View-Consistent MeshFlow for Stereoscopic Video Stabilization

Heng Guo, *Student Member, IEEE*, Shuaicheng Liu, *Member, IEEE*, Shuyuan Zhu, *Member, IEEE*, Heng Tao Shen, *Senior Member, IEEE*, and Bing Zeng, *Fellow, IEEE*

***Abstract*—This paper presents a method to stabilize shaky stereoscopic videos captured by hand-held stereo cameras. It is often problematic to apply traditional monocular video stabilization techniques directly to the stereoscopic views independently. This is mainly because some undesirable vertical disparities and inaccurate horizontal disparities are produced, which violates the original stereoscopic disparity constraint, leading to erroneous depth perceptions. In this paper, we show that the MeshFlow stabilization method for monocular videos can be extended for stereoscopic videos by taking additional disparity constraints during the stabilization. In particular, we first estimate disparities between two views. Then, we compute camera motions by the MeshFlow motion model, in which the camera paths can be extracted from the meshes at each view. Next, we smooth these paths of two views separately. After path optimization, we adjust the meshes of one view by our proposed joint disparity and stability mesh warp (JDSW), so that the temporal stability and the correct depth perception can be achieved simultaneously. We evaluate our method on various challenging stereoscopic videos with different camera motions and scene types. In the experiment, we adapt the objective quality assessment of single videos to evaluate our stereoscopic video stabilization. We further propose an objective evaluation method to assess the quality of the disparities in terms of the spatial and temporal coherence after the stabilization. The experimental results demonstrate the effectiveness of our method both quantitatively and qualitatively.**

***Index Terms*—Stereoscopic, video stabilization, MeshFlow, disparity.**

I. Introduction

STEREOSCOPIC 3D is becoming increasingly popular these years, as it can enhance the experiencing of depth by feeding different views to different eyes. With the popularity of stereoscopic devices, the demand of processing stereoscopic contents raises quickly, including stereoscopic cloning [1], [2], [3], warping [4], [5], inpainting [6], [7], panorama stitching [8], [9], and retargeting [10], [11]. These techniques mainly focus on the stereoscopic image processing where good performances have been achieved for various applications, while a few works have considered the stereoscopic video processing. In this work, we focus on a basic yet very important video processing task, the stereoscopic video stabilization.

Manuscript received Jan-09-2018; revised Jun-27-2018; accepted Aug-07-2018. This work has been supported by National Natural Science Foundation of China (61502079, 61672134 and 61720106004), the "111" Project (B17008). (Corresponding authors: Shuaicheng Liu and Bing Zeng; email: liushuaicheng@uestc.edu.cn; eezeng@uestc.edu.cn)

H. Guo, S. Liu, S. Zhu, and B. Zeng are with Institute of Image Processing, School of Information and Communication Engineering. H. Shen is with School of Computer Science and Engineering, University of Electronic Science and Technology of China, Chengdu, Sichuan, China.

http://www.liushuaicheng.org/tci/stereostb/stereoscopic.mp4 (55Mb, a 3D red-blue glass is needed for observing them and also for most images shown in this paper).

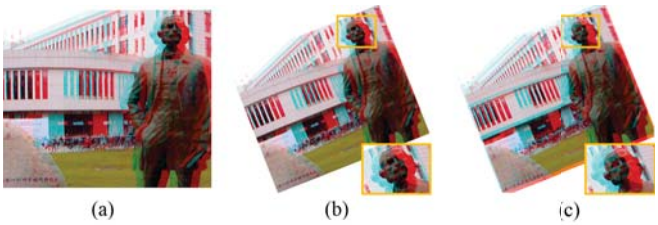


Fig. 1. Artifacts in stereoscopic image warping: (a) original stereoscopic image, (b) vertical disparities introduced by warping each view separately, and (c) results with the correct warping.

Videos (monocular or stereoscopic) captured by hand-held cameras often appear shaky and undirected. Digital video stabilization techniques improve the video quality by removing camera jitters, synthesizing videos with smoothed camera motions. Compared with monocular videos, the stabilization of stereoscopic videos is more challenging, because the shakiness in stereoscopic videos not only yields annoying jittery motions but also causes visual discomforts and dizziness [12].

The primary challenge in the stereoscopic video stabilization is to maintain a good disparity consistency. If we apply classic monocular video stabilization methods directly to each view of a stereoscopic video separately, the inherent disparities would be damaged, which yields the problematic depth perception, leading to 3D fatigue. Here, artifacts are mainly caused by two reasons. First, as proved in [5], applying the existing monocular image warping techniques directly to stereoscopic images would introduce vertical disparities (fake disparities). Fig. 1 shows such an example. If we warp the left and right views of a stereoscopic image (Fig. 1(a)) independently, vertical disparities will be introduced (Fig. 1(b)). Compared to that, Fig. 1(c) shows the warping result with correct disparities. Second, maintaining the temporal consistency is another challenge. If we stabilize each view inappropriately or put problematic disparity constraints during the stabilization, the temporal constraints could not be maintained. In the highlighted region of Fig. 2, the disparities of the status vary smoothly along the time in the original video (Fig. 2(a)). If two views were processed independently, as shown in Fig. 2(b), the disparities of the same region vary significantly between adjacent frames, which creates a hallucination as if the statue

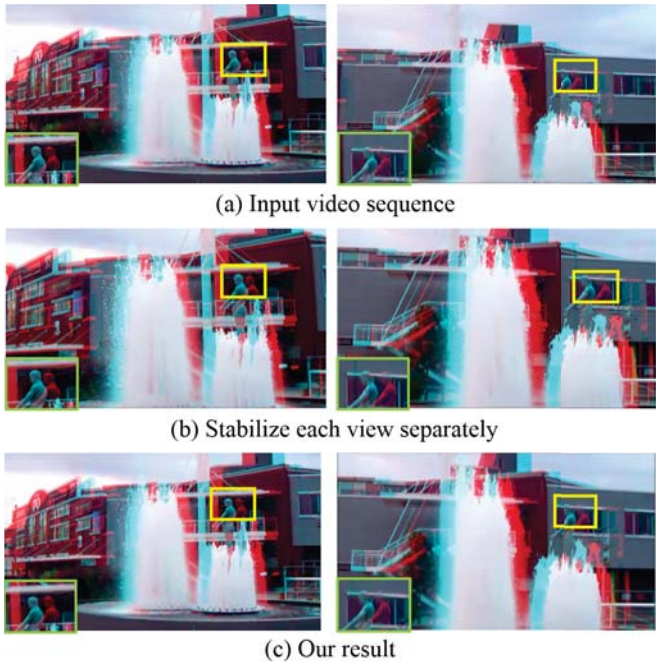


Fig. 2. Temporal artifacts caused by stabilizing each view separately: compared to the original stereoscopic video (a), the temporal coherence of disparities is damaged, as shown in (b), because of applying the monocular video stabilization method directly to each video, whereas (c) shows our result.

is moving forward and backward. Fig. 2(c) shows that our method is able to handle this problem by addressing the temporal coherence of disparities.

With regards to the spatial artifacts, Liu *et al.* [5] developed a technique to extend some existing image warping algorithms to stereoscopic images. In particular, they synthesize a target disparity map according to some user-specified warps. In this way, the original disparity distribution can be well kept to avoid introducing fake disparities. With regards to the temporal artifacts, Liu *et al.* [13] proposed a subspace approach to maintain the temporal consistency. They show that the feature trajectories from the left and right views of a stereoscopic video share a common subspace. As such, the stabilized stereoscopic video can be obtained by smoothing the common eigen-trajectories. However, this method requires long feature trajectories for the matrix factorization, which are hard to obtain (particularly when videos contain large camera shakes or quick camera rotations).

Compared with the subspace method [13], some parametric-based approaches [14], [15], [16], [17] can estimate camera motion by extracting feature correspondences between neighboring consecutive frames, which releases the requirement of long feature tracks. Therefore, these methods are robust to various challenging camera motions. However, these approaches still require the estimation of some parametric motion models, e.g., affines [14] or homographies [17], which are also hard to implement on embedded platforms, such as FPGA.

Recently, Liu *et al.* [16] proposed a MeshFlow motion model, which is a non-parametric model that can represent motions caused by non-trival depth variations. Different from parametric approaches that require either IRLS [18] or RANSAC [19] for the robust model fitting, MeshFlow relieves the complexity by calculating the sparse motions directly at grid mesh vertexes with median filters for the outlier rejection. In this way, MeshFlow can be estimated very efficiently under various platforms with the capability of representing accurately the spatially-variant motions for the high-quality video stabilization.

