# Joint Video Stitching and Stabilization From Moving Cameras

Heng Guo, Shuaicheng Liu, *Member, IEEE*, Tong He, Shuyuan Zhu, *Member, IEEE*, Bing Zeng, *Fellow, IEEE*, and Moncef Gabbouj, *Fellow, IEEE*

***Abstract*— In this paper, we extend image stitching to video stitching for videos that are captured for the same scene simultaneously by multiple moving cameras. In practice, videos captured under this circumstance often appear shaky. Directly applying image stitching methods for shaking videos often suffers from strong spatial and temporal artifacts. To solve this problem, we propose a unified framework in which video stitching and stabilization are performed jointly. Specifically, our system takes several overlapping videos as inputs. We estimate both *inter motions* (between different videos) and *intra motions* (between neighboring frames within a video). Then, we solve an optimal virtual 2D camera path from all original paths. An enlarged field of view along the virtual path is finally obtained by a space-temporal optimization that takes both inter and intra motions into consideration. Two important components of this optimization are that: 1) a grid-based tracking method is designed for an improved robustness, which produces features that are distributed evenly within and across multiple views and 2) a mesh-based motion model is adopted for the handling of the scene parallax. Some experimental results are provided to demonstrate the effectiveness of our approach on various consumer-level videos and a Plugin, named "Video Stitcher" is developed at Adobe After Effects CC2015 to show the processed videos.**

***Index Terms*— Video stitching, video stabilization, bundled paths, space-temporal optimization.**

## I. Introduction

IMAGES captured by a single device, such as a cell-phone, a DV or a tablet, often have a limited field of view. Image stitching can enlarge the view angle by combining contents from several overlapping images captured by multiple cameras, which is often referred to as panorama [1] or image mosaics [2]. In this work, we extend image stitching to video stitching for videos that are captured simultaneously for the same scene by multiple moving cameras.

Manuscript received May 2, 2016; revised July 12, 2016 and August 23, 2016; accepted August 23, 2016. Date of publication September 8, 2016; date of current version September 28, 2016. This work was supported in part by the National Natural Science Foundation of China under Grant 61502079, Grant 61370148, and Grant 61300091, in part by the Research Program for Scientific and Technological Innovation of Sichuan Province, China, under Grant 2014TD0006, and in part by the State Grid Corporation of China Research and Development Project under Grant 521999150031. The associate editor coordinating the review of this manuscript and approving it for publication was Prof. Jing-Ming Guo. *(Corresponding authors: Shuaicheng Liu; Bing Zeng.)*

H. Guo, S. Liu, T. He, S. Zhu, and B. Zeng are with the Institute of Image Processing, University of Electronic Science and Technology of China, Chengdu 611731, China (e-mail: liushuaicheng@uestc.edu.cn; eezeng@uestc.edu.cn).

M. Gabbouj is with the Department of Signal Processing, Tampere University of Technology, 33720 Tampere, Finland.

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TIP.2016.2607419

One challenge in video stitching under this circumstance is that videos captured by moving cameras often appear shaky. As a result, directly applying image stitching methods to shaky video frames in the frame-by-frame fashion would suffer from two drawbacks. First, strong perspective distortions would be resulted because of the shakiness in the captured videos. Second, the frame-by-frame stitching does not consider the temporal smoothness within the videos so that jitters would become visible. The existing solutions to this problem largely rely on additional hardware and require to constrain the movement between cameras. For instance, *FullView* camera type FC-110 (http://www.fullview.com/products.html) assembles 10 CCD cameras around its optical center into a glass container; *GoPano* (http://www.gopano.com/) mounts a spherical lens in front of a cell-phone camera for enlarged view angles; Perazzi *et al.* [3] proposed a method for panoramic video capturing through an unstructured camera array; and Li *et al.* [4] and Jiang and Gu [5] proposed to stitch videos under static cameras. In general, these hardware-based solutions are expensive and inconvenient.

In this work, we propose an all-software approach to jointly stitch and stabilize shaky videos captured by multiple moving cameras, thus creating a stablized video with an enlarged view angle. Fig. 1 illustrates two scenarios: (1) three persons are capturing a scene using cell-phones (Fig. 1(a)) and (2) videos are captured by two micro UAVs (Fig. 1(b)), whereas Fig. 1(c) and (d) show two stitched results corresponding to (a) and (b), respectively.

This is a kind of collaborative capturing that has already been explored in photography, including CamSwarm [6] (Photograph) and PanoSwarm [7] (Panorama imaging). We expect that our collaborative video capturing can share the same benefits. This is also related to robots/drones co-vision where some works have been done, e.g., dense 3D reconstruction from multiple cameras [8] and collaborative visual SLAM (CoSLAM) [9].

The key in video stitching is how to handle camera motions. We can stitch images of different views at every individual frame using various image stitching methods [10]–[12]. However, as aforementioned, the results tend to suffer from both spatial and temporal artifacts. On the other hand, video stabilization methods (e.g., [13]–[16]) can remove jittery and shaky motions. However, these methods cannot be extended directly for stabilizing a stitched video, e.g., perspective distortions in stitched frames, motion estimation under an enlarged view angle, and large motion diversities within a stitched frame are some major factors for the unsatisfactory result.


Fig. 1. Our system settings: videos are captured by moving devices. Our system produces a stitched and stablized video with an enlarged view angle. http://www.liushuaicheng.org/TIP/VideoStitching2016/videostitching.mp4.

To solve the problem, we formulate video stitching and stabilization into a unified framework such that spatial alignment and temporal stability can be achieved simultaneously. We recognize that image features form the basis for both stitching and stabilization. To achieve a high-quality performance, we would like to have rich features to cover all frames densely and uniformly within and across views. In summary, we are facing three challenges: (1) detection of rich image features to facilitate a joint stitching and stabilization, (2) stitched results being free from spatial artifacts (e.g., misalignments and distortions), and (3) keeping the temporal consistency in stitched results (i.e., free from jitters and shakiness).

With regards to the image features, we design a grid-based tracker. Different from traditional approaches that adopt a global threshold [17] over the whole frame, we divide each frame into regular grids and a local threshold is adopted for each grid. In this way, we can generate more features as compared with a global threshold, especially for textureless regions [18]. Then, the detected features are pruned for a balanced distribution before the KLT tracking [19]. Next, we extend the tracking to multiple views by introducing a plane-based homography-RANSAC. As a result, the features are scattered densely and uniformly not only within a single view but also across different views, which lays a solid foundation for subsequent steps.

As for the frame stitching, we adopt mesh-based warps [20], because they can handle spatially-variant motions caused by scene depth and parallax, thus producing high-quality stitching results. Specifically, we adopt the warp-based method in [10] that has been widely used for image stitching, with emphasis given to the manipulation of mesh structures [11], [21].

To enforce the temporal consistency, we choose the bundled-paths stabilization approach [16]. For video stitching, we need to modify it in order to take multiple inputs such that all videos can be smoothed simultaneously to a mutually-stabilized position with respect to stitching constraints. The bundled-paths approach is built upon mesh warps for motion estimation. Likewise, the mesh warps can represent spatially-variant motions, leading to high-quality stabilization results. More importantly, as both stitching and stabilization adopt the mesh structure, the involved constraints can be easily manipulated for a joint optimization.

The main contribution of this paper is a unified framework that facilitates the join video stitching and stabilization. To the best of our knowledge, this is the first framework that achieves spatial alignment and temporal stability simultaneously on the videos captured by multiple cameras that can move freely. Furthermore, our approach enjoys a higher efficiency and robustness due to the fast and rich feature tracking method and the 2D motion model. Finally, the mesh-based motion estimation enables us to handle scenes with parallax effectively.

The rest of this paper is organized as follows: Sec. II reviews the related works. Sec. III presents our method in detail. Discussions are provided in Sec. IV. Results are presented in Sec. V. Some conclusions are finally drawn in Sec. VI.

## II. Related Works

### A. Video Stabilization

According to motion models, video stabilization methods can be categorized as 3D, 2.5D, and 2D approaches.

The 3D methods estimate camera motions in the 3D space for stabilization [22]. To this end, Liu *et al.* developed a 3D stabilization system based on the full 3D reconstruction [15]; a depth camera [23] and a light field camera [24] were attempted for the 3D motion recovery. To avoid the expensive and brittle 3D reconstruction, the 2.5D methods utilize partial 3D information embedded in long feature tracks for stabilization [25], [26]. In general, 3D and 2.5D methods are not robust enough for consumer videos as they both require long feature tracks, which are hard to obtain in the presence of quick camera motions (e.g., quick rotations).

