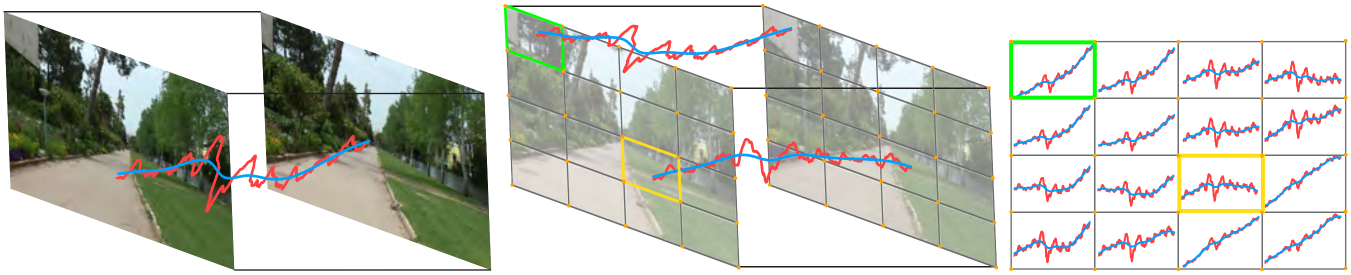
Bundled Camera Paths for Video Stabilization

Shuaicheng Liu* Lu Yuan† Ping Tan* Jian Sun†

*National University of Singapore †Microsoft Research Asia



(a) a single global path (b) our bundled paths

Figure 1: *Comparison between traditional 2D stabilization (a single global camera path) and our bundled camera paths stabilization. We plot the camera trajectories (visualized by the y-axis translation over time) and show the original path (*red*) and the smoothed path (*blue*) for both methods. Our bundled paths rely on a 2D mesh-based motion representation, and are smoothed in space-time.*

Abstract

We present a novel video stabilization method which models camera motion with a bundle of (multiple) camera paths. The proposed model is based on a mesh-based, spatially-variant motion representation and an adaptive, space-time path optimization. Our motion representation allows us to fundamentally handle parallax and rolling shutter effects while it does not require long feature trajectories or sparse 3D reconstruction. We introduce the 'as-similar-as-possible' idea to make motion estimation more robust. Our space-time path smoothing adaptively adjusts smoothness strength by considering discontinuities, cropping size and geometrical distortion in a unified optimization framework. The evaluation on a large variety of consumer videos demonstrates the merits of our method.

CR Categories: I.4.3 [Image Processing and Computer Vision]: Enhancement—Registration

Keywords: video stabilization, image warping, camera paths

Links: DL PDF

1 Introduction

A video captured with a hand-held device (*e.g.*, a cell-phone or a portable camcorder) often appears remarkably shaky and undirected. Digital video stabilization improves the video quality by removing unwanted camera motion. It is of great practical importance because the devices (mobile phones, tablets, camcorders) capable of capturing video have become widespread and online sharing is so ubiquitous.

Prior video stabilization methods synthesized a new stabilized video by estimating and smoothing 2D camera motion [Matsushita et al. 2006; Grundmann et al. 2011] or 3D camera motion [Liu et al. 2009; Liu et al. 2012]. In general, 2D methods are more robust and faster because they only estimate a linear transformation (affine or homography) between consecutive frames. But the 2D linear motion model is too weak to fundamentally handle the parallax caused by non-trivial depth variation in the scene. On the contrary, the 3D methods can deal with the parallax in principle and generate strongly stabilized results. However, their motion model estimation is less robust to various degenerations such as feature tracking failure, motion blur, camera zooming, and rapid rotation. Briefly, 2D methods are more robust but may sacrifice quality (*e.g.*, introducing unpleasant geometrical distortion or producing less stabilized output), while 3D methods can achieve high-quality results but are more fragile.

Some recent methods [Liu et al. 2011; Goldstein and Fattal 2012] have successfully combined the advantages of these two kinds of methods. Liu et al. [2011] applied a low-rank, subspace constraint on 2D feature trajectories, which is an effective simplification of 3D reconstruction. Goldstein and Fattal [2012] avoided 3D reconstruction by exploiting the 'epipolar transfer' technique. These methods relax the requirement from 3D reconstruction to 2D long feature tracking. Nevertheless, requiring long feature tracking (typically over 20 frames) makes it difficult to handle more challenging cases (*e.g.*, rapid motion, fast scene transition, large occlusion) in the consumer videos.

This paper aims at the same goal of robust high-quality result but from an opposite direction: we propose a more powerful 2D camera motion model. Specifically, we present *bundled camera paths* model which maintains multiple, spatially-variant camera paths. In other words, each different location in the video has its own camera path. This flexible model allows us to fundamentally deal with nonlinear motion caused by parallax and rolling shutter effects [Liang et al. 2008; Baker et al. 2010; Grundmann et al. 2012]. At the same time, the model enjoys the robustness and simplicity of 2D methods, because it only requires feature correspondences between two consecutive frames.

Our bundled camera paths model is built on two novel components: a warping-based motion representation (and estimation), and an adaptive space-time path smoothing. The *first component* represents the motion between two consecutive frames by mesh-based, spatially-variant homographies (Figure 1(b)) with a 'as-similar-as-possible' regularization constraint [Igarashi et al. 2005; Schaefer et al. 2006]. This constraint is critical because estimating a model with such a high degree of freedom is usually risky in the cases of insufficient features or large occlusions. To the best of our knowledge, this is the first work to employ the mesh-based 'as-similar-as-possible' regularization for spatially-variant motion estimation in video stabilization. Notice that the 'as-similar-as-possible' warping was used in [Liu et al. 2009; Liu et al. 2011] for video stabilization. But we directly use the mesh vertices as the motion model itself. No intermediate representation is used, such as 3D reconstruction [Liu et al. 2009] or subspace [Liu et al. 2011].

Based on the proposed motion representation, we construct a bundle of camera paths, each of which is the concatenation of local homographies at the same grid cell over time (Figure 1(b)). Our *second component* smooths all bundled camera paths as a whole to maintain both spatial and temporal coherences. Furthermore, to avoid excessive cropping/geometrical distortion and approximate cinematography favored path, we adopt a discontinuity-preserving idea similar to bilateral filtering [Tomasi and Manduchi 1998] to adaptively control the strength of smoothing.

For a quantitative evaluation, we provide a comprehensive dataset (including both public examples and our own video clips of different kinds of motions). We show that our new 2D method is comparable to or outperforms other competitive 2D or 3D methods.

2 Related Work

2D Methods estimate 2D transformations between consecutive video frames and smooth them over time to generate a steady video. Most previously developed methods apply an affine or homography model, and focus on the design of the smoothing algorithm. Earlier works [Morimoto and Chellappa 1998; Matsushita et al. 2006] apply low-pass filters to individual model parameters. Some methods assume prior motion models such as polynomial curves [Chen et al. 2008] for desired camera trajectories. Gleicher and Liu [2007] divide the original camera trajectory into multiple segments for subsequent individual smoothing. More recently, Grundmann et al. [2011] gracefully apply L_1-norm optimization to generate a camera path consisting of constant, linear and parabolic motions, which follow cinematography rules. Grundmann et al. [2012] further adopt a homography-array-based motion model to deal with rolling shutter effects. These two techniques have been integrated into Google YouTube. It is robust, follows cinematography rules, and performs well on many consumer videos.

Our method belongs to this category. But we use a spatially-variant model to represent the motion between video frames and design an appropriate smoothing technique for this model.

