SteadyFlow: Spatially Smooth Optical Flow for Video Stabilization

Shuaicheng Liu[1] Lu Yuan[2] Ping Tan[1] Jian Sun[2]

[1]National University of Singapore [2]Microsoft Research

Abstract

We propose a novel motion model, SteadyFlow, to represent the motion between neighboring video frames for stabilization. A SteadyFlow is a specific optical flow by enforcing strong spatial coherence, such that smoothing feature trajectories can be replaced by smoothing pixel profiles, which are motion vectors collected at the same pixel location in the SteadyFlow over time. In this way, we can avoid brittle feature tracking in a video stabilization system. Besides, SteadyFlow is a more general 2D motion model which can deal with spatially-variant motion. We initialize the SteadyFlow by optical flow and then discard discontinuous motions by a spatial-temporal analysis and fill in missing regions by motion completion. Our experiments demonstrate the effectiveness of our stabilization on real-world challenging videos.

1. Introduction

Video stabilization results heavily rely on the adopted motion model. Some methods assume a parametric 2D motion model (such as homography [20] or a mixture of homography [8, 18]) between consecutive frames. These methods are robust but have limited power to deal with spatially variant motion. Feature trajectories provide more flexible non-parametric 2D motion representation. Some recent methods (such as [12, 15, 7, 25]) achieve good stabilization results by smoothing feature trajectories. However, dealing with feature trajectories is complicated. Feature trajectories are often spatially sparse and unevenly distributed. They might end or start at any frame of the video. Furthermore, obtaining long trajectories is hard in consumer videos (*e.g.*, due to rapid camera panning or motion blur).

Dense 2D motion field (such as optical flow) is a more flexible and powerful motion model. When the optical flow is spatially smooth, we find that *smoothing feature trajectories can be well approximated by smoothing pixel profiles*, which are motion vectors collected at the same *pixel location* over time. In other words, we can smooth pixel profiles instead of smoothing feature trajectories. No feature tracking is required. All the pixel profiles begin at the first frame and end at the last frame. This is a much desired property for robust stabilization of consumer videos.

The optical flow of a general video could be rather discontinuous, especially on moving objects and strong depth edges. Therefore, we require to modify the raw optical flow to get a SteadyFlow. The SteadyFlow is a close approximation of the optical flow, by enforcing strong spatial smoothness, so that we can simply smooth the pixel profiles extracted from the SteadyFlow to stabilize the original video. We initialize the SteadyFlow by traditional optical flow and identify discontinuous motion vectors by a spatial-temporal analysis. These discontinuous flows are removed and missing regions are filled in by motion completion. We evaluate our method on different types of challenging videos. The experiment results demonstrate the robustness of our technique.

2. Related Work

Video stabilization techniques can be roughly divided as 2D and 3D methods. 2D stabilization methods use a series of 2D transformations (such as homography or affine models) to represent the camera motion, and smooth these transformations to stabilize the video. Early 2D methods mainly focus on the design of appropriate motion smoothing techniques. Some methods [21, 20, 10] used simple Gaussian smoothing. Chen et al. [3] fit polynomial curves to represent camera motion. Gleicher and Liu [6] broke camera trajectories into segments for individual smoothing. Grundmann et al.[9] applied an elegant L1-norm optimization for camera path smoothing. More recently, Liu et al. [18] adopted the 'as-similar-as-possible' warping to estimate a mixture-of-homography motion model, and designed an algorithm to smooth the bundled camera paths. 2D methods are often robust, but their motion models have limited power to describe complicated scenes (e.g. scenes with discontinuous depth variations or moving objects). We employ a more general 2D motion model with per-pixel flow to deal with non-linear motion like 3D stabilization methods.

3D methods estimate 3D camera motion for stabilization. Beuhler et al.[2] designed stabilization algorithms

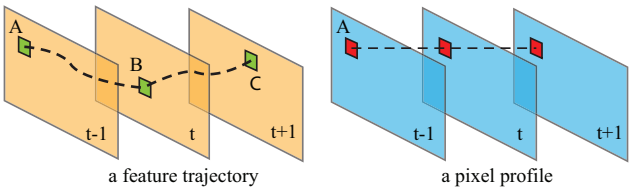


Figure 1. Feature trajectory vs. pixel profile. A feature trajectory tracks a scene point while a pixel profile collects motion vectors at the same pixel location.

from a projective 3D reconstruction with an uncalibrated camera. Liu et al.[14] developed a 3D stabilization system with 'content-preserving' warping. Zhou et al. [26] further introduced plane constraints to the system. Liu et al. [17] used a depth camera for robust stabilization. Generally speaking, these 3D methods produce superior results, especially on scenes with non-trivial depth changes. However, the requirement of 3D reconstruction (or a depth sensor) is demanding. Some recent methods relax this requirement by exploiting partial 3D information embedded in long feature tracks. Liu et al.[15] smoothed basis of feature trajectories in some subspaces for stabilization. Wang et al.[25] represented each trajectory as a Bezier curve and smoothed with a spatial-temporal optimization. Goldstein and Fattal[7] used 'epipolar transfer' to alleviate the strain on long feature tracks. Smoothing feature track is also used to stabilize a light-field camera [24]. Recently, Liu et al. [16] extended the subspace method to deal with stereoscopic videos. Nearly all 3D methods rely on robust feature tracking for stabilization. Long feature trajectories are difficult to obtain. We demonstrate that the effect of smoothing feature trajectories can be well approximated by smoothing pixel profiles, which do not require brittle tracking.

