Video Stitching with Extended-MeshFlow

Kai Chen, Jian Yao†, Binbin Xiang, Jingmin Tu
School of Remote Sensing and Information Engineering, Wuhan University, Wuhan, Hubei, China
†E-Mail: jian.yao@whu.edu.cn Web: http://cvrs.whu.edu.cn/

***Abstract*—In this paper, we present a method that stitches multiple videos captured with a fixed camera rig. We propose an Extended-MeshFlow motion model for video stitching. Firstly, uniform features are detected and matched at the overlapping region, from which the Exended-MeshFlow model is estimated. The model then warps the adjacent views to the common central view to eliminate the spatial misalignment. The motions located on the feature positions are interpolated to the mesh vertexes by Multilevel B-Spline Approximation(MBA). Collecting the motions on the vertexes form the vertex profiles, which are smoothed for temporal consistency. During the smoothing, only previous frames are required, thus the proposed method can stitch videos in an online mode. Experimental results on various of videos demonstrate that the proposed method can produce comparable stitching results in aspects of spatial alignment and temporal coherence.**

I. Introduction

Stitching multiple videos captured simultaneously from different views is a very practical problem, which is closely related to many applications, such as: panoramic hyperlapse photography [1], immersive virtual reality [2] and interactive mixed reality [3]. Two conditions are supposed to be satisfied when multiple videos are stitched together. On the one hand, considering the corresponding frames of adjacent videos, the spatial alignment should be ensured. On the other hand, for different temporal frames intra a video, the temporal consistency should be satisfied. Otherwise, severe jittering artifacts would be induced to stitching result. It is not easy to balance these two aspects, which makes video stitching challenging.

In the aspect of spatial alignment, video stitching is closely related to the problem of image stitching. Before giving an introduction of algorithms on video stitching, we briefly review some typical motion models that are proposed in the problem of image stitching and have been utilized in most existing video stitching methods. Early homography model [4] stitched image pairs under the assumption that the captured scenes were planar or the cameras were concentric, which was difficult to meet in practice. Therefore, seamline-based model [5]–[7] were proposed to reduce misalignment artifacts. This model stitched images by mosaicing different image components at the place in which minimal misalignments exist. It could be achieved by different seamline detection technologies, such as approaches based on Graph-Cuts [5] or dynamic programing [7]. Seamline-based model would fail to find any satisfactory seamline when images contained complex structures. For this reason, flow-based model [8] was proposed to cooperate with seamline-based model. But it usually required an estimation of dense optical flow, which was relatively time consuming. Recently, mesh-based model [9]–[11] was proposed to stitch images with large parallaxes. It divided images into many mesh grids and each grid shared a local homography model, which had been proved to be effective in elimination of parallax artifacts.

Existing video stitching systems could be divided into three categories according to the types of camera platforms. In the first kind of systems [12], [13], cameras are mounted on fixed stationary rigs (e.g., surveillance system). Under this condition, the position relationships between cameras and the background scenes of input videos both are consistent and can be pre-calibrated and pre-modeled before video stitching. Botao He and Shaohua Yu thus [13] designed a two-stage video stitching procedure for surveillance videos. It firstly aligned background scenes with a location-dependent warping method, followed by a change-detection based seamline detection approach to avert ghosting and misalignment artifacts caused by moving foregrounds.

As for the second kind of systems [14]–[17], the camera rig is mounted on some steady moving platforms (e.g., unmanned aerial vehicles or Google street view car). Namely, the relative positions among these cameras are still fixed while the background scenes are dynamic changing. Perazzi *et al.* [15] proposed a warping-based method to compensate the parallaxes between different views using weighted optical flow extrapolation, combined with constrained relaxation to make the stitched results close to the original input videos to ensure the temporal consistency. Rich360 [16] utilized a deformable spherical projection surface to minimize the spatial parallax from different cameras and performed a non-uniform spherical ray sampling to preserve as-rich-as-possible details for import image regions. Yuan *et al.* [17] designed an iterative warping method for camera array with different resolutions, which combined the advantages of mesh grid model and flow-based model and was robust to huge resolution gap.

