Smooth Globally Warp Locally: Video Stabilization using Homography Fields

William X. Liu, Tat-Jun Chin
School of Computer Science, The University of Adelaide, South Australia

***Abstract*—Conceptually, video stabilization is achieved by estimating the camera trajectory throughout the video and then smoothing the trajectory. In practice, the pipeline invariably leads to estimating update transforms that adjust each frame of the video such that the overall sequence appears to be stabilized. Therefore, we argue that estimating good update transforms is more critical to success than accurately modeling and characterizing the motion of the camera. Based on this observation, we propose the usage of homography fields for video stabilization. A homography field is a spatially varying warp that is regularized to be as projective as possible, so as to enable accurate warping while adhering closely to the underlying geometric constraints. We show that homography fields are powerful enough to meet the various warping needs of video stabilization, not just in the core step of stabilization, but also in video inpainting. This enables relatively simple algorithms to be used for motion modeling and smoothing. We demonstrate the merits of our video stabilization pipeline on various public testing videos.**

I. Introduction

The proliferation of video recording devices has resulted in large quantities of videos taken by amateurs, especially on popular video sharing websites. Many amateur videos are taken in an undirected and spontaneous manner, which yields low-quality videos with significant amounts of shakiness. There is thus a need for software tools that can conduct post hoc stabilization of recorded videos.

In theory, a video is stabilized by estimating the camera poses throughout the video, smoothing the trajectory to remove high-frequency components, then re-rendering the video from the new poses. In practice, most approaches avoid dealing explicitly with the camera trajectory, and directly estimate and filter the 2D motions (image transforms) between successive frames. From the smoothed motions, *update transforms* (2D warping functions) are obtained to adjust each frame of the video to "undo" the jerky motions.

Earlier methods relied on simple 2D image transforms (e.g., affine or projective) to model the camera motion [1], [2], [3]. Apart from efficiency in estimation, simple motion models also permit straightforward smoothing algorithms. However, the update transform is also customarily defined using the basic 2D transforms, which cannot preserve the image contents well. As a result the stabilized videos often appear distorted and "wobbly" [4]. Nevertheless, techniques based on simple 2D transforms can work well, especially if good motion planning is used [3], [5].

Recent works have proposed more sophisticated motion models and smoothing algorithms. Liu et al. [6] track local features in the video, then smooth the set of trajectories based on low-rank matrix factorization. Goldstein and Fattal [7] estimate fundamental matrices that encapsulate the epipolar constraint between successive frames, then generate virtual point trajectories from the epipolar relations which are then filtered. More recently, Liu et al. [8] spatially partition the video frame into a grid of subwindows, then estimate a chain of homographies across the video for each subwindow. The separate chains of homographies are then smoothed in a bundled manner to avoid drift; Fig. 1(a).

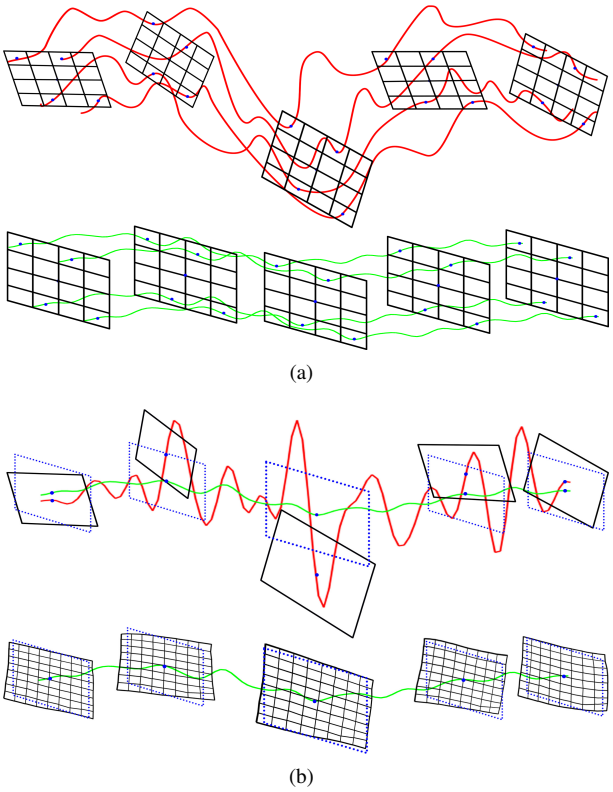


Fig. 1. (a) Given the input video, Liu et al. [8] spatially partition the video frame into subwindows (typically 16×16), then estimate a chain of homographies (red trajectories) for each subwindow. The multiple homography chains are then smoothed in a bundled fashion (green trajectories). (b) In our approach, the camera motion is estimated as a *single* 2D homography chain (red trajectory), which can be smoothed using standard techniques [2], [3] (green trajectory). Instead of applying frame-global update transforms (dotted frames), the smoothed parameters are used to estimate homography fields (shown as flexible grids), which are spatially varying warps regularized to be as projective as possible.

Contrary to the recent works, we argue that the key to effective video stabilization is in designing better update transforms - not in constructing complex motion models to precisely characterize and smooth the movement of each feature or pixel

in the video. To capture the camera motion, it is sufficient to estimate the frame global image motion - following [2], [3], we use simple 2D homographies in our approach. On the other hand, since update transforms will eventually need to be applied on all the frames, we argue that it is more fruitful to design warping functions that can adjust the frames without creating undesirable artifacts.

To construct the all-important update transform, we propose the usage of *homography fields*, which are spatially varying warps that are regularized to be as projective as possible [9]. This enables flexible and accurate warping that adheres closely to the underlying scene geometry. Our update transform is powerful enough to eliminate unwanted jerky motions, while at the same time prevent the warped sequence of frames from appearing wobbly or distorted. Crucially, homography fields can be integrated closely with any homography-based smoothing algorithm [2], [3]; this realizes a video stabilization pipeline that *smooths globally and warps locally*. Fig. 1(b) gives an overview of our approach.

Apart from removing shakiness, we show another crucial ability is video inpainting to fill in blank regions arising from video re-rendering. We show how this can be done more effectively using homography fields, compared to previous techniques [2]. Our work is one of the first to treat trajectory smoothing and video inpainting in a single unified pipeline.

A. Related work

a) Spatially varying warps: The spirit of our work is closely aligned with Liu et al. [4], who proposed content preserving warp (CPW) as an update transform for video stabilization (CPW was applied in [6], [7], [8]). The warp is designed to respect the contents of the image, while being as similar as possible to prevent wobbling in the output video. However, in its original form, CPW cannot account for video inpainting.