The 2D methods estimate a series of 2D linear transformations (e.g., affines or homographies) between adjacent frames and smooth these transformations for stabilization [13], [27]. Some priors are incorporated during the smoothing, such as polynomial curves [28] and cinematographical rules [14]. Some methods focus on the correction of rolling shutter effects [18], [29], while Liu *et al.* estimate the bundled camera paths by a mesh-based motion model for spatially-variant

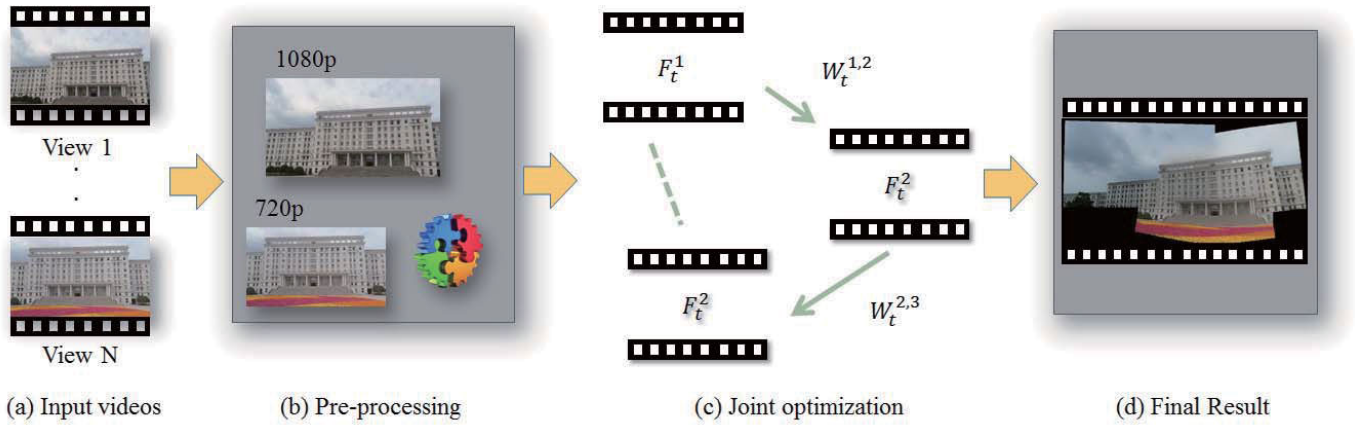



Fig. 2. Pipeline of our system: (a) input videos that are captured by different video recorders, (b) auto video synchronization, (c) the core of our system - joint optimization of stitching and stabilization, and (d) the final result.

motion representation [16]. In our work, the bundled-paths approach is adopted for stabilization.

### *B. Image Stitching*

Early image stitching methods adopt a single homography for image alignment [1], [30]. Then, Gao *et al.* proposed to use two homographies for image stitching when the scene could be modeled roughly by two planes (e.g., the ground and the sky) [31]. In general, there are two kinds of stitching strategies: seam-driven [12], [32], [33] and warp-based [10], [11], [21]. In the seam-driven category, Agarwala *et al.* proposed the photomontage that composites a photograph by cutting and joining multiple photographes seamlessly [33], while Zhang and Liu found a homography that leads to a minimum energy seam to stitch large-parallax images [12]. In the warp-based category, Zaragoza *et al.* proposed an as-projective-as-possible (APAP) mesh deformation that warps images by following a global projective transformation and allows local non-projective deviations [21], while Chang *et al.* proposed a shape-preserving half-projective (SPHP) method to correct distortions in non-overlapping regions [11]. In our work, we adopt mesh warps for view alignment.

### *C. Video Mosaics and Stitching*

For monocular video input, early methods [34], [35] found image mosaics from a single video and represent the video by mosaics for the efficient indexing. For multiple video inputs, Jiang *et al.* proposed an approach to stitch videos using content-preserving warps [5], Li *et al.* applied double seam selection to achieve efficient structure deformation [4], Hamza *et al.* stabilized panoramic videos captured on portable platforms [36], and Perazzi *et al.* calculated optical flow to warp different views [3]. However, these methods are either applicable only for static cameras or assembled with fixed rigs for portable capturing. On the other hand, El *et al.* [37], [38] and Lin *et al.* [39] proposed methods to stitch videos captured by individual moving cameras. However, El *et al.* mainly focused on the frame-based stitching without considering the temporal smoothness explicitly, whereas Lin *et al.* proposed a video stitching method that relies on the dense 3D reconstruction whose computation is very complicated. In contrast, our method can simultaneously stitch and stabilize videos efficiently, creating videos with not only enlarged view angles but also stabilized motions.

## III. Our Method

Figure 2 shows the pipeline of our framework. The first step of our algorithm is to capture multiple videos as the system's inputs by cameras that can move at a certain degree of freedom. Here, we enforce a rough synchronization, i.e., all cameras begin to record approaximately at the same time. In reality, the difference is usually within a fraction of one second, which is well acceptable for many cases, even for some dynamic scenes.

The second step is to do some pre-processing on the inputs to unify spatial and temporal resolutions. In our implementation, we choose the video with the smallest spatial resolution as the target, and the rest is resized to the target resolution. Similarly, we adjust the frame-rates of videos to the slowest one among all captured videos. The choice of the lowest values can avoid spatial up-sampling and temporal interpolation.

The third step is the joint optimization of video stitching and stabilization, which is the core of our system. At this step, we estimate two types of motions to bring into the optimization, i.e., *inter motions* (motions at the corresponding frames between different videos) and *intra motions* (motions within a video between neighboring frames). We denote the intra motions as $C_n(t)$, with $t$ standing for the time and $n$ denoting the view; whereas the inter motions are denoted as $T_{n,m}(t)$. (e.g., motion at time $t$ between the first and second footages is $T_{1,2}(t)$). We adopt the bundled-paths approach [16] as the baseline for stabilization. The uniqueness of our approach is that not only intra motions but also inter motions are considered during the optimization. The optimization mainly consists of three components: (1) a fast and rich feature tracking, (2) a mutually optimal camera path generation, and (3) a joint stitching and stabilization. The first component provides high-quality features for robust inter and intra motion estimations. The second one generates an optimal camera path lying among all the original paths to suppress

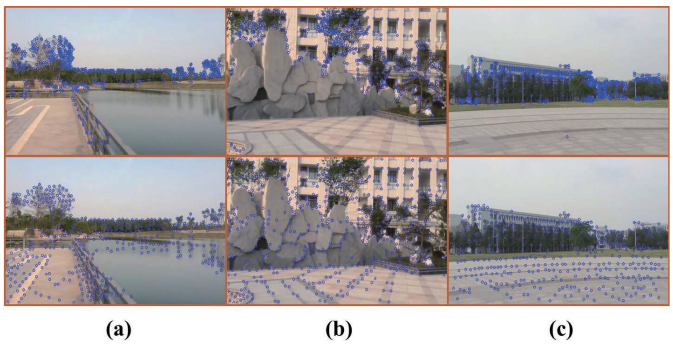



Fig. 3. The first row shows frames by naive feature detection method using FAST feature detector and KLT tracker. The second row shows frames by our grid-based uniform feature tracking strategy. It is noted that the feature distribution by our new tracker is more even.

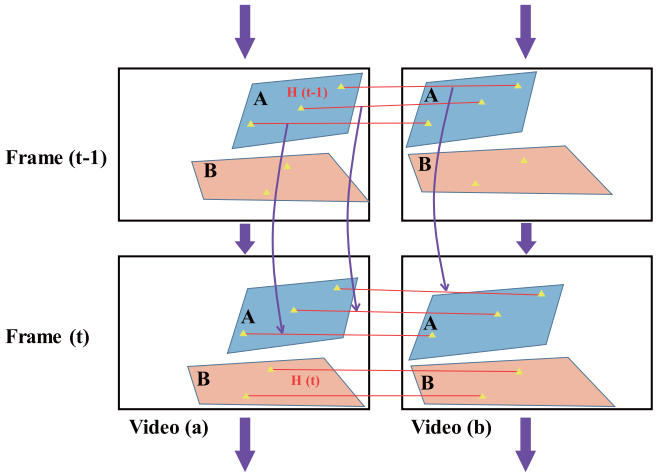



Fig. 4. Integration of feature tracking and feature matching. Without loss of generality, we assume that there are only two visible planes A and B in the scene. While there are only matches of plane A at frame $t-1$, we obtain matches of both planes at frame $t$ using tracking-matching combined method. Previously, matches on the plane B will be lost. Now, we can retain all matches from multiple planes.

perspective distortions. The third one uses the optimal camera path to carry out a joint stitching and stabilization.

### A. Fast and Rich Feature Tracking

In this section, we present the details regarding the feature tracking. Note that the features' quality plays an important role for both stabilization and stitching, because mesh-based motion estimation heavily relies on these features [16]. We believe that high-quality features should be detected to cover the frame densely and uniformly, and they should also last continuously in as many frames as possible. In the following, we first present the grid-based tracker for the case of a monocular video. Then, we upgrade it by the plane-based RANSAC for the case of multiple videos.

*1) Grid-Based Feature Detection:* Our grid-based tracker is built upon the FAST feature detector [40] and KLT tracker [19]. Traditionally, a global threshold often produces fewer features in poorly textured areas (e.g., the ground and the sky as shown in the top row of Fig. 3), because the threshold is biased by other highly textured regions [18]. Therefore, we need to adopt local thresholds for different regions to pull up more features. The bottom row of Fig. 3 shows our grid-based detection results.

Specifically, we divide a single frame into $5\times5$ regular grids and for each grid we use an independent FAST feature detector. Each detector will automatically choose an appropriate threshold for a local grid [17]. Moreover, the threshold of each detector is updated dynamically during tracking, based on the number of detected features. As a result, the thresholds vary both spatially and temporally according to the scene contents. Sometimes, rich texture regions may gather too many features (e.g., trees in the examples of Fig. 3's first row). Thus, we need to prune the features by abandoning some of them based on their FAST scores. On the other hand, some flat areas, such as the sky or the lake, may not be able to fully accommodate as many features as the ground, because these regions have almost zero gradients.

