MESHFLOW VIDEO DENOISING

Zhihang Ren, Jiajia Li, Shuaicheng Liu, Bing Zeng

School of Electronic Engineering
University of Electronic Science and Technology of China, Chengdu, China

ABSTRACT

We propose an efficient video denoising approach that produces clean videos by utilizing the recently proposed meshflow motion model for the camera motion compensation. The meshflow is a spatially-smooth sparse motion field with motion vectors located at the mesh vertexes. The model is very effective and efficient for the purpose of the multi-frames denoising due to its internal characteristics such as the lightweight, the nonparametric form, and the spatially-variant motion representation. Specifically, the meshflows are estimated between adjacent frames, which are used to align frames within a sliding time window. A denoised frame is generated by fusing of several registered frames in a spatial and temporal manner with outlier rejections. Various challenging examples demonstrate the effectiveness and practicability of the proposed approach.

Index Terms— Video denoising, meshflow.

1. INTRODUCTION

Videos captured in a low-light environment often suffer from strong noises that severely decrease the quality of videos. Image/video denoising approaches [1, 2, 3, 4, 5] aim at removing or suppressing noises for the video quality improvement. Some methods explore the sparsity of a multiresolution representation in the wavelet domain [6, 7] or utilize the non-local measurements for restorations [8, 9]. In general, multiple image denoising is superior to single image denoising as more observations are provided. Video frames often contain complementary information that can be fused for the strong denoising [10, 11]. In this work, we aim at the video denoising with videos captured by hand-held cameras. Please refer the project page for the videos and results[1].

Typically, there are two challenges with respect to a practical video denoising, i.e., the camera motion compensation and the handling of outliers during the temporal pixels fusing. The former refers to the image registration between adjacent video frames. As the camera is not stationary during the capturing, accurate motion estimations can increase the performance of various applications, such as the HDR [12], the video deblurring [13] and the video stabilization [14]. The latter can suppress the artifacts (e.g., ghosting) introduced by misalignments, which may be caused by various reasons, including dynamic objects, occlusions and insufficient descriptions of the motion model (e.g., a single homography).

Liu *et al.* proposed a multi-frame denoising approach that merges several images captured by a cell-phone burst model to a clean denoised image [15]. A parametric pyramid homography-flow motion model was proposed for the image sequence alignment. The denoised result was synthesized by a multiscale-based spatial-temporal fusing with the consistent pixel verification. More recently, Liu *et al.* proposed a meshflow motion model for the video stabilization [16]. The meshflow is a non-parametric motion model, which consists of the sparse motion vectors located on the vertexes of a grid mesh.

Inspired from these two works, we propose to use meshflow motion model for the video denoising. Compared with multiscale homography-flows, the estimation of the meshflow does not require the hierarchical structures and the level-by-level optimizations, which is more efficient and convenient, yielding comparable results if not superior. Based on the meshflow, we utilize the concepts of pixel-profiles [17, 18] instead of adopting real motion trajectories, for the efficient motion accumulation and the consistent pixels identification. By combining several novelties, we present a video denoising framework that not only runs efficiently but also produces high quality results when compared with the state-of-the-art methods, such as BM3D [5], VBM3D [19], BM4D [20], VBM4D [21] and Burst [15].

2. MESHFLOW DENOISING

Figure 1 shows our pipeline. For a given input noisy video, we estimate the meshflow (Sec. 2.1) between adjacent frames. To denoise a frame, we move a temporal window along the video with a radius of 5 frames. Frames within the window are warped towards the central frame according to the estimated motions by the meshflow (Sec. 2.2). With all the frames aligned to the central frame within the local window, we fuse them to denoise the central frame. The fusing includes the identification of outlier pixels. In Fig. 1(c), the green and the red dots denote inlier and outlier pixels. Details regarding the consistency check and the fusing process will be discussed in Sec. 2.3. The window is moved forward by one frame at each time and the video is denoised frame by frame.