For the task of image stitching, Zaragoza et al. [9] proposed as-projective-as-possible (APAP) warp to handle non-pure rotational viewpoints. Our work can be seen as an extension of [9] to video stabilization, with two novel contributions. First, we show how APAP warp can be integrated with any homography-based stabilization [1], [2] or camera motion planning [3] algorithm. Second, we devise a novel video inpainting algorithm based on homography fields, which outperforms state-of-the-art methods.

b) Video inpainting: Differing from inpainting for censored or damaged videos [10], inpainting for stabilized videos is an *extrapolation* problem, since the missing regions almost always occur at the sides of the video frames. The state-of-the-art method [2] eschews the usage of standard mosaicing techniques for video inpainting [11] (i.e., stitching neighboring frames to the current frame), due to assumption on scene planarity. Instead, Matsushita et al. propagate 2D motion vectors (from optic flow computations) to guide the inpainting. However, Sec. IV shows that this approach produces visible artifacts if the blank region is large, since it is difficult to extrapolate far without an underlying geometric model. In contrast, since our technique based on homography field is guided by multi-view projective constraints, it can extrapolate to large regions with fewer artifacts.

Note that approaches that avoid inpainting must either limit the magnitude of smoothing (as shown in some of the results in [8]) or aggressively crop the output video (such as [5]) to avoid significant blank regions in the output video.

II. Smooth Globally Warp Locally

Let the video frames be $I_1, I_2, \ldots, I_T$. In the ideal case where the camera is purely rotating about a point, the image motion can be modeled perfectly by a homography chain. Denote C_t as the homography that warps I_t to the first frame I_1. To undo jerky motions, Gleicher and Liu [3] suggested to warp each frame I_t by the update transform

$$\mathbf{B}_t = \mathbf{P}_t^{-1}\mathbf{C}_t, \tag{1}$$

where $\mathbf{P}_t$ is a homography that warps I_t to I_1 following a new camera path. Since both $\mathbf{P}_t$ and $\mathbf{C}_t$ are homographies, $\mathbf{B}_t$ is also a homography. Observe that if $\mathbf{P}_t = \mathbf{C}_t$ (no smoothing), the update transform is the identity mapping.

The transform $\mathbf{C}_t$ is recursively defined as

$$\mathbf{C}_t = \mathbf{C}_{t-1}\mathbf{H}_{t,t-1}, \tag{2}$$

where $\mathbf{C}_1$ is the identity matrix, and $\mathbf{H}_{t,t-1}$ is the homography that maps from I_t to I_{t-1}. Motion estimation is thus achieved by estimating $\mathbf{H}_{t,t-1}$ for all $t = 2, \ldots, T$, then chaining them in the right order. To estimate $\mathbf{H}_{t,t-1}$, we detect and match local features between I_t and I_{t-1} (using wide baseline [12] or dense techniques [13]), before applying the standard robust estimation approach.

In our framework, any homography-based stabilization and motion planning algorithm can be used to estimate $\{\mathbf{P}_t\}_{t=1}^T$. For concreteness, we follow the single path smoothing algorithm of Liu et al. [8]. Given $\{\mathbf{C}_t\}_{t=1}^T$, the smoothed path $\{\mathbf{P}_t\}_{t=1}^T$ is obtained by minimizing

$$O(\{\mathbf{P}_t\}) = \sum_t \|\mathbf{P}_t - \mathbf{C}_t\|^2 + \lambda \sum_{r\in\Omega_t} \|\mathbf{P}_t - \mathbf{P}_r\|^2, \tag{3}$$

where Ω_t contains the index of the neighbors of I_t; in our work, each frame is linked to the nearest 60 frames. The second term smooths the trajectory, while the first term encourages $\mathbf{P}_t$ to be close to $\mathbf{C}_t$. Parameter λ controls the strength of smoothing by trading off the two terms. See Liu et al. for details of setting λ and minimizing (3).

A. Spatially varying update transform

As we will show later, homography-based motion modeling and smoothing (estimating $\{\mathbf{C}_t\}_{t=1}^T$ and optimizing $\{\mathbf{P}_t\}_{t=1}^T$) is sufficient to capture and smooth the camera trajectory, even if it is jerky and highly nonsmooth. The key to produce visually good results lies in the update transform. Instead of a frame-global homography (1), we map each pixel $\mathbf{p}_*$ in I_t to the output frame via the *local homography*

$$\mathbf{B}_t^* = \mathbf{P}_t^{-1}\mathbf{C}_t^*. \tag{4}$$

Conceptually, $\mathbf{C}_t^*$ is a homography that warps $\mathbf{p}_*$ to the base frame I_1. Similar to (2), $\mathbf{C}_t^*$ can be defined recursively as

$$\mathbf{C}_t^* = \mathbf{C}_{t-1}^*\mathbf{H}_{t,t-1}^*, \tag{5}$$

where $\mathbf{C}_1^*$ is the identity matrix, and $\mathbf{H}_{t,t-1}^*$ is a pixel-centered homography that maps $\mathbf{p}_*$ from I_t to I_{t-1}. Note that the de-shaking adjustment is provided by a single globally smoothed homography chain $\{\mathbf{P}_t\}_{t=1}^T$.

The local homographies $\{\mathbf{B}_t^*\}$ for all pixels $\{\mathbf{p}_*\}$ in I_t constitute a homography field. Estimating the $\mathbf{B}_t^*$ for a pixel $\mathbf{p}_*$ amounts to estimating $\mathbf{C}_t^*$, which in turn requires estimating $\mathbf{H}_{t,t-1}^*$ for all t. Let $\{(\mathbf{p}_i, \mathbf{p}_i')\}_{i=1}^N$ be verified feature matches across I_t and I_{t-1}, where $\mathbf{p}_i = [x_i \ y_i]^T$ and $\mathbf{p}_i' = [x_i' \ y_i']^T$. Using the Moving DLT technique [9], we estimate $\mathbf{H}_{t,t-1}^*$ as

$$\arg\min_{\mathbf{h}} \sum_i \|w_i^* \mathbf{a}_i \mathbf{h}\|^2, \quad \text{s.t. } \|\mathbf{h}\| = 1, \tag{6}$$

where $\mathbf{h} \in \mathbb{R}^9$ is obtained by vectorizing a 3×3 homography matrix, and $\mathbf{a}_i \in \mathbb{R}^{2\times 9}$ contains monomials

$$\mathbf{a}_i = \begin{bmatrix} \mathbf{0}_{1\times 3} & -\tilde{\mathbf{p}}_i^T & y_i' \tilde{\mathbf{p}}_i^T \\ \tilde{\mathbf{p}}_i^T & \mathbf{0}_{1\times 3} & -x_i' \tilde{\mathbf{p}}_i^T \end{bmatrix} \tag{7}$$

from linearizing the homography constraint for the i-th datum $(\mathbf{p}_i, \mathbf{p}_i')$. Here, $\tilde{\mathbf{p}}_i$ is $\mathbf{p}_i$ in homogeneous coordinates.