By generating more features for low texture regions through adapting local thresholds and removing some features for rich texture areas, we can detect features that are distributed uniformly. Then, we send them into the KLT tracker. Note that features of a frame come from two sources: tracked features from previous frames and newly-detected features at the current frame. We run the grid-based feature detection only when the number of tracked features drops below a threshold (empirically set to 800).

*2) Integrating Feature Tracking With Feature Matching:* Feature tracking is usually applied for a single video. Normally, we cannot apply feature tracking between different views due to large view angle diversities (e.g., large scale differences). Feature matching works well under this situation. On the other hand, homography-based RANSAC is often adopted to reject outliers after feature matching. However, a single homography-RANSAC can only retain matches residing on a single plane. For a scene consisting of multiple planes (e.g., buildings and the ground), the estimation is only accurate for the dominant plane region that occupies the largest area.

To solve this problem, we propose to integrate the feature tracking with the feature matching, which can produce rich feature correspondences between two videos with multiple planes. Specifically, at frame $t-1$ in Fig. 4, plane A is the dominant plane because potentially there are three feature matches in plane A but only two feature matches in plane B. Thus, if we use homography $H(t-1)$, only matches of plane A can pass homography-RANSAC. Next, all features will be tracked from frame $t-1$ to frame $t$, including matched features of plane A and un-matched features of plane B. Now, we only apply feature matching to un-matched features at frame $t$. This time, the plane B becomes the dominant plane. Thus, we get homography $H(t)$ and matches of plane B are retained (frame $t$ of Fig. 4).

As a result, we obtain feature correspondences of all visible planes at each frame. This strategy, on one hand, fulfills the feature matching with the uniform distribution to facilitate a high-quality inter motion estimation; on the other hand, also increases the speed because matches can be tracked from the adjacent frame pair so that we no longer need to match all features exising in the current frame pair. Fig. 5 gives

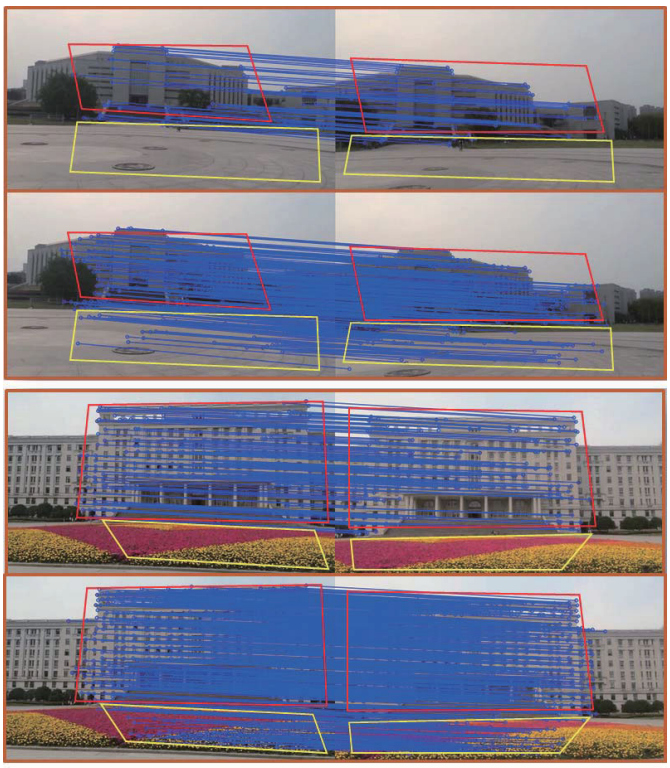


Fig. 5. We show two examples to illustrate the matched features. In each example, the first row shows feature matching result by single homography based RANSAC, and the second by our tracking-matching combined strategy. The red region and the yellow region indicate two distinct planes in the scene. Traditional matching method can only generate feature correspondences of a single plane, while our matching method can keep matches of multiple planes.

a visual comparison between the traditional matching method and our tracking-matching combined strategy. It can be seen that features on the ground have been matched successfully in our method.

### *B. 2D Stabilization*

Now, we have already obtained rich and uniform features within and across views. Below, we begin to discuss how to stablize and stitch videos jointly. For the completeness, we first introduce the bundled-path method [16] that stabilizes a single video. Then, we add the stitching constraints in the next section to consider the inter motions between multiple videos. We begin by reviewing the method in the case of a single path, followed by the case of bundled paths.

*1) Smooth a Single Path:* A single homography $F$ is estimated between neighboring frames in the original video. It is estimated by tracked features between adjacent frames. The camera path is defined as a concatenation of these homographies [14]:

$$C(t) = F(t)F(t-1)...F(1)F(0), F(0) = I. \tag{1}$$

Given the original path $\mathbf{C} = \{C(t)\}$, the smoothed path $\mathbf{P} = \{P(t)\}$ is obtained by minimizing the following energy function:

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

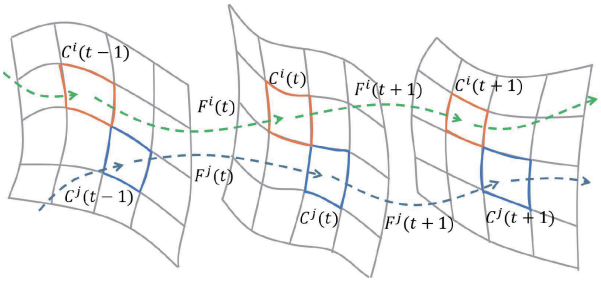



Fig. 6. Smoothing of bundled paths in which each cell has its own camera path.

where $\Omega_t$ denotes the neighborhood at frame $t$. The data term $\|P(t) - C(t)\|^2$ enforces the new camera path to be close to the original path so as to reduce croppings and distortions. $\|P(t) - P(r)\|^2$ is the smoothness term that stabilizes the path. The smoothing kernel $w_{t,r}$ gives higher weights to paths with temporal proximity and preserves motion continuity. It is set as the product of two Gaussians: the first Gaussian is a function of the frame distance and the second is a function of the difference in translation coefficients of the camera path $C(t)$. Similar to the bilateral filter, the smoothing is conducted adaptively. For example, quick camera motions, such as quick rotations and fast zoomings, will get smaller smoothing weights due to large variance of the camera path between frames, and thus they will be skipped to avoid excessive croppings. Note that $\lambda_t$ balances the smoothness for each frame. With a small $\lambda_t$, the result stays close to the original position. Note also that $\lambda_t$ is computed iteratively, by checking the cropping ratio and distortions for every frame. When $\lambda_t$ is set to 0, the result will roll back to the original input. The smoothed video is obtained by applying a warping transform $B_t$ on each frame of the input video, which is defined as $B(t) = C^{-1}(t)P(t)$.

*2) Smooth Bundled-Paths:* The algorithm divides each frame into $16 \times 16$ grids and estimates a camera path for each cell. The estimation is based on the "as-similar-as-possible" mesh warp [20], which warps the frame $t-1$ to the frame $t$. Fig. 6 illustrates the configuration. All paths are smoothed simultaneously by a space-time optimization:

$$\sum_i \mathcal{O}\left(\left\{P^i(t)\right\}\right) + \sum_t \sum_{j \in N(i)} \left\|P^i(t) - P^j(t)\right\|^2, \tag{3}$$

where $N(i)$ includes eight neighbors of the grid cell $i$. Both Eq. (2) and Eq. (3) are quadratic, and therefore can be minimized efficiently by linear system solvers. For more details, please refer to [16].

There are basically two benefits of choosing bundled-paths for stabilization: it can handle parallax and correct large perspective distortions that arise when a single homography is adopted. More discussions will be given in subsequent sections.

### *C. Joint Stitching and Stabilization*

In this section, we describe how to jointly stitch and stabilize multiple videos. The method consists of two steps: 1) finding an optimal camera path that facilitates the stitching

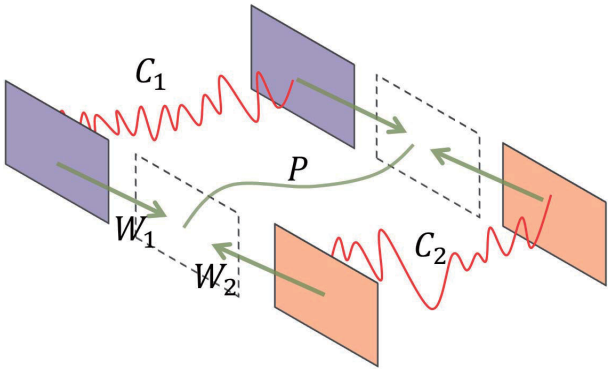



Fig. 7. Illustration of optimal camera path. Given two original camera paths $C_1$ and $C_2$, we seek an optimal camera path $P$ that yields the minimum distortions after frame-warping by $W_1$ and $W_2$.

with minimum distortions and 2) stabilizing videos along the obtained optimal camera path with considering the stitching constraints. Similarly, we first present the method in the case of a single path and then extend it to the case of multiple paths. Without loss of generality, we only describe in the case of two views, which can be extended easily to multiple views.