3. SteadyFlow Model

In this section, we introduce the concept of pixel profiles. We will further explain why a shaky video can be stabilized by directly smoothing the pixel profiles of the SteadyFlow. Then we demonstrate the SteadyFlow model and the advantages over feature trajectories.

3.1. Pixel Profiles vs. Feature Trajectories

A *pixel profile* consists of motion vectors collected at the same *pixel location*. In comparison, a feature trajectory follows the motion of a scene point. Figure 1 shows a feature trajectory and a pixel profile starting at the pixel A in frame $t-1$. The feature trajectory follows the movement from pixel A in frame $t-1$ to pixel B in frame t, and then to pixel C in frame $t+1$. In comparison, the pixel profile collects motions at a fixed pixel location A over time.

Figure 2. A simple static scene (from [7]) with gradual depth variations and its optical flow. This video can be stabilized by smoothing all the pixel profiles extracted from its optical flow.

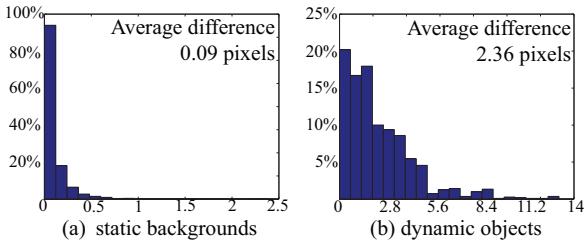


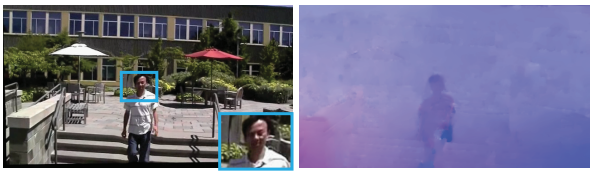
Figure 3. Histogram of the difference between feature trajectories and pixel profiles on static backgrounds and dynamic objects.

3.2. Stabilization by Smoothing Pixel Profiles

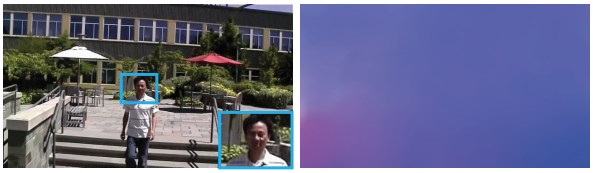
We begin with a simple example. Figure 2 shows an video of static scene with gradual depth changes. Its optical flow is spatially smooth as shown on the right side. We simply smooth all the pixel profiles extracted at every pixel location (the technique of smoothing will be presented in Section 5). In this way, we can obtain a well stabled output video. This suggests that a video can be stabilized by smoothing pixel profiles.

To understand that, we examine 108 videos in a publicly available dataset[1]. We compute optical flows between all consecutive frames on these videos. We also run a KLT tracker[19] to all videos to get feature trajectories. We further manually mark out moving objects in all video frames assisted by Adobe After Effect CS6 Roto brush. In this way, we collect 14,662 trajectories on static backgrounds and 5,595 trajectories on dynamic objects with the length no less than 60 frames. We compare the difference between a feature trajectory and the pixel profile which begins from the starting point of the trajectory. The difference is evaluated as the average of all motion vector differences between the feature trajectory and the pixel profile at corresponding frames. The histogram of this difference for all trajectories is shown in Figure 3. In Figure 3(a), we can see over 90% of feature trajectories on static backgrounds are very similar to their corresponding pixel profiles (less than 0.1-pixel motion difference). This suggests that smoothing the feature trajectories can be well approximated by smoothing the pixel profiles. In comparison, as shown in Figure 3 (b), the difference between a feature trajectory and its corresponding pixel profile is large on moving objects.

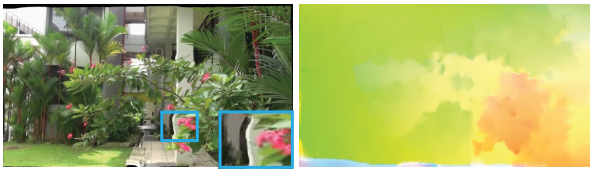
[1] http://137.132.165.105/SIGGRAPH2013/database.html



(a) smoothing pixel profiles collected from raw optical flow (with dynamic object)



(b) smoothing pixel profiles collected from our SteadyFlow



(c) smoothing pixel profiles collected from raw optical flow (with depth edge)

(d) smoothing pixel profiles collected from our SteadyFlow

Figure 4. Comparisons between optical flow and our SteadyFlow. (a) and (c): On the left side, we show the videos stabilized by smoothing the pixel profiles according to the raw optical flow. Please see the distortions highlighted in close-up views. The optical flow field is visualized on the right side. (b) and (d): Corresponding results according to our SteadyFlow.