The non-stationary weights $\{w_i^*\}_{i=1}^N$ are calculated as

$$w_i^* = \exp\left(-\|\mathbf{p}_i - \mathbf{p}_*\|^2 / \sigma^2\right). \tag{8}$$

As $\mathbf{p}_*$ is varied across the image domain, the weights vary smoothly and provide a "spatial chaining" effect on the set of homographies $\{\mathbf{H}_{t,t-1}^*\}$ between I_t and I_{t-1}. Collectively the homographies define a spatially varying warp.

To solve (6), we vertically stack $\mathbf{a}_i$ for all i in a matrix $\mathbf{A}$, build the $2N \times 2N$ weight matrix

$$\mathbf{W}^* = \text{diag}([w_1^* \ w_1^* \ w_2^* \ w_2^* \ \ldots \ w_N^* \ w_N^*]), \tag{9}$$

and take the least significant right singular vector of $\mathbf{W}^*\mathbf{A}$.

B. Efficient estimation

Theoretically each pixel $\mathbf{p}_*$ in I_t has a corresponding homography $\mathbf{H}_{t,t-1}^*$. In practice, neighboring pixels yield very similar homographies. Following [9], we solve (6) on a X by Y grid on I_t, and warp a pixel from I_t using its closest local homography. Note that each $\mathbf{H}_{t,t-1}^*$ can be solved independently, thus the estimation effort scales linearly with the grid size. For $X \times Y = 20 \times 20$ and $N = 480$, Matlab can solve for all $\{\mathbf{H}_{t,t-1}^*\}$ in about 0.7s.

We also further simplify $\mathbf{B}_t^*$ by redefining (5) as

$$\mathbf{C}_t^* = \mathbf{C}_{t-1}^* \mathbf{H}_{t,t-1}^* \approx \mathbf{C}_{t-1} \mathbf{H}_{t,t-1}^*, \tag{10}$$

i.e., the frame-global homography chain $\mathbf{C}_{t-1}$ (obtained from the smoothing step) is used to propagate the spatially varying warp to the base frame. In practice, this yields little noticeable difference in the warping results.

Overall, our approach requires estimating and smoothing only a single homography chain (i.e., the camera trajectory), while the frame updating is conducted locally depending on the video contents, i.e., smooth globally warp globally.

III. Video Inpainting with Homography Fields

To fill in the blank regions of an updated frame, we can stitch neighboring frames to the current frame. However, the original approach of [11] based on standard homographic warps can produce artifacts in nonplanar scenes [2]. Here, we propose a more effective video inpainting technique based on homography fields. Algorithm 1 summarizes our method, and details are as follows.

Algorithm 1 Video inpainting using homography fields.

Require: Target frame $\tilde{I}_t$, source frames $\mathcal{S} = \{\tilde{I}_s \mid s \neq t\}$, original feature matches between all frames.

while there are blank pixels in $\tilde{I}_t$ and $\mathcal{S}$ is nonempty **do**
  Remove from $\mathcal{S}$ the frame $\tilde{I}_s$ closest in time to $\tilde{I}_t$.
  Obtain and verify new matches between $\tilde{I}_t$ and $\tilde{I}_s$.
  for each blank pixel $\mathbf{p}_*$ in $\tilde{I}_t$ **do**
    Compute $\mathbf{H}_{t,s}^*$ that is centered on $\mathbf{p}_*$ by (6).
    Warp $\mathbf{p}_*$ to $\tilde{I}_s$ by calculating $\tilde{\mathbf{p}}_*' \sim \mathbf{H}_{t,s}^* \tilde{\mathbf{p}}_*$.
    if $\tilde{I}_s(\mathbf{p}_*')$ is not blank or out of bounds **then**
      Copy pixel colors from $\tilde{I}_s(\mathbf{p}_*')$ to $\tilde{I}_t(\mathbf{p}_*)$.
      if $\mathbf{p}_*'$ is a detected feature in $\tilde{I}_s$ **then**
        Designate $\mathbf{p}_*$ as a feature, and match $\mathbf{p}_*$ to the correspondences of $\mathbf{p}_*'$ in other frames.
      end if
    end if
  end for
end while

A. Sliding window RANSAC

At this stage, the input video frames have been updated $\tilde{I}_1, \tilde{I}_2, \ldots, \tilde{I}_T$ to remove jerky motions. Each $\tilde{I}_t$ is stored as a 2D image with blank regions. The feature matches used in motion estimation and smoothing have also been warped to the updated frames.

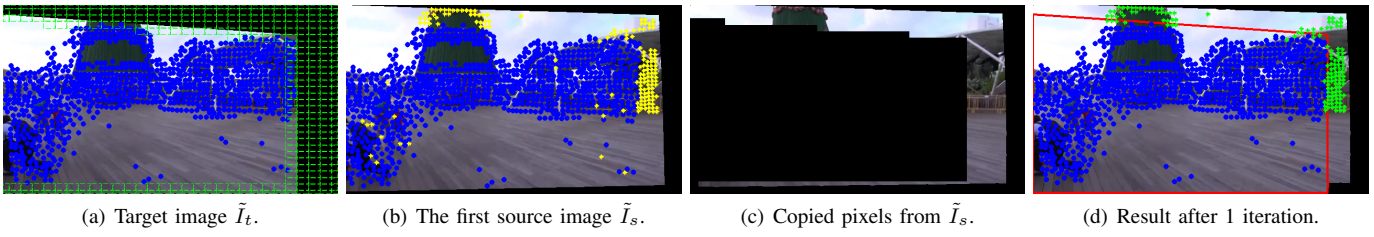
Given the target frame $\tilde{I}_t$ to inpaint and a source frame $\tilde{I}_s$ (a neighboring frame), we obtain additional feature matches across $\tilde{I}_t$ and $\tilde{I}_s$. RANSAC is then conducted in a sliding window fashion to remove outliers. Specifically, given a common subwindow of $\tilde{I}_t$ and $\tilde{I}_s$, RANSAC is applied to estimate a local homography using the feature matches in the subwindow. After processing all subwindows, any match that is not deemed an inlier in a subwindow is discarded. This method allows to retain more feature matches that would otherwise be discarded by standard RANSAC.