In this paper, we propose a robust stereoscopic video stabilization algorithm that removes the temporal jitters and preserves the disparities at the same time. We adopt MeshFlow for the motion estimation and upgrade it by adding the disparity regularization for stereoscopic videos. In particular, we begin by calculating the disparity map between two views using dense point correspondences. Then, we estimate the MeshFlow motion models within each view, from which the camera paths can be extracted. Next, we smooth the paths of two views separately. According to the smooth transform, we calculated the desired disparity based on the disparity preserving warp (DPW) [5]. After that, we adjust the meshes of one view by the proposed joint disparity and stability mesh warp (JDSW), which help to maintain the disparities and stabilities simultaneously. Guided by the target meshes of each view, we render the stabilized stereoscopic videos.

The main contribution of this paper can be summaried into the following three aspects,

1) We first introduce Meshflow into stereoscopic video stabilization problem, which enables our approach share higher efficiency and robustness over existing methods. In addition, the mesh-based motion representation helps us handle scenes with large parallax and non-trivial depth variations.
2) We propose a novel warping method named JDSW which considers disparity and stability jointly in the mesh warping. By conducting JDSW, we overcome the main challenge in stereoscopic video stabilization, that is, keeping the spatial and temporal coherence of the disparity.
3) We design an objective metric to evaluate the quality of the disparities after stabilization quantitatively.

The rest of this paper is organized as follows. Sec. II reviews the related works. Sec. III presents our method in detail. Some discussions are provided in Sec. IV. Some results are presented in Sec. V. Finally, Sec. VI concludes the paper.

II. Related Works

In this section, we present a brief review of the related works, including the traditional monocular video stabilization, the stereoscopic disparity manipulation, and the stereoscopic video stabilization.

A. Video stabilization.

Existing works on the monocular video stabilization can be roughly categorized as 2D [14], [15], [17], [20], 2.5D [21], [22], [23], and 3D [24], [25], [26], [27], [28] approaches according to their adopted motion models.

The 3D methods reconstruct 3D camera poses as well as 3D scene points, and smooth these 3D camera trajectories to stabilize the video [24], [29]. If 3D reconstructions are applicable for the video, the 3D methods often produce the best result as compared to other approaches. However, 3D reconstruction is often fragile due to the requirement of long feature tracks that are hard to obtain in videos containing occlusions, motion blurs, or quick camera motions. To recover the 3D camera motion robustly, one usually needs some extra hardware, such as the depth camera [26] or the light field camera [27]. Instead of applying the expensive and brittle 3D reconstruction, the 2.5D methods utilize partial 3D information that is embedded in long feature tracks for stabilization, such as the epipolar constraint [21] and the subspace constraint [22]. However, these 2.5D methods still require a certain length of feature tracks, causing the algorithm sensitive to the scene contents.

The 2D methods estimate a series of 2D linear transformations (e.g., affines or homographies) between consecutive frames to represent the camera motions and smooth these transformations to stabilize the video [15], [30]. Some priors are incorporated during the smoothing, such as the polynomial curves [31] and the cinematographical rules [14]. Compared to the 3D methods, the 2D motion representation does not encounter the problem of long feature tracks since it only extracts feature correspondences between adjacent neighboring frames. However, a single linear model is usually insufficient to model scenes with large depth variations [32]. To address this issue, Liu *et al.* [17] proposed a mesh-based motion model. By dividing each video frame into regular cells and computing a homography for each cell independently, this method achieves the capacity of non-linear motion representation. Later on, Liu *et al.* [16] proposed a MeshFlow motion model, which replaces the mesh homographies by 2D motion vectors. For the efficiency and robustness, we use MeshFlow for our motion estimation.

B. Stereoscopic disparity manipulation

It is crucial to maintain stereoscopic disparities for the high-quality stereoscopic image/video editing. Luo *et al.* [1] proposed a two-step iterative disparity adaptation process to reduce depth discontinuities and maintain global depth structures for the stereoscopic image cloning. Lo *et al.* [2] enabled the stereoscopic image copy-and-paste by an end-to-end system, which is robust to the disparity estimation with certain inaccuracy. Lang *et al.* [33] exploited a non-linear and locally adaptive algorithm for the disparity mapping, which is based on the visual saliency of scene elements to remap the disparity range. Lee *et al.* [11] proposed a layer-based stereoscopic resizing algorithm by using the mesh deformation. Wang *et al.* [6] developed a stereoscopic inpainting system that takes stereo images as input and fills the missing color and depth. Niu *et al.* [5] presented a technique to compute an optimal target disparity map based on some user-specified warps, which is free from the artifacts such as vertical disparities and 3D fatigues. Du *et al.* [4] developed a method to manipulate perspectives in stereoscopic image pairs. Didyk *et al.* [34] introduced a perceptual model of disparity for computer graphics, which defines a metric to compare a stereo image to an alternative stereo image, yielding the magnitude of the perceived disparity change. In order to keep correct perceptive depth, we adopt the method of [5] to warp disparities.

C. Stereoscopic video stabilization

There are a few works focusing on the stereoscopic video stabilization. Chu *et al.* [35] developed a mobile application for stereoscopic stabilization, which relies on extra devices including the gyroscope and accelerometer embedded on the smart phones to obtain the camera pose. However, since the camera motions of the left and right views are different from each other, it is unreasonable to estimate the motion of two views from the same sensor data. Also the gyroscope and accelerometer are influenced by surrounding temperature and their accumulative drifts make the motion data less reliable. Smith *et al.* [27] claims their light field video stabilization algorithm can be extended for stereoscopic videos. But their method ignores the disparity consistency between left and right views, which heavily affects the depth perceptual in stereoscopic videos. Liu *et al.* [13] proposed a method that considers the video stabilization and stereoscopic disparity maintaining jointly in the subspace video stabilization framework [22]. They show that the feature trajectories from the left view to the right view share a common subspace. In other words, the low-rank subspace theory not only stands for the traditional videos, but also holds for the stereoscopic videos. However, their work requires long feature tracking. Practically, long feature tracks are hard to obtain in casual videos due to quick camera motions. In contrast, our method only requires feature matches between neighboring frames, and therefore improves the robustness largely. Moreover, we design a novel mesh warping method JDSW to keep the spatial and temporal coherence of the disparities.

III. Our Method

Figure 3 shows our pipeline. We calculate disparities between the left and right videos and estimate the MeshFlow-based camera motions between neighboring frames within each video. The pipeline involves three types of operation: traditional monocular MeshFlow video stabilization, disparity warp, and JDSW warp. For the clarification and completeness, we begin by briefly revisiting the disparity warp [5] and MeshFlow stabilization [16], following which we describe the JDSW warp and finish the pipeline.

A. Disparity

In this section, we present the details regarding the disparity calculation and warping. Note that, we follow the same idea with [5] on disparity warping except for using DISflow[36] to accelerate the disparity estimation. The accuracy of the disparity plays an important role for the correct depth perception, which can influence the quality of JDSW in the subsequent step. In the following, we first present the estimation of the disparity map. Then, we discuss the details regarding the disparity warp.

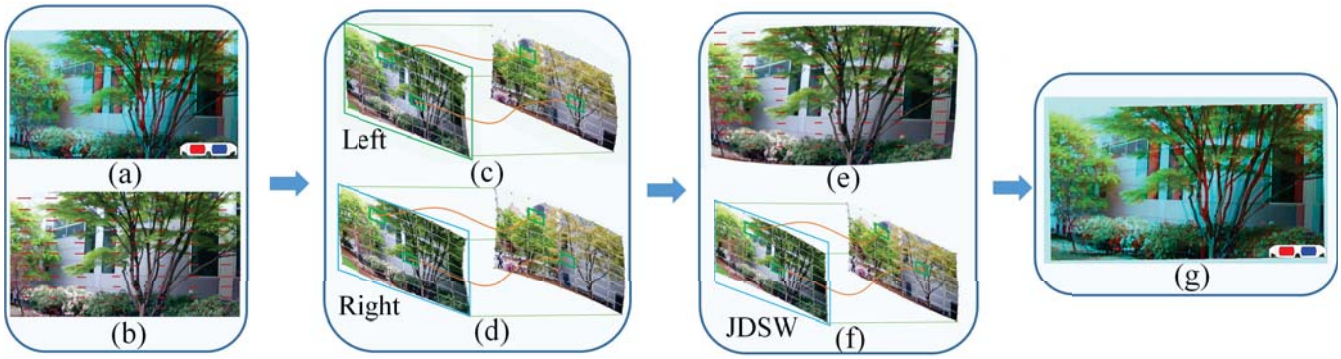


Fig. 3. Our system pipeline. (a) The input stereoscopic video. (b) Estimated disparities sampled for illustration. (c) and (d) are the left and right videos, which are smoothed by MeshFlow [16] separately. (e) The disparities are warped according to [5]. (f) Without loss of generality, we warp the right meshes using JDSW, and the left meshes are directly derived from stabilization. (g) shows the final result.