3.3. SteadyFlow

The analysis in Figure 3 (b) suggests that pixel profiles can be very different from feature trajectories sometimes. In Figure 4 (a) and (c), we show two videos with more complicated optical flow fields to study this problem further. As we can see, the flow vectors are discontinuous on the walking person and strong depth edges. If we smooth the pixel profiles of the raw optical flow, we observe severe image distortions, as illustrated in the close-up views. This indicates that smoothing pixel profile generates poor results on discontinuous flows.

We seek to modify the raw optical flow to get a SteadyFlow. The SteadyFlow should satisfy two properties. First, it should be close to the raw optical flow. Second, it should be spatially smooth to avoid distortions. With these properties, a video can be stabilized by smoothing all its pixel profiles collected from the SteadyFlow. In Figure 4 (b) and (d), we show the results by smoothing the pixel profiles generated from the SteadyFlow (shown on the right side). The results are free from artifacts.

Note that a simple Gaussian smoothing of the raw optical flow is insufficient, as the smoothing will propagate the motions on the moving objects to the background, which decreases the frame registration accuracy and generates temporal wobbles nearby the moving object. Instead, we identify, discard discontinuous flow vectors, and fill in missing flows to satisfy the two desired properties of SteadyFlow. The details will be presented in Section 4.2 and Section 4.3.

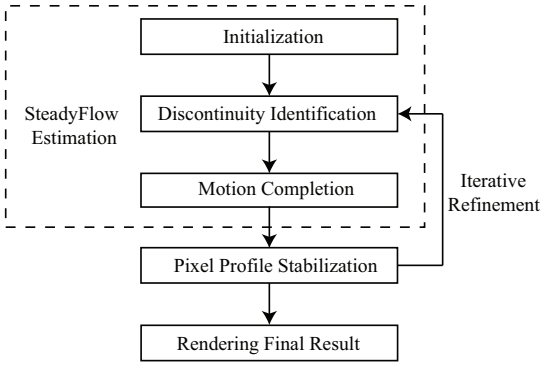


Figure 5. Pipeline of SteadyFlow stabilization.

3.4. Advantages over Feature Trajectories

In video stabilization, the pixel profiles are superior to the feature trajectories for several reasons. First, the pixel profiles are spatially and temporally dense. In comparison, feature trajectories are sparse, unevenly distributed, and would reach out of the video frame. So it is much harder to design a good filter to smooth feature trajectories. Second, accurate long feature trajectories are difficult to obtain. Though we might get dense feature trajectories by frame-by-frame tracing optical flow, these feature trajectories suffer from significant drifting errors [4]. Third, smoothing feature trajectories independently would introduce severe distortions. Some extra constraints (*e.g.*, subspace projection [15]) are required before smoothing. In comparison, as we will see later, pixel profiles can be smoothed individually as long as the flow field is spatially smooth.

Pixel profiles rely on the quality of optical flows. Optical flow estimation is often imprecise at textureless regions and object boundaries. In most of the time, textureless regions have few structure, so that they introduce little visible distortions. The inaccuracy of flows at object boundaries is largely alleviated by our discontinuity abolition and motion completion.

4. SteadyFlow Estimation

Our stabilization system pipeline is illustrated in Figure 5. We first initialize the SteadyFlow by a robust optical flow estimation. To enforce spatial smoothness, we then identify discontinuous motion vectors and overwrite them by interpolating the motion vectors from neighboring pixels. Then, pixel profiles based stabilization is applied on the SteadyFlow. We adopt an iterative approach to increase

the accuracy of SteadyFlow estimation. The final result is rendered according to the stabilized SteadyFlow.

4.1. Initialization

We initialize the SteadyFlow with a robust optical flow estimation. We first estimate a global homography transformation from the matched KLT features [19] between two video frames. We align them accordingly, and then apply the optical flow algorithm described in [13] to compute the residual motion field. The SteadyFlow is initialized as the summation of the residual optical flow and the motion displacements introduced by the global homography.

4.2. Discontinuity Identification

A possible solution to detect different motions is to adopt the motion segmentation techniques [23]. However, motion segmentation itself is a difficult problem. Many methods require long feature trajectories. Though there are two-frame-based motion segmentation techniques[5, 22], typically it is still challenging to deal with large foreground objects due to insufficient motion contrast between neighboring frames.