B. Feature propagation

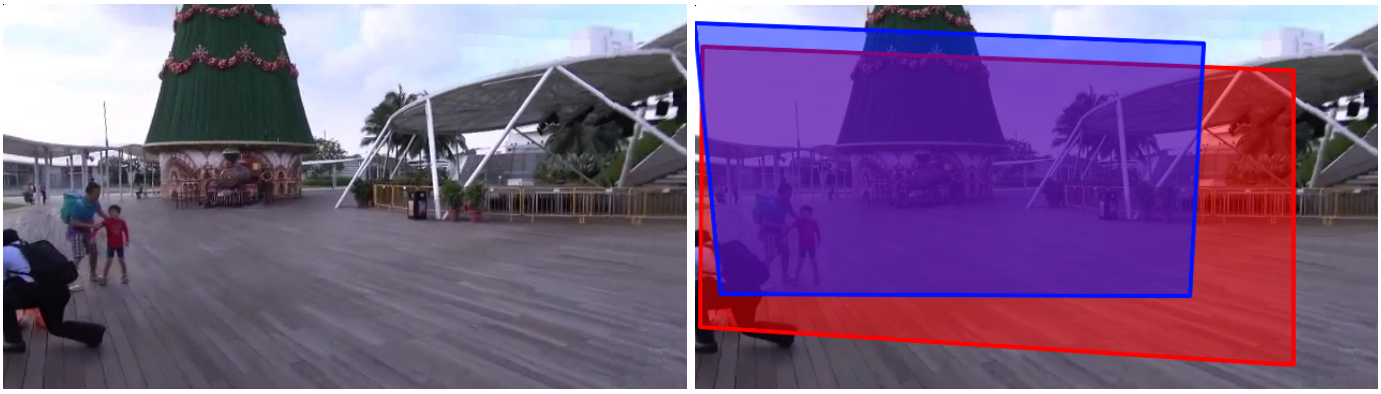
For each blank pixel $\mathbf{p}_*$ in $\tilde{I}_t$, a local homography $\mathbf{H}_{t,s}^*$ is estimated between $\tilde{I}_t$ and $\tilde{I}_s$ following (6). The source pixel $\mathbf{p}_*'$ is then obtained by warping $\mathbf{p}_*$ to $\tilde{I}_s$. If $\mathbf{p}_*'$ is a defined pixel, its color is copied to $\mathbf{p}_*$. To increase efficiency, $\tilde{I}_t$ can be divided into X by Y cells, and (6) is invoked on the center of the cells. A blank pixel is then warped using the local homography of the cell to which it belongs.

Similar to all feature-based methods, homography fields require good feature matches to produce accurate alignment. This means, however, that blank pixels far away from the defined regions (and available feature matches) may not be mapped optimally to the source image. Ideally there should be feature matches close to any blank pixel.

Due to the camera movement and the time-based order of choosing source frames in Algorithm 1, the blank regions in $\tilde{I}_t$ are usually inpainted in the order of their proximity to the defined regions; see Fig. 2(c). We take advantage of this condition to progressively grow the set of feature matches between the blank region and other source frames. If a copied pixel $\mathbf{p}_*'$ from $\tilde{I}_s$ is a detected feature, the inpainted pixel $\mathbf{p}_*$



(a) Target image $\tilde{I}_t$. (b) The first source image $\tilde{I}_s$. (c) Copied pixels from $\tilde{I}_s$. (d) Result after 1 iteration.



(e) Result after 20 iterations. (f) The blue and red shaded windows show the corresponding final output regions for $\tilde{I}_t$ by the methods of [14] and [8].

Fig. 2. Sample run of our video inpainting method (Algorithm 1). Blue points indicate verified feature matches between $\tilde{I}_t$ and $\tilde{I}_s$. Yellow points are feature points which appear in $\tilde{I}_s$ but not $\tilde{I}_t$. Green points indicate features propagated from $\tilde{I}_s$ to $\tilde{I}_t$. The final inpainting result is given in (e). In (f), we show the equivalent final output frames of two methods [14], [8] that do not conduct video inpainting.

in $\tilde{I}_t$ will also be designated as one. The correspondences of $\mathbf{p}'_*$ in the other frames will then be matched to $\mathbf{p}_*$, i.e., $\tilde{I}_t$ inherits the feature matches between $\tilde{I}_s$ and other frames. This ensures that a target blank pixel will always be close to feature matches; see Fig. 2(d).

Our feature propagation step is analogous to Matsushita et al.'s optical flow-based motion inpainting. In their approach, even though feature matches can be transferred to the blank regions as a byproduct, they will not benefit the subsequent motion propagation steps, since the optic flow vectors will not change. Sec. IV will compare both methods.

IV. Results

To evaluate the performance of the proposed approach, we have conducted experiments on the video clips used in [8], [14]. A variety of scenes and camera motions are contained in these videos. The reader is referred to the supplementary material[1] for the full video results.

Parameters settings for our approach are follows: the number of neighboring frames in Ω_t in (3) is 40; for homography field warps, the default grid size is $X \times Y = 20 \times 20$. In our pipeline, motion estimation, smoothing and frame updating take about 0.8s per frame, while inpainting takes $\approx$ 6.4s per frame.

The various components of our pipeline are benchmarked against several state-of-the-art videos stabilization approaches [2], [8], [5], [14].

[1] https://www.youtube.com/playlist?list=PLhRzYMNYGcCeUynmHIs1jR49ReqtYs60u

A. *Smooth globally warp locally*

An underlying premise of our approach is that 2D homographies are sufficient for motion estimation and smoothing. This follows the practice of influential works such as Matsushita et al. [2] and Gleicher and Liu [3]. However, the update transform must be more flexible than a standard homography warp, in order to deal with violations to the assumption of pure rotational motions and other distortions. To validate our idea, the following videos have been generated (see supplementary material):

- Video 1: stabilization by temporal local method [2] and frame updating via frame-global homography (1).
- Video 2: stabilization by temporal local method [2] and frame updating via homography field (4).
- Video 3: stabilization by single path smoothing (3) and frame updating via frame-global homography (1).
- Video 4: stabilization by single path smoothing (3) and frame updating via homography field (4).