1) Disparity estimation: Per-pixel dense disparity estimation has been studied for decades [37]. The existing approaches as well as the challenges have been well documented in [38]. In this work, our disparity comes from two sources. We extract sparse feature matches between two views [39]. The outliers are excluded by the homography-based RANSAC [32]. The sparse features can only cover the areas with rich textures. To deal with the poorly-textured regions, we further calculate the dense optical flow. We sample the densely-matched correspondences uniformly (every five pixels) to cover the textureless regions. For the efficiency, we adopt a recent fast optical flow method [36]. The correspondences from the sparse features and the dense flows are treated equally in our system. The former usually possesses the high precision and improves the accuracy of the disparity for the regions with rich textures, while the latter can provides constraints for non-textured regions, which is also very important for the high-quality disparity warp. As such, we obtain a set of points with disparity values. For a point in the left view p_i^l and its matched point p_i^r in the right view, the disparity can be obtained as:

$$d_i = p_i^l - p_i^r, \tag{1}$$

where d_i donates the *i*-th disparity.

2) Disparity warping: The original disparities can be easily damaged if the image warp is applied separately on two views. As shown in [5], when applying an image warp, the desired disparity is linearly correlated with the original disparity in the local image regions. Therefore, the target disparity map can be obtained by minimize the following energy:

$$\sum_{d_i} \sum_{d_j \in N(d_i)} \|(\hat{d}_i - \hat{d}_j) - s_i(d_i - d_j)\|^2 \tag{2}$$
$$s.t. \quad \hat{d}_{min} = s d_{min},$$

where d_i and $\hat{d}_i$ are the original and the desired warped disparities of a point i[1], and $N(d_i)$ denotes the neighboring points locating around the *i*-th point. Here, the window size is set to 30×30, s_i is a scaling factor obtained from a similarity transformation that is fitted from neighboring pixels centered at i before and after the image warp. In addition, the boundary condition is set to the point with the smallest disparity. d_{min} denotes the minimum magnitude of the input disparity and s represents the scaling factor around this point. In particular, to calculate the local scaling factor s_i, a 3×3 window centred at i is defined. Image pixel locations are extracted within this local window before and after the image warp. The optimal similarity transformation is obtained by minimizing the energy:

$$arg \min_{H_s} \sum_{c} \|H_s c - \hat{c}\|^2 \tag{3}$$

where c and $\hat{c}$ denote the extracted corresponding points before and after the warping, respectively. H_s is confined to a similarity transform and s_i is extracted from the scaling coefficient of H_s. For more details, please refer to [5].

Notably, the warping function described in [5] is a user-specified image warp. In our scenario, the image warp comes from the MeshFlow video stabilization [16].

[1]We use $(\hat{\cdot})$ to denote disparities or points in the warped coordinates.

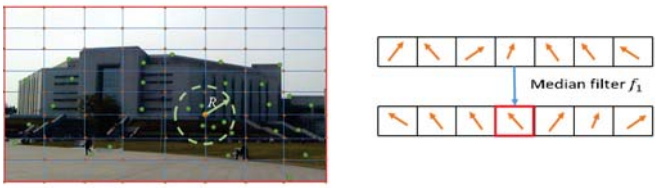


Fig. 4. Motion propagation in MeshFlow [16]: we assign the neighboring feature motion flow to the motion list of a vertex, and then the motion of the vertex can be obtained after the median filtering.

B. MeshFlow video stabilization

In this part, we first introduce the principle of the MeshFlow motion model [16] and then discuss the details of how to smooth the camera motion adaptively.

1) MeshFlow motion model: MeshFlow[16] is a regular sparse motion field with the motion vectors located at the mesh vertexes. It describes the camera motion between two

consecutive video frames. The estimation of MeshFlow contains three steps:

- Extract sparse features between consecutive frames and calculate the motion vectors at each point.
- Divide the video frame into regular grid mesh and determine the motion vector at mesh vertexes according to the neighboring feature motions.
- Smooth the motion vectors by a spatial median filter to remove noise.

To start with, we first extract feature correspondences between neighboring frames. We detect FAST features [40] and find the corresponding points in the adjacent frame using the KLT tracking algorithm [41]. Rather than using a global RANSAC [32] to remove the outliers, we adopt the sub-region RANSAC, where the frame is divided into 4×4 subimage and a local homography-based RANSAC is used to refine the feature set in each subimage. Suppose that $(p, \hat{p})$ is the corresponding feature point between two frames, the motion flow of p can be computed as: $F_p = p - \hat{p}$.

Notably, the distribution of the features plays an important role during the estimation of MeshFlow. Recently, Guo *et al.* [42] proposed an approach to stitch multiple individually captured videos with common contents. They proposed a strategy to extract uniformly-distributed features within and across views by adopting a local threshold adaptation and a plane-based verification. We borrow the similar idea for our stereo video feature extraction.

Now, we describe how to estimate the motion at the mesh vertexes. We divide the frame into 16×16 regular mesh grids. Each mesh vertex can gather some motion vectors from its nearby feature motions. We set a circle region for each vertex, as indicated by the dashed circle shown in Fig. 4. Empirically, the radius of the circle is set to the width of the mesh grid. For each vertex, the feature points located inside are pushed into a motion list, as shown in the right part of Fig. 4. We then use a median filter to smooth the motion vectors in the list and assign the result to the vertex as its motion vector.

Notice that the sparse motion field obtained in this way tends to suffer from some noise, caused by the tracking errors and dynamic moving objects. To enforce a further regularization, we apply another 3×3 median filter on these vertexes to produce a spatially-smoothed sparse motion field - the so called MeshFlow.

After calculating MeshFlow for every consecutive video frames, we can extract the *vertex profiles* as our camera trajectory [16], [20]. A vertex profile collects motions at a spatial vertex location along the time. Let us use $F_i(t)$ to denote the motion vector at the i-th vertex at time t. The camera motion $\mathbf{C}$ along the time can be defined as the accumulation of the consecutive motions:

$$C_i(t) = F_i(t) + F_i(t-1) ... + F_i(1) + F_i(0), F_i(0) = 0, \quad (4)$$

where $C_i(t)$ represents the camera trajectory at the spatial location of the *i*-th vertex. Given the original camera path $\mathbf{C} = C(t)$, we then adopt the adaptive path optimization method to obtain the smoothed camera path.

2) Adaptive path smooth: We first describe our adaptive smooth strategy for a single camera path, and then extend it to multiple camera paths.

Given an original camera path $\mathbf{C} = C(t)$, we obtain the smoothed path $\mathbf{P} = P(t)$ by minimizing the following energy:

$$\begin{aligned} \mathcal{O}(\{P(t)\}) = & \sum_t \|P(t) - C(t)\|^2 \\ & + \sum_t (\lambda_t \sum_{r \in \Omega_t} \omega_{t,r}(C) \cdot \|P(t) - P(r)\|^2), \end{aligned} \quad (5)$$

where Ω_t denotes the temporal neighborhood at frame t. $\|P(t) - C(t)\|^2$ is the data term that enforces the optimal camera path to be close to the original path to reduce the cropping and distortion. $\|P(t) - P(r)\|^2$ is the smoothness term that stabilizes the path. The smooth kernel $w_{t,r}$ is a Gaussian weight that is set to $exp(-\|r-t\|^2/(\Omega_t/3))$. λ_t is a parameter that balances the smoothness for each frame. It is set to 1 initially and refined during the iterative minimization of the energy in Eq. (5) such that the cropping and the stability can be balanced, and the wobbling distortions can be suppressed. Please refer to [17] for more discussions.