[1]Project page: http://www.liushuaicheng.org/ICIP/2017/index.html

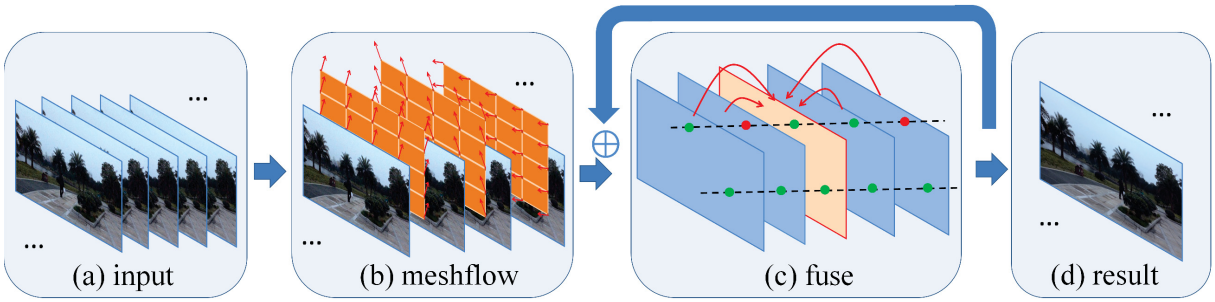

Fig. 1. Our system pipeline. (a) The input noisy video. (b) The meshflow is estimated between adjacent frames. (c) All the neighboring frames within a local temporal window are warped towards the central frame according to the meshflow. The denoised frame is generated by fusing of warped frames with consistent pixels identification. (d) The video is denoised frame by frame with a moving temporal window.

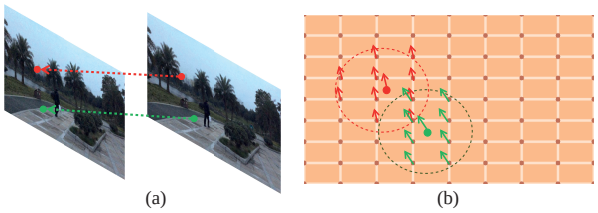

Fig. 2. Meshflow motion model. (a) We show two matched feature pairs. The arrow indicates the motion. (b) Motion of a feature point is propagated to the nearby vertexes.

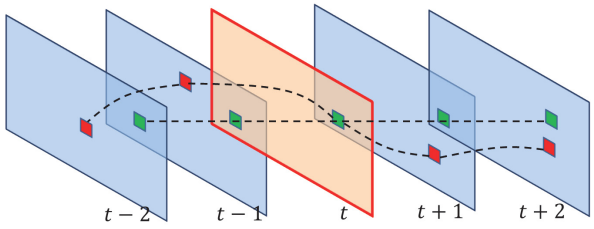

Fig. 3. Motion tracks (red) versus pixel profiles (green). We adopt pixel profiles for the motion accumulation.

2.1. Meshflow Motion Model

The meshflow is a sparse motion field. It is often used to estimate motions between neighboring frames, which is originally proposed in [16]. We introduce the meshflow in this section for the sake of completeness.

Rich Features. We detect FAST image features [22] and track them by KLT [23] to the adjacent frame. The meshflow estimation prefers an uniform and dense feature coverage. Therefore, the detection threshold is tuned locally and automatically for different image regions. More details regarding the rich feature pruning can be found in [24].

Motion Propagation. Each matched pair of features yields a motion vector as shown in Fig. 2(a). In our implementation, we calculate motions between a current frame and its previous frame. A regular mesh is placed onto the current frame and the motion at the feature point is duplicated to the nearby mesh vertexes as shown in Fig. 2(b). Notably, some of the vertexes may receive multiple motion vectors contributed from different feature points.

Median Filters. Each mesh vertex should only have one unique motion vector, which is picked from the motion candidates at each vertex by a median filter. Another median filter is applied spatially to reject motion outliers caused by mismatched features and dynamic objects. To this end, we obtain a smoothly varying sparse motion field. More discussions can be found in [16].

Notations. As we calculate the motions backward, the meshflow at frame t is denoted as F_t with motions pointing from frame t to frame $t-1$. The meshflow at the first frame is set to 0.

2.2. Motions Accumulation

Figure 3 shows an example of motion tracking and motion accumulation. To denoise frame t (Fig. 3 red frame border), we need to warp the neighboring frames (Fig. 3 blue frame border) towards the central frame t. For example, we want to warp frame $t-2$ to frame t. One solution is to estimate meshflow directly between them. But it is not efficient as it introduces extra feature detection and tracking. Our solution is to use previously estimated adjacent meshflow F_t to approximate the non-adjacent jumps.