Observe that significant distortions and wobbliness remain in Videos 1 and 3, more so in Video 1 since the temporal local method is not as prolific in smoothing very shaky videos. Good results are obtained in Videos 2 and 4, *even though the same homography-based stabilization approaches are used*. This supports our intuition that homography chains are sufficient to characterize the camera trajectory, and the "trick" for successful video stabilization is good frame updating.

Video 5 shows the result of [8] on the same input video above. Comparing Videos 4 and 5, it is evident that our

approach can provide the same quality. Practically, however, our approach is simpler and more efficient since only one homography chain needs to be estimated and stabilized. Note that since our pipeline is not tied to a specific stabilization and motion planning algorithm, any homography-based technique can be exploited by our pipeline.

B. Video inpainting

Since most recent video stabilization approaches do not conduct video inpainting, Matsushita et al. [2] remains state-of-the-art. First, we compare with methods that do not conduct inpainting at all [14], [8]. As shown on the top right of Figs. 3–8, the stabilized output frames of such methods must sacrifice significant image contents in order to avoid blank regions in the video. Observe that the amount of discarded contents is larger if the video is stabilized more aggressively (Figs. 7 and 8). This is expected since the stabilized path deviate more from the original trajectory.

If the frame to be inpainted has small blank regions (Fig. 3; see also the target frames in [2]), both Matsushita et al. and our approach can satisfactorily inpaint the video. The target frames in Figs. 4–8 have large blank regions - again, this condition occurs if the video is very shaky and significant amounts of smoothing must be applied. On such challenging cases, it can be observed that Matsushita et al.'s motion propagation often introduces artifacts in locations far away from the originally defined pixels. Note that the default number of neighboring frames used in [2] is 12, which can inpaint very limited blank region. Instead, we used our neighboring size 40. In contrast, homography fields can depend on projective regularization and feature propagation for guidance to more accurately fill-in large blank regions.

C. Overall results

Videos 6 to 22 in the supplementary material are the results generated with our overall video stabilization pipeline. Note that the blue boundaries in the videos mark the image region of stabilized frames before inpainting. To compare our results with results of the state-of-the-art methods, please refer to the videos of [8][2] (also included in the supplementary material[3]) and [5], [14][4] (the latter has been implemented as the video stabilizer on YouTube).

V. Conclusion

We have presented a new 2D video stabilization method that stabilizes globally and warps locally. Our thesis is that global homography is sufficient for motion representation and camera path stabilization. The key to effective video stabilization is the construction accurate update transforms. To this end, we proposed the usage of homography fields which is a kind of spatially varying warp. Compared with state-of-the-art methods, the proposed method can generate equally good results with a much simpler pipeline. Based on homography fields, we also proposed a video inpainting method for stabilized videos. Our inpainting algorithm allows the decrease of cropping ratio and preserve content.

[2] http://www.liushuaicheng.org/SIGGRAPH2013/index.htm

[3] https://www.youtube.com/playlist?list=PLhRzYMNYGcCehIMvvY9vl-y3F0M0lK0mJ

[4] http://www.youtube.com/playlist?list=PLhRzYMNYGcCdF4LPbuiHmMBYhxerNMGB_

A. Limitations and future work

Our video stabilization and inpainting method rely on the existence of sufficient feature matches across successive frames. With denser feature points, our method is able to produce stabilized frames closer to the real scene and inpainted images with less distortion. However, if the feature points are not widely spread across the entire image area, especially around the area with clear and definite geometric structure, our methods will not produce very satisfactory results. To alleviate this problem, we will be investigating spatially smooth optic flow methods [15].

Another issue of inpainting in some input videos is the occasional sudden exposure changes between neighboring frames. This yields unsightly "strips" in the inpainted regions. To deal with exposure changes, we will investigate exposure normalization schemes [16].

References

[1] C. Morimoto and R. Chellappa, "Evaluation of image stabilization algorithms," in *ICASSP*, 1998. 1, 2

[2] Y. Matsushita, E. Ofek, W. Ge, X. Tang, and H.-Y. Shum, "Full-frame video stabilization with motion inpainting," *IEEE TPAMI*, vol. 28, no. 7, pp. 1150–1163, 2006. 1, 2, 3, 4, 5, 6, 7, 8

[3] M. L. Gleicher and F. Liu, "Re-cinematography: Improving the camerawork of casual video," *ACM Transactions on Multimedia Computing, Communications, and Applications*, vol. 5, no. 1, pp. 2:1–2:28, 2008. 1, 2, 4

[4] F. Liu, M. Gleicher, H. Jin, and A. Agarwala, "Content-preserving warps for 3d video stabilization," in *ACM SIGGRAPH*, 2009. [Online]. Available: http://doi.acm.org/10.1145/1576246.1531350 1, 2

[5] M. Grundmann, V. Kwatra, and I. Essa, "Auto-directed video stabilization with robust l1 optimal camera paths," in *CVPR*, 2011. 1, 2, 4, 5

[6] F. Liu, M. Gleicher, J. Wang, H. Jin, and A. Agarwala, "Subspace video stabilization," *ACM Trans. Graph.*, vol. 30, no. 1, pp. 4:1–4:10, 2011. 1, 2

[7] A. Goldstein and R. Fattal, "Video stabilization using epipolar geometry," *ACM Trans. Graph.*, vol. 31, no. 5, pp. 126:1–126:10, 2012. 1, 2

[8] S. Liu, L. Yuan, P. Tan, and J. Sun, "Bundled camera paths for video stabilization," *ACM SIGGRAPH*, 2013. 1, 2, 4, 5, 6, 7, 8

[9] J. Zaragoza, T.-J. Chin, M. Brown, and D. Suter, "As-projective-as-possible image stitching with moving dlt," in *CVPR*, 2013. 2, 3

[10] Y. Wexler, E. Shechtman, and M. Irani, "Space-time completion of video," *IEEE TPAMI*, vol. 29, no. 3, pp. 463–476, 2007. 2

[11] A. Litvin, J. Konrad, and W. C. Karl, "Probabilistic video stabilization using kalman filtering and mosaicking," in *In IS&T/SPIE Symposium on Electronic Imaging, Image and Video Communications and Proc*, 2003, pp. 663–674. 2, 3

[12] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *IJCV*, vol. 60, no. 2, pp. 91–110, 2004. 2

[13] N. Sundaram, T. Brox, and K. Keutzer, "Dense point trajectories by GPU-accelerated large displacement optical flow," in *ECCV*, 2010. 2

[14] M. Grundmann, V. Kwatra, D. Castro, and I. Essa, "Effective calibration free rolling shutter removal," *ICCP*, 2012. 4, 5, 6, 7, 8

[15] S. Liu, Y. Ping, P. Tan, and J. Sun, "SteadyFlow: spatially smooth optical flow for video stabilization," in *CVPR*, 2014. 5

[16] Y. Hwang, J.-Y. Lee, I. S. Kweon, and S. J. Kim, "Color transfer using probabilistic moving least squares," in *CVPR*, 2014. 5

Fig. 3. Selected video inpainting result 1. (top left) Updated input frame $\tilde{I}_t$; (top right) Blue and red shaded windows respectively indicate equivalent final output frames of Grundmann et al. [14] and Liu et al. [8] who do not conduct video inpainting; (bottom left) Matsushita et al. [2]'s result using motion inpainting; (bottom right) Our result using homography fields.