In the third kind of systems [18]–[25], the input videos are captured with freely moving cameras. Video stitching gets more complex in this situation as the cameras move independently and the parallax between videos may be caused by large position and pose differences of the cameras or by complex depth variations. Wei Jiang and Jinwei Gu [20] proposed a spatial-temporal warping-based method to locally align frames from different videos while maintaining the temporal consistency, followed by a spatial-temporal seamline detection method based on 3D Graph-Cuts to eliminate large parallax induced by moving foregrounds. A more typical approach is to stitch videos in an unified video stabilization

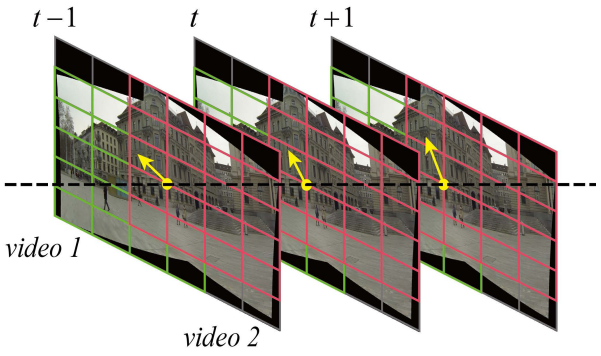


Fig. 1. Illustration of proposed Extended-MeshFlow. The yellow arrow on each mesh vertex indicates the estimated motion vector defined on this vertex. All vectors defined on the same vertex over time consisting of a vertex profile (as the black dotted line indicates).

framework [21], [22], [25]. Guo *et al.* [21] eliminated spatial parallax by 2D camera path optimization, combined with a spatial-temporal mesh-based motion model to ensure the temporal coherency. Lin *et al.* [22] applied a similar motion model to stitch videos. The difference was that the smooth camera paths were generated from a sparse 3D reconstruction result.

In this paper, we concentrate on the first two situations, where the cameras are mounted on a fixed rig. Compared with the case of freely moving cameras, the problem of video stitching with camera rig gets less complex, but is more widely applied in practice. As the camera rig is generally imperfect, video stitching under this condition is still a problem that has not been completely solved. We therefore propose an Extended-MeshFlow motion model for video stitching. In the following paper, we firstly present the estimation of the proposed Extended-MeshFlow model in Section II. An adaptive smoothing method for Extended-MeshFlow will be described in Section III, which is specially designed for the preservation of temporal consistency. Various experimental results presented in Section IV demonstrate the superiority of our proposed method. Similar to many existing methods [15], [16], we assume that the input videos have been pre-projected onto some common surface based on the pre-calibrated parameters of the camera rig.

II. Extended-MeshFlow

MeshFlow was firstly proposed by Liu *et al.* [26] for the application of video stabilization. We extend this model to Extended-MeshFlow for video stitching. As shown in Fig. 1, for one thing, these two models are similar because they both only estimate motions at mesh vertexes. For another, they are different as one estimates motions between temporal frames intra a video for the purpose of video stabilization, while the other estimates motions between corresponding frames of adjacent videos for the purpose of misalignment elimination. This difference leads to a different model estimation method. Without loss of generality, we take the case of stitching two videos as an example to describe the proposed model estimation method.

A. Model Estimation

At first, an uniform $M \times N$ mesh grid is placed onto each video frame. The SURF features [27] are detected and matched in each overlapping region. For a pair of video frame at time t, suppose that $\{p^{(1)}, p^{(2)}\}$ is a pair of matched points. $p^{(1)}$ and $p^{(2)}$ denote 2D image coordinates of feature points on view1 and view2 respectively. The displacement between $\{p^{(1)}, p^{(2)}\}$ can be computed as $d\left(p^{(1)}, p^{(2)}\right) = \|p^{(1)} - p^{(2)}\|$, which is to be eliminated in video stitching. In order to achieve this objective and reduce warping distortion at the same time, we propose to simultaneously warp $p^{(1)}$ and $p^{(2)}$ to their midpoint $\tilde{p}$ on the mid-view. That is, for each pair of matched points, we define their motion vectors $v_{p^{(1)}}$ and $v_{p^{(2)}}$ as follows:

$$\begin{cases} v_{p^{(1)}} = \tilde{p} - p^{(1)} = \dfrac{p^{(2)} - p^{(1)}}{2} \\ v_{p^{(2)}} = \tilde{p} - p^{(2)} = \dfrac{p^{(1)} - p^{(2)}}{2}. \end{cases} \tag{1}$$

Multiple pairs of matched points form multiple pairs of motion vectors, which are defined on feature positions and are distributed in the overlapping region unevenly. In order to estimate motion vectors defined on regular mesh vertexes from these irregular motion vectors, we formulate it into the scatter data interpolation problem and apply the Multilevel B-Splines Approximation (MBA) [28] to solve it. MBA is an effective s-catter data interpolation algorithm, which has been widely used in image warping, data interpolation and geometric modeling. It circumvents the tradeoff between smoothness and accuracy by a multilevel fashion and achieves a smooth and accurate estimation result. We thus can get a spatial smooth estimation of motions even without additional smoothing filtering.