*1) Optimal Camera Path Generation:* The optimal camera path should stay in between of all original camera paths and also as close as possible to these original paths, because large distortions would occur if a stabilized position is far away from those original positions. As a result, the optimal path is obtained by minimizing the following energy function:

$$\begin{aligned}\mathcal{O}(\{P(t)\}) = &\sum_t \lambda_1(t)\|P(t) - C_1(t)\|^2 \\ &+ \sum_t \lambda_2(t)\|P(t) - C_2(t)\|^2 \\ &+ \sum_t \Big(\sum_{r\in\Omega_t} \hat{\omega}_{t,r}(C_1, C_2)\cdot\|P(t) - P(r)\|^2\Big), \end{aligned} \quad (4)$$

where $C_1(t)$ and $C_2(t)$ represent two original camera paths and $P(t)$ is the optimized path.

The above energy function consists of three terms: two data terms to enforce the similarities between the optimal path and the original paths as well as one smooth term to encourage smooth transitions between neighboring frames. The smoothing kernel $\hat{w}_{t,r}$ is set to be the product of three Gaussians:

$$G_t(\|r-t\|)\cdot G_{m1}(\|C_1(r) - C_1(t)\|)\cdot G_{m2}(\|C_2(r) - C_2(t)\|),$$

where $G_t()$ gives higher weights to temporal nearby frames, and $G_{m1}$ and $G_{m2}$ measure the changes of camera poses. We use the translational coefficients of the original paths to calculate the camera differences. In our implementation, $\Omega_t$ is set to 60 frames and the standard deviations of $G_{m1}$ and $G_{m2}$ are set to 10.

Fig. 7 shows the configuration with two capturing cameras. The original paths of two videos are shown in the jittered curves (in red). The optimal path is obtained after the optimization (in green) so that frames will end up with the minimum distortion when they are warped towards the optimal path by $W_1(t)$ and $W_2(t)$.

Different from Eq. (2) where we put a $\lambda_t$ in front of the smoothness term to control the strength of smoothing, we now put $\lambda_1(t)$ and $\lambda_2(t)$ in front of two data terms. They control the frame distortions after the frames are warped to their optimal/destined positions. In principle, each frame has its own $\lambda_1(t)$ and $\lambda_2(t)$, yielding two extra variables that need to be solved during the optimization. However, it is too complex and considered as unnecessary [16]. The alternative way, which is more efficient, is to adjust them adaptively in each iteration. That is, we first run the optimization with a fixed $\lambda_{\{1,2\}}(t) = \lambda$ (empirically set to 3) and then check the distortion of every frame; if the distortion score of any frame is beyond a threshold (empirically set to 1.03), we decrease $\lambda(t)$ by a step $\lambda(t)/10$ and run the optimization again.

The distortion score, also referred to as the wobble score [16], is derived from the warping transforms (homographies) $W_{\{1,2\}}(t)$. The anisotropic scaling of $W_{\{1,2\}}(t)$ measures the distortion, which can be computed by the ratio of the two largest eigenvalues of the affine part of $W_{\{1,2\}}(t)$.

Eq. (4) is quadratic and thus can be solved by any linear system. Similar to [16], we use a Jacobi-based iterative solver [41]. The update rule is defined as:

$$\begin{aligned}P^{(\xi+1)}(t) = &\frac{\lambda_1(t)}{\gamma}C_1(t) + \frac{\lambda_2(t)}{\gamma}C_2(t) \\ &+ \sum_{r\in\Omega_t, r\neq t} \frac{2\hat{\omega}_{t,r}}{\gamma}P^{(\xi)}(r), \end{aligned} \quad (5)$$

where $\gamma = \lambda_1(t) + \lambda_2(t) + 2\sum_{t\in\Omega_t, r\neq t}\hat{\omega}_{t,r}$ and $\xi$ is the iteration index. We set $P^{(0)}(t) = C_1(t)$ for initialization.

Figure 8 shows the influence of $\lambda_{1,2}(t)$. By setting $\lambda_1(t) = 1$ and $\lambda_2(t) = 0$, $P(t)$ tends to $C_1(t)$, i.e., we warp the right frame towards the left frame. The right frame absorbs all perspective distortions, resulting in a frame stretching as shown in Fig. 8 (a). Fig. 8 (b) shows the opposite case, where the left frame warp towards the right frame. In Fig. 8 (c), the left and right frames are warped to the optimal position, in which the perspective distortions are equally shared, thus leading to the best visual quality. Notably, some of perspective distortions can still be observed in Fig. 8 (c) due to the use of a single homography. Such distortions can be further suppressed by mesh-based warps as discussed in Sec. IV.

It is worth to emphasize that the optimal path may or may not be the exact middle position of two frames. How much a frame can deviate from its original position is determined by the scene depth. If the scene is flat, the corresponding frames can move a large distance without any distortions. However, if the scene contains large depth variations, especially when the variations are not continuous, the corresponding frames can barely move [42]. In the meantime, although the cameras film the same scene, the captured contents are different, leading to different tolerable deviations for different frames. For instance, the left frame can deviate a large distance while the right frame is restricted, or both frames can move freely. In the case when both frames are confined, equal weights will be yielded so that two frames are transformed to their middle position for a balanced distortion.

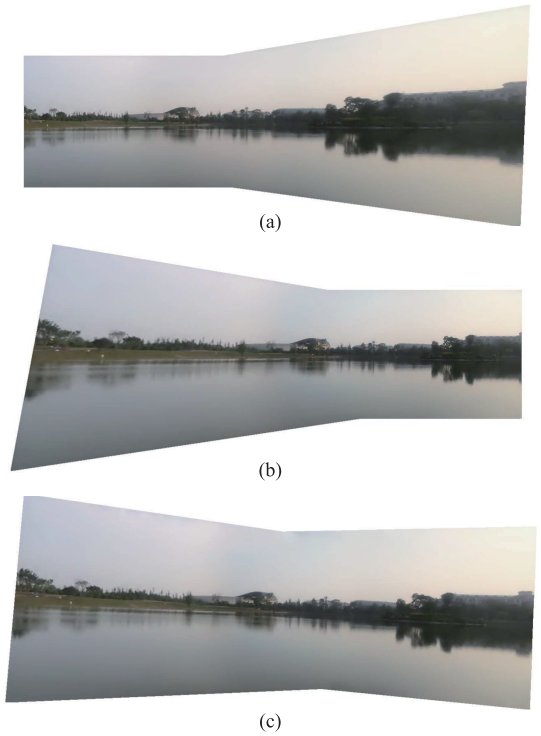


Fig. 8. (a) Right frame warps towards left frame. (b) Left frame warps towards right frame. (c) Left and right frames warp towards to the optimal position. The importance of an optimal path is demonstrated clearly: both (a) and (b) suffer from strong perspective distortions. (The warping used here is based on a single homography; whereas the mesh-based warps can further suppress the perspective distortions.)

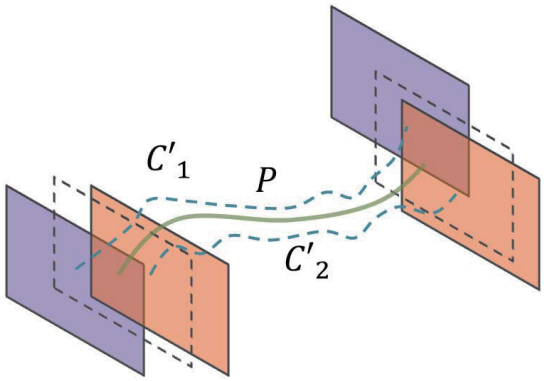



Fig. 9. Frames are warped towards the optimal camera path $P$, yielding two new camera paths $C_1'$ and $C_2'$.

*2) Single-Path Stitching and Stabilization:* After the optimal path is determined, the next step is to bring the original frames to this new path. Intuitively, the final result can be generated by transforms $W_{\{1,2\}}(t) = C_{\{1,2\}}^{-1}(t)P(t)$. However, it usually will produce unsatisfactory result, as $P(t)$ is obtained without consideration of frame alignments. Alternatively, we warp the original frames to the new position according to $W_{\{1,2\}}(t)$. The new position will be used as the initialization for the joint stitching and stabilization.