We introduce a novel spatial-temporal analysis to identify pixels with discontinuous flow vectors. These pixels are viewed as 'outlier' pixels. We use an outliers mask $\mathbf{M}_t(p)$ to record if pixel p at frame t is 'outlier' (*i.e.*, $\mathbf{M}_t(p) = 0$) or not (ie, $\mathbf{M}_t(p) = 1$). In the spatial domain, we threshold the gradient magnitude of raw optical flow to identify discontinuous regions. Once the magnitude at p is larger than the threshold (0.1 in our experiment), p is considered as 'outlier'. The spatial analysis can only detect boundary pixels on moving objects, because the motion vectors within a moving object is often coherent, though they are different from the motions on the background. Therefore, we will further adopt a temporal analysis to identify them.

The temporal analysis examines the accumulated motion vectors $c_t(p) = \sum_t u_t(p)$, where $u_t(p)$ is the motion vector on pixel p at frame t, to decide if p is 'outlier'. It is based on the observation that, in a stable video, the accumulated motion vectors $c_t(p)$ should be smooth over time, except on moving objects and strong depth edges. Figure 6 shows a stabilized video and the accumulated motion vectors at two pixel positions. The pixel (marked by a white star) always lies on the static background. Its accumulated motion vectors generate a smooth trajectory over time (shown in Figure 6 (b)). In comparison, as shown in Figure 6 (a), the trajectory of accumulated motion vectors at the other pixel (marked by a white dot) has significant amount of high frequencies at the beginning, because a moving person passes through that pixel in the first few frames. Its trajectory becomes smooth when the person moves away. We compute the outlier mask $\mathbf{M}_t(p)$ as:

$$\mathbf{M}_t(p) = \begin{cases} 0, & (\|c_t(p) - \mathcal{G} \otimes c_t(p)\| > \varepsilon) \\ 1, & otherwise. \end{cases} \quad (1)$$

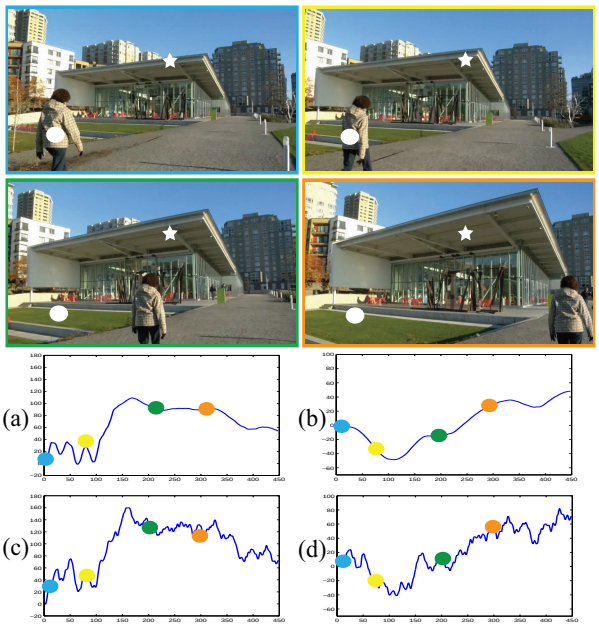


Figure 6. We identify discontinuous motion vectors by analyzing if the trajectory of accumulated motion vectors on a pixel profile is temporally smooth. We show four frames from a stabilized video. (a) and (b) are the trajectories of the accumulated motion vectors evaluated at the pixels marked by white dot and white star. (c) and (d) are the trajectories at the corresponding positions on the input video. The temporal locations of these 4 frames are denoted in the trajectories by dots with the same color as the frame border.

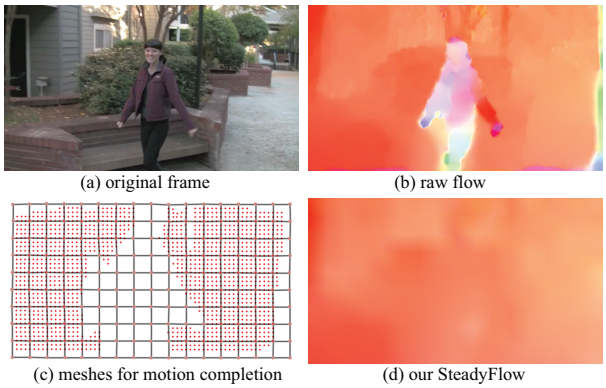
where $\mathcal{G}$ is a Gaussian filter (with default standard deviation 3) and ε uses adaptive threshold (described in Section 4.4).

4.3. Motion Completion

We collect all the pixels with discontinuous motions to form a outlier mask. Motion vectors within the mask are discarded. We then complete it in a similar way as [17] by the 'as-similar-as-possible' warping [14, 15]. Basically, we take the pixels on the mask boundary as control points, and fill in the motion field by warping 2D meshes grids with the grid size 40×40 pixels. Mathematically, it amounts to minimizing the energy $E(V) = E_d(V) + E_s(V)$. We take the same smoothness E_s as described in [14] to maintain the rigidity of the grid. The data term E_d is defined as:

$$E_d(V) = \sum_p \mathbf{M}(p) \cdot ||V\pi_p - (p + u_p)||. \quad (2)$$

Here,the grids vertices are indicated by V. The vector u_p is the initial optical flow at the pixel p, such that $(p, p + u_p)$ form a pair of control points. The parameter π_p is the bilinear coordinate, *i.e.*, $p = V_p\pi_p$, where V_p is the 4 grid vertices enclosing p. For more detailed explanation and justification, please refer to [17, 14]. This energy is minimized by solving a sparse linear equations system. We use bilinear interpolation to compute the motion vector of every pixel



