Selfie Video Stabilization

Jiyang Yu and Ravi Ramamoorthi

University of California, San Diego
jiy173@eng.ucsd.edu, ravir@cs.ucsd.edu

Abstract. We propose a novel algorithm for stabilizing selfie videos. Our goal is to automatically generate stabilized video that has optimal smooth motion in the sense of both foreground and background. The key insight is that non-rigid foreground motion in selfie videos can be analyzed using a 3D face model, and background motion can be analyzed using optical flow. We use second derivative of temporal trajectory of selected pixels as the measure of smoothness. Our algorithm stabilizes selfie videos by minimizing the smoothness measure of the background, regularized by the motion of the foreground. Experiments show that our method outperforms state-of-the-art general video stabilization techniques in selfie videos.

1 Introduction

Selfie video has become one of the major video types thanks to the recent development of social media. However, selfie videos taken by amateurs are usually shaky due to the lack of stabilizing equipment. Recent state-of-the-art works have been developed for general video stabilization tasks and integrated into commercial tools such as Adobe Warp Stabilizer[1] and the YouTube video stabilizer[2]. However, selfie videos usually have properties that create difficulties for existing methods. We show several example frames from typical selfie videos in Fig. 1, in which these properties are demonstrated:

(a) Large non-rigid occlusion from face and body close to the camera;
(b) Selfie videos usually come with strong motion blur/out-of-focus background;
(c) Foreground motion does not coincide with background motion.

General video stabilization methods fail in selfie videos for several reasons. First, most of these works depend on tracking 2D feature points. Existing 3D stabilization approaches require Structure from Motion(SfM) to estimate an initial camera path and build a sparse 3D scene. 2D methods also need to find frame motion using features. Therefore these methods are sensitive to inaccurate tracking of feature points. In Fig. 1(b), we show the example frames with blurred background and lack of sharp corners. In these videos, feature point detection is less reliable and the subsequent feature tracking is error-prone.

Second, it is also difficult to obtain long and error-free feature tracks in selfie videos with strong shake. The feature tracking becomes brittle due to the

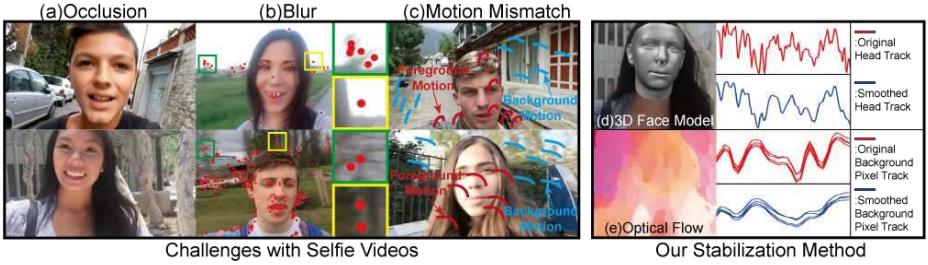

Fig. 1. *Selfie videos have several properties that cause difficulties for traditional video stabilization methods: (a) face and body significantly occludes the background; (b) bad feature detection caused by motion blur/out of focus, insets show areas where feature points are hard to track accurately; (c) foreground and background motion mismatch; the foreground motion(red) can be different from background motion(blue) due to the dynamics of face and body; Our method uses (d) a 3D face model to analyze the motion in the foreground and (e) optical flow to analyze the motion in the background. The video is stabilized with respect to both foreground and background.*

significant occlusion imposed by human face and body. Having noticed feature tracking as a general shortcoming in video stabilization, some methods tried to avoid using features by analyzing the pixel profiles using optical flow[3]. However, optical flow based algorithms still have failure cases when the occluding object dominates the foreground, which is likely to happen in selfie videos(Fig. 1(a)). Our algorithm takes advantage of optical flow to track the background pixels. Unlike Liu et.al[3] which uses optical flow to synthesize new frames, we only warp the frame with 2D projective transformations and a grid-based warp field. This guarantees the rigidity over the entire frame. To avoid tracking points and generating long trajectories, we only use small segments of these trajectories so that the foreground occlusion has minimal impact on the stabilization. We further discuss the advantages of the strategy in Sec. 7.

Third, general video stabilization only stabilizes with respect to part of the scene. This is not always desired in selfie videos. Both foreground (face and body) and background are important regions that need to be stabilized. To our knowledge, ours is the first method that utilizes the face geometry information in the video stabilization task(Fig. 1(d)). Our algorithm can automatically plan the optimal motion so that both the foreground and background motion are smoothed(Fig. 1(d)(e)). In summary, our contributions include:

Foreground motion from 3D face model: We utilize 3D human face information to gain knowledge about foreground motion in selfie videos(Sec. 4).

Novel background motion tracking: Our method uses optical flow to find dense correspondences on the background, and therefore does not require good feature detection and tracking. We only use temporal motion information and

are robust to occlusions in the scene(Sec. 5).

Optimal foreground/background stabilization: By considering foreground motion, our method can stabilize selfie videos with respect to foreground and background simultaneously(Sec. 6).