3D Methods often rely on robust feature tracking for stabilization. Beuhler et al. [2001] perform stabilization with a projective 3D reconstruction of the scene from an uncalibrated camera. Liu et al. [2009] develop the first successful 3D video stabilization system and are the first to introduce 'content-preserving' warping for stabilization.

Since 3D reconstruction is difficult, recent methods directly smooth the trajectories of tracked features. Liu et al. [2011] smooth some basis trajectories (preferably longer than 50 frames) of the subspace formed by the feature tracks. This method achieves similar quality to 3D reconstruction-based methods, while reducing the requirement from 3D reconstruction to long feature tracking. It has been transferred to Adobe After Effects as a feature called "Warp Stabilizer". Goldstein and Fattal [2012] utilize an "epipolar transfer" technique to avoid the fragile 3D reconstruction. This technique also alleviates the strain on long feature tracks. But it still requires moderate feature track length (typically over 20 frames). Feature track smoothing is also used in light-field camera video stabilization work [Smith et al. 2009]. To address the occlusion issue, Lee et al. [2009] introduce feature pruning to choose robust feature trajectories for smoothing.

Nearly all methods involving feature tracking face a common obstacle – in many consumer videos obtaining long feature tracks is fragile due to occlusion, motion blur or rapid camera motion. Our method does not encounter this issue since it only computes relative motion between consecutive frames.

Motion Estimation computes the transition between two images with view overlap. Optical flow algorithms [Lucas and Kanade 1981] model this transition by individual displacement vectors at every pixel. When there is no parallax, this transition can be represented elegantly by a global homography transformation [Hartley and Zisserman 2003]. Local alignment [Shum and Szeliski 2000] or a dual-homography model [Gao et al. 2011] can reduce alignment error caused by parallax. Szeliski and Shum [1996] represent motion using a mixture of spline models with spatially variant spatial support to facilitate registration. Lin et al. [2011] estimate a smoothly varying affine field to align images of large viewpoint changes. This model can be potentially used for video stabilization. However, its current motion estimation technique is slow (may take 8 minutes to process a 720p frame).

Our motion model is essentially a mesh-based, spatially-variant homography model, inspired by recent image warping techniques [Igarashi et al. 2005; Schaefer et al. 2006; Liu et al. 2009]. We extend the "as-similar-as-possible" idea from image synthesis to motion estimation, and apply it to video stabilization. It is very efficient to estimate our motion model (may take only 50 milliseconds to process a 720p frame).

Rolling Shutter Removal estimates and corrects inter-row motion caused by the row-parallel readout, *i.e.*, electronic rolling shutter [Nakamura 2005] mainly in CMOS sensors. Prior works design different parametric inter-row motion models, including a per-row translation model [Liang et al. 2008; Baker et al. 2010] and 3D rotation model [Forssén and Ringaby 2010]. Recently, Grundmann et al. [2012] proposed a calibration-free homography mixture model, which shows significant improvement. Karpenko et al. [2011] use dedicated hardware – the gyroscope on mobile devices, to correct the rolling shutter effects in real-time.

Similar to [Grundmann et al. 2012], our method corrects rolling shutter effects without any prior calibration. Our warping-based model naturally handles the rolling shutter effects as a special kind of spatially variant motion. So we do not need a separate rolling shutter correction step in our stabilization.

3 Bundled Camera Paths

In this section, we introduce our warping-based motion model and bundled camera paths.

3.1 Warping-based Motion Model

We propose using an image warping model to represent the motion between consecutive video frames, which provides stronger modeling power than conventional single, 2D linear transformations. We adopt the warping model in [Igarashi et al. 2005; Liu et al. 2009],

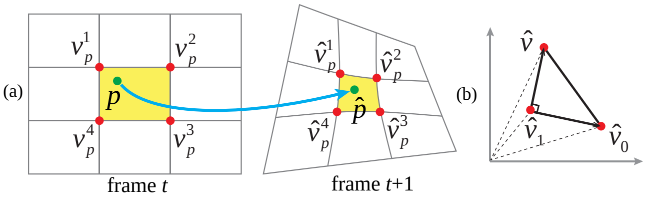


Figure 2: *(a) Parameterization of the motion between two frames by a regular grid mesh, where a pair of matched features (p, $\hat{p}$) should be represented by the same bilinear interpolation of their four enclosing vertices. (b) The as-similar-as-possible term requires each triangle $\hat{v}, \hat{v}_0, \hat{v}_1$ to follow a similarity transformation.*

though more general models such as 'moving-least-square' [Schaefer et al. 2006] or parameterized optical flow [Nir et al. 2008] might be used.

Model At each frame, we define a uniform grid mesh as illustrated in Figure 2. The motion is represented by an (unknown) warping of the grid mesh to register two frames (in fact, their corresponding feature points). We require matched features (*e.g.*, p and $\hat{p}$ in Figure 2) to share the same bilinear interpolation of the four corners of the enclosing grid cell after warping. At the i-th grid cell, the warping from frame t to frame $t+1$ introduces a homography $F_i(t)$, which can be determined from the motion of the four enclosing vertices. Thus, the warping-based motion model is actually a set of spatially-variant homographies on a 2D grid.

Note that this highly flexible model is able to handle parallax. It is between global homography and per-pixel optical flow. However, estimating a model with such a high degree of freedom is very risky because we may not have sufficient features (due to textureless regions or occlusions) in every cell.

Regularization To address this challenge, we propose imposing a shape-preserving (*i.e.*, "as-similar-as-possible" [Igarashi et al. 2005]) constraint. The combination of the shape-preserving and mesh representation together provides two kinds of regularizations: 1) for each cell, the fitted homography should be biased toward a reduced similarity (or rigid) transformation; 2) the intrinsic connection of the mesh (two neighboring mesh cells share two vertices) enforces a first-order continuity constraint. They can help to propagate or fill in information from regions with sufficient features to other regions.

Finally, we estimate the motion by minimizing two energy terms: a data term for matching features, and a shape-preserving term for enforcing regularization.

3.2 Model Estimation

We first describe our basic method by following [Liu et al. 2009], and later extend it for better robustness in the next subsection.

Data term As shown in Figure 2, suppose $\{p, \hat{p}\}$ is the p-th matched feature pair from frame t to frame $t+1$. The feature p can be represented by a 2D bilinear interpolation of the four vertices $V_p = [v_p^1, v_p^2, v_p^3, v_p^4]$ of the enclosing grid cell: $p = V_p w_p$, where $w_p = [w_p^1, w_p^2, w_p^3, w_p^4]^\top$ are interpolation weights that sum to 1. We expect that the corresponding feature $\hat{p}$ can be represented by the same weights of the warped grid vertices $\hat{V}_p = [\hat{v}_p^1, \hat{v}_p^2, \hat{v}_p^3, \hat{v}_p^4]$. Therefore the data term is defined as

$$E_d(\hat{V}) = \sum\nolimits_p ||\hat{V}_p w_p - \hat{p}||^2. \qquad (1)$$

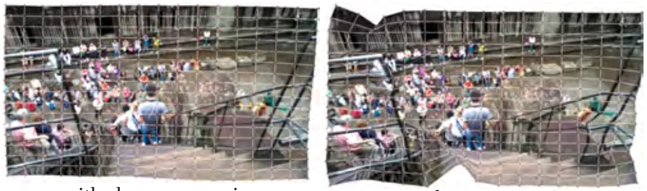


Figure 3: *Comparison of motion estimation with and without the shape-preserving term.*

Here $\hat{V}$ contains all the warped grid vertices. Solving $\hat{V}$ determines the warping of the grid.