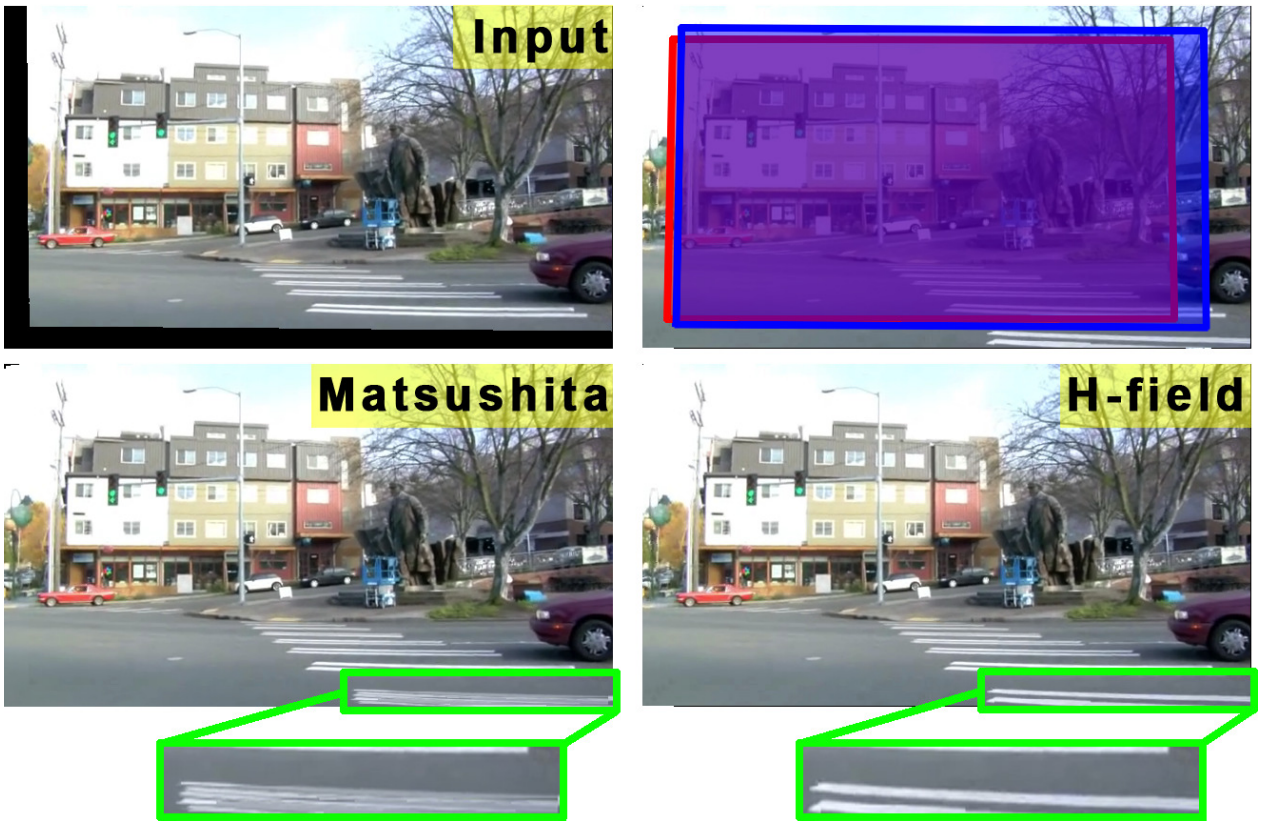


Fig. 4. Selected video inpainting result 2. (top left) Updated input frame $\tilde{I}_t$; (top right) Blue and red shaded windows respectively indicate equivalent final output frames of Grundmann et al. [14] and Liu et al. [8] who do not conduct video inpainting; (bottom left) Matsushita et al. [2]'s result using motion inpainting; (bottom right) Our result using homography fields.