Labeled selfie video dataset: We provide a selfie video dataset (Fig. 7) of 33 videos, labeled with properties such as dynamic occlusion and lack of background features (Fig. 9) that significantly affect the video stabilization task. The dataset can be used to compare different methods, and will be a useful resource for the field. We make the dataset, code and benchmark per Fig. 9 publicly available online at `http://viscomp.ucsd.edu/projects/ECCV18VideoStab`.

2 Related Work

General video stabilization can be broadly categorized into 2D methods and 3D methods, according to their proposed camera motion models.

2D Stabilization General 2D video stabilization techniques compute 2D motion and generate stabilized video by applying the smoothed motion to original video frames. Some approaches use simple camera motion models. Grundmann et al.[2] proposed a constrainable L1-optimization framework which solves the smoothed camera path composed of constant, linear and parabolic motion. Gleicher and Liu[4] assume the scene is largely planar and use homography to synthesize new frames. Liu et al.[5] divide the frame space into a grid mesh and allow spatially-variant motion. Some methods smooth the motion by imposing non-trivial constraints. Liu et al.[6] smooth 2D feature tracks by enforcing low-dimensional subspace constraints. Wang et al.[7] smoothes feature tracks while maintaining the spatial relations among them. Goldstein and Fattal[8] uses epipolar constraints when warping the video frames into synthesized frames. There are also explorations of non feature-point based approaches: Liu et al.[3] solves for a smooth per-frame optical flow field and stabilizes the video by smoothing pixel profiles instead of smoothing feature tracks.

3D Stabilization Some methods sparsely reconstruct the 3D scene. The sparse 3D information is used to guide the synthesis of new frames. These methods generate better results than 2D methods by modeling physically accurate 3D camera motions but are less robust under non-ideal conditions, e.g. large occlusion, motion blur, and rolling shutter. Liu et al.[1] first uses structure from motion to find feature points' 3D positions, reprojects them onto the smoothed camera path and warps the original frames according to reprojected feature points. There are also methods that render new frames using 3D information: Buehler et al.[9] uses image-based rendering to synthesize new views; Smith et al.[10] utilize the light field camera to stabilize video; Sun[11] uses depth information. Due to the non-rigid occlusion in selfie videos(Fig. 1(a)), 3D methods that are based on structure from motion can be error-prone. 3D methods that use depth information are also not directly applicable to selfie videos, since depth is not available

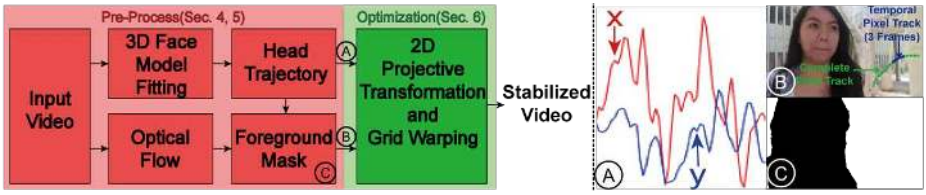

Fig. 2. *Pipeline of our method. Ⓐ: By fitting a 3D face model, we find the head trajectory in the selfie video(Sec. 4); Ⓑ: Optical flow is used to track background pixels for 3 neighboring frames; Ⓒ: The foreground mask is computed from the head trajectory and is used to find the background pixels(Sec. 5). The 2D projective transformation and a grid-based warp field is estimated to remove the undesired motion of both foreground and background(Sec. 6).*

in most cases.

Face Modeling Human face modeling has been intensely studied. We will only summarize works that are closely related to our work. A widely used early work(Blanz and Vetter[12]) models face shape across people as a parametric PCA space learned from a database of laser scans. Cao et al.[13] models faces by assembling Kinect face scans as a tensor with identity and expression dimensions. Many follow-up works apply these models in image manipulation (Cao et al.[13], Fried et al.[14]), image/video re-animation(Blanz et al.[15], Thies et al.[16]), face tracking(Cao et al.[17]), facial performance capture(Cao et al.[18], Shi and Tomasi[19]), face reconstruction and rigging(Garrido et al.[20], Ichim et al.[21]). However, these works mainly focus on images/videos captured under ideal conditions(still or stable camera). Our work explores the possibility of utilizing 3D face models in the analysis of selfie videos captured with camera shake. We blend the models in Blanz and Vetter [12] and Cao et al. [13], to use as the reference model for the face fitting process, which will be discussed in Sec. 4.

3 Overview

In this section, we provide an overview of our approach for stabilizing selfie videos(Fig. 2). We seek to stabilize the selfie video with respect to both foreground and background. We analyze the foreground motion by modeling the face using a 3D face model, and analyze the background motion by optical flow. Fitting the 3D face model to selfie videos provides the head trajectory. We transform each frame according to the head positions so that the foreground regions are roughly aligned across the entire video. Since the foreground regions are aligned, the accumulated motion in this region will be smaller than background regions. Therefore the foreground and background regions can be separated. Details regarding this process will be discussed in Sec. 4. The background is defined as the white region in the foreground mask shown in Fig. 2. We randomly select pixels in the background that satisfy certain conditions, and use the optical flow to track their motion. Because of occlusion, our method only tracks pixels for 3 neighboring frames. We discuss details of pixel selection and tracking in Sec. 5.