Shape-preserving term We use the same shape-preserving term as [Liu et al. 2009] involving all vertices in $\hat{V}$,

$$E_s(\hat{V}) = \sum_{\hat{v}} \|\hat{v} - \hat{v}_1 - sR_{90}(\hat{v}_0 - \hat{v}_1)\|^2, \; R_{90} = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix}, \qquad (2)$$

where $s = \|v - v_1\| / \|v_0 - v_1\|$ is a known scalar computed from the initial mesh. This shape-preserving term requires the triangle of neighboring vertices v, v_0, v_1 to follow a similarity transformation.

Linearly combining two terms forms our final energy $E(\hat{V})$:

$$E(\hat{V}) = E_d(\hat{V}) + \alpha E_s(\hat{V}), \qquad (3)$$

where α is an important weight to control the amount of regularization. We will discuss how to adaptively determine it later. Since the energy $E(\hat{V})$ is quadratic, the warped mesh $\hat{V}$ can be easily solved by a sparse linear system solver.

Estimating homographies After having a new mesh, we can estimate each local homography $F_i(t)$ in the grid cell i of frame t by solving a linear equation:

$$\hat{V}_i = F_i(t) V_i, \qquad (4)$$

where V_i and $\hat{V}_i$ are the four vertices before and after the warping.

Figure 3 shows the warped mesh grid according to the estimated motion. Left and right are the results with and without the shape-preserving term. It is clear that the regularization term helps maintain a smooth varying mesh representation.

3.3 Robust Estimation

We further generalize our motion estimation to make it more robust.

Outlier rejection We reject incorrectly matched features at two scales. At the coarse scale (the whole image), we apply RANSAC algorithm [Fischler and Bolles 1981] to fit a global homography $\bar{F}(t)$ and discard features by a relatively large threshold on fitting error (6% image width). At the fine scale (4 × 4 sub-images), we apply RANSAC again to reject features by a relatively small threshold (2% image width).

Pre-warping To facilitate the warping estimation, we use global homography $\bar{F}(t)$ to bring matching features closer. We then solve the warping to estimate the residual motion, which generates a homography $F_i'(t)$ at each grid cell. The final homography $F_i(t)$ is simply computed as $F_i'(t) \times \bar{F}(t)$. Note that this coarse-to-fine strategy has been used in [Liu et al. 2009] for image synthesis and proven effective in motion estimation literature [Brox et al. 2004].

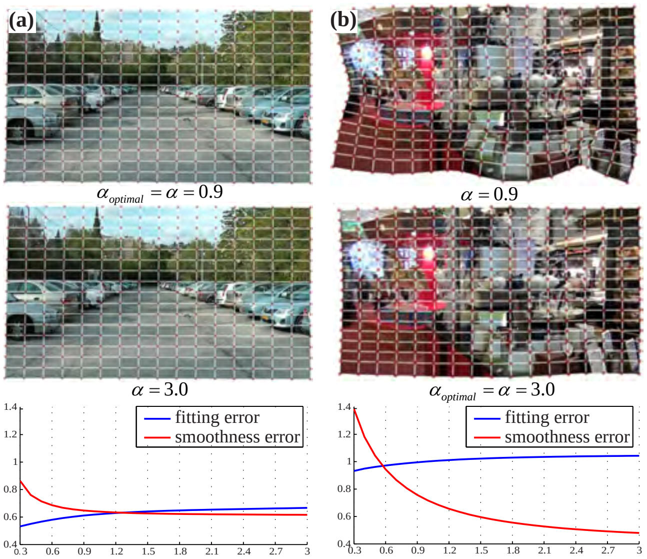


Figure 4: *Our method automatically chooses an appropriate α for different scenes: (a) a scene free of occlusion; (b) a scene with severe occlusion.*

Adaptive regularization A good regularization should be adaptive to image content. For example, if reliable features are uniformly distributed over the whole image, we should trust the data term more and use a smaller weight α in Equation 3 for a weaker regularization. But when there is occlusion or insufficient features, we prefer stronger regularization as the data term is less reliable. To implement this strategy, we adaptively set α per frame, based on two errors: fitting error e_h and smoothness error e_s.

The fitting error e_h is the average residual of the feature matching under the estimated homographies, i.e., $e_h = \frac{1}{n}\sum_p \|F_p \times p - \hat{p}\|^2$, where F_p is the homography in the cell containing p, and n is the number of feature pairs. The smoothness error e_s measures the similarity (L_2 distance) between neighboring local homographies by $e_s = \beta \sum_{j \in \Omega_i} \|F_i - F_j\|^2$, where Ω_i consists of the neighboring cells of i. Here, the homography matrix is normalized so that sum of all its elements is one. We empirically set $\beta = 0.01$, since it makes the scale of e_h and e_s similar on most of the examples. Then we define the combined error as $e = e_h + e_s$. We equally discretize α into 10 values between 0.3 and 3. We perform the model estimation using every discretized value and select the model with minimum error e.

As shown in Figure 4(a), for simple scenes with smooth depth variation, neighboring cells tend to have similar homographies. So we choose a small α(=0.9) to better minimize the data error. On the contrary, for scenes with large occlusion (Figure 4(b)), neighboring local homographies are less similar. The smoothness error can be significantly reduced by increasing α. So our system will automatically choose a large α(=3.0) to ensure consistent local motion.

Finally, we show an example in Figure 5 to verify the strength of the regularization of our method. In this example, we compare two meshes estimated using all features and a subset of features. Two similar results indicate our method can robustly deal with regions of insufficient features.

3.4 Bundled Camera Paths

With estimated local homographies, we can define a bundle of spatially-variant camera paths for the whole video. Let $C_i(t)$ be the camera pose of the grid cell i at frame t. It can be written as:

$$C_i(t) = C_i(t-1)F_i(t-1), \Rightarrow C_i(t) = F_i(0)F_i(1)\cdots F_i(t-1),$$

where $\{F_i(0), ..., F_i(t-1)\}$ are estimated local homographies at the same grid cell i, as shown in Figure 6 (a). We call these spatially-variant paths as "bundled camera paths". In the next section, we describe how we smoothen these bundled paths for video stabilization.

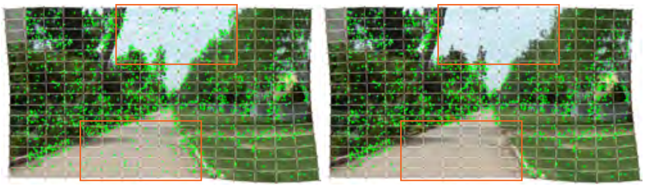


Figure 5: *Left: the estimated warping mesh from all feature points. Right: we exclude all the features in the orange box when estimating the warping model. A similar mesh can be obtained despite the lack of features.*

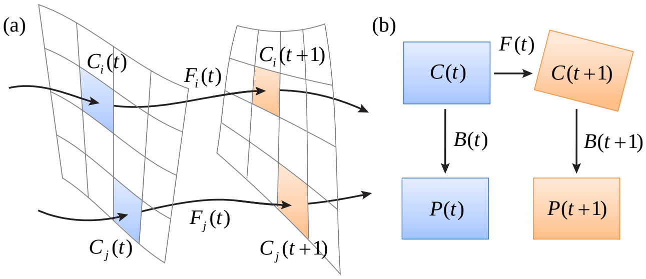


Figure 6: *(a) Bundled camera paths. (b) Relationships among original path $\{C(t)\}$, smoothed path $\{P(t)\}$, and transformations $\{B(t)\}$*

4 Path Optimization

We first describe our smoothing method for a single camera path, and extend it to a bundle of camera paths.