(a) original frame (b) raw flow (c) meshes for motion completion (d) our SteadyFlow

Figure 7. Example of motion completion. (a) A frame from input video. (b) The raw optical flow. (c) Warped mesh estimated from background samples. The white region shows the outlier mask. (d) SteadyFlow after rewrite the discontinuous motion vectors.

according to the motion of the grid vertices.

Figure 7 shows the estimated SteadyFlow. The missing regions in the flow field (white regions in Figure 7 (c)) corresponds to dynamic objects,depth edges (*e.g.*, flows on tree branches) and image boundary pixels with inaccurate raw optical flows. The motion vectors in the missing regions are interpolated from their neighboring pixels. In this way, we generate the SteadyFlow as shown in Figure 7 (d).

The raw optical flow field might also be smoothed by strong Gaussian smooth. However, Gaussian smooth propagates the foreground motion to background pixels. This makes the frame registration fail at background and causes strong temporal wobble artifacts in the stabilized video.

4.4. Iterative Refinement

Note that our temporal analysis for the estimation of outliers mask requires a stable video. As shown in Figure 6 (c) and (d), the trajectories generated on the original shaky video is discontinuous everywhere. In practice, we obtain an initial outlier mask $\mathbf{M}_t$ estimated from the shaky video only by spatial analysis of discontinuous flow vectors. Then we apply an iterative scheme to alternatively refine the outlier mask $\mathbf{M}_t$. At each iteration, the first step is to exclude outliers and fill in the missing regions of the input SteadyFlow according to the mask $\mathbf{M}_t$. The motion completion is described in Section 4.3. The second step is to stabilize the SteadyFlow, which will be described in Section 5. In the third step, the stabilized SteadyFlow is then used to further refine $\mathbf{M}_t$ by temporal analysis of discontinuous flow vectors as described in Section 4.2. Since our temporal analysis is more suitable for stable videos, we may consider adaptive threshold $(1+\alpha^{1/n})\varepsilon$ used in Equation 1 to assign a conservative threshold in the beginning. Here, n is the iteration index and $\alpha = 20, \varepsilon = 0.2$ is used in our experiment. We iterate the whole three steps to finally generate the stabilized result. We use 5 iterations in our experiments empirically.

5. Pixel Profiles based Stabilization

We here derive the stabilization algorithm that smoothes the pixel profiles extracted from the SteadyFlow. Let $\mathbf{U}_t$, $\mathbf{S}_t$ be the SteadyFlow estimated from frame t to frame $t-1$ in the input video and stabilized video respectively. The smoothing is achieved by minimizing the following objective function similar to [18]:

$$\mathcal{O}(\{\mathbf{P}_t\}) = \sum_t \left(\|\mathbf{P}_t - \mathbf{C}_t\|^2 + \lambda \sum_{r\in\Omega_t} w_{t,r} \|\mathbf{P}_t - \mathbf{P}_r\|^2 \right), \tag{3}$$

where $\mathbf{C}_t = \sum_t \mathbf{U}_t$ is the field of accumulated motion vectors of the input video. Similarly, we have $\mathbf{P}_t = \sum_t \mathbf{S}_t$ of the stabilized video. The first term requires the stabilized video staying close to its original to avoid excessive cropping, while the second term enforces temporal smoothness.

There are three differences from path optimization in [18]. First, since SteadyFlow itself enforces strong spatial smoothness, we do not require any spatial smoothness constraint in Equation 3. Second, the weight $w_{t,r}$ only involves the spatial Gaussian function $w_{t,r} = exp(-||r - t||^2/(\Omega_t/3)^2)$ rather than a bilateral weight. To adaptively handle different motion magnitudes, we adopt an adaptive temporal window Ω_t in our smoothing (to be discussed in Section 5.1). Third, the $\mathbf{P}$ and $\mathbf{C}$ here are non-parametric accumulated motion vectors instead of parametric models (*i.e.*, homographies).