B. Robust Estimation

Applying the MBA algorithm to estimate the Extended-MeshFlow model for each video frame is sensitive to noises and outliers. For this reason, we further design two steps to obtain a more robust estimation result.

At first, we adopt an outlier rejection method under the assumption of local motion coherency [29], [30], which thinks that matched points in a small neighborhood tend to have consistent motions. We therefore represent each matched pair into a local motion descriptor and identify outliers in the local motion space. More details about the proposed outlier rejection method are referred to [30].

Secondly, we propagate the matched correspondences from time $t-1$ to t as feature tracking methods are more robust than the one of feature matching. Specifically, we track the matched features of view1 and view2 from frame $t-1$ to frame t respectively using KLT tracking method [31]. During the tracking, the matching relationship of each pair of features is preserved and propagated to current frame. Once the number of propagated matches is less than a pre-defined threshold, we discard them and re-detect and match features on current frames.

III. Video Stitching with Extended-MeshFlow

Each input video keeps an Extended-MeshFlow, which uses a set of motion vectors defined on regular mesh vertexes to represent a local motion model and aims to warp adjacent video frames to the common central view to eliminate the residual misalignment after pre-projection. In order to balance the spatial alignment and temporal consistency at the same time, vertex profiles are firstly extracted from the estimated Extended-MeshFlow model. Then, similar to [26], [32], the Extended-MeshFlow of each video is temporally optimized by adaptively smoothing all vertex profiles that this video holds. During the optimization, only previous frames are required. Thus, as the smoothing processes frame-by-frame, the input video frames can be stitched together according to the smoothed models after each optimization, which actually is an online mode.

A. Vertex Profiles Generation

A vertex profile represents local motions of its neighboring image regions. As shown in Fig. 1, it is a collection of all motion vectors defined on the same mesh vertex over the whole video sequences. Under this definition, a vertex keeps a vertex profile and each video holds a set of vertex profiles, which are the bases of our temporally video stitching algorithm. In our proposed method, only the latest τ frames will be utilized to optimize the motions on current video frames. That is, the max length of a vertex profile is τ. As we design our video stitching method in an online mode, the future frames will not be used to optimize current frames, namely, at time t, only the frames after $t-\tau$ and before t will be utilized.

B. Adaptive Smoothing of Vertex Profiles

Temporally motion smoothing is an essential step for many video-based applications. Kim *et al.*. [33] applied a half Gaussian kernel to smooth the motion trajectory. Liu *et al.* [26] applied regression analysis to establish the relationships between various complex motions and adaptive smoothing weights. In our proposed model, motions indicate compensations of parallaxes, which should not be smoothed indiscriminately. For example, if some moving objects suddenly appear into the video, which would lead to a abrupt change of parallax. Under this circumstances, an over-smoothing motion model would lead to misalignment artifacts at the position of moving objects. We thus propose our adaptive smoothing strategy.

For the frame at time t, the vertex profiles it holds can be denoted as $\mathbf{V}(t) = \{V_i(t)|i = 1, 2, ..., m\}$. m is the number of vertexes in current frame and $V_i(t)$ denotes the profile the i-th vertex keeps. $V_i(t)$ can be further denoted as $V_i(t) = \{v_i^{\phi-1}(t_0), v_i^{\phi-1}(t_0+1), ..., v_i^{\phi-1}(t-1), v_i(t)\}$, where t_0 is the starting time of current vertex profile and the superscript $\phi-1$ indicates the latest optimization result, namely, for each vertex, its motion profile $V_i(t)$ consisting of the optimized results of previous frames in last optimization $\{v_i^{\phi-1}(t_0), v_i^{\phi-1}(t_0+1), ..., v_i^{\phi-1}(t-1)\}$ and current unoptimized result $\{v_i(t)\}$. The objective of temporally smoothing at time t is to obtain the optimized vertex profiles $\widetilde{\mathbf{V}}(t) = \{\widetilde{V}_i(t)|i = 1, 2, ..., m\}$, where $\widetilde{V}_i(t) = \{v_i^{\phi}(t_0), v_i^{\phi}(t_0+1), ..., v_i^{\phi}(t-1), v_i^{\phi}(t)\}$. The optimized vertex profiles should ensure frames alignment and temporal smoothness at the same time, which can be achieved by leveraging the temporally smoothness and the similarity of the original motion:

$$\begin{aligned}\mathcal{O}(\widetilde{\mathbf{V}}(t)) = &\sum_{i=1}^{m}\sum_{r=t_0}^{t-1}\|v_i^{\phi}(r) - v_i^{\phi-1}(r)\|^2 \\ &+ \sum_{i=1}^{m}\|v_i^{\phi}(t) - v_i(t)\|^2 + \sum_{i=1}^{m}\sum_{r=t_0}^{t-1}\|v_i^{\phi}(t) - v_i^{\phi}(r)\|^2.\end{aligned} \tag{2}$$

In (2), the first term is used to constrain the similarity between results in previous optimization and the one of curren optimization. The second term is used to constrain the similarity of the original motion at time t and the third term is the smooth term, which imposes a temporally smoothing constraints on each vertex profile. As the Extended-MeshFlow itself enforces strong spatial coherence by MBA interpolation, we do not append any additional spatial constraints, namely, each vertex profile can be smoothed independently. Therefore, for each vertex, the energy function can be simplified as:

$$\begin{aligned}\mathcal{O}(\widetilde{V}_i(t)) = &\sum_{r=t0}^{t-1}\|v_i^{\phi}(r) - v_i^{\phi-1}(r)\|^2 \\ &+ \|v_i^{\phi}(t) - v_i(t)\|^2 + \sum_{r=t_0}^{t-1}\|v_i^{\phi}(t) - v_i^{\phi}(r)\|^2.\end{aligned} \tag{3}$$

For an adaptive smoothing, we permit the abrupt changes that may be resulted from moving objects and ensure the stitching result without obvious jittering artifacts at the same time. We thus add an adaptive weight term to get our final energy function:

$$\begin{aligned}\mathcal{O}(\widetilde{V}(t)) = &\sum_{r=t0}^{t-1}\|v^{\phi}(r) - v^{\phi-1}(r)\|^2 \\ &+ \|v^{\phi}(t) - v(t)\|^2 + \sum_{r=t_0}^{t-1}\omega_{t,r}\|v^{\phi}(t) - v^{\phi}(r)\|^2.\end{aligned} \tag{4}$$

In (4), the lower index i has been removed for clarity. The adaptive weight is obtained by multiplying two parts: $\omega_{t,r}^{(t)}$ and $\omega_{t,r}^{(v)}$, which are calculated as:

$$\omega_{t,r}^{(t)} = exp(-\|r-t\|^2/\sigma^2) \tag{5}$$

$$\omega_{t,r}^{(v)} = 1 - \frac{\|v(t) - v^{\phi-1}(r)\|}{\|v(t)\| + \|v^{\phi-1}(r)\| + \varepsilon}, \tag{6}$$

where σ is used to normalize the half Gaussian kernel and ε is a small value to prevent the numerator from being divided by zero ($\varepsilon = 0.001$ in our implementation). In our formulation, $\omega_{t,r}^{(t)}$ is used to measure time differences and $\omega_{t,r}^{(v)}$ is used to measure motion deferences. Combining them together can avoid the over-smoothing of temporal motion changes. In our implementation, the value of τ is set to 30 and σ is set to $\frac{1}{3}\tau$. The final energy function defined by (4) is quadratic and can be solved by any sparse linear system efficiently.

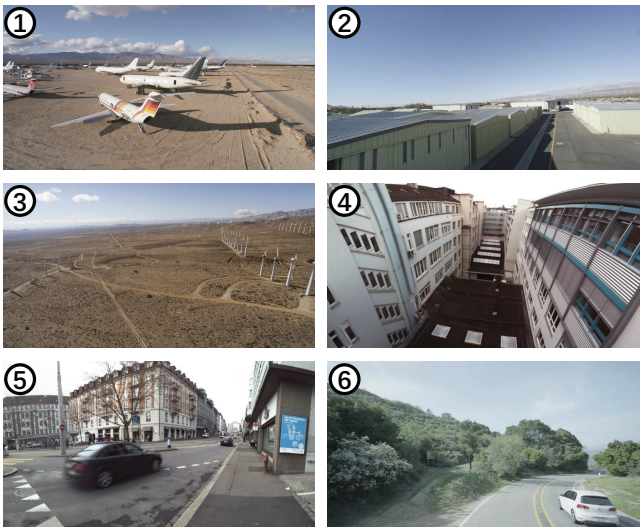


Fig. 2. The dataset used in our experiment. Group **01-03** and Group **06** consist of five videos and group **04-05** have fourteen views.