Eq. (5) minimizes a single camera trajectory. To minimize all paths for all vertexes, a spatial constraint is imposed [17]:

$$\sum_i \mathcal{O}(\{P_i(t)\}) + \sum_t \sum_{j \in N(i)} \|P_i(t) - P_j(t)\|^2, \quad (6)$$

where $N(i)$ includes eight neighbors of the *i*-th vertex profile. The first term comes from the Eq. (5). The second term enforces the similarities between neighboring paths. Notably, in practice, when the motion fields are with strong spatial smoothness, it is of less importance to enforce the neighboring similarities as Eq. (6) [16], [20]. We have tested the approach of with and without the similarity term, and the results are visually similar.

$$B_i(t) = P_i(t) - C_i(t), \quad (7)$$

After the path optimization, we calculate the stability transform $B_i(t)$ by Eq. (7) to move each vertex *i* to its stabilized position. Then guided by the stabilized mesh in each frame, we render the left view to a smooth output. After that, the result of the right view will be synthesized by JDSW which will be presented in Sec. III-C.

C. Joint disparity and stability mesh warp (JDSW)

The Joint disparity and stability mesh warp (JDSW) is one of the contributions in this work, where we use disparity constraints and stability constraints to guide the mesh warping. In this way, the temporal stability and the correct depth perception can be achieved simultaneously. Before stepping into this part, we would like first introduce some notations.

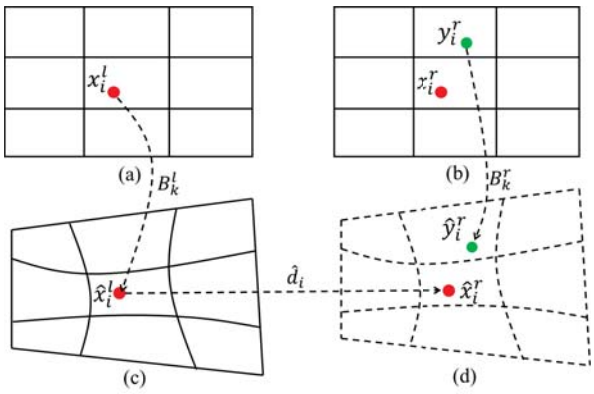


Fig. 5. Diagram of 'JDSW'. The blue dot and the red dot denote disparity point and motion point, respectively. The left mesh (a) is warped to (c) by MeshFlow video stabilization. (d) denotes the target right mesh warped from (a) using 'JDSW'

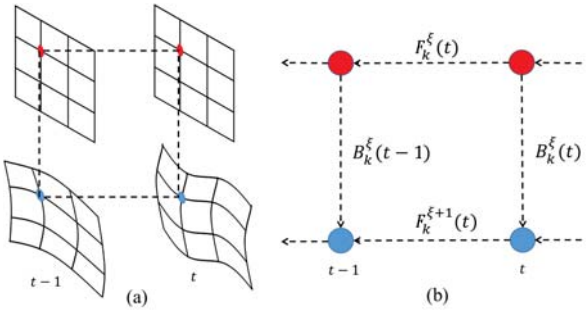


Fig. 6. Mesh configuration and motion relationship. (a) meshes (left or right) at $t-1$ and t before and after warp. (b) The unknown motions $F_k^{\xi+1}(t)$ can be derived as $(F_k^{\xi}(t) + B_k^{\xi}(t-1) - B_k^{\xi}(t))$.

1) Notations: After video stabilization, we obtain $B_k^l(t)$ and $B_k^r(t)$ at vertex k for both views. Let us omit the index t for simplicity. The left mesh can be warped using B_k^l as shown in Fig. 5(a) and (c). However, we cannot warp the right mesh using B_k^r directly, which could damage the disparities between two views. An original disparity point pair is defined as (x_i^l, x_i^r), with disparity of d_i, and shown as red dots in (a) and (b) of Fig. 5, where $d_i = [d_i, 0]^T$. Notably, we always use k, h to index mesh cells and i, j to index points. We further denote $\hat{x}_i^l$ as the transformed point of x_i^l using corresponding B_k^l ($\hat{x}_i^l = x_i^l + B_k^l$). The disparity point $\hat{x}_i^r$ (Fig. 5(d)) can be obtained using $\hat{x}_i^l$ and the corresponding warped disparity $\hat{d}_i$ ($\hat{x}_i^r = \hat{x}_i^l + \hat{d}_i$), which is estimated in Sec. III-A1-A. We further denote motion points as y_i^r, which are uniformly sampled (every five pixels) in the right frame (green dot in Fig. 5(b)). The warped motion point of the right view $\hat{y}_k^r$ (Fig. 5(d)) can be obtained by using B_k^r as $\hat{y}_i^r = y_i^r + B_k^r$.

The green dot $\hat{y}_i^r$ indicates the stabilized position to which the right mesh should move, while the red dot $\hat{x}_i^r$ encodes the correct disparities that should also be satisfied. We seek for a mesh warp that best satisfies two requirements (dotted mesh in Fig. 5(d)). Note that there are many disparity points and motion points, we only show one of them in Fig. 5 for illustration.

2) Disparity constraints $\{(x_i^r, \hat{x}_i^r)\}$*:* The disparity constraints are encoded in the point pairs $(x_i^r, \hat{x}_i^r)$, which come from the initial disparity correspondences (x_i^l, x_i^r), the warped disparity $\hat{d}_i$, and the warping transform B_k^l, where $\hat{x}_i^r$ is calculated as $\hat{x}_i^r = x_i^l + B_k^l + \hat{d}_i$. By enforcing disparity constraints, the right view tend to move to the positions which preserve desired disparity with the left view.

3) Stability constraints $\{(y_i^r, \hat{y}_i^r)\}$*:* Stability constraints aim to keep the temporal smoothness in the right view. For a point y_i^r in the original frame (Fig. 5(b)), its stabilized point $\hat{y}_i^r$ (Fig. 5(d)) can be obtained as: $\hat{y}_i^r = y_i^r + B_k^r$, where B_k^r is the corresponding updating vector from stabilization.

To obtain the warped mesh that takes both stability and disparity into considerations, we take all disparity constraints $\{(x_i^r, \hat{x}_i^r)\}$ and stability constraints $\{(y_i^r, \hat{y}_i^r)\}$ as the control points, and assign these two sets with equal weights. Then, we apply the MeshFlow model estimation (Sec. III-B1) by using these jointed control point correspondences. The estimation process can automaticallpy seek a balance between these two sets and find the optimal mesh warp. More discussions and some comparisons with the traditional content preserving warp (CPW) [24], [43] will be given in the Sec. IV-C.

After getting optimal meshes from JDSW, we render the right view using mesh warping. Combined with the synthesized video of the left view, a stabilized and disparity-consistent stereoscopic video result is finally obtained. The framework of our algorithm can then be concluded as follows.

Algorithm 1 Framework of Stereoscopic Video Stabilization.

Input:
Stereoscopic video sequence including left views $\{L_1, L_2, ...L_N\}$ and right views $\{R_1, R_2, ...R_N\}$;

Output:
Stabilized stereoscopic video sequence;

Extract matched features between consecutive frames for left and right views separately;
for each $i \in [1, N]$ **do**
  Extract the DISflow[36] between L_i and R_i, calculate the original disparity d_i;
  Estimate the camera motions C_i^L and C_i^R based on Meshflow[16] for left and right views;
end for
Smooth the camera path iteratively and obtain stabilization mesh transform B_i^L and B_i^R;
for each $i \in [1, N]$ **do**
  Calculate the warped disparity $\hat{d}_i$ based on B_i^L;
  Conduct JDSW on the right view and obtain mesh transform $\hat{B}_i^R$;
  Render the left view L_i based on B_i^L;
  Render the right view R_i based on $\hat{B}_i^R$;
end for

IV. Discussions

MeshFlow has been utilized in two places in our system, i.e., the video stabilization and the JDSW warp. The former estimates the camera motions between adjacent frames while

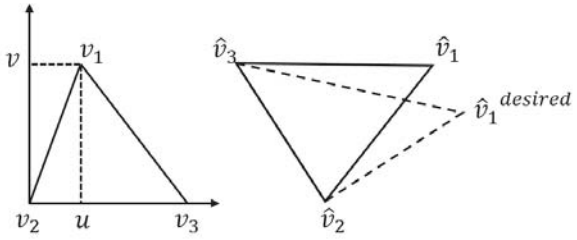


Fig. 7. The rigidity term in CPW. (a) the local coordinate system before warping. (b) $\hat{v_1}$ should be represented by $\hat{v_2}$ and $\hat{v_3}$ under the same local coordinates after warping.

the latter warps images with disparity and stability constraints. In the stabilization, MeshFlow is used as a motion model. In the JDSW warp, MeshFlow works as the image warping model. In [44], we have utilized the bundled camera paths [17] for the motion recovery and adopted the content preserving warp (CPW) [24] for the image warp (frame rendering), which involves the techniques that are similar to the as-rigid-as-possible mesh deformation [43]. In the following, we first present how to conduct JDSW using CPW, and then compare the performances of CPW and MeshFlow with respect to both the JDSW warp and motion estimation.