Likewise, we can further obtain iterative solution by:

$$\mathbf{P}_t^{(\xi+1)} = \frac{1}{\gamma} \left(\mathbf{C}_t + \lambda \sum_{r\in\Omega_t, r\neq t} w_{t,r} \mathbf{P}_r^{(\xi)} \right), \tag{4}$$

where the scalar $\gamma = 1 + \lambda \sum_r w_{t,r}$ and ξ is an iteration index (by default, $\xi = 10$). After optimization, we will warp the original input video frame to the stabilized frame by a dense flow field $\mathbf{B}_t = \mathbf{P}_t - \mathbf{C}_t$. We can further derive the relationship between $\mathbf{B}_t$ and $\mathbf{U}_t$, $\mathbf{S}_t$ as:

$$\mathbf{U}_t + \mathbf{B}_{t-1} = \mathbf{B}_t + \mathbf{S}_t \Rightarrow \mathbf{S}_t = \mathbf{U}_t + \mathbf{B}_{t-1} - \mathbf{B}_t. \tag{5}$$

5.1. Adaptive Window Selection

Our smoothing technique requires a feature trajectories to be similar to its corresponding pixel profile within the temple window Ω_t. We adaptively adjust the size of Ω_t to deal with motion velocity changes in the video. Specifically, as shown in Figure 8, the SteadyFlow is assumed to be spatially smooth within a window (denoted by the yellow box) of the size$(2\tau+1)\times(2\tau+1)$, centered at pixel A. Within the window, smoothing the feature trajectory (denoted by the solid red line) can be well approximated by

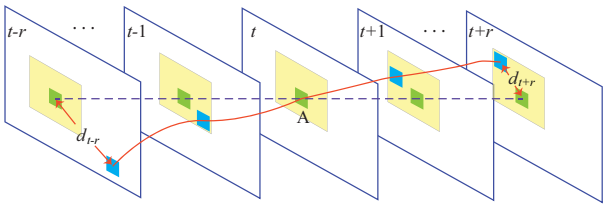


Figure 8. Estimation of adaptive temporal window size Ω_t in Equation 3. The window size Ω_t is selected such that the feature trajectory (denoted by the red line) is always within the predetermined yellow box.

smoothing the pixel profile (denoted by the dish blue line). Once the trajectory goes outside the window, *i.e.*, $d_{t-r} > \tau$ ($\tau = 20$ in our implementation), it would introduce non-negligible errors to the approximation. So we estimate Ω_t for each pixel in a pixel profile to ensure the feature trajectory started at that pixel is within $(2\tau+1)\times(2\tau+1)$ for all frames in Ω_t. The feature trajectory here is approximated by tracing the optical flows. For instance, in Figure 8, the window for point A is $\Omega_t(A) = [t-1, t+r]$. To avoid spatial distortion, it is necessary to choose a global smooth window Ω_t for all pixels in the frame t. So we take the intersection of the windows at all pixels to determine the final temporal support for frame t. With the help of dynamic window, we can handle videos with quick camera motion e.g. quick rotation, fast zooming.

6. Results

We evaluated our method on some challenging examples from publicly available videos in prior publications to facilitate comparisons. These example videos include large parallax, dynamic objects, large depth variations, and rolling shutter effects. All comparisons and results are provided in the project page[2].

Our system takes 1.5 second to process a video frame (640×360 pixels) on a laptop with 2.3GHz CPU and 4G RAM. The computation bottleneck is the optical flow estimation (1.1 second per frame), which could be significantly speed up by GPU implementations. Our outliers mask estimation takes 0.34 second on each frame. It is independent per-pixel computation and can be parallelized easily.

Videos with Large Dynamic Objects This is a challenging case for previous 2D video stabilization methods. A large portion of corresponding image features are on the foreground moving objects. Previous methods often rely on RANSAC to exclude these points to estimate the background 2D motion.

Figure 9 shows a synthetic example. We compared our method with a simple 2D technique that adopts homography fitting with RANSAC for motion estimation. In Figure 9 (a), we can see that RANSAC cannot exclude all the outliers, which cause distortions in the results as shown in (c). In comparison, our SteadyFlow estimation can exclude all the undesirable motion vectors on the foreground object (see Figure 9 (b)) and produce better stabilization result in (d). To further know how our SteadyFlow estimation extract outlier masks for this example, Figure 10 shows the intermediate masks at each iteration.

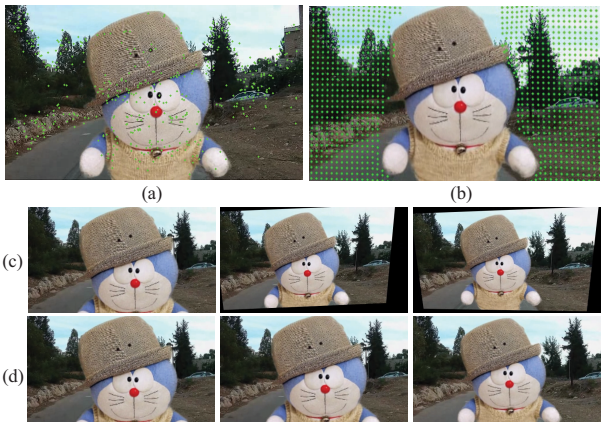


Figure 9. Comparison with single homography based stabilization. (a) Inliers after RANSAC based homography fitting. (b) Inlier motion vectors after our outlier mask detection. (c) and (d) are results from the single homography based method and our method respectively.