The goal of video stabilization is to warp the original video frame so that the undesired frame motions are cancelled. We model the frame motion as a combination of global motion and local motion. The global motion refers to the 2D projective transformation of a frame. Since the frame content is the result of multiple factors, e.g., camera projection, camera distortion, rolling shutter and the 3D structure of the scene, simple 2D projective transformation cannot represent the camera motion accurately. Therefore, we use local motion to refer to any residual motion. Motivated by this analysis, we design our stabilization algorithm as a single joint optimization that simutaneously stabilizes foreground head motion and the background's global and local motion. We will describe details of our joint optimization algorithm in Sec. 6.

4 Foreground Tracking

Since the head and body are attached, we believe that the head motion can well represent the entire foreground motion. We don't explicitly track the body in this work, but implicitly separate the foreground and background by accumulating the optical flow. Details will be discussed in Sec. 5. Here we seek to find the position of the head in each frame. Since multiple faces could exist in the selfie video, we only track the dominant face in the video. A regular 2D face detector can provide the face bounding box for each frame, but is not accurate enough for tracking the exact head position. A 2D facial landmark detector provides more accurate detection of the face, but is easily affected by head rotation and facial expression. To find the actual head position invariant to head rotation and facial expression, we use the 3D position of the head and reproject it to the image space as the head position. This requires modeling the face and gaining knowledge about the shape of the face in the selfie video.

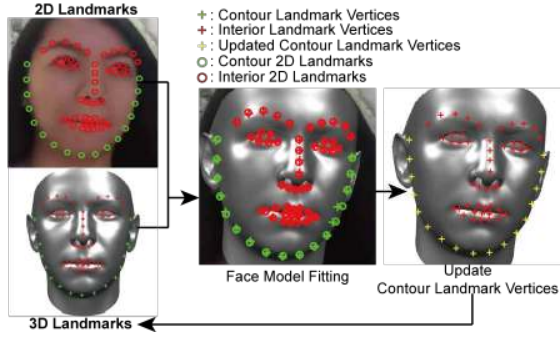

Fig. 3. *The vertices used as contour 3D landmarks are fixed in the fitting process. The fitted face is rendered and new contour 2D landmarks are detected. The projected vertices closest to the detected 2D contour landmarks are selected as 3D contour landmarks for the next iteration.*

3D Face Model We utilize the linear face model proposed by Blanz and Vetter[12] and the bilinear face model proposed by Cao et al.[13]. Note that although the bilinear face model is more widely used in recent researches, their model was built based on a head mesh with relatively sparse vertices compared to Blanz and Vetter[12]. Our facial landmark based algorithm, which we will discuss later in this section, needs a dense face mesh in the face fitting algorithm. Therefore, we extend the linear face model of Blanz and Vetter[12] by transferring the ex-

pressions from Cao et al.[13]. Our extended linear face model is parameterized as follows:

$$\boldsymbol{F} = \boldsymbol{\mu} + \boldsymbol{U}_s\boldsymbol{\Sigma}_s\boldsymbol{c}_s + \boldsymbol{U}_e\boldsymbol{\Sigma}_e\boldsymbol{c}_e \tag{1}$$

where $\boldsymbol{\mu}$ encodes the vertex position of the mean face, $\boldsymbol{U}_s$ are the principal components of face shape, diagonal matrix $\boldsymbol{\Sigma}_s$ contains standard deviations of principal components and $\boldsymbol{c}_s$ is the weight vector that combines principal components. In (1), the third term $\boldsymbol{U}_e$ represents the expression principal components. It is generated as follows: we average the shape dimension of the bilinear face model[13] and use deformation transfer[22] to deform the mean linear face model with the bilinear face model's expressions. We extract principal components $\boldsymbol{U}_e$ of these expression deformed face meshes using regular PCA.

Face Fitting Algorithm Our face model fitting algorithm is a purely landmark based algorithm. For a video with T frames, we detect facial landmarks L_t in each frame using Bulat and Tzimiropoulos[23]. The unknown parameters include the 3D rotation $\boldsymbol{R}_t \in SO(3)$, the 3D translation $\boldsymbol{T}_t \in \mathbb{R}^3$, per-frame facial expression coefficient $\boldsymbol{c}_{e,t}$ and the shape parameter $\boldsymbol{c}_s$. We also assume a simple perspective camera projection:

$$\boldsymbol{P} = \begin{bmatrix} f & 0 & w/2 \\ 0 & f & h/2 \\ 0 & 0 & 1 \end{bmatrix} \tag{2}$$

where we assume same unknown focal length in horizontal and vertical direction, known fixed optical center at the center of the frame(w and h represents frame width and height respectively), and zero skew. Denoting the 3D transformation matrix as $\boldsymbol{K}_t = [\boldsymbol{R}_t\ \boldsymbol{T}_t]$ where $\boldsymbol{R}_t \in \mathbb{R}^{3\times3}$ and $\boldsymbol{T}_t \in \mathbb{R}^3$, the 3D face model is fitted by solving:

$$\min_{\substack{\boldsymbol{P},\boldsymbol{R}_t,\boldsymbol{T}_t,\\ \boldsymbol{c}_s,\boldsymbol{c}_{e,t}}} \sum_{t=0}^{T-1} \left(\left\| L_t - \boldsymbol{P}\boldsymbol{K}_t\widehat{\boldsymbol{F}}_t \right\|^2 + \lambda_1 \left\| \boldsymbol{c}_{e,t} \right\|^2 \right) + \lambda_2 T \left\| \boldsymbol{c}_s \right\|^2 \tag{3}$$