A. CPW for JDSW

Let V denotes the mesh vertices for the input mesh. The mesh is warped by optimizing the following energy:

$$E(\hat{V}) = \lambda_1 E_d(\hat{V}) + \lambda_2 E_s(\hat{V}) + \lambda_3 E_r(\hat{V}) \tag{8}$$

where $\hat{V}$ are the unknown mesh vertices; $E_d(\hat{V})$, $E_s(\hat{V})$, and $E_r(\hat{V})$ account for the disparity term, the stability term, and the rigidity term, respectively, with λ_1, λ_2 and λ_3 being the associated weights ($\lambda_1 = 5, \lambda_2 = 1, \lambda_3 = 1$) [44].

1) Disparity term: The disparity term comes form the disparity constraints $\{(x_i^r, \hat{x}_i^r)\}$. We represent each point x_i^r by its 2D bilinear interpolation of four vertexes $V_i = [v_i^1, v_i^2, v_i^3, v_i^4]$ of the enclosing grid cell: $x_i^r = V_i w_i$, where $w_i = [w_i^1, w_i^2, w_i^3, w_i^4]^T$ are the interpolation weights that sum to 1. For the warped vertexes $\hat{V}_i = [\hat{v}_i^1, \hat{v}_i^2, \hat{v}_i^3, \hat{v}_i^4]$, we use the same weights to represent $\hat{x}_k^r$ after warping. The disparity term is defined as:

$$\begin{aligned} E_d(\hat{V}) &= \sum_i \|\hat{V}_i w_i - \hat{x}_i^r\| \\ &= \sum_i \|\hat{V}_i w_i - (B_k^l + x_i^l + \hat{d}_i)\| \end{aligned} \tag{9}$$

2) Stability term: Similarly, the stability term comes from the motion constraints $\{(y_i^r, \hat{y}_i^r)\}$. The motion points y_i^r are represented by their corresponding enclosing grid cell as $y_i^r = V_i w_i$. The stability term is then defined as:

$$\begin{aligned} E_s(\hat{V}) &= \sum_i \|\hat{V}_i w_i - \hat{y}_i^r\| \\ &= \sum_i \|\hat{V}_i w_i - (B_k^r + y_i^r)\| \end{aligned} \tag{10}$$

TABLE I
AVERAGED DIFFERENCE BETWEEN MESHFLOW AND CPW IN JDSW.

1	2	3	4	5	6
0.395	0.763	1.266	1.274	1.816	0.949
7	8	9	10	11	12
0.855	0.785	0.767	0.446	0.933	1.157
					unit:pixel

TABLE II
EFFICIENCY COMPARISON BETWEEN MESHFLOW AND CPW IN JDSW.

index	1	2	3	4	5	6
Meshflow	8.87	5.43	7.38	8.77	8.52	8.11
CPW	120.05	64.93	109.12	138.01	128.84	112.18
index	7	8	9	10	11	12
Meshflow	7.29	7.06	6.65	7.19	6.87	7.43
CPW	103.68	97.73	106.01	102.79	91.08	108.98
						unit:ms

3) Rigidity term: It enforces the spatial smoothness during the mesh deformation. Each grid cell is divided into two triangles. Fig. 7 shows one of the triangles before and after warping. For each triangle, v_1 can be represented by the other two vertices v_2 and v_3 in a local coordinate system. Let (u, v) be the normalized local coordinates of v_1(Fig. 7(a)). We encourage $\hat{v}_1$ to be still represented by $\hat{v}_2$ and $\hat{v}_3$ under the same local coordinates after warping(Fig. 7(b)). Then, the following distance should be minimized respect to $\hat{v_1}, \hat{v_2}$, and $\hat{v_3}$ [24], [43]:

$$\|\hat{v}_1 - (\hat{v}_2 + u(\hat{v}_3 - \hat{v}_2) + vR_{90}(\hat{v}_3 - \hat{v}_2))\|^2, \tag{11}$$

where u, v are the same values computed before warping, $R_{90} = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix}$, and $E_r(\hat{V})$ consists of all such cost from all triangles collected from every grid cells. Eq. (8) is quadratic and can be minimized by a sparse linear system.

B. MeshFlow vs. CPW in JDSW

Both MeshFlow and CPW can conduct the JDSW warp. The former directly moves each mesh vertexes according to their surrounding motion vectors while the latter solves a global energy to determine the warped vertexes. Here, we design an experiment to compare their performances. We collect 12 examples as shown in Fig. 8. For each example, we warp the frame by MeshFlow and CPW separately both with the same mesh resolution 16×16 and the same warping constraints. Then, we compute the average difference (measured by pixels) of all vertexes warped by these two approaches. Table I shows the results. It is clear from these results that the differences are marginal. In the meantime, by observing the resulted video visually, one can also notice that these differences are negligible. In Table II, we show the time usage when computing JDSW in each frame based on CPW and Meshflow. Obviously, Meshflow is more than 14 times faster than CPW in average, that is also fit in motion estimation module. Therefore, with negligible differences between the meshes obtain from these two warping methods, we prefer to choose Meshflow for efficiency.

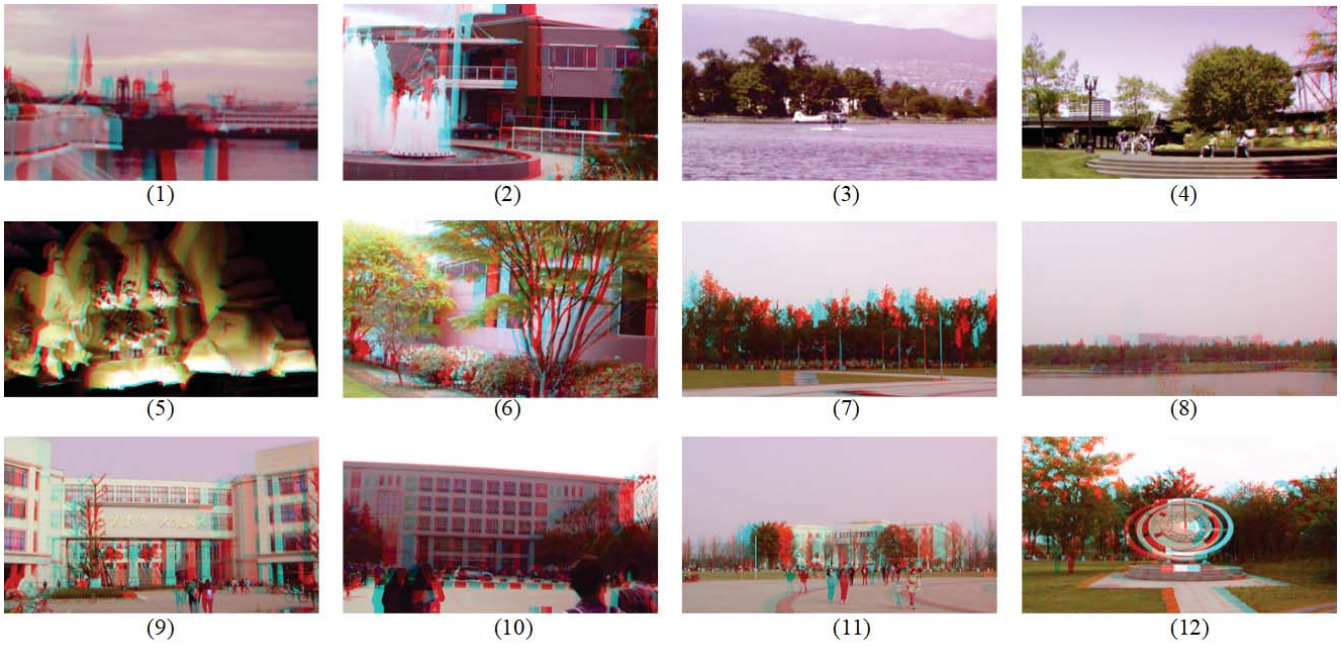


Fig. 8. Our results on various challenging videos with different scene types and camera motions. These examples can be found under the project page: http://www.liushuaicheng.org/tci/stereostb/index.html

C. MeshFlow vs. CPW in motion estimation

As the approach in [44] adopts the bundled paths [17] for the motion estimation, it requires CPW to warp two adjacent video frames (warp frame t towards frame $t-1$). Then, it is necessary to compare the performances of MeshFlow and CPW with respect to the motion estimation. Here, we also use 12 examples in Fig. 8. We warp the adjacent frames by these two approaches. Similarly, the warping constrains and the mesh resolution (16×16) remains the same for both methods. We record the average vertexes differences with the same strategy. Table III summaries the results. Again, we can hardly observe these differences visually.