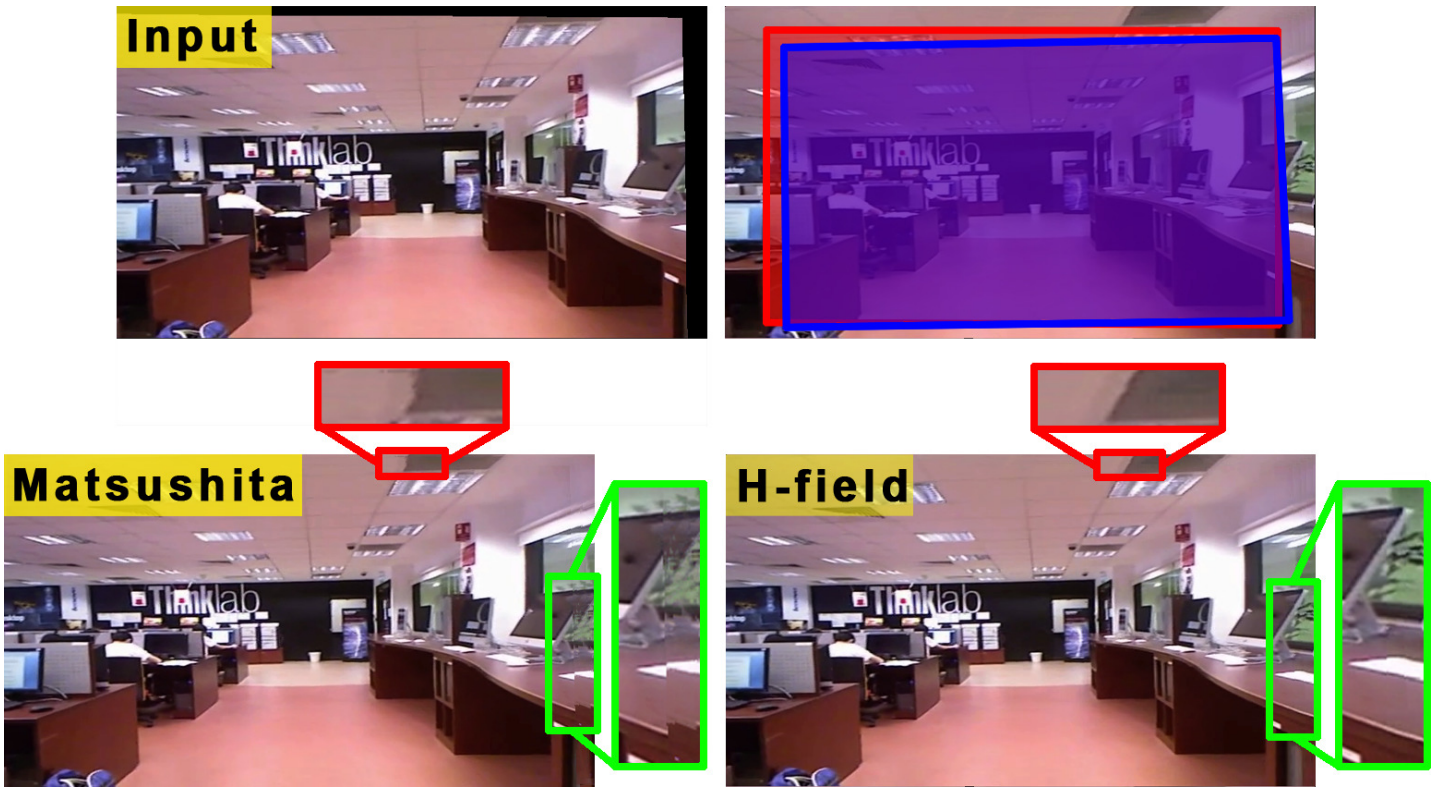


Fig. 5. Selected video inpainting result 3. (top left) Updated input frame $\tilde{I}_t$; (top right) Blue and red shaded windows respectively indicate equivalent final output frames of Grundmann et al. [14] and Liu et al. [8] who do not conduct video inpainting; (bottom left) Matsushita et al. [2]'s result using motion inpainting; (bottom right) Our result using homography fields.

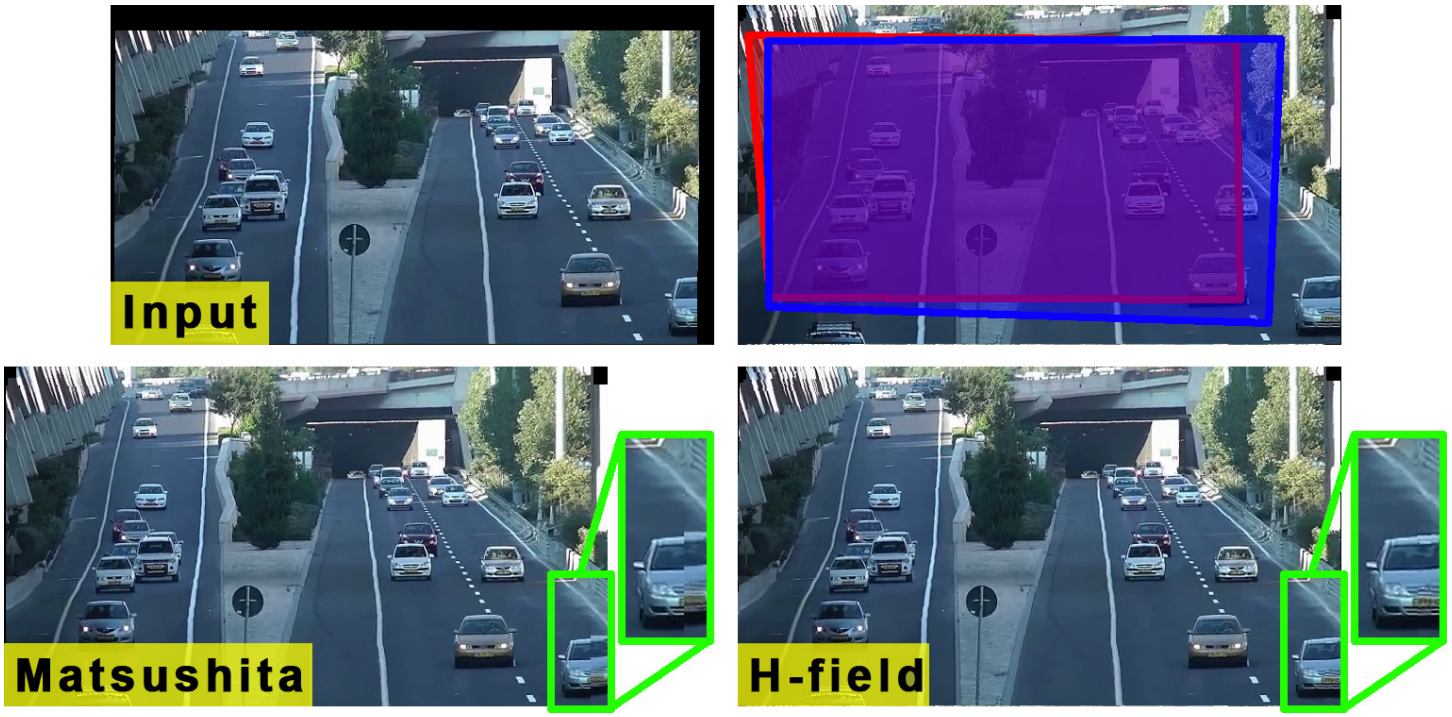


Fig. 6. Selected video inpainting result 4. (top left) Updated input frame $\tilde{I}_t$; (top right) Blue and red shaded windows respectively indicate equivalent final output frames of Grundmann et al. [14] and Liu et al. [8] who do not conduct video inpainting; (bottom left) Matsushita et al. [2]'s result using motion inpainting; (bottom right) Our result using homography fields.

