. 187–194.

[14] F. Shi, H.-T. Wu, X. Tong, and J. Chai, "Automatic acquisition of high-fidelity facial performances using monocular videos," *ACM TOG*, vol. 33, no. 6, pp. 222, 2014.

[15] J. Thies, M. Zollhofer, M. Stamminger, C. Theobalt, and M. Nießner, "Face2face: Real-time face capture and reenactment of rgb videos," in *CVPR*, 2016, pp. 2387–2395.

[16] 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 TOG*, vol. 35, no. 3, pp. 28, 2016.

[17] C. Cao, Q. Hou, and K. Zhou, "Displaced dynamic expression regression for real-time facial tracking and animation," *ACM TOG*, vol. 33, no. 4, pp. 43, 2014.

[18] A. Jourabloo and X. Liu, "Large-pose face alignment via CNN-based dense 3D model fitting," in *CVPR*, 2016, pp. 4188–4196.

[19] 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*, 2017, vol. 2, p. 5.

[20] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, "Mobilenets: Efficient convolutional neural networks for mobile vision applications," *arXiv preprint arXiv:1704.04861*, 2017.

[21] S. Agarwal, K. Mierle, and Others, "Ceres solver," http://ceres-solver.org, Accessed: 2018-12-23.