where $\widehat{\boldsymbol{F}}_t$ represents the landmark vertices controlled by $\boldsymbol{c}_s$ and $\boldsymbol{c}_{e,t}$ as in (1), and λ_1 and λ_2 are regularization values that prevent the optimization from getting in local minima. The optimization can be easily solved as an unconstrained non-linear least squares problem. We use all the 199 shape principal components and 46 expression principal components in the fitting process. We use $\lambda_1 = 5$ and $\lambda_2 = 5$ in our experiment. The centroids of fitted face meshes are projected using the solved projection matrix $\boldsymbol{P}$, resulting in the head trajectory.

Facial landmark update In Bulat and Tzimiropoulos[23], the contour 2D landmarks are defined by the face silhouette. The face silhouette depends on the pose of the head; therefore the corresponding contour landmark vertices need to be updated during optimization (3). However, this requires computing and rendering the whole mesh for facial landmark detection. To avoid this cost, we only update the contour landmark vertices between two iterations of optimization: we first fix the landmark vertices and use them in the face model fitting, then fix the estimated parameters and update the contour landmark vertices.

The update of landmark vertices is demonstrated in Fig. 3. We first render the current face mesh, and detect 2D landmarks using the rendered image. We update the landmark vertices' indices by projecting all the visible vertices to the image plane and find the closest ones to the detected 2D landmarks. These closest vertices are used as contour landmark vertices in the next iteration. Note that the 2D-3D correspondence is established by finding vertices closest to landmarks. Therefore, a denser mesh will result in more accurate correspondence. This explains why we extend the denser linear face model(Blanz and Vetter[12]) instead of using the sparse bilinear face model(Cao et al.[13]) directly.

Disscussion General video stabilization methods have difficulties when occlusion occurs in the video. Some methods either try to exclude the occlusion region by detecting discontinuity in motions(Liu et al.[3]) or let users remove features belonging to the foreground(Bai et al. [24]). The novelty of our method is that it also considers the motion in the foreground. Due to the dynamics of faces, feature based analysis is easily affected by head poses and facial expressions. We use the 3D face model to track the foreground face, so that the foreground can be analyzed even with large non-rigidity. In Fig. 4 we show that by implementing the contour landmark update scheme, our face fitting algorithm also achieves results comparable to the methods that use 3D facial landmarks estimated using non-rigid structure-from-motion(Shi et al. [19]) or 2D facial landmarks with additional light and shading constraints(Thies et al. [16]). Note that our method uses only 2D landmarks and thus is simpler than state-of-the-art methods.

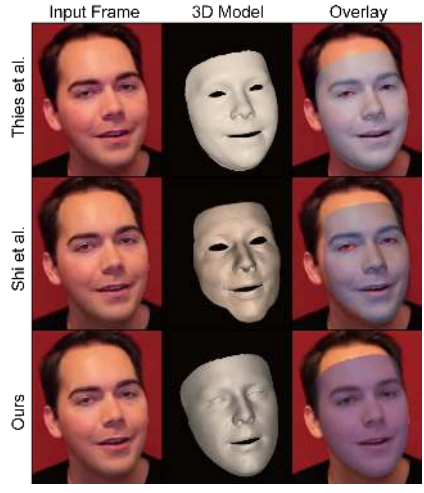

Fig. 4. *Comparison of our 3D face fitting result to Shi et al. [19] and Thies et al. [16]. Our method achieves comparable results without using complex structure-from-motion and shading constraints.*

5 Background Tracking

While we can track the foreground motion using a 3D face model, we also need to analyze the background motion so that both these regions can be considered in the stabilization process. We use the optical flow proposed by Kroeger et al. [25] to track a group of background pixels in each frame. The optical flow can be inaccurate in specific regions due to motion blur/out-of-focus and occlusion. However, minor inaccuracies in small regions can be ignored since our goal is to analyze the global camera motion. In addition, to minimize the impact of

occlusion in the scene, we only track each pixel for 3 neighboring frames. We will discuss how this temporal motion information is used in our stabilization process in Sec. 6.

Not all pixels can be used to track the background motion. Obviously, pixels falling in the foreground region should not be selected. Face fitting described in Sec. 4 provides the head positions in each frame. We first translate all the frames so that the head positions in each frame are aligned to the same point, which leads to a head-aligned video. We perform optical flow between each frame of the head-aligned video. The accumulated optical flow forms a map that encodes the accumulated motion magnitude of each pixel. Since the video is aligned with respect to head position, the accumulated magnitude of optical flow will be smaller in the face and body region, but larger in the background region.

We show an example of a motion map in Fig. 5A. After computing the motion map, we use K-means to divide the pixels into two clusters. The cluster with smaller values is considered as foreground. The randomly selected pixels in this cluster will not be used in the stabilization.