There are two choices. Intuitively, we should trace the motion path to form a motion track (Fig. 3 red squares). However, there are some problems such as the tracks can move outside of a frame. Moreover, we need to check the values of tracks to decide the subsequent locations, which is inefficient. Our solution is to adopt the pixel-profile strategy [17]. A pixel profile collects motion vectors at a fixed spatial location along the time (Fig. 3 green squares). It is shown in [17] that the motions at the pixel-profile is a very good approximation of the corresponding motion track. Profiles can lead to several advantages such as full coverage (no border issues) and parallel computing (each pixel location is computed independently). Therefore, we adopt the profiles to accumulate

Without consistency With consistency

Fig. 4. With and without consistency check. Please notice the ghosting effects in the left example of without consistency.

adjacent motions for the non-adjacent frames:

$$P_s^t = \begin{cases} -\sum_{i=s}^{t} F_i, & if \quad s < t \\ \sum_{i=s}^{t} F_i, & if \quad s > t \end{cases} \tag{1}$$

where s and t are frame index and P_s^t is a meshflow pointing from s to t. Notably, the accumulations are conducted for the sparse motion fields. We warp the images according to the meshflow by a mesh-based image warp. To this end, all frames within a local window are warped to the central frame.

2.3. Fusing

With all frames aligned to the target frame, we average the pixels at the same spatial location for the denoising. Some of the pixels cannot be averaged if they correspond to misalignments, occlusions or dynamic objects. The fusing of these inconsistent pixels can result in severe "ghosting" effects. For each pixel location, we compare the intensities of the pixels at the target frame with the associated candidates at the warped frames. Pixels with intensity difference smaller than an empirical threshold $\tau = 20$ (intensity ranges from 0$\sim$255) are treated as consistent pixels. Other strategies have been explored in [15]. Here, we keep our method simple yet effective.

3. EXPERIMENTS

We conduct some experiments with respect to several aspects to validate the effectiveness of our design.

Pixel consistency. Figure 4 shows a comparison with regards to with and without consistency verification in the denoising results. Please notice the ghosting effects of the moving person highlighted by the red arrow.

Motion accumulation. Figure 5 shows a comparison between direct motion calculation and motion accumulation as discussed in Sec. 2.2. Both methods can produce comparable results. However, the direct calculation requires additional feature detection and tracking between non-adjacent frames.

Panning and Walking We test our performances on various camera motion types. The videos captured during panning and walking are more challenging than those filmed without large camera movements. Because fewer frames can be registed within the local window due to the scene variation. By experiments, we show that our method can successfully handle these camera movements.

Direct motion calculation Motion accumulation

Fig. 5. Comparison between direct motion calculation and motion accumulation. Both approaches yield similar results.

4. RESULTS

We run our method on an Intel i7 4.0GHz CPU with 16G RAM. Our unoptimized C++ implementation can achieve 260ms on average to process a frame with resolution of 1920×1080. Specifically, our method takes 27ms, 21ms, 38ms, 25ms and 149ms to track features, estimate mesh flow, accumulate motion, check consistency and fuse pixels, respectively. The method can be further accelerated by the GPU, especially for the fusing which is highly parallelizable.

We compare our method with the methods BM3D [5], VBM3D [19], BM4D [20], VBM4D [21] and Burst [15]. All these methods are quite slow. They usually need more than a minute to process a frame, except the burst denoising [15], which reports the speed as 480ms to process a frame under the similar image resolution. Fig. 6 shows some results. Residual noises can still be observed in the compared examples. Please notice the missing of the electric wire in the first example. The temporal issues of the compared examples can be viewed in the accompany video.

We collect some data from the project page of the burst denoising [15] for comparisons. Fig. 7 shows two examples. The burst denoising approach targets at the image denoising, where 10 noisy images are the input to the system and 1 clean image is the output. We generate our results with the same inputs as [15]. Our method can produce comparable results. Notably, the pixel consistency strategy of [15] tends to reject moving objects, which maybe reasonable when the output is a single image. Whereas, our video denoising must keep the dynamic objects as illustrated in the right example of Fig. 7.

5. CONCLUSION

We have presented a method that utilizes meshflow motion model for the video denoising. With the motion accumulations and the pixel consistency verification, our method can achieve strong denoising results under a very fast speed. Our method is robust to different type of camera motions and scene types. Various challenging cases demonstrate the effectiveness of the proposed method.

6. ACKNOWLEDGE

This work has been supported by National Natural Science Foundation of China (61502079 and 61370148).

Fig. 6. Comparisons with various methods, including (b) BM3D [5], (c) VBM3D [19], (d) BM4D [20] and (e) VBM4D [21]. Please zoom in for clearer comparisons and refer to the video for comparisons of the temporal consistencies. Our method can keep image details as highlighted by the rectangle in the first example while suppress the noises effectively as highlighted in the second example.