In this way, we compare the performances of MeshFlow and CPW experimently. We show that they can produce comparable results. Also, as discussed in the next section, MeshFlow runs 25 times faster compared with CPW. Moreover, it is a non-parametric motion model that can be easily transplanted to embedded platforms. Therefore, we replace CPW with MeshFlow, which is more efficient and lightweight.

TABLE III
AVERAGED DIFFERENCE BETWEEN MESHFLOW AND CPW IN MOTION ESTIMATION.

1	2	3	4	5	6
0.165	0.597	0.413	0.134	0.504	0.615
7	8	9	10	11	12
0.574	0.678	0.401	0.437	0.489	0.574
					unit:pixel

V. EXPERIMENTS

We run our method on a PC with Intel i5 2.4GHz CPU and 12G RAM. For a stereoscopic video with resolution of 1280×720, we extract $400 \sim 550$ FAST features [40] at each frame. We track them using KLT [41]. We divide each frame into 16×16 mesh grids and estimate the motion using MeshFlow. Our algorithm takes 83.1 milliseconds to process a frame (12fps). Specifically, we spend 2.13ms, 2.01ms, 1.23ms to extract features, to estimate MeshFlow motions, and to smooth camera trajectories, respectively. With respect to the disparity manipulation, we take 6.58ms to calculate disparities by the DIS optical flow [36] and 42.2ms to warp disparities of a frame. Moreover, JDSW and the frame rendering consume 7.45ms and 22.8ms. If we replace MeshFlow by CPW in the motion estimation and JDSW, the time consumption were 51.103ms and 98.54ms respectively. Compared with the subspace method [13] and the video stitching [42], which achieve 4fps and 1.5fps at 640×480 resolution, our method achieves a much higher efficiency. We use the 2D sparse motion flow rather than the parametric homography to represent the camera motion. Therefore, we can release computations from the model fitting and the matrix operations, such as the inverse and multiplications. As the subspace method [13] computes the mesh vertexes by optimizing trajectories in the transformed subspace while the video stitching method [42] optimizes a joint energy function, they are not as efficient as ours that only involves light-weighted computations, such as the vector addition/substraction and the median filtering.

The memory usage of our method is flexible with the amount of extracted features and mesh grid division. Based on the above parameter setting, our algorithm needs up to 530MB to process a 1280×720 stereoscopic video with 600 frames.

We captured several stereoscopic shaky videos with the resolution of 1280×720. Fig. 8 shows the thumbnails of

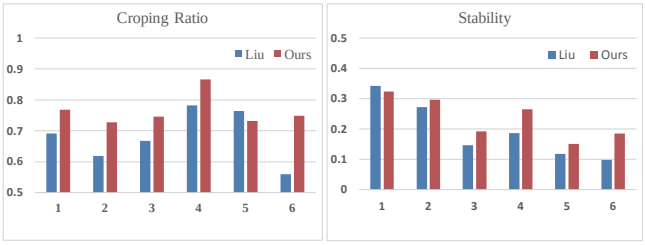


Fig. 9. Quantitative comparasion with the subspace method [13] on publicly available data.

these videos. The first 6 examples are collected from [13] and the rest are captured by ourselves. We stabilize them using the proposed approach. The 7th and 8th videos contain quick camera rotations; the 9th,10th, and 11th examples contains dynamic moving objects; the 8th contains large portions of poor textured regions. The results show that our method can handle these challenging cases robustly.

A. Quality evaluation of the stabilization

To evaluate our method quantitatively and objectively, we follow the idea of [17]. Specifically, the following two values can be defined to measure the quality of the stabilization, the cropping ratio and the stability.

1) Cropping ratio: After the video stabilization, each frame is transformed to a new position. We find a maximum rectangle to crop the video frames which ensures each frame do not contain empty regions. The cropping ratio is the percentage of the remaining area after cropping over the original area. The maximum value is one.

2) Stability: We extract feature trajectories of the original video as well as the stabilized video of each view. For those trajectories whose lengths are longer than 20 frames, we calculate their Fourier transforms. We set the 2nd to 7th components in the frequency spectrum as the low frequency portion and the others are high frequency part. Then, the stability is defined as the decrease percentage of high-frequency components in the stabilized video compared to its original shaky video. A higher stability score implies that the video is more smooth.

3) Comparisons: We compare our method with [13] that extends the subspace method [22] to stabilize the stereoscopic videos. The comparison is carried out on 6 videos provided in [13]. The thumbnails of these videos are shown at the first 6 examples in Fig. 8. Fig. 9 shows the results in terms of the cropping ratio and the stability. As shown in the figure, our method achieves a higher stability and keeps a larger visual contents in most of the cases.

B. Quality evaluation of the disparity

In this section, we propose an approach to evaluate the quality of the disparities in the stabilized video. The evaluation metric contains two parts: vertical disparity variance and the temporal and spatial coherence of the horizontal disparity.

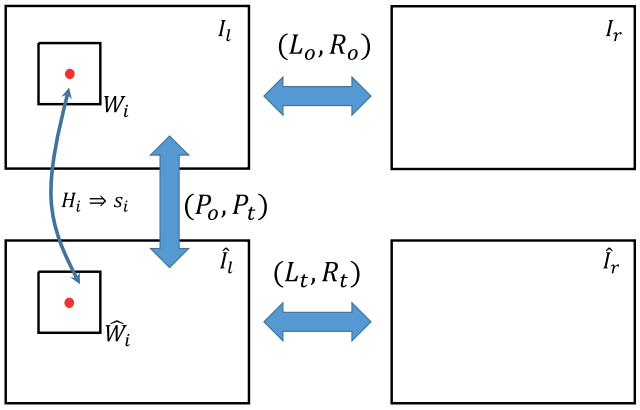


Fig. 10. The quality measurement for disparity in the stereoscopic video.

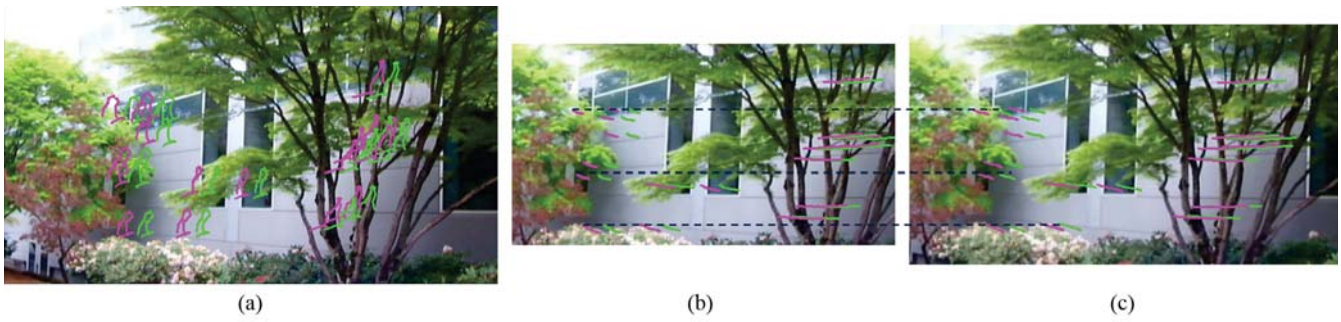
TABLE IV
AVERAGE VERTICAL DISPARITY

index	1	2	3	4	5	6
Liu [13]	0.0	0.082	0.001	0.0	0.102	0.008
Our result	0.0	0.0	0.0	0.0	0.081	0.001
W/O JDSW	0.002	0.361	0.102	0.009	0.243	0.015
						unit:pixel

1) Vertical disparity variance: For a regular stereoscopic video, the vertical disparity should be zero. After image warping, problematic disparity maybe introduced as shown in Fig. 1(b). Therefore, we calculate the average vertical disparities in the stabilized stereoscopic results. Based on dense feature correspondences between left and right view, we can obtain the disparities of the stabilized stereoscopic videos. Then we record the average of the vertical disparity along all the video frames in Table IV. JDSW can effectively remove the vertical disparity introduced after stabilization and our results are better than [13] on this metric.