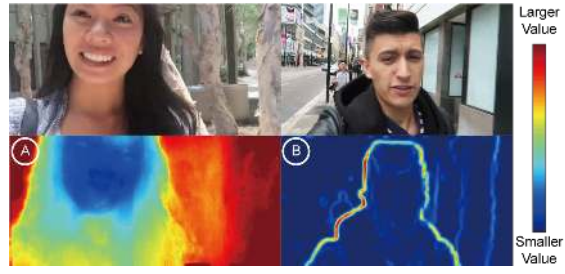

Fig. 5. Ⓐ*: Accumulated optical flow. A large value indicates the background area.* Ⓑ*: Example moving standard deviation of optical flow. Large values indicate the edges of objects in the scene.*

Moreover, pixels near the occluding boundary should not be selected. Although our method does not require long feature tracks, we still need to track pixels using optical flow. The downside of tracking with optical flow is that tracking loss caused by occlusion is not easily detectable. To tackle this problem, we want to remove the pixels that are near the occluding boundary.

The objects in the scene can be distinguished by the direction of their motions. We use the standard deviation of the optical flow σ_F in a 21×21 neighborhood to measure the local variation in the optical flow. An example of standard deviation of the optical flow is shown in Fig. 5B. The foreground boundary has a larger variation in terms of the optical flow direction. In the stabilization, we only use the pixels with σ_F smaller than a threshold value. We use 0.3 as the threshold in all of our tests.

6 Stabilization

The goal of video stabilization is to warp the original video frame so that the undesired frame motions are cancelled. We model the frame motion as a combination of global motion and local motion. The global motion refers to the 2D projective transformation of a frame. Since the frame content is the result of multiple factors, e.g. camera projection, camera distortion, rolling shutter and

the 3D structure of the scene, a simple 2D projective transformation cannot represent the camera motion accurately. Therefore, we use local motion to refer to any residual motion.

Motivated by this analysis, we design our algorithm to stabilize the global motion using the whole frame 2D projective transformation and stabilize the local motion using the per-frame grid warping.

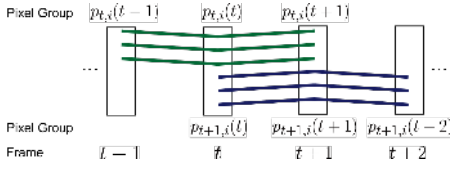

Fig. 6. *Our method tracks background pixels for 3 neighboring frames.*

Smoothness Measure In selfie videos, the human appears as a large occlusion near the camera, making the trajectory of a pixel fragile. As a consequence, obtaining long feature tracks is difficult. Instead of tracking a pixel over multiple frames, we only track a pixel for 3 frames that are necessary for estimating the second derivative at time t. To demonstrate our idea, we use a single pixel in the scene as an example. Assume a pixel p is tracked over a time period. The trajectory it forms is denoted by $p(t)$. To evaluate the smoothness of this trajectory, we use the integral of squared second derivative or acceleration over the time period. This metric is commonly used in cubic spline fitting algorithms for optimal smoothness. By using this metric, we allow the frames to move to some extent but not try to completely eliminate the low frequency shake. This also helps in generating a larger output frame size when the camera motion is large, which is very common in selfie videos. Details of this effect will be discussed in Sec. 7. For a set of selected background pixels (which pixels we choose for this purpose is discussed in Sec. 5), the smoothness of the background motion can be written as:

$$E_s(t) = \sum_{i=1}^{N_t} \|\widehat{p}_{t,i}(t+1) - 2\widehat{p}_{t,i}(t) + \widehat{p}_{t,i}(t-1)\|^2 \tag{4}$$

where $p_{t,i}$ is the i^{th} pixel tracked from $t-1$ to $t+1$, and $\widehat{p}$ is the new trajectory formed by transforming the original trajectory p. To guarantee the robustness, we track N_t pixels that are randomly selected in the frame at time $t-1$. We illustrate the tracking of background pixels in Fig. 6.

Frame Transformation We seek to find a per-frame 2D projective transformation along with a per-frame grid warp field to transform $p_{t,i}$ to $\widehat{p}_{t,i}$ so that the objective (4) is minimized.

For the grid warping, we use the same bilinear interpolation representation as Liu et al.[1]. Each point is represented by a combination of four vertices of its enclosing grid cell:

$$p_{t,i} = w_{t,i}^T V_t \tag{5}$$

where V_t is a vector of the four vertices of the original grid cell that $p(t)$ is in; and w_t is the weight which sums to 1. Denote the output grid as $\widehat{V}$ and the 2D projective transformation as H. The warped scene point $\widehat{p}$ can be calculated

using the same weights:

$$\widehat{p}_{t,i} = w_{t,i}^T H_t \widehat{V}_t \tag{6}$$

Regularization In selfie videos, the foreground that contains face and body should also be stabilized. The motion of the foreground is not always consistent with the motion of the background. To account for the foreground motion, we also consider the head trajectory:

$$E_h(t) = N_t \left\| \widehat{h}(t+1) - \widehat{h}(t) \right\|^2 \tag{7}$$

where $h(t)$ is the head trajectory and $\widehat{h}(t)$ is the transformed head trajectory at time t. The head trajectory was obtained via fitting a 3D face model to the video as described in Sec. 4.