Figure 9 shows the scenario that two frames are warped towards the optimal camera path $P$, which produces two new videos with camera paths $C_1'$ and $C_2'$, respectively. Then, our job is to find their smoothed camera paths, $U_1$ and $U_2$, which not only inherit the merit of $C'$ (to reduce the perspective distortion) but also take the alignment constraints between different views into consideration. Moreover, they also follow a smooth transition between adjacent frames so as to enforce the temporal smoothness. To achieve this goal, the energy function is defined as:

$$\begin{aligned}\mathcal{O}(\{U_1(t), U_2(t)\}) = &\sum_t \|U_1(t) - P(t)\|^2 \\ &+ \sum_t \|U_2(t) - P(t)\|^2 \\ &+ \sum_t \Big(\sum_{r\in\Omega_t} \omega_{t,r}(C_1') \cdot \|U_1(t) - U_1(r)\|^2\Big) \\ &+ \sum_t \Big(\sum_{r\in\Omega_t} \omega_{t,r}(C_2') \cdot \|U_2(t) - U_2(r)\|^2\Big) \\ &+ \beta \sum_t \|U_1(t)T_{1,2}(t) - U_2(t)\|^2, \end{aligned} \tag{6}$$

where $\omega_{t,r}$ is defined similarly as in Eq. (2). The last term in the above equation aligns two videos during path smoothing, where $\beta$ controls the strength of alignment ($\beta$ is empirically set at 1), and the inter camera motion $T_{1,2}(t)$ is a transformation to align the left and right frames at time $t$ between $U_1$ and $U_2$. Notice that $T_{1,2}(t)$ is the best fitting homography obtained from feature matches of the corresponding left and right frames. We adopt an alternate optimization strategy, i.e., we fix one path and optimize the other. The update rule for the Jacobi-based iterative solver of $U_1$ is:

$$\begin{aligned} U_1^{(\xi+1)}(t) = &\frac{1}{\gamma_1}P(t) + \frac{1}{\gamma_1}T_{1,2}^{(\xi)}(t)U_2^{(\xi)}(t) \\ &+ \sum_{r\in\Omega_t, r\neq t} \frac{2\omega_{t,r}(C_1')}{\gamma_1}U_1^{(\xi)}(r), \end{aligned} \tag{7}$$

where $\gamma_1 = 1 + T_{1,2}^{2(\xi)}(t) + 2\sum_{t\in\Omega_t, r\neq t}\omega_{t,r}(C_1')$ and $\xi$ denotes the iteration index. Note that, as we adopt the iterative solver, $T_{1,2}(t)$ is varying during each iteration. Here, $T_{1,2}^{(\xi)}(t)$ is computed according to feature matches of the current corresponding frames between the left and right videos, whose position is updated by $U_1^{(\xi)}(t)$ and $U_2^{(\xi)}(t)$ after each iteration. Likewise, the update rule for $U_2$ is:

$$\begin{aligned} U_2^{(\xi+1)}(t) = &\frac{1}{\gamma_2}P(t) + \frac{1}{\gamma_2}U_1^{(\xi)}(t)T_{1,2}^{(\xi)}(t) \\ &+ \sum_{r\in\Omega_t, r\neq t} \frac{2\omega_{t,r}(C_2')}{\gamma_2}U_2^{(\xi)}(r), \end{aligned} \tag{8}$$

where $\gamma_2 = 2 + 2\sum_{t\in\Omega_t, r\neq t}\omega_{t,r}(C_2')$, At initialization, we set $U_{1,2}^{(0)}(t) = C_{1,2}'(t) = C_{1,2}^{-1}(t)W_{1,2}(t)$.

After the optimization, we obtain two camera paths $U_{\{1,2\}}(t)$, and the optimized result is generated by warp transforms $B_1(t) = C_1^{-1}(t)U_1(t)$ and $B_2(t) = C_2^{-1}(t)U_2(t)$.

The single path stitching method works reasonably well when the scene is flat, where the motion can be accurately described by a single homography. However, when the scene consists of multiple planes, a single homography is not sufficient. To handle scenes with depth variations, we extend the single path method into the case with multiple paths.

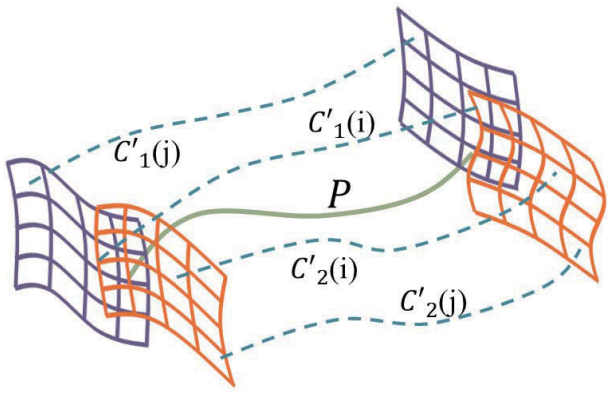



Fig. 10. Frames are divided into regular cells and each cell has its own camera path $C\{i\}$. After warping towards the optimal common camera path $P$, each cell yields a new camera path $C'\{i\}$.

*3) Bundled-Paths Stitching and Stabilization:* Figure 10 shows the configuration of bundled-paths during the stitching and stabilization. Here, $P(t)$ generated by Eq. (4) is still adopted, which servers as the initialization for the optimization. Both left and right videos are divided into $16 \times 16$ regular cells. Each cell contains its own camera path, denoted as $C^i_{\{1,2\}}(t)$. They are warped towards the optimal camera path $P(t)$ by transformation $(C^i_{\{1,2\}}(t))^{-1}P(t)$, which yields the new camera paths $C^{i'}_{\{1,2\}}(t)$. The first-order continuity is preserved during this operation, as all cells are transformed by the same $P(t)$. In other words, the cells would not be splitted after transformation. Then, we want to find the optimal paths $U^i_{\{1,2\}}(t)$, which not only align two views but also have a smoothed motion. To this goal, the energy is defined as:

$$\sum_i \mathcal{O}(\{U^i_1(t), U^i_2(t)\}) + \sum_t \sum_{j\in N(i)} \|U^i_1(t) - U^j_1(t)\|^2 + \sum_t \sum_{j\in N(i)} \|U^i_2(t) - U^j_2(t)\|^2, \quad (9)$$

where $N(i)$ includes eight neighbors of the cell $i$. The first term is the objective function in Eq. (6) for each local path, and the second and third terms together enforce the spatial smoothness between neighboring local paths. Again, we adopt the alternate optimization with Jacobi-based iterations. The update rule for $U^i_1(t)$ is defined as:

$$U^{i,(\xi+1)}_1(t) = \frac{1}{\gamma'_1}P(t) + \frac{1}{\gamma'_1}T^{i,(\xi)}_{1,2}(t)U^{i,(\xi)}_2(t) + \sum_{r\in\Omega_t, r\neq t} \frac{2\omega_{t,r}(C^{i'}_1)}{\gamma'_1}U^{i,(\xi)}_1(r) + \sum_{j\in N(i), j\neq i} \frac{2}{\gamma'_1}U^{j,(\xi)}_1(t), \quad (10)$$

where

$$\gamma'_1 = T^{2(i,(\xi))}_{1,2}(t) + 2\sum_{t\in\Omega_t, r\neq t} \omega_{t,r}(C^{i'}_1) + 2N(i) - 1. \quad (11)$$

Likewise, the update rule for $U^i_2(t)$ is:

$$U^{i,(\xi+1)}_2(t) = \frac{1}{\gamma'_2}P(t) + \frac{1}{\gamma'_2}U^{i,(\xi)}_1(t)T^{i,(\xi)}_{1,2}(t) + \sum_{r\in\Omega_t, r\neq t} \frac{2\omega_{t,r}(C^{i'}_2)}{\gamma'_2}U^{i,(\xi)}_2(r) + \sum_{j\in N(i), j\neq i} \frac{2}{\gamma'_2}U^{j,(\xi)}_2(t), \quad (12)$$

where

$$\gamma'_2 = 1 + 2\sum_{t\in\Omega_t, r\neq t} \omega_{t,r}(C^{i'}_2) + 2N(i). \quad (13)$$

Different from Eq. (6) in which $T_{1,2}(t)$ is a single homography matrix, here, $T^i_{1,2}(t)$ is derived from the mesh warps [15] between two views. Specifically, a cell $i$ contains four vertices, denoted as $V_i = \{v^1_i, v^2_i, v^3_i, v^4_i\}$. After warping, these vertices move to a new place, denoted as $\hat{V} = \{\hat{v}^1_i, \hat{v}^2_i, \hat{v}^3_i, \hat{v}^4_i\}$. $T_i(t)$ can be obtained by solving a linear equation $\hat{V}_i = T^i_{1,2}(t)V_i$. Clearly, $T^i_{1,2}(t)$ encodes the alignment constraints, which is estimated during each iteration.

*4) Multiple Inputs:* Suppose that there are $N > 2$ inputs, the process of solving the optimal path $P(t)$ as described in Eq. (4) is now revised as:

$$\mathcal{O}(\{P(t)\}) = \sum_{k=1}^{N}\sum_t \lambda_k(t)\|P(t) - C_k(t)\|^2 + \sum_t\Big(\sum_{r\in\Omega_t}\hat{\omega}_{t,r}(C_1, .., C_N)\cdot\|P(t) - P(r)\|^2\Big). \quad (14)$$

Here, $\hat{\omega}_{t,r}(C_1, .., C_N)$ takes all paths into consideration. The Jacobi-based update rule can be derived accordingly. Likewise, Eq. (6) can be revised to take $N$ inputs:

$$\begin{aligned}\mathcal{O}(\{U_1(t), .., U_N(t)\}) &= \sum_{k=1}^{N}\sum_t \|U_k(t) - P(t)\|^2 \\ &+ \sum_{k=1}^{N}\sum_t\Big(\sum_{r\in\Omega_t}\omega_{t,r}(C'_k)\|U_k(t) - U_k(r)\|^2\Big) \\ &+ \beta\sum_{k}^{N-1}\sum_t \|U_k(t)T_{k,k+1}(t) - U_{k+1}(t)\|^2 \\ &+ \beta\sum_t \|U_N(t)T_{N,1}(t) - U_1(t)\|^2. \end{aligned} \quad (15)$$

Note that the first two terms in the above equation remain the same as in Eq. (6). For frames that do not overlap, we ignore the alignment term accordingly. When extending Eq. (15) into bundled-paths, we add the term $\sum_t\sum_{j\in N(i)}\|U^i_k(t) - U^j_k(t)\|^2$ for each video to constrain the similarity of sub-paths. Empirically, the optimization converges in 15 iterations.