Fig. 7. Comparison with the Burst image denoising approach [15]. The examples are collected form the datapage of [15]. Our method can produce comparable results with improved efficiency. Moreover, in the right example, our method can keep the dynamic moving person, which is missing in [15], as highlighted by the red arrow.

References

[1] P. Chatterjee, N. Joshi, S. Kang, and Y. Matsushita, "Noise suppression in low-light images through joint denoising and demosaicing," in *Proc. CVPR*, 2011, pp. 321–328.

[2] J. Chen, C. Tang, and J. Wang, "Noise brush: interactive high quality image-noise separation," *ACM Trans. Graph.*, vol. 28, no. 5, pp. 146, 2009.

[3] K. Dabov, A. Foi, V. Katkovnik, and K. Egiazarian, "Color image denoising via sparse 3d collaborative filtering with grouping constraint in luminance-chrominance space," in *Proc. ICIP*, 2007, vol. 1, pp. I–313.

[4] X. Chen, B. Sing, J. Yang, and J. Yu, "Fast patch-based denoising using approximated patch geodesic paths," in *Proc. CVPR*, 2013, pp. 1211–1218.

[5] K. Dabov, A. Foi, V. Katkovnik, and K. Egiazarian, "Image denoising by sparse 3-d transform-domain collaborative filtering," *IEEE Trans. on Image Processing*, vol. 16, no. 8, pp. 2080–2095, 2007.

[6] E. Balster, Y. Zheng, and R. Ewing, "Combined spatial and temporal domain wavelet shrinkage algorithm for video denoising," *IEEE Trans. on Circuits and Syst. for Video Tech.*, vol. 16, no. 2, pp. 220–230, 2006.

[7] V. Zlokolica, A. Pizurica, and W. Philips, "Wavelet-domain video denoising based on reliability measures," *IEEE Trans. on Circuits and Syst. for Video Tech.*, vol. 16, no. 8, pp. 993–1007, 2006.

[8] A. Buades, B. Coll, and J. Morel, "A non-local algorithm for image denoising," in *Proc. CVPR*, 2005, vol. 2, pp. 60–65.

[9] A. Buades, B. Coll, and J. Morel, "A review of image denoising algorithms, with a new one," *Multiscale Modeling & Simulation*, vol. 4, no. 2, pp. 490–530, 2005.

[10] J. Chen and C. Tang, "Spatio-temporal markov random field for video denoising," in *Proc. CVPR*, 2007, pp. 1–8.

[11] N. Kalantari, E. Shechtman, C. Barnes, S. Darabi, D. B Goldman, and P. Sen, "Patch-based high dynamic range video," *ACM Trans. Graph.*, vol. 32, no. 6, pp. 202, 2013.

[12] M. Granados, K. Kim, J. Tompkin, and C. Theobalt, "Automatic noise modeling for ghost-free hdr reconstruction," *ACM Trans. Graph.*, vol. 32, no. 6, pp. 201, 2013.

[13] F. Tan, S. Liu, L. Zeng, and B. Zeng, "Kernel-free video deblurring via synthesis," in *Proc. ICIP*, 2016, pp. 2683–2687.

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

[15] Z. Liu, L. Yuan, X. Tang, M. Uyttendaele, and J. Sun, "Fast burst images denoising," *ACM Trans. Graph.*, vol. 33, no. 6, pp. 232, 2014.

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

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

[18] S. Liu, M. Li, S. Zhu, and B. Zeng, "Codingflow: Enable video coding for video stabilization," *IEEE Trans. on Image Processing*, vol. 26, no. 7, pp. 3291–3302, 2017.

[19] K. Dabov, A. Foi, and K. Egiazarian, "Video denoising by sparse 3d transform-domain collaborative filtering," in *In Proc. European Signal Process. Conf. EUSIPCO*, 2007.

[20] M. Maggioni, V. Katkovnik, K. Egiazarian, and A. Foi, "Nonlocal transform-domain filter for volumetric data denoising and reconstruction," *IEEE Trans. on Image Processing*, vol. 22, no. 1, pp. 119–133, 2013.

[21] M. Maggioni, G. Boracchi, A. Foi, and K. Egiazarian, "Video denoising, deblocking, and enhancement through separable 4-d nonlocal spatiotemporal transforms," *IEEE Trans. on Image Processing*, vol. 21, no. 9, pp. 3952–3966, 2012.

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

[23] J. Shi and C. Tomasi, "Good features to track," in *Proc. CVPR*, 1994, pp. 593–600.

[24] 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*, vol. 25, no. 11, pp. 5491–5503, 2016.