C. Final Panoramic Video Generation

After each optimization, the motions on current frames are temporally smoothed. The motions actually are a set of vectors defined on regular mesh vertexes. We thus can warp current frames according to the mesh warp [34], followed by the standard multi-band blending algorithm [35] to get a seamless panoramic video frame. Traverse all video frames can obtain the entire stitching video.

IV. Experimental Results

We demonstrate the effectiveness of the proposed video stitching method on the dataset proposed by Perazzi *et al.* [15]. As shown in Fig. 2, it consists of six groups of videos with a lot of challenging scenes, such as: complex backgrounds and moving cars. All the videos are captured from $5-14$ cameras that are mounted on a fixed rig and videos have been pre-projected onto a sphere surface based on the pre-calibrated camera parameters. We design our experiments in three aspects: Firstly, we make a comparison between results before and after our proposed method to validate the proposed algorithm. Secondly, we compare our stitching results with three popular commercial softwares: VideoStitch Studio [1], Autopano 2.5 and Antupano 3.0 [2]. Thirdly, we compare our proposed method with the one of [15].

TABLE I
A Comparison Between Stitching Results With and Without Our Proposed Method.

No.	Alignment Error		Consistency Score	
	without	with	without	with
01	11.21	**9.90**	21.09	**25.25**
02	10.24	**9.06**	20.12	**23.09**
03	10.32	**9.49**	27.80	**28.53**
04	14.30	**10.41**	28.20	**28.96**
05	14.39	**10.69**	28.67	**28.96**
06	14.65	**12.34**	**22.60**	22.27

[1] http://www.video-stitch.com/studio
[2] http://www.kolor.com/autopano/

Fig. 3. Comparative results of videos in the aspect of spatial alignment quality. **Top**: Video frames after pre-projection. **Bottom**: Stitching results with the proposed algorithm.

All experiments are conducted with an un-optimized C++ implementation of our method on a PC with Intel Core i7-4770 3.4GHz CPU and 16GB memory.

A. Validation on Proposed Method

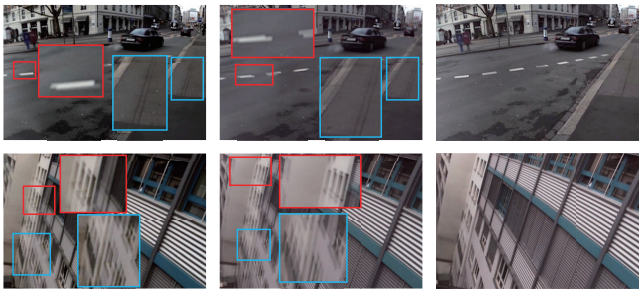
For each group of videos, we use the same metric in [10] to assess the spatial alignment quality of video stitching results. We compute the RMSE of one minus normalized cross correlation (NCC) over a neighborhood π of 5×5 window for pixels in the overlapping region according to (7):

$$RMSE\left(I_i, I_j\right) = \sqrt{\frac{1}{N}\sum_{\pi}\left(1.0 - NCC\left(\mathbf{x}_i, \mathbf{x}_j\right)\right)}, \qquad (7)$$

where N is the number of pixels in current overlapping region π and $\mathbf{x}_i$ and $\mathbf{x}_j$ are the pixels in video frame I_i, I_j respectively. We compute the RMSE for every frame and average them to obtain the final spatial alignment metric.

In order to measure temporal consistency of the stitching result, we design a similar evaluation metric that is proposed in [34]. We firstly divide the stitching result into many small video batches and each batch contains up to 30 frames. We then adopt the frequency-based method proposed in [34] to measure the temporal consistency of feature trajectories within every video batch. Taking the minimum value among all trajectories in a batch yields the consistency score of this batch and we select the average value of the smallest five scores among all video batches as the final temporal consistency score of the stitching result.

We compute the metrics of spatial alignment and temporal consistency for all six groups of videos. In the measurement of spatial alignment quality, we compare the metric values between videos after pre-projection and after the proposed stitching procedure. At the same time, we compare the temporal consistency of videos that are stitched without and with the proposed temporal constraint. All results have been listed in Table. I and Fig. 3. We can see that with the proposed stitching method, both spatial and temporal quality have been improved, which demonstrates the effectiveness of our proposed method.