In Fig. 11, we visualize tracks of some sampled feature points. In original stereoscopic video, feature tracks are shaky and have high frequency jitters. After video stabilization, we can see the feature tracks are smooth. At the same time, the dashed lines shown in Fig. 11 imply that both [13] and our method hardly introduce vertical disparity in the stabilized stereoscopic video. In addition, since the thumbnails are in same scale, we prove that our stabilization algorithm can keep more visual content compared to [13].

2) Spatial and temporal coherence of the disparity: As shown in Fig. 10, the input stereoscopic video frames at time t are denoted as I_l and I_r, and the stabilized frame are denoted as the $\hat{I}_l$ and $\hat{I}_r$. We calculate three dense optical flow fields between these frames using the approach developed in [36]. The flow fields between I_l and I_r, $\hat{I}_l$ and $\hat{I}_r$, I_l and $\hat{I}_l$ are denoted as (L_o, R_o), (L_t, R_t), and (P_o, P_t), respectively. We extract the x component from the first two fields as the disparities. The evaluation metric is derived by comparing these disparities before and after the stabilization. The third field captures the warping transformation between the pixels of the original frame and the stabilized frame, which provides a local scale factor during the comparison.



(a) (b) (c)

Fig. 11. Visualization of the feature tracks. (a) original stereoscopic video. (b) Stabilized video obtained from [13]. (c) Stabilized video obtained from our method. The purple tracks are located in the left view and the green ones are belong to the right view. The thumbnails shown here are in same scale.

TABLE V
SPATIAL COHERENCE

($\times 10^3$)	1	2	3	4	5	6
[13]	13.36	14.26	32.53	55.69	**0.101**	18.83
Our result	**8.59**	**11.91**	**27.93**	**15.84**	0.89	**14.18**
W/O JDSW	29.31	19.55	39.93	46.84	1.12	21.49

TABLE VI
TEMPORAL COHERENCE

($\times 10^3$)	1	2	3	4	5	6
[13]	**8.33**	**13.47**	154.47	79.056	90.87	30.24
Our result	8.50	17.88	**62.71**	**69.88**	**54.41**	**10.44**
W/O JDSW	11.87	23.23	101.38	87.76	123.49	13.19

In particular, we further extract SIFT matches [39] between the original frame and the stabilized frame of the left view (Fig. 10, the red points). For each SIFT match, we set a local window (typically radius is set to 30) centered at each matched pair (Fig. 10, W_i and $\hat{W}_i$). Then, we collect the dense matches from (P_o, P_t) within the window and fit a local homography H_i from these dense matches. We extract the scale component s_i from the similarity transform H_i. Next, we collect all disparities d_i and $\hat{d}_i$ from (L_o, R_o) and (L_t, R_t) within the local window W_i and $\hat{W}_i$. The disparity value at the window centers (the red points) are denoted as d_c and $\hat{d}_c$.

As suggested by [5], the correct disparities are linearly correlated within a local region before and after the warp. Therefore, we calculate the differences of the disparities with respect to the center of each window and compare these differences by allowing a local scale change s_i:

$$M_i = |1 - \frac{\sum_{j \in W_i} \|d_j - d_c\|}{s_i \cdot \sum_{j \in \hat{W}_i} \|\hat{d}_j - \hat{d}_c\|}| \qquad (12)$$

We use the average of M_i at all SIFT matches to denote the disparity measure in the frame t. With respect to the spatial coherence, we use the mean value of the whole video. With regards to the temporal coherence, we adopt the standard deviation of all frames. Smaller value indicates the higher disparity quality.

In Tables V and VI, we compare the disparity in terms of the spatial coherence and the temporal coherence between our work and the subspace stereoscopic video stabilization [13]. Notice that, as the resulted values are relatively small, we have amplified them by 10^3 for a clearer illustration. To demonstrate the effectiveness of JDSW, we also conduct the evaluation without JDSW. These comparisons show that our method (with JDSW)is better than [13] in maintaining the spatial and temporal disparity coherence.

TABLE VII
USER STUDY

index	1	2	3	4	5	6
[13]	36%	36%	**64%**	**44%**	32%	36%
Ours	**38%**	**60%**	36%	40%	**44%**	**44%**
Comparable	36%	4%	8%	16%	24%	20%

C. User Study

To give a subjective evaluation on the visual quality of different methods, we conduct a user study based on our results and [13]. Specifically, we invited 25 viewers to vote video (1)-(6) shown in Fig. 8 with respect to the following two aspects: (1) stability, (2) content preserved. The evaluation result is shown in Table VII. The number shows the percentage that the viewers select the videos. The 'Comparable' includes the viewers who consider both of the videos are equally good. In general, our method is comparable with [13]. [13] obtains more stable results in video (3) and (4). Because, our 2D-based stabilization method force the smoothed camera motions not to be far away from its original camera position to avoid overcropping.

VI. CONCLUSION

We have presented in this paper a method for the stereoscopic video stabilization, which combines the merits of the MeshFlow video stabilization [16] and the disparity preserving warp (DPW) [5]. By dividing each video frame into cells and applying MeshFlow, we can handle scenes with large parallax effectively. To address the problem of the disparity coherence, we proposed a novel warping method (i.e., JDSW), which jointly takes disparities and stabilities into considerations during the mesh warp. We further proposed an objective metric to evaluate the quality of the disparities of the stabilized videos. The effectiveness of our method has been demonstrated on various stereoscopic videos.

REFERENCES

[1] S. Luo, I. Shen, B. Chen, W. Cheng, and Y. Chuang. Perspective-aware warping for seamless stereoscopic image cloning. *ACM Trans. Graphics*, 31(6):182:1–182:8, 2012.

[2] W. Lo, J. Baar, C. Knaus, M. Zwicker, and M. Gross. Stereoscopic 3d copy & paste. 29(6):147, 2010.

[3] R. Tong, Y. Zhang, and K. Cheng. Stereopasting: interactive composition in stereoscopic images. *IEEE Trans. on Vis. and Comput. Graph.*, 19(8):1375–1385, 2013.

[4] S. Du, S. Hu, and R. Martin. Changing perspective in stereoscopic images. *IEEE Trans. on Vis. and Comput. Graph.*, 19(8):1288–1297, 2013.

[5] Y. Niu, W. Feng, and F. Liu. Enabling warping on stereoscopic images. *ACM Trans. Graphics*, 31(6):183, 2012.

[6] L. Wang, H. Jin, R. Yang, and M. Gong. Stereoscopic inpainting: Joint color and depth completion from stereo images. In *Proc. CVPR*, pages 1–8, 2008.

[7] T. Mu, J. Wang, S. Du, and S. Hu. Stereoscopic image completion and depth recovery. *The Vis. Comput.*, 30(6-8):833–843, 2014.

[8] S. Peleg, M. Ben-Ezra, and Y. Pritch. Omnistereo: Panoramic stereo imaging. *IEEE Trans. Pattern Anal. Mach. Intell.*, 23(3):279–290, 2001.

[9] F. Zhang and F. Liu. Casual stereoscopic panorama stitching. In *Proc. CVPR*, 2015.

[10] T. Basha, Y. Moses, and S. Avidan. Geometrically consistent stereo seam carving. In *Proc. ICCV*, pages 1816–1823, 2011.

[11] K. Lee, C. Chung, and Y. Chuang. Scene warping: Layer-based stereoscopic image resizing. In *Proc. CVPR*, pages 49–56, 2012.

[12] M. Lambooij, M. Fortuin, I. Heynderickx, and W. IJsselsteijn. Visual discomfort and visual fatigue of stereoscopic displays: A review. *Journal of Imaging Science and Technology*, 53(3):30201–1, 2009.

[13] F. Liu, Y. Niu, and H. Jin. Joint subspace stabilization for stereoscopic video. In *Proc. ICCV*, pages 73–80, 2013.

[14] M. Grundmann, V. Kwatra, and I. Essa. Auto-directed video stabilization with robust l1 optimal camera paths. In *Proc. CVPR*, pages 225–232, 2011.

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

[16] S. Liu, P. Tan, L. Yuan, J. Sun, and B. Zeng. Meshflow: Minimum latency online video stabilization. In *Proc. ECCV*, pages 800–815, 2016.

[17] S. Liu, L. Yuan, P. Tan, and J. Sun. Bundled camera paths for video stabilization. *ACM Trans. Graphics*, 32(4):78, 2013.

[18] P. Holland and R. Welsch. Robust regression using iteratively reweighted least-squares. *Communications in Statistics-theory and Methods*, 6(9):813–827, 1977.