4.1 Optimizing a Single Path

A good camera path smoothing should consider multiple competing factors: removing jitters, avoiding excessive cropping, and minimizing various geometrical distortions (shearing/skewing, wobble). To reach a desired balance, we propose an optimization-based framework taking all factors into account.

Formulation Given an original path $\mathbf{C} = \{C(t)\}$, we seek an optimized path $\mathbf{P} = \{P(t)\}$ by minimizing the following function:

$$\mathcal{O}(\{P(t)\}) = \sum_t \left(\|P(t) - C(t)\|^2 + \lambda_t \sum_{r \in \Omega_t} \omega_{t,r}(\mathbf{C}) \cdot \|P(t) - P(r)\|^2 \right), \tag{5}$$

where Ω_t are the neighborhood at frame t. The other terms are:

- data term $\|P(t) - C(t)\|^2$ enforcing the new camera path to be close to the original one to reduce cropping and distortion;
- smoothness term $\|P(t) - P(r)\|^2$ stabilizing the path;
- weight $\omega_{t,r}(\mathbf{C})$ to preserve motion discontinuities under fast panning/rotation or scene transition;

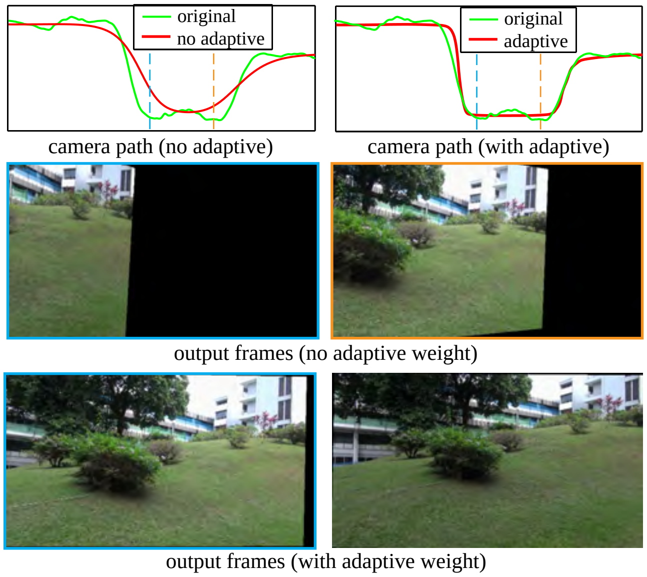


Figure 7: *Comparison of with and without adaptive weights $G_m()$ for a video with rapid camera panning. The camera paths on the top plot the x-translation over time.*

- parameter λ_t to balance the above two terms.

Since Equation 5 is quadratic, we can solve it with any linear system solver. Here, we use a Jacobi-based iterative solver [Bronshtein and Semendyayev 1997]:

$$P^{(\xi+1)}(t) = \frac{1}{\gamma}C(t) + \sum_{r\in\Omega_t, r\neq t} \frac{2\lambda_t\omega_{t,r}}{\gamma}P^{(\xi)}(r), \qquad (6)$$

where $\gamma = 1 + 2\lambda_t \sum_{r\in\Omega_t, r\neq t}\omega_{t,r}$, and ξ is an iteration index. At initialization, $P^{(0)}(t) = C(t)$. Once we obtain the optimized path $\mathbf{P}$, we compute the warping transform $B(t) = C^{-1}(t)P(t)$ to warp the original video frame to the stabilized result (Figure 6(b)).

Discontinuity-preserving The adaptive weight $\omega_{t,r}$ is important to preserve motion discontinuity. We follow the idea of bilateral filter [Tomasi and Manduchi 1998] and design it by two Gaussian functions:

$$\omega_{t,r} = G_t\left(\|r-t\|\right) \cdot G_m\left(\|C(r) - C(t)\|\right), \qquad (7)$$

where $G_t()$ gives larger weight to the nearby frames. $G_m()$ measures the changes of two camera poses.

We use a large kernel to ensure successful suppression of both high-frequency jitters (*e.g.*, handshake) and low-frequency bounces (*e.g.*, walking). In our implementation, we set Ω_t to 60 neighboring frames and the standard deviation of $G_t()$ to 10. In contrast, previous low-pass filtering based methods [Matsushita et al. 2006] typically need a smaller amount of support (*e.g.*, 10 frames) to avoid aggressive cropping and distortion. But such a small kernel is often insufficient in suppressing low frequency bounces.

The reason why we can use a larger kernel lies in $G_m()$. In video stabilization, for rapid camera motion (e.g, caused by fast panning or scene transition), an inappropriate amount of smoothing may lead to excessive cropping, as shown in Figure 7. In this case, the camera pans quickly, and naïve Gaussian smoothing (second row) causes the camera path to significantly deviate from its original path, as indicated by the dashed lines in the left plot on top. The corresponding frames shown on the second row will require large cropping. Our adaptive term $G_m()$ preserves the sudden camera motions to a certain degree. The result from our adaptive smoothing (bottom row) produces much less cropping.

To measure the camera motion, we use the change in translation components $\mu_x(t), \mu_y(t)$ extracted from the camera pose $C(t)$, namely $|\mu_x(t) - \mu_x(r)| + |\mu_y(t) - \mu_y(r)|$. The frame translation $\mu_x(t), \mu_y(t)$ can describe most camera motions in practice except for an in-plane rotation or scale around the principal axis.

Cropping and distortion control The above adaptive term $\omega_{t,r}$ can give us a certain amount of ability to control cropping and distortion. However, the user may want to have strict control on the cropping ratio and distortion. In principle, we could formulate a constrained optimization to address this issue. But it may be too complex to be solved or reproduced.

In this work, we resort to a simple but effective method - adaptively adjust the parameter λ_t for each frame. We first run the optimization with a global fixed $\lambda_t = \lambda$ (empirically set to 5) and then check the cropping ratio and distortion of every frame. For any frame that does not satisfy the user requirements (cropping ratio or distortion is smaller than a pre-defined threshold), we decrease its parameter λ_t by a step $(1/10\lambda_t)$ and re-run the optimization. Note, according to Equation 6, a smaller λ will make the optimized path closer to the original one, which has less cropping and distortions. The procedure is iterated until all frames satisfy the requirements.

We measure the cropping ratio and distortion from the warping transform $B(t) = C^{-1}(t)P(t)$. The anisotropic scaling of $B(t)$ measures the distortion. It can be computed by the ratio of the two largest eigenvalues of the affine part of $B(t)$ [Hartley and Zisserman 2003]. We use $B(t)$ to compute the overlapping area of the original video frame and the stabilized frame. The cropping ratio is the ratio of this area and the original frame area. In our experiments, we require the cropping ratio to be larger than 0.8, and the distortion score to be larger than 0.95 for all examples. In principle, we can further measure the perspective distortion by the two perspective components in $B(t)$. But we empirically find they are always too small when compared with the affine components and do not include them.

4.2 Optimizing Bundled Paths

Our motion model generates a bundle of camera paths. If these paths are optimized independently, neighboring paths could be less consistent, which may generate distortion in the final rendered video. Hence, we do a space-time optimization of all paths by minimizing the following objective function

$$\sum_i \mathcal{O}\left(\{P_i(t)\}\right) + \sum_t \sum_{j\in N(i)} \|P_i(t) - P_j(t)\|^2, \qquad (8)$$

where $N(i)$ includes eight neighbors of the grid cell i.