*5) Result Synthesis:* After optimization, each mesh cell gets a warp transform $B^i_k(t) = C^{i,-1}_k(t)U_k(t)$. Then, we get the optimized result of each input video by warping all cells according to the desired positions. The final video


Fig. 11. Different composite strategy: overlaying vs. blending. We overlay one frame on top of the other frame to avoid any potential misalignment within overlapping region.

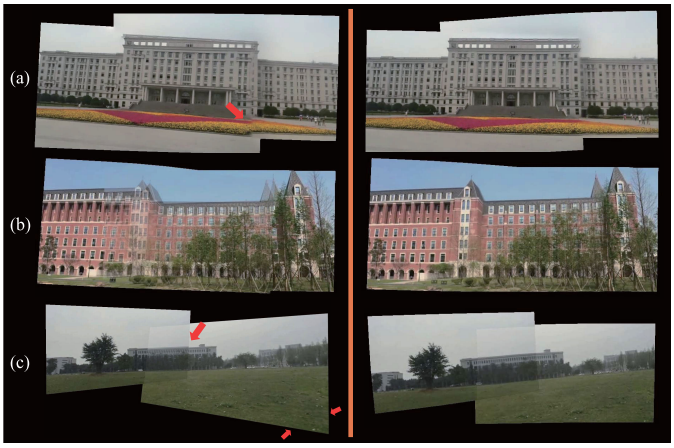



Fig. 12. Each row shows an example by excluding one component of our framework. The left and right columns show the results without and with one component, respectively: (a) without/with rich feature tracking, (b) without/with stitching constraints, and (c) without/with meshed-based structures. The artifacts are highlighted by red arrows.

is synthesized by merging each optimized video frame-by-frame. In this procession, we adopt the multi-band blending algorithm [43] to remove stitching seams.

## IV. Discussions

We discuss our method in several aspects in this section to show its effectiveness.

### A. Overlaying Frames

We adopt the overlaying strategy in our final composition, where one frame is laid on top of the other. The main reason of adopting this strategy is to hide any potential misalignment within the overlapping area for an improved robustness. Fig. 11 shows an example where the linear blending strategy introduces severe ghosting effects due to the misalignments. Clearly, the overlaying strategy is a good choice as long as the overlapping boundary regions can be well aligned. For this purpose, we add more weights to feature points that are close to overlapping boundaries. With emphasis on these boundaries, some misalignments can be allowed in the central area of overlapping regions, because they are covered and cannot be observed. By adopting this strategy, we sacrifice the registration quality of central areas to achieve better alignments at overlapping boundaries. A similar scheme has been reported in [39].

### B. Functionality of Different Components

We evaluate our framework by turning off each component one by one. Fig. 12 shows some results. In Fig. 12 (a), we turn off the rich feature tracking. As a result, most of the features are located on the building with only a few being placed on the ground. The corresponding result suffers from misalignments. In Fig. 12 (b), we turn off the alignment terms (i.e., the last two terms in Eq. (15)). It is clear that two views cannot be aligned in the absence of inter motions. In Fig. 12 (c), we apply a single homography for stitching and stabilization, rather than mesh warps. Not only misalignments arise at the stitching boundary, but also large perspective distortion appears at the right frame border (highlighted by two arrows in Fig. 12 (c)). For a clearer illustration, we use linear blending for examples (b) and (c). These results suggest that every component is important to ensure a reliable and high-quality video stitching.


Fig. 13. To compare the direct stitch and our method in terms of the temporal smoothness, we use temporal images for illustration: (a) a green line in the stitched video is selected to illustrate the temporal smoothness; (b) the green line is plotted along the temporal domain using the direct method, where one video is transformed to the other without any consideration of temporal smoothness; (c) the green line plotted for our method. Clearly, the direct stitching method yields high-frequency jitters.

### C. Direct Stitching

The most straightforward idea for video stitching is to apply image stitching method directly in the frame-by-frame fashion. However, such a direct stitching is problematic due to the ignoring of temporal smoothness. Here, we demonstrate this problem by an example, with the results shown in Fig. 13. First, we fix a vertical line (shown in green) in Fig. 13 (a). When the video is played, we only record the set of pixels along this fixed line and concatenate all these sets according their time-ordering so as to generate Fig. 13 (b) and (c). Note that the horizontal axis of Fig. 13 (b) and (c) represents the time-line, while the vertical axis represent the pixels covered by the fixed-line. Fig. 13 (b) shows the result of direct stitching. As can be seen, it is jittering. Fig. 13 (c) shows our result which is much smoother.

## V. Experiments

The input videos are captured by several moving cameras (cellphones or UAVs). For cellphones, each camera is held by one person who can move freely, while two UAVs with independent flights are used for the other case. Our system can automatically adjust them to the same resolution and frame-rate. We run our method on a PC with Intel i5 2.4GHz CPU and 8G RAM. The boundary seam is eliminated by multi-band blending implemented in OpenCV. For a video of $1280 \times 720$ resolution, our unoptimized system takes 511 milliseconds to stitch two frames (i.e., about 2 fps). In particular, we spend 123ms, 94ms, 31ms, 137ms and 126ms for feature extraction, motion estimation, path optimization, frame warping and


Fig. 14. Video stitching results on various scenes. Refer to the supplementary files for the videos.

TABLE I
THE STABILITY AND STITCHING SCORES OF EXAMPLES IN FIG. 14

| | #1 | #2 | #3 | #4 | #5 | #6 | #7 | #8 |
|---|---|---|---|---|---|---|---|---|
| Stability (original/result) score | 0.67/0.83 | **0.37/0.78** | 0.65/0.82 | 0.71/0.88 | 0.68/0.84 | 0.63/0.91 | 0.59/0.85 | 0.86/0.90 |
| Stitching score | 0.99 | 0.54 | 1.07 | 1.01 | 0.43 | 1.04 | **1.17** | 1.15 |

multi-bad blending, respectively. For the same resolution, the running time reported in [39] is several minutes per frame, which involves the time-consuming dense 3D reconstruction.

To verify whether our method is capable of stitching various kinds of videos, we have tried many examples, with eight of them being shown in Fig. 14. Here, we do not confine the camera motions during the capturing so that some of them are captured during walking and some with camera pannings or rotations. The 2nd and 5th examples focus on scenes of a lake and the sky, while the 3rd example contains dynamic moving objects. The results show that our system can handle these challenging cases robustly. The 4th, 6th, and 7th examples film scenes consisting of two dominant plane structures (the ground and building). The detected features are equally distributed using our rich feature tracking. The feature distribution of the 6th example is shown in Fig. 3(b). The single homography stitching result of the 2nd example is given in Fig. 8(c). The mesh warps can suppress perspective distortions efficiently. The 8th example is captured by UAVs. Our method can stitch and stabilize all these videos quite successfully.

### A. Objective Evaluations

We propose two objective metrics for the evaluation of stability and stitching quality.

*1) Stability Score:* Evaluates the smoothness of the final stitched video. It is originally proposed in [16] to access the stability of a single video. Here, we borrow the idea for evaluations of stitched videos. Specifically, we track features on the stitched video and retain tracks with length greater than 20 frames. Then, we analyze these feature tracks in the frequency domain. We take a few of the lowest frequencies (2nd to 6th without DC component) and calculate the energy percentage over full frequencies. Averaging from all tracks yields the final score. Notably, this metric is better suited for the evaluation of the same video processed by different approaches [16]. *The stability score should stay close to* 1 *for a good result.*

*2) Stitching Score:* Evaluates the stitching quality. We calculate the feature reprojection error for the evaluation. Specifically, for each frame, the distance between a pair of matched features is calculated after the features being transformed. *A small distance indicates a good alignment* [20]. Averaging distances from all feature pairs gives the stitching score of one frame. We take the worst score (the largest value) among all frames as the final stitching score. Notably, as the overlaying strategy is adopted, we only involve features near the stitching boundary for the evaluation.

Table I summaries the stability and stitching scores of examples in Fig. 14. For the stability score, we show the score of input shaky videos (averaged from two inputs) and the score of the stitched result. As indicated by these scores, the stability


Fig. 15. Comparison with the method [39]: the left example consists of three views and the right one combines two views. For both examples, our result is more stable than method [39]. Left example stability: 0.82 (Lin's) vs. 0.87 (ours). Right example stability: 0.77 (Lin's) vs. 0.84 (ours). Please refer to the supplementary files for a clearer visual comparison.

has been largely improved in all examples. On the other hand, in Fig. 13(b), the direct stitching method leads to a decrease of the stability from 0.57 (original) to 0.43 (stitched).

For the stitching score, a visually visible misalignment often produces scores above 5. For example, the stitching score of a failure case in Fig. 17 is 6.67. The largest value in our results is 1.17, corresponding to the 7th example in Fig. 14.

### *B. The Parameter $\beta$*

The most important parameter in our system is the $\beta$ of Eq. (15), which balances the strength of stitching and stabilization. It is set to 1 empirically in all examples presented above. Here, we would like to elaborate the effectiveness of adopting different $\beta$ values: 0.1, 1, and 10 (refer to the supplementary video for the example). Accordingly, the stability score is 0.87, 0.85, and 0.56, respectively, whereas the stitching score is 5.39, 1.05, and 0.88, respectively. When $\beta = 0.1$, although the results have the highest stability score, it also corresponds to the poorest stitching score. On the contrary, larger $\beta$ can produce high quality stitched results, but the video suffers from strong shakiness. The empirical value $\beta = 1$ seems to be a good tradeoff between the stitching and stabilization.