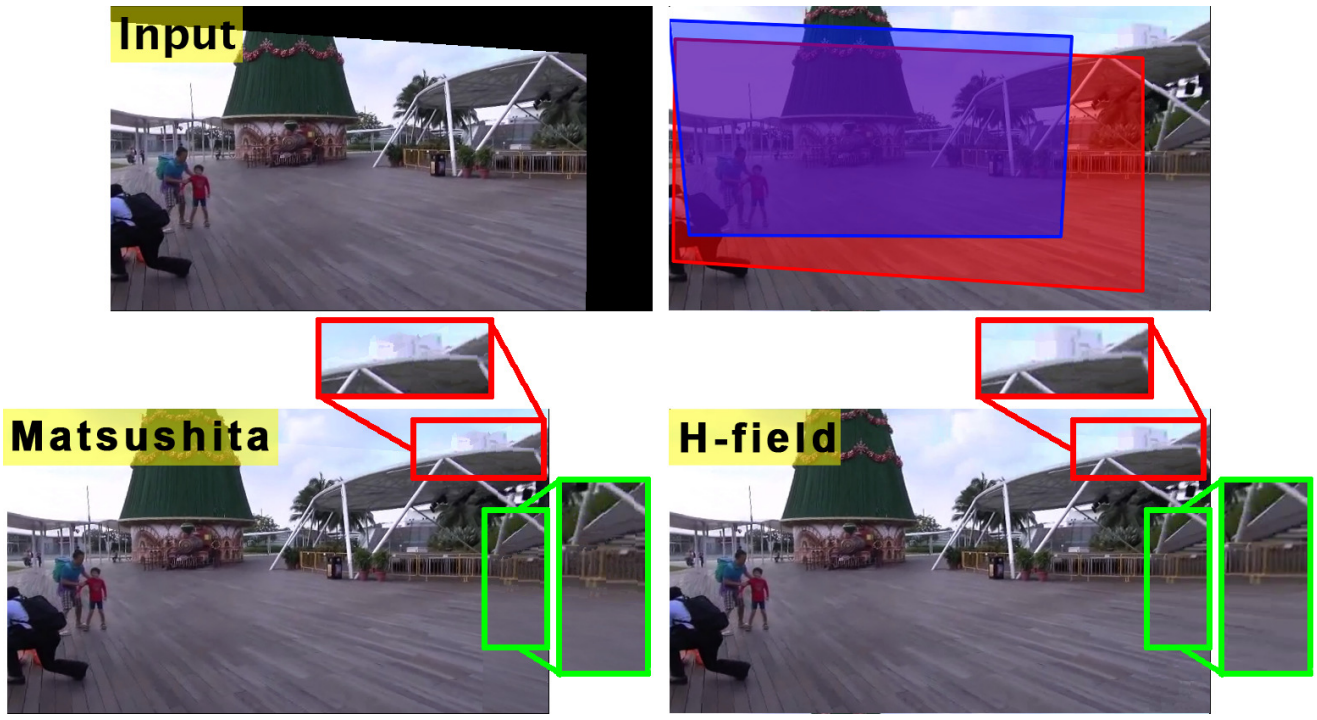


Fig. 7. Selected video inpainting result 5 (same data used in Fig. 2). (top left) Updated input frame $\tilde{I}_t$; (top right) Blue and red shaded windows respectively indicate equivalent final output frames of Grundmann et al. [14] and Liu et al. [8] who do not conduct video inpainting; (bottom left) Matsushita et al. [2]'s result using motion inpainting; (bottom right) Our result using homography fields.

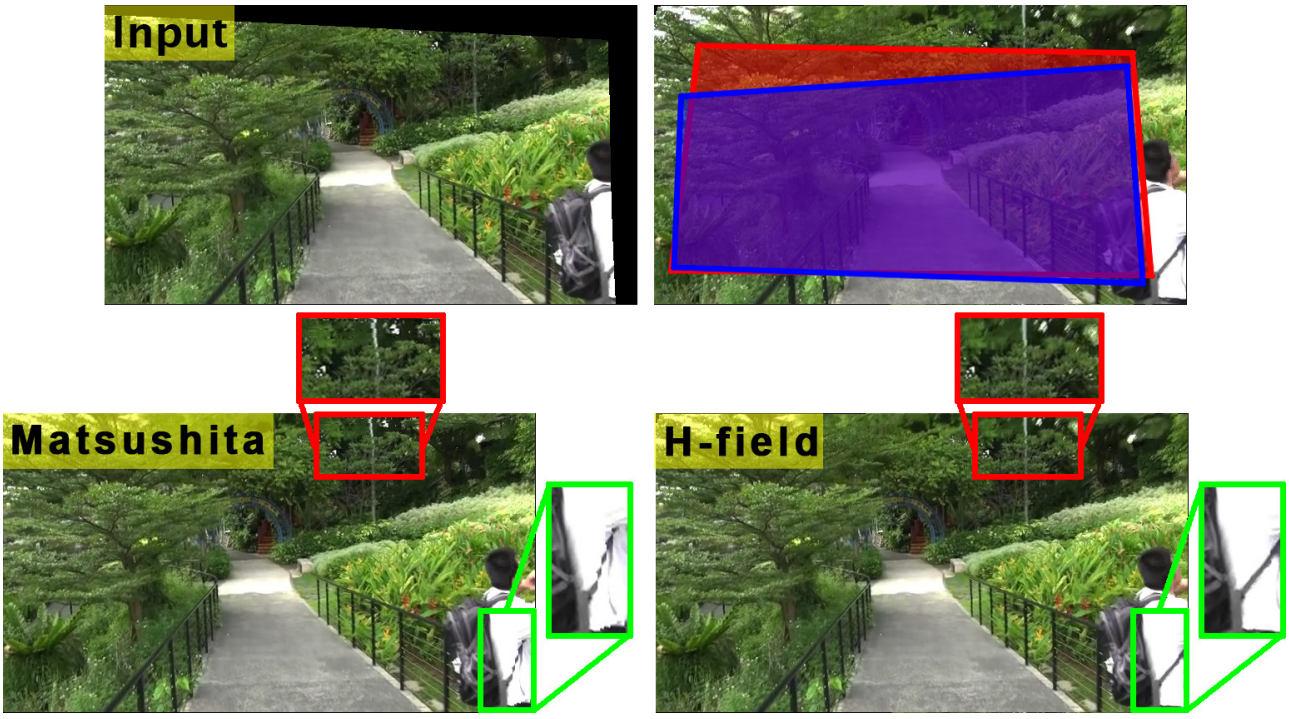


Fig. 8. Selected video inpainting result 6. (top left) Updated input frame $\tilde{I}_t$; (top right) Blue and red shaded windows respectively indicate equivalent final output frames of Grundmann et al. [14] and Liu et al. [8] who do not conduct video inpainting; (bottom left) Matsushita et al. [2]'s result using motion inpainting; (bottom right) Our result using homography fields.