(a) (b) (c)

Fig. 4. Comparisons with commercial softwares VideoStitch Studio and Autopano 2.5. (a) Results from VideoStitch Studio. (b) Results from Autopano 2.5. (c) Results from our method.

t-1 t t+1

Fig. 5. Comparisons with Autopano 3.0. First row: Results from D.WARP under the "Prioritize Space" mode. Second row: Results from D.WARP under the "Prioritize Time" mode. Third row: Results from our method.

B. Comparison with Commercial Softwares

We compare the proposed algorithm with three popular commercial softwares: VideoStitch Studio, Autopano 2.5 and Autopano 3.0. The first two softwares compute a stitching model according to the relative positions and poses among cameras and stitch all video frames based on the same stitching model, which is not robust enough when the scene contains moving objects or obvious depth variations.

Autopano 3.0 is the improved version of Autopano 2.5. A parallax elimination tool named D.WARP is appended to Autopano 3.0, which has two typical operation modes: "Prioritize Space" and "Prioritize Time". The first mode corrects as much parallax artifacts as possible with less considerations on temporal consistency, while the second one eliminates parallaxes with great respect to temporal coherency. Because of the watermarks of stitching results produced by commercial softwares, we fail to compare the results quantitatively. Therefore, we compare our stitching results with the one of these three softwares qualitatively and the intuitive comparative results are presented in Fig. 4 and Fig. 5.

Fig. 4 shows two comparative results with the captured scenes containing huge depth variations and moving objects. We can see that the stitching results produced by VideoStitch Studio and Autopano 2.5 are deteriorated by severe ghosting, while our results are free from obvious misalignment artifacts. Fig. 5 presents the stitching results of three consecutive frames produced by Autopano 3.0 under two operation modes and produced by our method. Autopano 3.0 fails to sufficiently reduce misalignments especially when it is required to consider the temporal consistency. Our proposed method balance the spatial alignment and temporal coherency at the same time and gets a better performance.

C. Comparison with Perazzi's method

The method proposed by Perazzi *et al.* [15] is one of state-of-the-art video stitching methods, which stitches videos without using future frames and is just similar to our proposed method. We conduct comparative experiments with it on the 1-th, 2-th, 3-th and 6-th groups of videos. In order to compare with their results, we slightly modify our programme to make it workable on the case of five unstructured cameras. Fig. 6 shows one of our experimental results. We can see that our method can obtain similar results even in some challenging scenes but is more efficient than Perazzi's method. During our experiment, it usually takes approximately 1 second to stitch a pair of adjacent video frames with up to $2K$ resolutions and takes about 6 seconds to output a stitched frame with 5 views, while Perazzi's method applies weighted optical flow extrapolation combined with constrained relaxation, which is achieved by solving a Poisson equation and thus usually requires tens of seconds to compute one frame with high resolution.

V. Conclusion

This paper propose an Extended-MeshFlow motion model for video stitching with videos captured by a fixed camera rig. Uniform features are firstly detected and matched at the overlapping region, from which the Exended-MeshFlow model is estimated using Multilevel B-Spline Approximation. The Extended-MeshFlow uses a set of motion vectors defined on regular mesh vertexes to represent a local motion model and aims to warp adjacent video frames to the common central view to eliminate the residual misalignment after video pre-projection. Collecting the motions on the vertexes form the vertex profiles, which are adaptively smoothed for the purpose of temporal consistency. Lastly, mesh-based deformation and multi-band blending algorithms are utilized to obtain the final stitching result. Experimental results demonstrate that the proposed video stitching method with Extended-MeshFlow achieves a balance between spatial alignment and temporal consistency and has a better performance compared with three advanced commercial softwares and one state-of-the-art video stitching method.

Acknowledgment

This work was partially supported by the National Natural Science Foundation of China (Project No. 41571436), the National Key Research and Development Program of China (Project No. 2017YFB1302400), the Hubei Province Science and Technology Support Program, China (Project No. 2015BAA027), the National Natural Science Foundation of China under Grant 91438203, and LIESMARS Special Research Funding.