The first term is the objective function in Equation 5 for each single path, and the second term enforces the smoothness between neighboring paths. This optimization is also quadratic and the optimum result can be obtained by solving a large sparse linear system. Again, our solution is updated by a Jacobi-based iteration [Bronshtein and Semendyayev 1997]:

$$P_i^{(\xi+1)}(t) = \frac{1}{\gamma'}\Big(C_i(t) + \sum_{\substack{r\in\Omega_t \\ r\neq t}} 2\lambda_t w_{t,r} P_i^{(\xi)}(r) + \sum_{\substack{j\in N(i) \\ j\neq i}} 2P_j^{(\xi)}(t)\Big),$$

where

$$\gamma' = 2\lambda_t \sum_{r\in\Omega_t, r\neq t} w_{t,r} + 2N(i) - 1.$$

We typically iterate 20 times to optimize camera paths.

During optimization, the motion-adaptive term $G_m(\cdot)$ is evaluated at individual cells, since different cells have different motion. In comparison, λ_t is determined from the global path (generated by concatenating the pre-warping global homographies), because it controls the overall cropping and distortion. Then, we use λ_t to optimize the camera paths in all cells.

Result synthesis After path optimization, we compute the warping matrix $B_i(t)$ for each cell i by $B_i(t) = C_i^{-1}(t)P_i(t)$. We then apply $B_i(t)$ to warp the i-th cell at the t-th frame to generate the final output video. Usually, applying $B_i(t)$ directly generates good results. This is because our motion estimation ensures first order smoothness of the original paths. Furthermore, the bundled optimization in Equation 8 requires nearby optimized paths to be similar. Thus, the smoothness is naturally satisfied by $B_i(t)$ most of the time. Sometimes, there are slight distortions (*e.g.*, seams of about 1-pixel width), in which case we perform a bilinear interpolation to fix them.

4.3 Correcting Rolling Shutter Effects

Our bundled paths model can naturally handle rolling shutter effects without pre-calibration. The principle of our method is similar to that of [Grundmann et al. 2012]. Our system does rolling shutter correction while simultaneously stabilizing the video. In a shaky video, a rolling shutter causes spatially variant high frequency jitters. When smoothing the camera paths, we simultaneously rectify rolling shutter effects and other jitters caused by camera shake.

5 Results

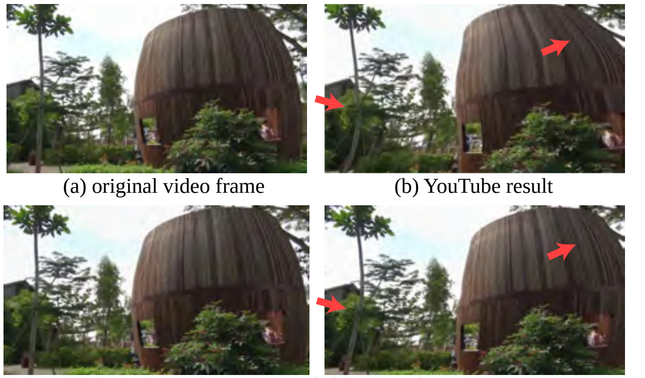
We run our method on an Intel i7 3.2GHZ Quad-Core machine with 8G RAM. We extract 400-600 SURF features [Bay et al. 2008] per frame. For motion estimation, we always divide the video frame to 16 × 16 cells. For a video of 1280 × 720 resolution, our unoptimized system takes 392 milliseconds to process a frame (around 2.5fps). Specifically, we spend 300ms, 50ms, 12ms and 30ms to extract features, estimate motion, optimize camera paths and render the final result. All original and result videos are provided on our webpage[1].

5.1 Algorithm Validation

We first verify the effectiveness of different components of the proposed approach.

A Global Path vs. Bundled Paths For the example in Figure 1, the result according to a global path has remaining jitters in some image regions. This is because the parallax makes the global homography motion model invalid, therefore some image regions cannot be stabilized very well. But our bundled paths can handle this kind of typical situation. Please refer to our accompanying video for a visual comparison.

Spatially-variant Homographies vs. Homography Mixture Grundmann et al. [2012] proposed a homography mixture model for rolling shutter correction. They divide a video frame into a 1D array of horizontal blocks, and use a Gaussian mixture of homographies for each block. This model is beyond a single 2D transformation and able to partially handle parallax.



(a) original video frame

(b) YouTube result

(c) our result

(d) homography mixtures (our implementation)

Figure 8: *Comparison with the homography mixture models in [Grundmann et al. 2012]. (a) A sample frame in the original video. (b) The output frame produced by YouTube Stabilizer. (c) The result produced by our method. (d) The result produced using our implementation of homography mixture [Grundmann et al. 2012] (with the same bundled path smoothing).*

Compared with our 2D mesh-based, spatially-variant homographies, this model has two limitations: 1) it does not address horizontal depth variation; 2) it uses weaker feature points (which apply lower threshold level for feature detection) and a simple Gaussian mixture for the regularization. Weaker feature points may result in larger fitting errors and the ability to use simple Gaussian smoothing is limited.

Figure 8 shows a comparison of these two models. In this example, the scene has horizontal depth variation and the sky region lacks feature points. Figure 8 (a) is the result of using YouTube Stabilizer (integrated Homography Mixture feature). We can observe severe geometrical distortions. To further verify our observation, we replace our spatially-variant model with the homography mixture model (our implementation) in our framework and generate the result in Figure 8 (d), where we observe similar distortion. In comparison, our warping-based motion estimation can fundamentally handle depth variation (not limited to vertical direction). Our result (Figure 8 (c)) does not suffer from such distortion. Please also see the comparison in the accompanying video.

Rolling Shutter Handling Figure 9 compares our methods with [Grundmann et al. 2012] on two example videos from their paper. Our model accounts for frame distortions such as skew (left example) and local wobble (right example). More examples are included in the supplementary video, which shows we achieve similar results on correcting rolling shutter distortion as [Grundmann et al. 2012].

5.2 Quantitative Evaluation

To quantitatively evaluate and measure the result from different aspects, we define three objective metrics.

Cropping and distortion Our first two metrics measure *cropping ratio* and *global distortion*. We first fit a global homography at each frame between input video and output video. We then compute the cropping ratio and distortion for each frame. The cropping ratio can be directly computed from the scale component of the homography. There is one global cropping ratio for the whole sequence, and each frame provides an estimation. We average these estimations at all frames as the final metric. The distortion is computed as defined in Section 4.1. Because any distortion in a single frame will destroy the perfection of the whole result, we choose their minimum across the whole sequence as the final metric. This "worst-case" metric

[1] http://www.ece.nus.edu.sg/stfpage/eletp/Projects/Stabilization/StabilizationSig13.html

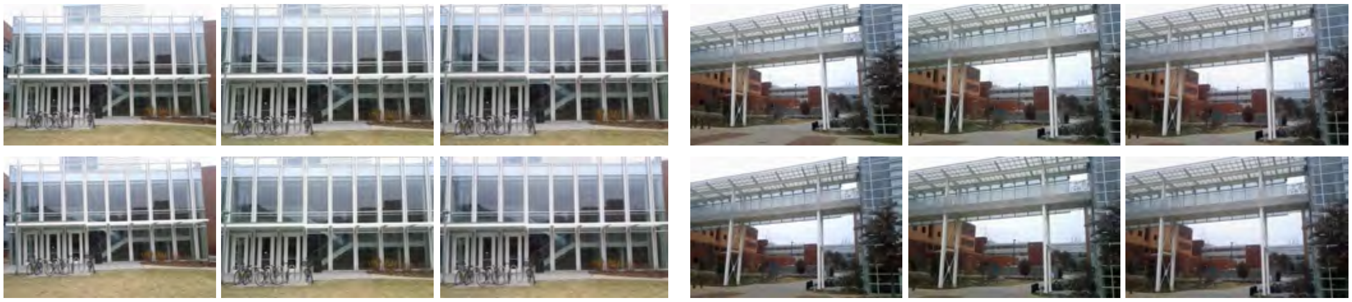


Figure 9: *Two rolling shutter removal examples using our method and [Grundmann et al. 2012]. Our results are on par with that from [Grundmann et al. 2012]. Please see video for a full comparison.*

allows us to easily see whether the whole result video is completely successful. For a good result, both metrics should be close to 1.