Moreover, to avoid undesired deformation caused by grid warping, we use the Laplacian of the grid to measure the rigidity of the warping:

$$E_V(t) = \Delta(\widehat{V}_t) \tag{8}$$

Optimization Our final objective function is a combination of the smoothness measure and the regularization term:

$$\min_{H_t, \widehat{V}_t} \quad \sum_{t=1}^{T-2} E_s(t) + \lambda_a E_h(t) + \lambda_b E_V(t) \tag{9}$$

Due to the high degree of freedom of the unknowns, the objective function has a complex landscape. Therefore, we first fix the grid $\widehat{V}_t$ and solve for 2D perspective transformation H_t, then use the result as an initialization and refine by running the full optimization (9).

We use Matlab's nonlinear least-squares solver to solve this optimization problem. For each frame at time t, the error terms E_s, E_h and E_V are only affected by 3 frames at $t-1$, t and $t+1$. This leads to a sparse jacobian matrix. Therefore this problem can be efficiently solved.

7 Results

In this section, we show example frames of selfie video stabilization, along with the visual comparison of the input video, our result, Liu et al.[1] and Grundmann et al.[2]. We also show that our method achieves better quantitative results than the comparison methods. Finally we discuss the advantages of our method over general video stabilization methods. Our results are generated with fixed parameters $\lambda_a = 1$ and $\lambda_b = 5$. On average, our Matlab code takes 15 min in all: 3 min for head fitting, 1 min for optical flow, 8 min for optimization and 3 min for warping and rendering the video on a desktop computer with an Intel i7-5930K CPU@ 3.5GHz. We did not focus on speed in this work and we believe that our optimization can be implemented on the GPU in future.

Fig. 7. *Example video stills from our dataset. The labels represent the example indices in Fig. 9.*

Fig. 8. *Visual comparison of input video, our result, Grundmann et al.[2] result and Liu et al.[1] result. The video stills are scaled by the same factor. Our method generates results with larger field of view and does not introduce visible distortion. We recommend readers to watch the accompanying video for more visual comparison. Labels represent example indices in Fig. 9.*

Test Set We collected 33 selfie video clips from the Internet, which is the first such dataset of selfie videos. A subset of our example clips are shown in Fig. 7. We label each video with properties that affect video stabilization: dynamic occlusion, multiple faces, large foreground motion, lack of background features, dynamic background and motion/defocus blur (Fig. 9). We will make our test set available publicly for comparison of different methods, and believe it will be a useful resource for the community.

Visual Comparison In Fig. 8, the video stills are scaled by the same factor so that their sizes can be compared. Our results have a larger field of view compared to Liu et al.[1] and Grundmann et al.[2], which is often desired in stabilizing selfie videos. This is because the movement of the camera is large in these examples. The methods proposed by Liu et al.[1] and Grundmann et al.[2] over-stabilize the background, resulting in a small overlap region among frames. To obtain a rectangular video, most of the regions have to be cropped. Our method considers the foreground and background motion together and allows the frame to move in a low frequency sense. Therefore we avoid over-stabilization with respect to either foreground or background. Also note that our result preserves the original shape of the face and body, while the Liu et al.[1] result contains large distortions

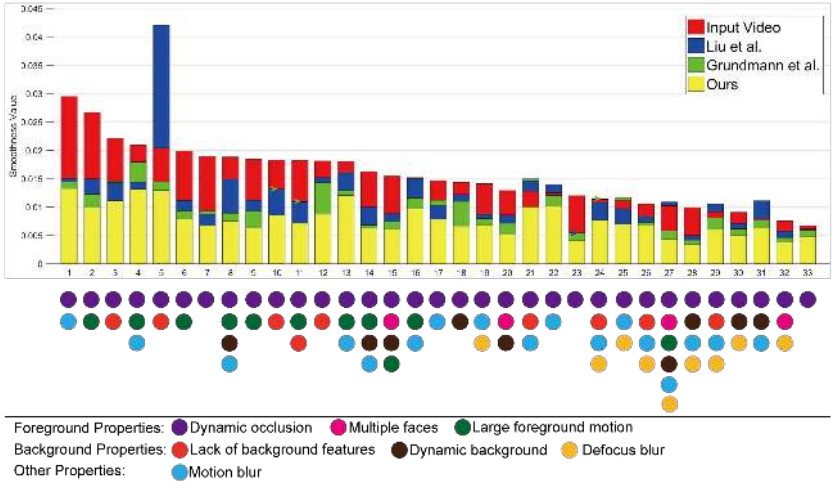

Fig. 9. *Smoothness comparison of input video, our result, Liu et al. result[1] and Grundmann et al. result[2]. The horizontal axis represents the examples, and the height of the bar represents the smoothness value. Colored arrows are added where the bars overlap. The labeled properties are visualized as colored dots below each example.*

on the face. Since the dynamics are hard to show with images, we recommend readers to watch the accompanying video for the visual comparison of the results.

Our method is not sensitive to λ_b, but by changing the head regularization value λ_a in (9), we can control the algorithm to mainly stabilize the foreground or the background. We also included an example stabilized with different λ_a values in the accompanying video.