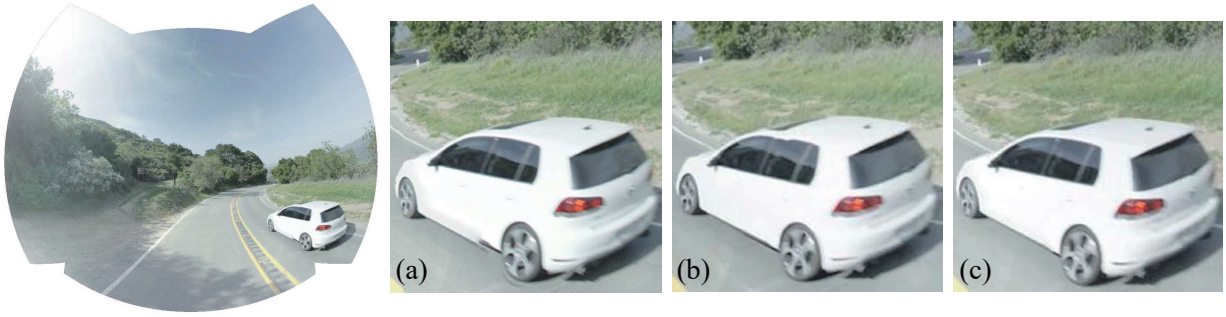


Fig. 6. Comparison with Perazzi's method [15]. (a) Results from pre-projection + multiband. (b) Results from Perazzi's method. (c) Results from our method.

REFERENCES

[1] T. Halperin, Y. Poleg, C. Arora, and S. Peleg, "Egosampling: Wide view hyperlapse from egocentric videos," *IEEE Transactions on Circuits and Systems for Video Technology*, 2017. 1

[2] W.-T. Lee, H.-I. Chen, M.-S. Chen, I. Shen, B.-Y. Chen *et al.*, "High-resolution 360 video foveated stitching for real-time vr," in *Computer Graphics Forum*, vol. 36, no. 7. Wiley Online Library, 2017, pp. 115–123. 1

[3] T. Rhee, L. Petikam, B. Allen, and A. Chalmers, "Mr360: Mixed reality rendering for 360 panoramic videos," *IEEE Transactions on Visualization and Computer Graphics*, vol. 23, no. 4, pp. 1379–1388, 2017. 1

[4] M. Brown and D. G. Lowe, "Automatic panoramic image stitching using invariant features," *International journal of computer vision*, vol. 74, no. 1, pp. 59–73, 2007. 1

[5] L. Li, J. Yao, X. Lu, J. Tu, and J. Shan, "Optimal seamline detection for multiple image mosaicking via graph cuts," *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 113, pp. 1–16, 2016. 1

[6] A. Eden, M. Uyttendaele, and R. Szeliski, "Seamless image stitching of scenes with large motions and exposure differences," in *IEEE International Conference on Computer Vision and Pattern Recognition*, 2006. 1

[7] Y. Xiong and K. Pulli, "Fast panorama stitching for high-quality panoramic images on mobile phones," *IEEE Transactions on Consumer Electronics*, vol. 56, no. 2, 2010. 1

[8] J. Jia and C.-K. Tang, "Image stitching using structure deformation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 30, no. 4, pp. 617–631, 2008. 1

[9] J. Zaragoza, T.-J. Chin, M. S. Brown, and D. Suter, "As-projective-as-possible image stitching with moving dlt," in *IEEE International Conference on Computer Vision and Pattern Recognition*, 2013. 1

[10] S. Li, L. Yuan, J. Sun, and L. Quan, "Dual-feature warping-based motion model estimation," in *IEEE International Conference on Computer Vision*, 2015. 1, 4

[11] K. Lin, N. Jiang, S. Liu, L.-F. Cheong, and M. D. J. Lu, "Direct photometric alignment by mesh deformation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 2405–2413. 1

[12] B. He, G. Zhao, and Q. Liu, "Panoramic video stitching in multi-camera surveillance system," in *Image and Vision Computing New Zealand (IVCNZ), 2010 25th International Conference of*. IEEE, 2010, pp. 1–6. 1

[13] B. He and S. Yu, "Parallax-robust surveillance video stitching," *Sensors*, vol. 16, no. 1, p. 7, 2015. 1

[14] M. Zheng, X. Chen, and L. Guo, "Stitching video from webcams," *Advances in Visual Computing*, pp. 420–429, 2008. 1

[15] F. Perazzi, A. Sorkine-Hornung, H. Zimmer, P. Kaufmann, O. Wang, S. Watson, and M. Gross, "Panoramic video from unstructured camera arrays," in *Computer Graphics Forum*, vol. 34, no. 2. Wiley Online Library, 2015, pp. 57–68. 1, 2, 4, 5, 6