Stability The third metric measures the *stability* of the result. Designing a good metric is non-trivial because it is hard to compare two different videos. We suggest an empirically good metric using frequency analysis on estimated 2D motion from a video. Our basic assumption is that the more energy is contained in the low frequency part of the motion, the more stable a video is.

Computationally, we estimate our bundled camera paths to approximate the true motion (optical flow) in a video. We do not smooth out anything after the estimation. Then, we extract translation and rotation components from each path. Each component is a 1D temporal signal. Finally, we evaluate the energy percentage of the low frequency components (expect for DC component) in these 1D signals to measure the stability.

Specifically, we take a few of the lowest (empirically set as from the 2nd to the 6th) frequencies and calculate the energy percentage over full frequencies (excluded by the DC component). Similar to the distortion, we take the smallest measurement among the translation and rotation as the final metric. For a good result, the metric should approach 1 here as well.

5.3 Comparison with Publicly Available Results

The purpose of this comparison is to test whether our results are comparable with (if not better than) previous "successful" results in [Liu et al. 2009; Liu et al. 2011; Goldstein and Fattal 2012; Grundmann et al. 2011]. We collect eleven test videos from these papers (thumbnails in Figure 10), and compare our results with their published results (all from authors' project webpages).

Overall, all methods generate similar stability both subjectively and quantitatively (Figure 10) on these examples, while our results are slightly better on some videos in terms of cropping ratio and distortion.

For video (2)-(4), 3D stabilization [Liu et al. 2009] achieves the best stability and distortion scores. It suggests that 3D methods are the first choice (in term of stability and distortion error), when the 3D motion can be successfully estimated. Although our results are slightly worse in stability, the visual difference is quite small (please verify from the supplementary video). Furthermore, the aggressive smoothing in 3D methods sometimes leads to an output FOV that is too small as demonstrated by the cropping score. Our method manages to provide a good trade-off. For video (5-9), [Liu et al. 2011], [Goldstein and Fattal 2012], and our method achieve similar stability, while our method is slightly better in cropping and distortion. For video (10-11)[2], our method outperforms the L1-optimization [Grundmann et al. 2011] in stability (slightly), cropping ratio, and distortion scores.

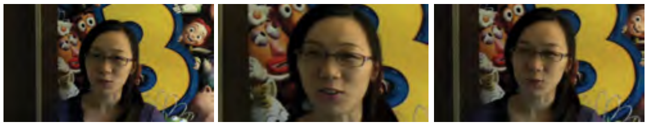


Figure 11: *Comparison with a failure case of prior methods.*

Figure 11 highlights the most challenging video (10) in this dataset. Liu et al. [2011] refer this example as a failure case because a single subspace cannot account for the feature trajectories on both the face and the background. Their results have visible distortion. [Grundmann et al. 2011] produced better result on this example. But in the video result, we still observe large temporal distortion on the background region. (See our accompanying video.) In comparison, our method can successfully handle this example (achieve best in terms of all three metrics) because the warping-based motion model can represent this complicated motion.

5.4 Comparison with the State-of-the-Art Systems

Due to no publicly available implementation of previous works, we compare our system with two well-known commercial systems – YouTube Stabilizer and 'Warp Stabilizer' in Adobe After Effects CS6. The YouTube Stabilizer is based on the combination of the L_1-norm path optimization [Grundmann et al. 2011] and homography mixtures [Grundmann et al. 2012]. The 'Warp Stabilizer' in Adobe After Effects is largely based on subspace stabilization [Liu et al. 2011]. We understand that commercial products are often different from a given research system. But we believe these two systems represent the essential elements of research conducted in this field, and the comparison makes sense for examining strengths or weaknesses and robustness (for various videos using a set of fixed parameters) of our system.

Dataset We assemble a comprehensive dataset of 174 short videos ($10 \sim 60$ seconds) from previous publications, Internet, and our own captures. To know the strength and weakness of a method in different situations, we roughly divide our data into 7 categories based on camera motion and scene type. They are: (I) *simple*, (II) *quick rotation*, (III) *zooming*, (IV) *large parallax*, (V) *driving*, (VI) *crowd*, and (VII) *running*.

[2] To better measure stability on background motion (caused by camera shake), we use a manual foreground mask to exclude foreground motion.

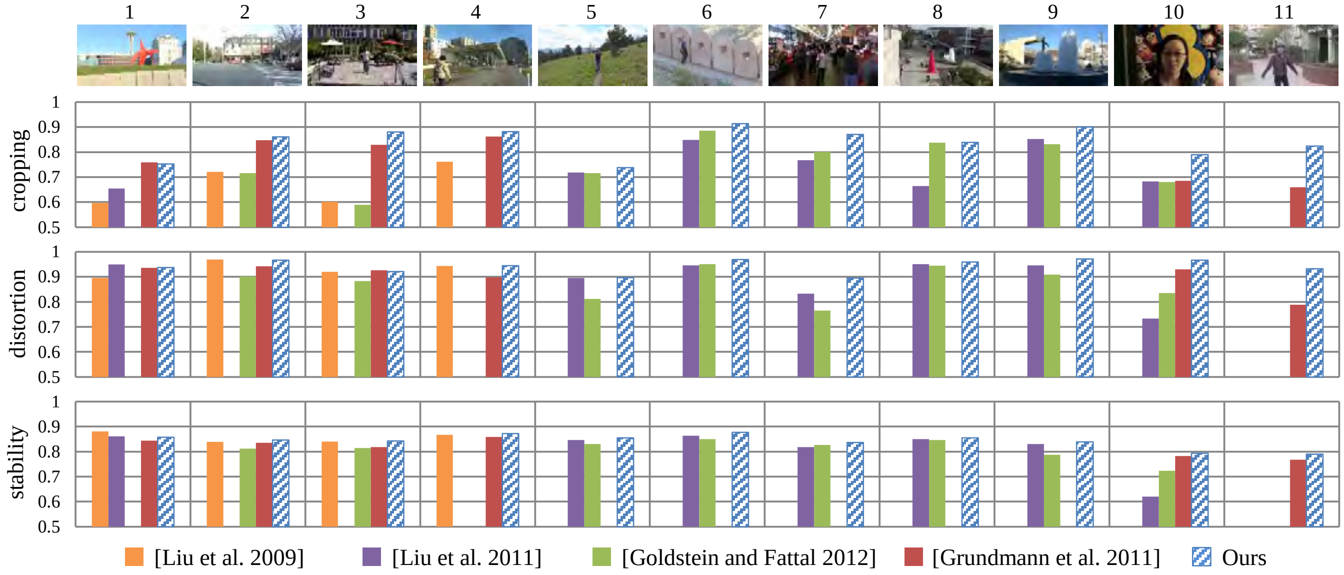


Figure 10: *Quantitative comparison with existing stabilization techniques on publicly available data.*

YouTube Stabilizer is a parameter-free online tool. But 'Warp Stabilizer' is an interactive system, and the user might carefully tune a few parameters. Here, we wish to examine its robustness as an automatic tool by fixing its parameters. We use the example videos in [Liu et al. 2011] to decide the best parameters. Finally, we choose the default parameters (smoothness: 50%, 'Smooth Motion' and 'Subspace Warp') to produce results.