In addition, we borrow four videos (shown in Figure 11) with remarkable dynamic objects from [15], [7] and [18], which are reported as failure cases. The large moving object (a person) in the first video (shown in Figure 11 (a)) breaks feature trajectories, and makes feature-track-based method (like [15]) fail. The examples in Figure 11 (b) and (c) clearly consist of two motion layers. For both examples, our method can identify distracting foreground objects despite their large image size. This ensures a successful SteadyFlow estimation and superior stabilization results. The recent 'Bundled-Paths' method [18] fits a 2D grid of homographies to model more complicated 2D motion. This method enforces stronger spatial smoothness at dynamic scenes, which reduces their model representation capability. Thus, it produces artifacts on the example shown in Figure 11 (d). In comparison, our SteadyFlow is powerful to exclude dynamic objects and can maintain the ability of modelling complicated motion. As a result, we can produce better results.

Videos with Large Depth Change We further evaluate our method on two videos with large depth changes, one video come from [14] and another captured by ourselves. Our 2D method achieved results of similar visual quality to 3D method. The video thumbnails are shown in Fig-

[2]http://137.132.165.105/CVPR2014/index.html

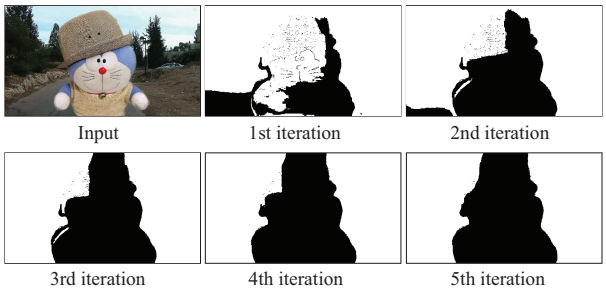


Figure 10. Estimated masks during each iteration of optimization on a synthetic example.

Figure 11. Failure examples reported in (a) and (b) Subspace stabilization [15], (b) Epipolar [7], (d) Bundled-Paths [18].

Figure 12. Two videos with large depth change for the comparison with traditional 2D stabilization.

Figure 13. Two rolling shutter videos borrowed from [8].

ure 12. We compared our results with that of a traditional 2D method [20] (using our implementation). As can be seen from the accompany video, the results from [20] contain jitters at some image regions. We further compare with indoor videos captured by Kinect[17]. Please refer to our project page for more comparisons.

Videos with Rolling Shutter Effects Rolling shutter effects of CMOS sensors cause spatial variant motions in videos. Our method can model rolling shutter effects as spatially variant high frequency jitters. It can simultaneously rectify rolling shutter effects when smoothing camera shakes. Figure 13 shows two rolling shutter videos borrowed from [8]. Our method produced similar quality as other state-of-art techniques [1, 11, 8, 18].

Figure 14. Comparison with YouTube Stabilizer. The red arrow indicates structure distortions in YouTube results.

Figure 15. Comparison with Adobe After Effects CS6 'Warp Stabilizer'. We can notice the global shearing/skewing in 'Warp Stabilizer' results.

Comparison with State-of-art System We further compared our system with two well-known commercial systems on our captured videos. One system is the YouTube Stabilizer, which is built upon the L_1-optimization method [9] and the homography mixture method [8]. We uploaded our videos to YouTube and downloaded the automatically stabilized results. Another system is the Adobe After Effects CS6 'Warp Stabilizer', which is based on the subspace stabilization method [15]. Since it is an interactive tool, we try our best to generate results with the best perceptual quality.

Figure 14 shows the comparison with YouTube Stabilizer. We can see remarkable structure distortions at the pole, which has discontinuous depth changes. In comparison, our SteadyFlow estimation masks out these depth changes and fill in by the neighboring motions. Thus our result is stable and free from distortions.

Figure 15 shows the comparison with the 'Warp Stabilizer' in After Effects CS6. In this example, the moving train makes the feature-trajectory-based subspace analysis fail. As a result, shearing/skewing distortions are visible in their result. Our SteadyFlow estimation excludes motion vectors on the train to obtain a spatially coherent motion field for stabilization. Our result is free from distortions, though it might not be physically correct.

7. Limitations

During the experiment, we noticed that the size of the foreground is crucial to a successful result. Our spatial-

Figure 16. Failure cases. Videos contain dominant foreground.

temporal analysis fails to distinguish foreground and background when videos contain dominant foreground objects. These objects consistently occupy more than half area of a frame and exsit for a long time. The stabilization will be applied on the foreground instead of background, or keep switching. Figure 16 shows two failure cases and some more in the project page.

8. Conclusion

We propose a novel motion representation, SteadyFlow, for video stabilization. Due to the strong spatial coherence in the SteadyFlow, we can simply smooth each motion profile independently without considering the spatial smoothness [18] or subspace constraint [17]. Our method is more robust than previous 2D or 3D methods. Its general motion model allows stabilizing challenging videos with large parallax, dynamic objects, rolling-shutter effects, etc.

9. Acknowledgement

This work is partially supported by the ASTAR PSF project R-263-000-698-305.