Quantitative Comparision To evaluate the level of smoothness of the videos, we compute the average squared magnitude of second derivative of tracks of all pixels in each frame. The smoothness measure is defined as:

$$S = \frac{1}{|\Omega|} \sum_{t=1}^{T-2} \sum_{i} \|\omega_{t,i}(t+1) - 2\omega_{t,i}(t) + \omega_{t,i}(t-1)\|^2 \tag{10}$$

where we track all the pixels $\boldsymbol{\omega}_t = \{\omega_{t,1}, \omega_{t,2}...\}$ in the frame at time t, and Ω is the set of all the pixels $\{\boldsymbol{\omega}_1, \boldsymbol{\omega}_2...\boldsymbol{\omega}_{T-1}\}$. Since we sum the second derivatives of all the pixel tracks, a smaller smoothness measure indicates that the frames are changing in a more stabilized way. In (10), we use the optical flow to track the pixels. To eliminate the effect of different video sizes, we normalize the optical flow with the frame size on horizontal and vertical directions respectively. We show smoothness comparison for these examples in Fig. 9. Note that a lower bar indicates a better result. For better comparison, we sorted the examples by their original smoothness value. Our final results achieve better smoothness compared to the results of Liu et al.[1] and Grundmann et al.[2] in all of the examples.

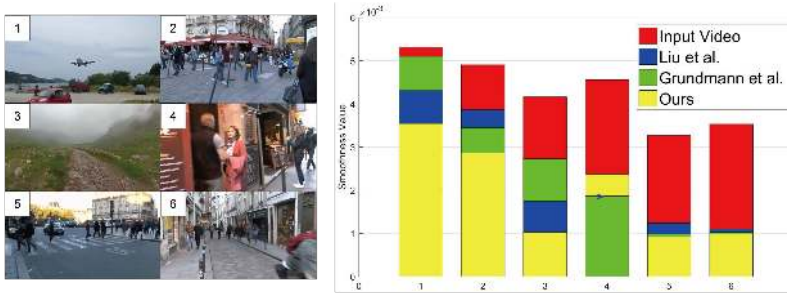

Fig. 10. *Example video stills from our test set, and smoothness comparison on general videos, showing our result, Liu et al.[1] result and Grundmann et al.[2] result. Numbers on video stills indicate the example indices on the bar graph. Colored arrows are added where the bars overlap.*

Advantages Our method has some advantages over other general video stabilization methods in the selfie video case. Traditional 2D and 3D methods usually rely on feature tracks[1, 2, 6, 10], making them vulnerable to insufficient feature counts in selfie videos. Since our method uses optical flow to track the motion, we achieve significantly better result in videos with few background features (examples 3, 5, 10, 11, 12, 21, 24, 26 and 29 in Fig. 9). Note that the feature point based general video stabilization methods fail in some of the low feature count cases (examples 5, 21 and 29 in Fig. 9), resulting in an even higher smoothness value than the input video. Our method is also robust to videos with motion blur and defocus blur, which are very common properties in selfie videos.

It is hard to obtain long feature tracks in selfie videos with large foreground motion. Note that 3D methods like Liu et al.[1] cannot perform accurate structure from motion when there is dynamic occlusion. Therefore Liu et al.[1] in general does not perform well in large foreground motion cases (examples 2, 4, 6, 8, 9, 11, 13, 14, 15, 16 and 27 in Fig. 9). Using only fragments of pixel trajectories over 3 frames, our method is robust to large occlusions near the camera. This strategy also helps handle dynamic background (examples 8, 14, 15, 18, 20, 27, 28, 30 and 31 in which multiple non-dominant faces or moving objects exist).

Finally, our method provides a novel application of 3D face modeling: track the foreground motion in selfie videos. Current 2D video stabilization methods focus on detecting non-rigid regions and do not consider the motion in these regions. In selfie videos, the foreground occupies a large portion of the frames and cannot be ignored. Our method automatically plans the motion so that both foreground and background motion are smoothed. The foreground motion also helps regularize the video stabilization process. In all of the examples, our method avoids over stabilizing the background and produces results with significantly larger field of view.

Generalization Our method also applies to stabilizing general videos. We can simply ignore the E_h term in (9) and perform the optimization for the entire background region. We also collect 6 general videos shown in Fig. 10 and compare

the smoothness of our result against Liu et al.[1] and Grundmann et al.[2]. Note that we only use 3 neighbouring frames to track the frame motion and only local motion information is available. Therefore, our method faces a harder problem in general video stabilization. However, Fig. 10 shows that our method still achieves comparable results in general video cases.

Failure Cases Our frame motion model does not apply to videos with complex motions, e.g. strong rolling shutter effect and fisheye effect. We also include a selfie video taken with a fisheye camera in the accompanying video, in which our method does not perform well. Our method does not explicitly correct motion blur. Therefore our results on videos with strong motion blur (mostly because of low illumination) will have unsatisfactory appearance. Our result of example 4 in the selfie video dataset belongs to this category. Note that Fig. 9 shows that we still generate better results for example 4 compared to Liu et al.[1] and Grundmann et al.[2].

8 Conclusion, Limitations and Future Work

We proposed a novel video stabilization technique for selfie videos. Our method analyzes the motion of foreground(face and body) using a 3D face model and the motion of background by temporally tracking the pixels using optical flow. We achieve visually and quantatively better results than the state-of-the-art general video stabilization methods. Our method also exhibits robustness under different situations(e.g., large foreground occlusion, blur due to motion or out-of-focus and foreground/background motion mismatch).