[19] M. Fischler and R. Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395, 1981.

[20] S. Liu, L. Yuan, P. Tan, and J. Sun. Steadyflow: Spatially smooth optical flow for video stabilization. In *Proc. CVPR*, pages 4209–4216, 2014.

[21] A. Goldstein and R. Fattal. Video stabilization using epipolar geometry. *ACM Trans. Graphics*, 31(5), 2012.

[22] F. Liu, M. Gleicher, J. Wang, H. Jin, and A. Agarwala. Subspace video stabilization. *ACM Trans. Graphics*, 30(1):4, 2011.

[23] S. Liu, B. Xu, C. Deng, S. Zhu, B. Zeng, and M. Gabbouj. A hybrid approach for near-range video stabilization. *IEEE Trans. on Circ. and Syst. for Video Tech.*, 27(9):1922–1933, 2017.

[24] F. Liu, M. Gleicher, H. Jin, and A. Agarwala. Content-preserving warps for 3d video stabilization. *ACM Trans. Graphics*, 28:44, 2009.

[25] Z. Zhou, H. Jin, and Y. Ma. Plane-based content-preserving warps for video stabilization. In *Proc. CVPR*, pages 2299–2306, 2013.

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

[27] B. Smith, L. Zhang, H. Jin, and A. Agarwala. Light field video stabilization. In *Proc. ICCV*, pages 341–348, 2009.

[28] K. Lin, S. Liu, L.-F. Cheong, and B. Zeng. Seamless video stitching from hand-held camera inputs. In *Computer Graphics Forum*, volume 35, pages 479–487, 2016.

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

[30] Carlos M. and Rama C. Evaluation of image stabilization algorithms. In *Proc. ICASSP*, volume 5, pages 2789–2792, 1998.

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

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

[33] M. Lang, A. Hornung, O. Wang, S. Poulakos, A. Smolic, and M. Gross. Nonlinear disparity mapping for stereoscopic 3d. *ACM Trans. Graphics*, 29(4):75, 2010.

[34] P. Didyk, T. Ritschel, E. Eisemann, K. Myszkowski, and H. Seidel. A perceptual model for disparity. 30(4):96, 2011.

[35] Chu C.-H. Video stabilization for stereoscopic 3d on 3d mobile devices. *2014 IEEE International Conference on Multimedia and Expo (ICME)*, pages 1–6, 2014.

[36] T. Kroeger, R. Timofte, D. Dai, and L. Van. Fast optical flow using dense inverse search. In *Proc. ECCV*, pages 471–488, 2016.

[37] D. Scharstein and R. Szeliski. A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. *International Journal Computer Vision*, 47(1-3):7–42, 2002.

[38] H. Hirschmüller and D. Scharstein. Evaluation of stereo matching costs on images with radiometric differences. *IEEE Trans. Pattern Anal. Mach. Intell.*, 31(9):1582–1599, 2009.

[39] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal Computer Vision*, 60(2):91–110, 2004.

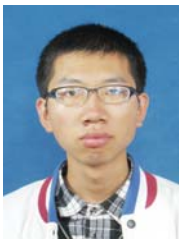
[40] M. Trajković and M. Hedley. Fast corner detection. *Image and vision computing*, 16(2):75–87, 1998.

[41] J. Shi and C. Tomasi. Good features to track. In *Proc. CVPR*, pages 593–600, 1994.

[42] H. Guo, S. Liu, T. He, S. Zhu, B. Zeng, and M. Gabbouj. Joint video stitching and stabilization from moving cameras. *IEEE Trans. on Image Processing*, 25(11):5491–5503, 2016.

[43] T. Igarashi, T. Moscovich, and J. Hughes. As-rigid-as-possible shape manipulation. *ACM Trans. Graphics*, 24(3):1134–1141, 2005.

[44] H. Guo, S. Liu, S. Zhu, and B. Zeng. Joint bundled camera paths for stereoscopic video stabilization. In *Proc. ICIP*, pages 1071–1075, 2016.



Heng Guo received his M.S. degree in School of Information and Communication Engineering and BEng degree in Electronic Information Engineering, both from University of Electronic Science and Technology of China (UESTC) in 2018 and 2015, respectively. He is now pursuing his Ph.D degree in Osaka University, Osaka, Japan. His research interests include computer vision and computer graphics. He is a student member of IEEE.

Shuaicheng Liu received his Ph.D. and M.S. degrees from National University of Singapore (NUS), Singapore, in 2014 and 2010, respectively, and his B.E. from Sichuan University, Chengdu, China, in 2008. In 2014, he joined University of Electronic Science and Technology of China (UESTC) and is currently an Associate Professor with the Institute of Image Processing, School of Information and Communication Engineering. His research interests include computer vision and computer graphics. He is a member of IEEE.

Shuyuan Zhu (S'08-A'09-M'13) received the Ph.D. degree from the Hong Kong University of Science and Technology, Hong Kong, in 2010. From 2010 to 2012, he worked at HKUST and Hong Kong Applied Science and Technology Research Institute Company Limited, respectively. In 2013, he joined University of Electronic Science and Technology of China and is currently a Professor with School of Information and Communication Engineering. Dr. Zhus research interests include image/video compression, image processing and compressive sensing. He has over 60 research publications and received the Top 10% paper award at IEEE ICIP-2014 and the Top 10% paper award at VCIP-2016. Dr. Zhu was the special session chair of image super-resolution at IEEE DSP-2015. He served as the committee member for IEEE ICME-2014, VCIP-2016, and PCM-2017. He is a member of IEEE CAS society.

Heng Tao Shen is a Professor of National Thousand Talents Plan and the director of Center for Future Media at the University of Electronic Science and Technology of China (UESTC). He obtained his BSc with 1st class Honours and PhD from Department of Computer Science, National University of Singapore (NUS) in 2000 and 2004 respectively. He then joined the University of Queensland (UQ) as a Lecturer, Senior Lecturer, Reader, and became a Professor in late 2011. His research interests mainly include Multimedia Search, Computer Vision, and Big Data Management on spatial, temporal, and multimedia databases. He has published over 150 peer reviewed papers, most of which are in prestigious international venues of interests. For his outstanding research contributions, he received the Chris Wallace Award in 2010 conferred by Computing Research and Education Association, Australasia, and the Future Fellowship from Australia Research Council in 2012. He is an Associate Editor of IEEE Transactions on Knowledge and Data Engineering, and has organized ICDE 2013 as Local Organization Co-Chair, and ACM Multimedia 2015 as Program Committee Co-Chair.

Bing Zeng (M91-SM13-F16) received his BEng and MEng degrees in electronic engineering from University of Electronic Science and Technology of China (UESTC), Chengdu, China, in 1983 and 1986, respectively, and his PhD degree in electrical engineering from Tampere University of Technology, Tampere, Finland, in 1991.

He worked as a postdoctoral fellow at University of Toronto from September 1991 to July 1992 and as a Researcher at Concordia University from August 1992 to January 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 the summer of 2013, through Chinas 1000-Talent-Scheme. At UESTC, he leads the Institute of Image Processing to work on image and video processing, 3D and multi-view video technology, and visual big data.

During his tenure at HKUST and UESTC, he graduated more than 30 Master and PhD students, received about 20 research grants, filed 8 international patents, and published more than 250 papers. Three representing works are as follows: one paper on fast block motion estimation, published in IEEE Transactions on Circuits and Systems for Video Technology (TCSVT) in 1994, has so far been SCI-cited more than 1000 times (Google-cited more than 2100 times) and currently stands at the 7th position among all papers published in this Transactions; one paper on smart padding for arbitrarily-shaped image blocks, published in IEEE TCSVT in 2001, leads to a patent that has been successfully licensed to companies; and one paper on directional discrete cosine transform (DDCT), published in IEEE TCSVT in 2008, receives the 2011 IEEE CSVT Transactions Best Paper Award. He also received the best paper award at China-Com three times (2009 Xi'an, 2010 Beijing, and 2012 Kunming).

He served as an Associate Editor for IEEE TCSVT for 8 years and received the Best Associate Editor Award in 2011. He was General Co-Chair of IEEE VCIP-2016, held in Chengdu, China, in November 2016. He is currently on the Editorial Board of Journal of Visual Communication and Image Representation and serves as General Co-Chair of PCM-2017. He received a 2nd Class Natural Science Award (the first recipient) from Chinese Ministry of Education in 2014 and was elected as an IEEE Fellow in 2016 for contributions to image and video coding.