References

[1] S. Baker, E. P. Bennett, S. B. Kang, and R. Szeliski. Removing rolling shutter wobble. In *Proc. CVPR*, 2010. 7

[2] C. Buehler, M. Bosse, and L. McMillan. Non-metric image-based rendering for video stabilization. In *Proc. CVPR*, 2001. 1

[3] B.-Y. Chen, K.-Y. Lee, W.-T. Huang, and J.-S. Lin. Capturing intention-based full-frame video stabilization. *Computer Graphics Forum*, 27(7):1805–1814, 2008. 1

[4] T. Crivelli, P.-H. Conze, P. Robert, M. Fradet, and P. Pérez. Multi-step flow fusion: towards accurate and dense correspondences in long video shots. In *Proc. BMVC*, 2012. 3

[5] R. Dragon, Hannover, B. Rosenhahn, and J. Ostermann. Multi-scale clustering of frame-to-frame correspondences for motion segmentation. In *Proc. ECCV*, 2012. 4

[6] M. L. Gleicher and F. Liu. Re-cinematography: Improving the camera dynamics of casual video. In *Proc. of ACM Multimedia*, 2007. 1

[7] A. Goldstein and R. Fattal. Video stabilization using epipolar geometry. *ACM Trans. Graph. (TOG)*, 31(5):126:1–126:10, 2012. 1, 2, 6, 7

[8] M. Grundmann, V. Kwatra, D. Castro, and I. Essa. Calibration-free rolling shutter removal. In *Proc. ICCP*, 2012. 1, 7

[9] M. Grundmann, V. Kwatra, and I. Essa. Auto-directed video stabilization with robust l1 optimal camera paths. In *Proc. CVPR*, 2011. 1, 7

[10] A. Karpenko, D. Jacobs, J. Baek, and M. Levoy. Digital video stabilization and rolling shutter correction using gyroscopes. In *Stanford CS Tech Report*, 2011. 1

[11] A. Karpenko, D. E. Jacobs, J. Baek, and M. Levoy. Digital video stabilization and rolling shutter correction using gyroscopes. In *Stanford Computer Science Tech Report CSTR 2011-03*, 2011. 7

[12] K.-Y. Lee, Y.-Y. Chuang, B.-Y. Chen, and M. Ouhyoung. Video stabilization using robust feature trajectories. In *Proc. ICCV*, 2009. 1

[13] C. Liu. Beyond pixels: Exploring new representations and applications for motion analysis. *Doctoral Thesis. Massachusetts Institute of Technology*, 2009. 4

[14] F. Liu, M. Gleicher, H. Jin, and A. Agarwala. Content-preserving warps for 3d video stabilization. *ACM Trans. Graph. (Proc. of SIGGRAPH)*, 28, 2009. 2, 4, 6

[15] F. Liu, M. Gleicher, J. Wang, H. Jin, and A. Agarwala. Subspace video stabilization. *ACM Trans. Graph.*, 30, 2011. 1, 2, 3, 4, 6, 7

[16] F. Liu, Y. Niu, and H. Jin. Joint subspace stabilization for stereoscopic video. In *Proc. ICCV*, 2013. 2

[17] S. Liu, Y. Wang, L. Yuan, J. Bu, P. Tan, and J. Sun. Video stabilization with a depth camera. In *Proc. CVPR*, 2012. 2, 4, 7, 8

[18] S. Liu, L. Yuan, P. Tan, and J. Sun. Bundled camera paths for video stabilization. *ACM Trans. Graph. (Proc. of SIGGRAPH)*, 32(4), 2013. 1, 5, 6, 7, 8

[19] B. D. Lucas and T. Kanade. An iterative image registration technique with an application to stereo vision. In *Proc. of IJCAI*, 1981. 2, 4

[20] Y. Matsushita, E. Ofek, W. Ge, X. Tang, and H.-Y. Shum. Full-frame video stabilization with motion inpainting. *IEEE Trans. Pattern Anal. Mach. Intell.*, 28:1150–1163, 2006. 1, 7

[21] C. Morimoto and R. Chellappa. Evaluation of image stabilization algorithms. In *Proc. of IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 2789 – 2792, 1998. 1

[22] M. Narayana, A. Hanson, and E. Learned-Miller. Coherent motion segmentation in moving camera videos using optical flow orientations. In *Proc. ICCV*, 2013. 4

[23] J. Shi and J. Malik. Motion segmentation and tracking using normalized cuts. In *Proc. ICCV*, 1998. 4

[24] B. M. Smith, L. Zhang, H. Jin, and A. Agarwala. Light field video stabilization. In *Proc. ICCV*, 2009. 2

[25] Y.-S. Wang, F. Liu, P.-S. Hsu, and T.-Y. Lee. Spatially and temporally optimized video stabilization. *IEEE Trans. on Visualization and Computer Graphics*, 17:1354–1361, 2013. 1, 2

[26] Z. Zhou, H. Jin, and Y. Ma. Plane-based content-preserving warps for video stabilization. In *Proc. CVPR*, 2013. 2