Quantitative Comparison For each category, we compute the average metrics and standard deviation of three systems (Figure 12 (a)). We discuss the results with regard to each system in detail below.

All three systems perform well in category (I) "*simple*", since this category contains videos with relatively smooth camera motion and mild depth variations. Though our method has a minor advantage, the users can safely choose any of three to get a desired result.

Among the remaining categories, we want to highlight the category (IV) "*large parallax*". The three systems achieve similar stability, while our system is clearly better in terms of distortion. We show two examples in Figure 12 (b) and (c) for visual comparison of our system and the YouTube Stabilizer. These examples show the limitation of a 1D array of homography mixtures – it cannot model depth changes in horizontal direction. Warp Stabilizer also generates some shearing/skewing artifacts in some video frames, though in principle this 3D method should be able to handle parallax. Figure 12 (d) shows such an example (please note the shearing of the bookshelf). This is probably due to the subspace analysis failure caused by occlusion. Our method succeeds in all of these examples. Comparison in this category clearly demonstrates the advantages of our warping-based motion model in dealing with a large parallax.

Categories (II–III) contain quick rotation or zooming, which are challenging cases for methods requiring long feature tracking. 'Warp Stabilizer' often generates significant cropping. Figure 12(e) is such an example. To alleviate this problem, we try to interactively tune its smoothing parameters. When applying a weaker smoothing, however, we find its result becomes shaky. In comparison, our method generates stable results with much less cropping. For categories (V–VII), the three systems generate similar stability levels ('Warp Stabilizer' is slightly better in category VII), while our system is consistently better with respect to either cropping ratio or distortion control.

We notice that our method generates relatively smaller standard deviations of the three metrics for all categories. It suggests that our method generates more consistent results from various inputs.

User Study We further conduct a user study with 40 participants to evaluate and compare our method with the YouTube Stabilizer and the 'Warp Stabilizer' in Adobe AfterEffects CS6. Every participant is required to evaluate results on 28 different input videos (randomly sampled from our dataset), in which there are 4 videos for each category mentioned above (The 4 video are prepared in the way that two of them compare our result to YouTube Stabilizer, and the other two to 'Warp Stabilizer'). In the user study, we use the scheme of forced two-alternative choice. Every participant is asked to pick a better one between the results of our method and YouTube Stabilizer, or between the results of our method and the 'Warp Stabilizer'. These videos are displayed to the subjects in a random order. The subjects are unaware of the video categories. Neither do they know which technique is used to produce the stabilized results. Figure 13 (a) shows such an interface for the user study. The original video is displayed on the top. The two stabilized ones are shown side-by-side below. Users can simultaneously play input video and both two results to better examine the difference. And these videos can be played back and forth, or be paused at a certain frame to help users carefully make their decision. The user can also play each of these videos individually to examine their quality without other distractions. We ask users to disregard differences in aspect ratio, or sharpness since each one may undergo different video codecs or further post-processing which makes uniform treatment difficult.

The user study results are shown in 13 (b). For each category, we show the average percentage of user preference. In general, the majority of all users showed significant preference towards our results when compared to any of the other two systems respectively. In particular, the participants prefer the overall quality of our results for category (IV) "*large parallax*" over YouTube Stabilizer (72% vs. 28%) and 'Warp Stabilizer' (69% vs. 31%). The result is consistent with our metric evaluation. For category (II–III) containing quick rotation or zooming, users show a strong bias in preference toward our results over 'Warp Stabilizer' (93% vs. 7% for rotation,

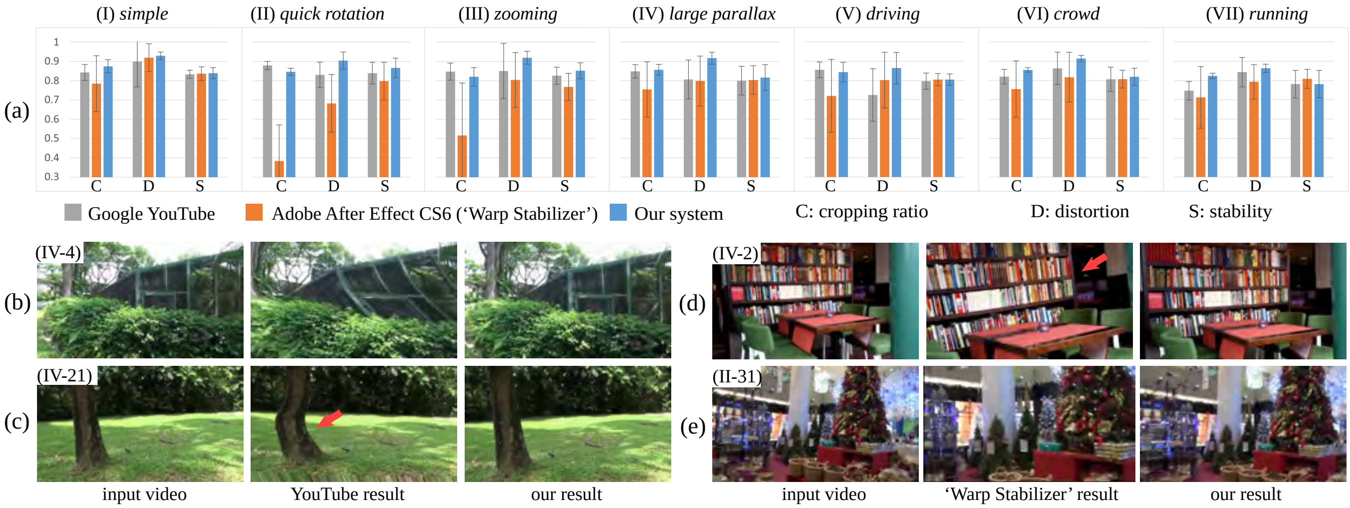


Figure 12: *Comparisons with two popular systems: YouTube Stabilizer and Adobe After Effect "Warp Stabilizer". Top: quantitative comparisons by three metrics: cropping (C), distortion (D) and stability (S). Bottom: some sample video frames for visual comparisons.*

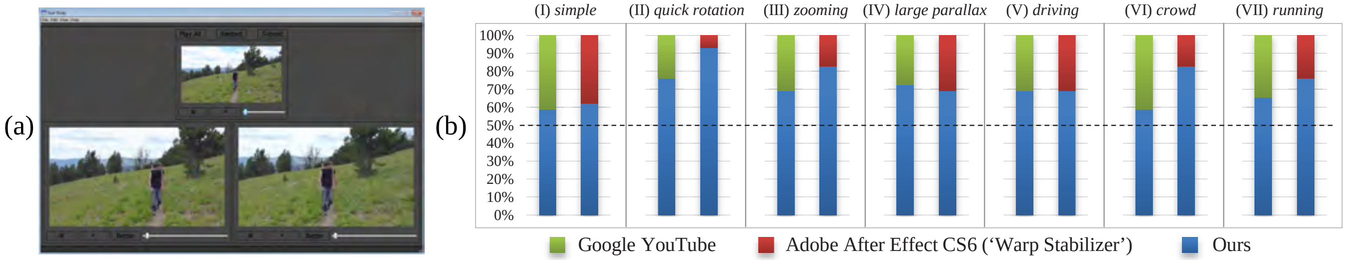


Figure 13: *(a) Pair-wise comparison interface for user study. (b) User study results by comparing our method with two popular systems: YouTube Stabilizer and Adobe After Effect "Warp Stabilizer".*

83% vs. 17% for zooming). This is possibly due to the significant cropping in the results of 'Warp Stabilizer'. For categories (V–VII), more participants prefer our results to the other two systems, although the three systems generate similar stability levels according to our stability metric. It is likely because of the superior distortion and cropping control in our method. In category (I) "*simple*", users express similar preference toward three results.