### *C. Compare With Other Method*

We do not compare our method with the methods proposed in [3]–[5] that rely on fixed rigs or the methods in [37] and [38] that stitch videos regardless of the temporal smoothness. We compare our method with the method proposed in [39]. This method also stitches multiple synchronized videos captured by hand-held cameras and is based on the dense 3D reconstruction (to calculate 3D camera paths and dense 3D point clouds). A common virtual path is synthesized in between of the original 3D paths and the result is rendered by warping each individual views toward the virtual path. The warping is guided by various constraints to minimize distortions and increase temporal stabilities.

Note that the method in [39] requires camera intrinsic parameters that are obtained by a careful calibration. Moreover, the 3D reconstructions (both sparse and dense) require consistently not only large overlaps among multiple videos but also high-quality videos (i.e., capturing with high-end DVs) so as to avoid the rolling shutter effects. On the other hand, the rolling shutter effects would be difficult to avoid when capturing with cell-phones or micro UAVs. Because of this, the method presented in this paper do not rely on the 3D reconstruction, but is a 2D approach with improved efficiency and robustness.

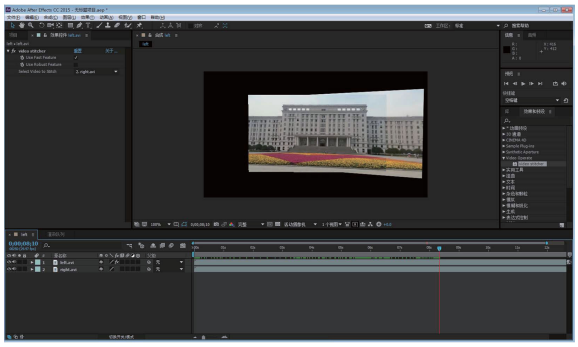


Fig. 16. We developed a plugin "Video Stitcher" at Adobe After Effects CC2015. We can drag in videos and drag the effect for stitching. We have captured the screen of the whole process, refer to our supplementary videos.

Fig. 17. A failure case where misalignments on the ground can be seen. It is caused by strong motion blurring, which significantly damages the feature quality, resulting in incorrect inter motions.

Fig. 15 provides a visual comparison. It can be seen that we have produced comparable results in terms of the stitching quality as well as the stability. Sometimes, the results in [39] contain some distortions along the frame border due to the inaccuracy of reconstructed 3D points at these regions. In contrast, our result is basically free from these distortions (see, e.g., the left bottom region of the left example in Fig. 15, and refer to the supplementary files for a clearer comparison).

### *D. "Video Stitcher" as AE Plugin*

To demonstrate the practice of our technique, we have further developed a Plugin "Video Stitcher" at Adobe After Effects CC2015, as shown in Fig. 16. For the improved robustness and stitching quality, we integrate our grid-based tracker into the plugin. We also provide the options for different feature descriptors. We would like to share this plugin to the public in the near future.

### *E. Limitations and Future Works*

We observed some limitations of our approach, which form the direction for our future works. Fig. 17 shows one

failure example (the corresponding video is provided in the supplementary files). Here, we can see some misalignments clearly in the ground region. The reason is as follows: this video contains a strong motion blurring, especially the left one, which severely lowers the quality of the matched features (in terms of both the number and accuracy) and thus leads to the failure of mesh warping. In general, a large depth variation and an inaccurate motion estimation would lead to misalignments. To solve this problem, video deblurring [44] would be a possible solution.

In this work, we have not considered an accurate video synchronization, whereas videos are synchronized roughly by users. Users start recording approximately at the same time. Empirically, we find this manual synchronization works well in many cases. Certainly, an advanced synchronization could further improve the performance, such as [45]. Alternatively, a special APP can be designed and installed on cell-phones, which triggers the video recording at the same time [7].

## VI. Conclusion

We have presented in this paper a unified framework that jointly stabilizes and stitches multiple videos captured by moving cameras. To this end, we propose to estimate both inter motions between different cameras and intra motions within a video. Based on them, the joint video stabilization and stitching is formulated into a constrained optimization problem, in which inter motions enforce the spatial alignment and intra motions maintain the temporal smoothness, respectively. By dividing each video frame into cells so as to facilitate the bundled-path approach, our method is capable of handling scenes with parallax. Extensive simulations and some supplementaries on various videos are provided to show the effectiveness of our method.

## References


[1] R. Szeliski and H.-Y. Shum, "Creating full view panoramic image mosaics and environment maps," in *Proc. 24th Annu. Conf. Comput. Graph. Interact. Techn.*, 1997, pp. 251–258.

[2] H.-Y. Shum and R. Szeliski, "Systems and experiment paper: Construction of panoramic image mosaics with global and local alignment," *Int. J. Comput. Vis.*, vol. 36, no. 2, pp. 101–130, 2000.

[3] F. Perazzi *et al.*, "Panoramic video from unstructured camera arrays," *Comput. Graph. Forum*, vol. 34, no. 2, pp. 57–68, May 2015.

[4] J. Li, W. Xu, J. Zhang, M. Zhang, Z. Wang, and X. Li, "Efficient Video Stitching Based on Fast Structure Deformation," *IEEE Trans. Cybern.*, vol. 45, no. 12, pp. 2707–2719, Dec. 2015.

[5] W. Jiang and J. Gu, "Video stitching with spatial-temporal content-preserving warping," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. Workshops (CVPRW)*, Jun. 2015, pp. 42–48.

[6] Y. Wang, J. Wang, and S.-F. Chang. (2015). "Camswarm: Instantaneous smartphone camera arrays for collaborative photography." [Online]. Available: https://arxiv.org/abs/1507.01148

[7] Y. Wang, J. Wang, and S.-F. Chang. (2015) "Panoswarm: Collaborative and synchronized multi-device panoramic photography." [Online]. Available: https://arxiv.org/abs/1507.01147

[8] H. Jiang, H. Liu, P. Tan, G. Zhang, and H. Bao, "3D reconstruction of dynamic scenes with multiple handheld cameras," in *Proc. ECCV*, vol. 7573. 2012, pp. 601–615.

[9] D. Zou and P. Tan, "Coslam: Collaborative visual slam in dynamic environments," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 2, pp. 354–366, Feb. 2013.

[10] W.-Y. Lin, S. Liu, Y. Matsushita, T.-T. Ng, and L.-F. Cheong, "Smoothly varying affine stitching," in *Proc. CVPR*, Jun. 2011, pp. 345–352.

[11] C.-H. Chang, Y. Sato, and Y.-Y. Chuang, "Shape-preserving half-projective warps for image stitching," in *Proc. CVPR*, Jun. 2014, pp. 3254–3261.

[12] F. Zhang and F. Liu, "Parallax-tolerant image stitching," in *Proc. CVPR*, 2014, pp. 3262–3269.

[13] 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.*, vol. 28, no. 7, pp. 1150–1163, Jul. 2006.

[14] M. Grundmann, V. Kwatra, and I. Essa, "Auto-directed video stabilization with robust L1 optimal camera paths," in *Proc. CVPR*, 2011, pp. 225–232.

[15] F. Liu, M. Gleicher, H. Jin, and A. Agarwala, "Content-preserving warps for 3D video stabilization," *ACM Trans. Graph.*, vol. 28, no. 3, p. 44, 2009.

[16] S. Liu, L. Yuan, P. Tan, and J. Sun, "Bundled camera paths for video stabilization," *ACM Trans. Graph.*, vol. 32, no. 4, p. 78, 2013.

[17] M. Trajković and M. Hedley, "Fast corner detection," *Image Vis. computing*, vol. 16, no. 2, pp. 75–87, 1998.

[18] M. Grundmann, V. Kwatra, D. Castro, and I. Essa, "Calibration-free rolling shutter removal," in *Proc. ICCP*, 2012, pp. 1–8.

[19] J. Shi and C. Tomasi, "Good features to track," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. Comput. Soc.*, Jun. 1994, pp. 593–600.

[20] T. Igarashi, T. Moscovich, and J. F. Hughes, "As-rigid-as-possible shape manipulation," *ACM Trans. Graph.*, vol. 24, no. 3, pp. 1134–1141, Jul. 2005.

[21] J. Zaragoza, T. Chin, Q. Tran, M. S. Brown, and D. Suter, "As-projective-as-possible image stitching with moving DLT," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 36, no. 7, pp. 1285–1298, Jul. 2014.

[22] C. Buehler, M. Bosse, and L. McMillan, "Non-metric image-based rendering for video stabilization," in *Proc. IEEE CVPR*, vol. 2, Dec. 2001, pp. 609–614.

[23] S. Liu, Y. Wang, L. Yuan, J. Bu, P. Tan, and J. Sun, "Video stabilization with a depth camera," in *Proc. CVPR*, 2012, pp. 89–95.

[24] B. M. Smith, L. Zhang, H. Jin, and A. Agarwala, "Light field video stabilization," in *Proc. ICCV*, 2009, pp. 341–348.

[25] F. Liu, M. Gleicher, J. Wang, H. Jin, and A. Agarwala, "Subspace video stabilization," *ACM Trans. Graph.*, vol. 30, no. 1, p. 4, 2011.