[16] J. Lee, B. Kim, K. Kim, Y. Kim, and J. Noh, "Rich360: optimized spherical representation from structured panoramic camera arrays," *ACM Transactions on Graphics (TOG)*, vol. 35, no. 4, p. 63, 2016. 1, 2

[17] X. Yuan, L. Fang, Q. Dai, D. J. Brady, and Y. Liu, "Multiscale gigapixel video: A cross resolution image matching and warping approach," in *IEEE International Conference on Computational Photography*, 2017. 1

[18] M. El-Saban, M. Izz, and A. Kaheel, "Fast stitching of videos captured from freely moving devices by exploiting temporal redundancy," in *Image Processing (ICIP), 2010 17th IEEE International Conference on*. IEEE, 2010, pp. 1193–1196. 1

[19] 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 *Image Processing (ICIP), 2011 18th IEEE International Conference on*. IEEE, 2011, pp. 1481–1484. 1

[20] W. Jiang and J. Gu, "Video stitching with spatial-temporal content-preserving warping," in *IEEE International Conference on Computer Vision and Pattern Recognition Workshops*, 2015. 1

[21] H. Guo, S. Liu, T. He, S. Zhu, B. Zeng, and M. Gabbouj, "Joint video stitching and stabilization from moving cameras," *IEEE Transactions on Image Processing*, vol. 25, no. 11, pp. 5491–5503, 2016. 1, 2

[22] K. Lin, S. Liu, L.-F. Cheong, and B. Zeng, "Seamless video stitching from hand-held camera inputs," in *Computer Graphics Forum*, vol. 35, no. 2. Wiley Online Library, 2016, pp. 479–487. 1, 2

[23] B.-S. Kim, K.-A. Choi, W.-J. Park, S.-W. Kim, and S.-J. Ko, "Content-preserving video stitching method for multi-camera systems," *IEEE Transactions on Consumer Electronics*, vol. 63, no. 2, pp. 109–116, 2017. 1

[24] Z. Cui, O. Wang, P. Tan, and J. Wang, "Time slice video synthesis by robust video alignment," *ACM Transactions on Graphics (TOG)*, vol. 36, no. 4, p. 131, 2017. 1

[25] Y. Nie, T. Su, Z. Zhang, H. Sun, and G. Li, "Dynamic video stitching via shakiness removing," *IEEE Transactions on Image Processing*, vol. 27, no. 1, pp. 164–178, 2018. 1, 2

[26] S. Liu, P. Tan, L. Yuan, J. Sun, and B. Zeng, "Meshflow: Minimum latency online video stabilization," in *European Conference on Computer Vision*, 2016. 2, 3

[27] H. Bay, T. Tuytelaars, and L. Van Gool, "Surf: Speeded up robust features," *European Conference on Computer Vision*, 2006. 2

[28] S. Lee, G. Wolberg, and S. Y. Shin, "Scattered data interpolation with multilevel b-splines," *IEEE transactions on visualization and computer graphics*, vol. 3, no. 3, pp. 228–244, 1997. 2

[29] J. Yao and W.-K. Cham, "Robust multi-view feature matching from multiple unordered views," *Pattern Recognition*, vol. 40, no. 11, pp. 3081–3099, 2007. 2

[30] L. Li, J. Yao, R. Xie, M. Xia, and W. Zhang, "A unified framework for street-view panorama stitching," *Sensors*, vol. 17, no. 1, p. 1, 2016. 2

[31] J. Shi *et al.*, "Good features to track," in *IEEE International Conference on Computer Vision and Pattern Recognition*, 1994. 2

[32] S. Liu, L. Yuan, P. Tan, and J. Sun, "Steadyflow: Spatially smooth optical flow for video stabilization," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2014, pp. 4209–4216. 3

[33] K. Kim, M. Grundmann, A. Shamir, I. Matthews, J. Hodgins, and I. Essa, "Motion fields to predict play evolution in dynamic sport scenes," in *IEEE International Conference on Computer Vision and Pattern Recognition*, 2010. 3

[34] S. Liu, L. Yuan, P. Tan, and J. Sun, "Bundled camera paths for video stabilization," *ACM Transactions on Graphics (TOG)*, vol. 32, no. 4, pp. 1–10, 2013. 4

[35] P. J. Burt and E. H. Adelson, "A multiresolution spline with application to image mosaics," *ACM Transactions on Graphics (TOG)*, vol. 2, no. 4, pp. 217–236, 1983. 4