After the user study, we also ask all participants to articulate the criteria for their feedbacks. We conclude the main criteria for unacceptable videos: 1) the video gets a smaller field of view or even contains frames with visible empty (black) area; 2) the video presents structure distortions in individual frames; 3) the motions in some video frames vibrate or oscillate; 4) the scene transition looks abrupt or not smoothed in the video. From these criteria, our proposed metrics can be partially related with human preferences. And both quantitative evaluation and user study results consistently indicate our system performs better than the other two systems.

5.5 Limitations and Discussion

We find that when 3D reconstruction is successful, 3D methods often generate the best results. However, our system is more robust as we do not require feature tracking, and it produces comparable or only slightly worse results. It is interesting to note that our adaptive path optimization can also be applied to path smoothing for 3D methods [Liu et al. 2009; Liu et al. 2011; Goldstein and Fattal 2012], which often use low-pass filtering (Gaussian smoothing), or curve fitting for path planning. In comparison, our adaptive camera path smoothing technique can automatically adjust the smoothness strength by considering discontinuity and distortion. We show such an example video on our project webpage.

There are cases where the warping-based motion model fails to handle severe occlusions or dis-occlusions, especially when combined with rolling shutter effects. Figure 14 shows two such examples. Our warping-based motion model chooses a large α to enforce strong coherence between grid cells. In this way, we can minimize the geometrical distortion, but at the same time, we sacrifice motion accuracy and eventually the stability of the result. In general, we find geometrical distortion is more disruptive than some slight remaining jitters.

Our path optimization does not strictly follow cinematography rules, which may be desirable in certain applications. But our discontinuity-preservation optimization produces visually pleasing results in most examples. If necessary, we could apply the strategy in [Gleicher and Liu 2007] as a post-process to solve this problem. We also do not deal with motion blur. Sometimes, the stabilized results contain visible blur artifacts. This problem can be addressed by the recent work [Cho et al. 2012].

6 Conclusion

We have presented a new 2D video stabilization method with a bundled camera paths model. The proposed method can simultaneously generate comparable results to 3D methods while keeping merits of 2D methods. Using image warping techniques for motion representation is an interesting finding in this paper. In the future, we would

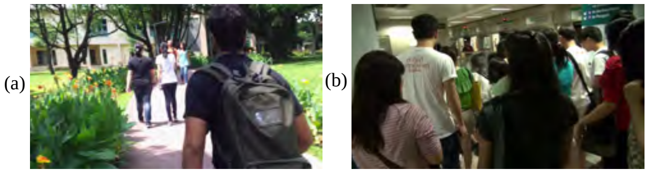


Figure 14: *Two failure cases. The left is due to severe occlusion together with rolling shutter effects. The right is caused by the crowd.*

extend this kind of representation to other video-based applications.

Acknowledgements

We thank all the reviewers for their helpful discussions, Kaimo Lin and other subjects for their help in the user study, Nathan Holdstein and Jiangyu Liu for their help in proofreading. This work is also partially supported by the Singapore project R-263-000-620-112.

References

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

Bay, H., Ess, A., Tuytelaars, T., and Van Gool, L. 2008. Speeded-up robust features (surf). *Comput. Vis. Image Underst. 110*, 3, 346–359.

Bronshtein, I. N., and Semendyayev, K. A. 1997. *Handbook of Mathematics*. Springer-Verlag, New York, NY, USA.

Brox, T., Bruhn, A., Papenberg, N., and Weickert, J. 2004. High accuracy optical flow estimation based on a theory for warping. In *Proc. ECCV*.

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

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

Cho, S., Wang, J., and Lee, S. 2012. Video deblurring for hand-held cameras using patch-based synthesis. *ACM Trans. Graph. (Proc. of SIGGRAPH) 31*, 4.

Fischler, M. A., and Bolles, R. C. 1981. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Commun. ACM 24*, 6, 381–395.

Forssén, P.-E., and Ringaby, E. 2010. Rectifying rolling shutter video from hand-held devices. In *CVPR*.

Gao, J., Kim, S. J., and Brown, M. S. 2011. Constructing image panoramas using dual-homography warping. In *Proc. CVPR*.

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

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

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

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

Hartley, R., and Zisserman, A. 2003. *Multiple View Geometry in Computer Vision*, 2 ed. Cambridge University Press, New York, NY, USA.

Igarashi, T., Moscovich, T., and Hughes, J. F. 2005. As-rigid-as-possible shape manipulation. *ACM Trans. Graph. (Proc. of SIGGRAPH) 24*, 3, 1134–1141.

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

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

Liang, C.-K., Chang, L.-W., and Chen, H. H. 2008. Analysis and compensation of rolling shutter effect. In *IEEE Trans. on Image Processing*.

Lin, W.-Y., Liu, S., Matsushita, Y., Ng, T.-T., and Cheong, L.-F. 2011. Smoothly varying affine stitching. In *Proc. CVPR*.

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

Liu, F., Gleicher, M., Wang, J., Jin, H., and Agarwala, A. 2011. Subspace video stabilization. *ACM Trans. Graph. 30*.

Liu, S., Wang, Y., Yuan, L., Bu, J., Tan, P., and Sun, J. 2012. Video stabilization with a depth camera. In *Proc. CVPR*.

Lucas, B. D., and Kanade, T. 1981. An iterative image registration technique with an application to stereo vision. In *Proc. of the International Joint Conference on Artificial Intelligence (IJCAI)*, 674–679.

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

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

Nakamura, J. 2005. *Image Sensors and Signal Processing for Digital Still Cameras*. CRC Press, Inc.

Nir, T., Bruckstein, A. M., and Kimmel, R. 2008. Over-parameterized variational optical flow. *Int. J. Comput. Vision (IJCV) 76*, 2, 205–216.

Schaefer, S., McPhail, T., and Warren, J. 2006. Image deformation using moving least squares. *ACM Trans. Graph. (Proc. of SIGGRAPH) 25*, 3, 533–540.

Shum, H.-Y., and Szeliski, R. 2000. Construction of panoramic image mosaics with global and local alignment. *Int. J. Comput. Vision (IJCV) 36*, 2, 101–130.

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

Szeliski, R. 1996. Motion estimation with quadtree splines. *IEEE Trans. Pattern Anal. Mach. Intell. 18*, 12, 1199–1210.

Tomasi, C., and Manduchi, R. 1998. Bilateral filtering for gray and color images. In *Proc. ICCV*, 839–846.