[26] A. Goldstein and R. Fattal, "Video stabilization using epipolar geometry," *ACM Trans. Graph.*, vol. 31, no. 5, p. 126, Aug. 2012.

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

[28] B.-Y. Chen, K.-Y. Lee, W.-T. Huan, and J.-S. Lin, "Capturing intention-based full-frame video stabilization," *Comput. Graph. Forum*, vol. 27, no. 7, pp. 1805–1814, Oct. 2008.

[29] S. Baker, E. Bennett, S.-B. Kang, and R. Szeliski, "Removing rolling shutter wobble," in *Proc. CVPR*, 2010, pp. 2392–2399.

[30] M. Brown and D. G. Lowe, "Automatic panoramic image stitching using invariant features," *Int. J. Comput. Vis.*, vol. 74, no. 1, pp. 59–73, Aug. 2007.

[31] J. Gao, S. J. Kim, and M. S. Brown, "Constructing image panoramas using dual-homography warping," in *Proc. CVPR*, 2011, pp. 49–56.

[32] J. Gao, Y. Li, T.-J. Chin, and M. S. Brown, "Seam-driven image stitching," in *Proc. Eurographics*, 2013, pp. 45–48.

[33] A. Agarwala *et al.*, "Interactive digital photomontage," *ACM Trans. Graph.*, vol. 23, no. 3, pp. 294–302, 2004.

[34] M. Irani, P. Anandan, and S. Hsu, "Mosaic based representations of video sequences and their applications," in *Proc. ICCV*, 1995, pp. 605–611.

[35] M. Irani and P. Anandan, "Video indexing based on mosaic representations," *Proc. IEEE*, vol. 86, no. 5, pp. 905–921, May 1998.

[36] A. Hamza, R. Hafiz, M. M. Khan, Y. Cho, and J. Cha, "Stabilization of panoramic videos from mobile multi-camera platforms," *Image Vis. Computing*, vol. 37, pp. 20–30, May 2015.

[37] M. El-Saban, M. Izz, and A. Kaheel, "Fast stitching of videos captured from freely moving devices by exploiting temporal redundancy," in *Proc. ICIP*, 2010, pp. 1193–1196.

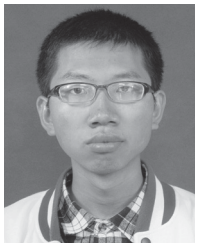
[38] M. El-Saban, M. Izz, A. Kaheel, and M. Refaat, "Improved optimal seam selection blending for fast video stitching of videos captured from freely moving devices," in *Proc. ICIP*, 2011, pp. 1481–1484.

[39] K. Lin, S. Liu, L.-F. Cheong, and B. Zeng, "Seamless video stitching from hand-held camera inputs," in *Comput. Graph. Forum*, vol. 35, no. 2, pp. 479–487, May 2016.

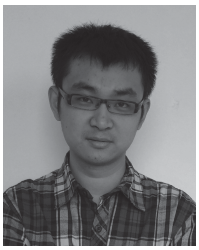
[40] E. Rosten and T. Drummond, "Machine learning for high-speed corner detection," in *Proc. ICCV*, 2006, pp. 430–443.

[41] I. Bronshtein and K. Semendyayev, *Handbook of Mathematics*. New York, NY, USA: Springer-Verlag, 1997.

[42] S. Liu, L. Yuan, P. Tan, and J. Sun, "Steadyflow: Spatially smooth optical flow for video stabilization," in *Proc. CVPR*, 2014, pp. 4209–4216.
[43] P. J. Burt and E. H. Adelson, "A multiresolution spline with application to image mosaics," *ACM Trans. Graph.*, vol. 2, no. 4, pp. 217–236, 1983.
[44] S. Cho, J. Wang, and S. Lee, "Video deblurring for hand-held cameras using patch-based synthesis," in *Proc. ACM SIGGRAPH*, vol. 31, no. 4, 2012, pp. 64-1–64-9.
[45] O. Wang, C. Schroers, H. Zimmer, A. S.-H. Markus Gross, and A. Sorkine-Hornung, "Videosnapping: Interactive synchronization of multiple videos," *ACM Trans. Graph.*, vol. 33, no. 4, p. 77, 2014.



**Heng Guo** received the B.Eng. degree in electronic information engineering from the University of Electronic Science and Technology of China (UESTC) in 2015. He is currently pursuing the master's degree with the Institute of Image Processing, School of Electronic Engineering, UESTC. His research interests include computer vision and computer graphics.



**Shuaicheng Liu** (M'15) received the B.E. degree from Sichuan University, Chengdu, China, in 2008, and the M.S. and Ph.D. degrees from the National University of Singapore, Singapore, in 2014 and 2010, respectively. In 2014, he joined the University of Electronic Science and Technology of China, where he is currently an Associate Professor with the Institute of Image Processing, School of Electronic Engineering. His research interests include computer vision and computer graphics.

**Tong He** received the B.Eng. degree in computer science from the University of Electronic Science and Technology of China (UESTC) in 2016. He is currently pursuing the Ph.D. degree with the University of California–Los Angeles (UCLA) Vision Laboratory, UCLA. His research interests include deep learning, machine learning, object detection, feature tracking, and SLAM. He received the UESTC Graduation Award.

**Shuyuan Zhu** (M'13) received the Ph.D. degree from the Hong Kong University of Science and Technology (HKUST), Hong Kong, in 2010. From 2010 to 2012, he was with HKUST and Hong Kong Applied Science and Technology Research Institute Company Limited, respectively. In 2013, he joined the University of Electronic Science and Technology of China, where he is currently an Associate Professor with the School of Electronic Engineering. He has over 40 research publications. His research interests include image/video compression, image processing, and compressive sensing. He received the Top 10 Paper Award at the IEEE ICIP 2014. He was the special session Chair of image super-resolution at the IEEE DSP 2015. He served as the Committee Member for the IEEE ICME 214, and serves as the Committee Member for the IEEE VCIP 2016. He is a Member of IEEE CAS society.

**Bing Zeng** (F'16) received the B.Eng. and M.Eng. degrees in electronic engineering from the University of Electronic Science and Technology of China (UESTC), Chengdu, China, in 1983 and 1986, respectively, and the Ph.D. degree in electrical engineering from the Tampere University of Technology, Tampere, Finland, in 1991. He was a Post-Doctoral Fellow with the University of Toronto from 1991 to 1992, and a Researcher with Concordia University from 1992 to 1993. He then joined the Hong Kong University of Science and Technology (HKUST). After 20 years of service at HKUST, he returned to UESTC in 2013, through Chinas 1000-Talent-Scheme. He leads the Institute of Image Processing, UEST, where he is involved in image and video processing, 3-D and multiview video technology, and visual big data. During his tenure at HKUST and UESTC, he graduated more than 30 Master and Ph.D. students, received about 20 Research Grants, filed eight international patents, and authored more than 200 papers. Three representing works are as follows: one paper on fast block motion estimation, published in the IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS FOR VIDEO TECHNOLOGY (TCSVT) in 1994, has so far been SCI-cited more than 870 times (Google-cited more than 1960 times) and currently stands at the seventh position among all the papers published in this transactions; one paper on smart padding for arbitrarily-shaped image blocks, published in the IEEE TCSVT in 2001, leads to a patent that has been successfully licensed to companies; and one paper on directional discrete cosine transform, published in the IEEE TCSVT in 2008, and received the 2011 IEEE CSVT Transactions Best Paper Award. He also received the best paper award at China-Com three times (2009 Xian, 2010 Beijing, and 2012 Kunming). He received the second Class Natural Science Award (the first recipient) from the Chinese Ministry of Education in 2014 and was elected as an IEEE Fellow in 2015 for contributions to image and video coding. He served as an Associate Editor of the IEEE TCSVT for eight years and received the Best Associate Editor Award in 2011. He is currently on the Editorial Board of Journal of Visual Communication and Image Representation and serves as the General Co-Chair of IEEE VCIP-2016 to be held in Chengdu, China, in 2016.

**Moncef Gabbouj** (F'11) received the B.S. degree in electrical engineering in 1985 from Oklahoma State University, Stillwater, and the M.S. and Ph.D. degrees in electrical engineering from Purdue University, West Lafayette, Indiana, in 1986 and 1989, respectively.

He was the Academy of Finland Professor from 2011 to 2015. He held several visiting professorships at different universities. He is currently a Professor of Signal Processing with the Department of Signal Processing, Tampere University of Technology, Tampere, Finland, and the TUT-Site Director of the NSF I/UCRC funded Center for Visual and Decision Informatics. He has authored over 650 publications and supervised 42 Ph. D. and 58 Master theses. His research interests include multimedia content-based analysis, indexing and retrieval, machine learning, nonlinear signal and image processing and analysis, voice conversion, and video processing and coding.

Dr. Gabbouj is a Member of the Academia Europaea and the Finnish Academy of Science and Letters. He received the 2015 TUT Foundation Grand Award, the 2012 Nokia Foundation Visiting Professor Award, the 2005 Nokia Foundation Recognition Award, and several best paper awards. He was the Chairman of the IEEE CAS TC on DSP and Committee Member of the IEEE Fourier Award for Signal Processing. He served as a Distinguished Lecturer for the IEEE CASS. He served as an Associate Editor and a Guest Editor of many IEEE, and international journals.