Our method requires optical flow to track pixels in the video, and therefore suffers from the overhead of computing optical flow for neighboring frames. Another limitation of our method is that we require that facial landmarks can be detected in most of the frames. In our experiments, we linearly interpolate the head position for frames in which no face was detected. If the faces are undetectable in many consecutive frames, simply interpolating head positions will yield inaccurate estimation of the foreground motion. These limitations can be resolved by applying a more efficient optical flow technique and a more robust facial landmark detector. Our frame motion model does not apply to videos with complex motion. Our method also does not correct motion blur. Therefore for night-time videos or videos taken under dark lighting conditions, our method does not produce satisfactory results.

Since our method utilizes the 3D face model in selfie videos, one future work would be using 3D information to estimate 3D camera motion, so that the 3D video stabilization can be applied to selfie videos with large dynamic occlusions. The 3D face model also enables other future works, including manipulating the shape and expression of the face in selfie videos or high quality 3D reconstruction of face and body from selfie videos.

Acknowledgements. We thank Nima Khademi Kalantari for the supplementary video voice over. This work was supported in part by Sony, the Ronald L. Graham endowed Chair, and the UC San Diego Center for Visual Computing.

References

1. Liu, F., Gleicher, M., Jin, H., Agarwala, A.: Content-preserving warps for 3D video stabilization. ACM Trans. Graph. **28**(3) (July 2009)
2. Grundmann, M., Kwatra, V., Essa, I.: Auto-directed video stabilization with robust l1 optimal camera paths. In: IEEE CVPR. (2011)
3. Liu, S., Yuan, L., Tan, P., Sum, J.: Steadyflow: Spatially smooth optical flow for video stabilization. In: IEEE CVPR. (2014)
4. Gleicher, M.L., Liu, F.: Re-cinematography: Improving the camerawork of casual video. ACM Trans. Multimedia Comput. Commun. Appl. **5**(1) (October 2008)
5. Liu, S., Yuan, L., Tan, P., Sun, J.: Bundled camera paths for video stabilization. ACM Trans. Graph. **32**(4) (July 2013)
6. Liu, F., Gleicher, M., Wang, J., Jin, H., Agarwala, A.: Subspace video stabilization. ACM Trans. Graph. **30**(1) (February 2011)
7. Wang, Y.S., Liu, F., Hsu, P.S., Lee, T.Y.: Spatially and temporally optimized video stabilization. IEEE Trans. Visual. and Comput. Graph. **19**(8) (Aug 2013)
8. Goldstein, A., Fattal, R.: Video stabilization using epipolar geometry. ACM Trans. Graph. **31**(5) (September 2012)
9. Buehler, C., Bosse, M., McMillan, L.: Non-metric image-based rendering for video stabilization. In: IEEE CVPR. (2001)
10. Smith, B.M., Zhang, L., Jin, H., Agarwala, A.: Light field video stabilization. In: IEEE ICCV. (2009)
11. Sun, J.: Video stabilization with a depth camera. In: IEEE CVPR. (2012)
12. Blanz, V., Vetter, T.: A morphable model for the synthesis of 3D faces. In: ACM SIGGRAPH. (1999)
13. Cao, C., Weng, Y., Zhou, S., Tong, Y., Zhou, K.: Facewarehouse: A 3D facial expression database for visual computing. IEEE Trans. Visual. and Comput. Graph. **20**(3) (March 2014)
14. Fried, O., Shechtman, E., Goldman, D.B., Finkelstein, A.: Perspective-aware manipulation of portrait photos. (2016)
15. Blanz, V., Basso, C., Poggio, T., Vetter, T.: Reanimating Faces in Images and Video. Computer Graphics Forum **22**(3) (2003)
16. Thies, J., Zollhöfer, M., Stamminger, M., Theobalt, C., Nießner, M.: Face2face: Real-time face capture and reenactment of rgb videos. In: IEEE CVPR. (2016)
17. Cao, C., Hou, Q., Zhou, K.: Displaced dynamic expression regression for real-time facial tracking and animation. ACM Trans. Graph. **33**(4) (July 2014)
18. Cao, C., Bradley, D., Zhou, K., Beeler, T.: Real-time high-fidelity facial performance capture. ACM Trans. Graph. **34**(4) (July 2015)
19. Shi, F., Wu, H.T., Tong, X., Chai, J.: Automatic acquisition of high-fidelity facial performances using monocular videos. (2014)
20. Garrido, P., Zollhöfer, M., Casas, D., Valgaerts, L., Varanasi, K., Pérez, P., Theobalt, C.: Reconstruction of personalized 3D face rigs from monocular video. ACM Trans. Graph. **35**(3) (May 2016)
21. Ichim, A.E., Bouaziz, S., Pauly, M.: Dynamic 3D avatar creation from hand-held video input. ACM Trans. Graph. **34**(4) (July 2015)
22. Sumner, R.W., Popović, J.: Deformation transfer for triangle meshes. In: ACM SIGGRAPH. (2004)
23. 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: IEEE ICCV. (2017)

24. Bai, J., Agarwala, A., Agrawala, M., Ramamoorthi, R.: User-assisted video stabilization. In: EGSR. (2014)
25. Kroeger, T., Timofte, R., Dai, D., Gool, L.V.: Fast optical flow using dense inverse search. In: ECCV. (2016)