Full-frame Video Stabilization

Yasuyuki Matsushita Eyal Ofek Xiaoou Tang Heung-Yeung Shum
Microsoft Research Asia
Beijing Sigma Center, No.49, Zhichun Road, Haidian District
Beijing 100080, P. R. China
{yasumat,eyalofek,xitang,hshum}@microsoft.com

Abstract

Video stabilization is an important video enhancement technology which aims at removing annoying shaky motion from videos. We propose a practical and robust approach of video stabilization that produces full-frame stabilized videos with good visual quality. While most previous methods end up with producing low resolution stabilized videos, our completion method can produce full-frame videos by naturally filling in missing image parts by locally aligning image data of neighboring frames. To achieve this, **motion inpainting** *is proposed to enforce spatial and temporal consistency of the completion in both static and dynamic image areas. In addition, image quality in the stabilized video is enhanced with a new practical deblurring algorithm. Instead of estimating point spread functions, our method transfers and interpolates sharper image pixels of neighbouring frames to increase the sharpness of the frame. The proposed video completion and deblurring methods enabled us to develop a complete video stabilizer which can naturally keep the original image quality in the stabilized videos. The effectiveness of our method is confirmed by extensive experiments over a wide variety of videos.*

1. Introduction

Video enhancement has been steadily gaining in importance with the increasing prevalence of digital visual media. One of the most important enhancements is video stabilization, which is the process for generating a new compensated video sequence where undesirable image motion caused by jittering is removed.

A major problem of current software video stabilizers is that missing image areas appear in the stabilized video due to the compensation of the motion path as shown in Fig. 1. This problem has been handled by either trimming the video to obtain the portion that appears in all frames or constructing image mosaics by accumulating neighboring frames to fill up the missing image areas. The former approach has the problem of reducing the original video resolution. Moreover, sometimes due to severe camera-shake, there might be no common area among neighboring frames. The latter approach works well for static and planar scenes, but produces visible artifacts for dynamic or non-planar scenes.

In this paper, we propose a practical and robust video completion and deblurring method which aims at generating full-frame stabilized videos with good visual quality. At the heart of the completion algorithm, we propose a new technique, *motion inpainting*, to propagate local motion data which is used for natural stitching of image mosaics. These methods enable us to develop a high quality video stabilizer that maintains the visual quality of the original video after stabilization.

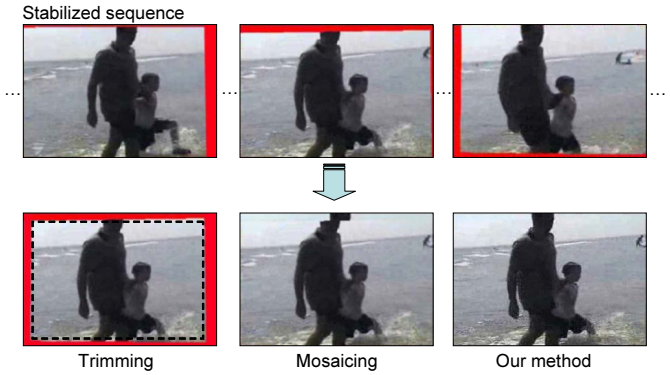


Figure 1: *Top row: stabilized image sequence. Red area represents the missing image area. Bottom row: from left to right, result of trimming (dotted rectangle becomes the final area), mosaicing and our method.*

1.1. Prior work

Motion deblurring has been studied extensively in the literature [1, 2, 3]. In the context of video frame deblurring, the point spread function (PSF) is estimated by assuming rather simple camera motion models[4, 5]. Although these methods are effective in some cases, the assumption of the simple motion models does not hold in many practical situations. In addition, deblurring methods which use deconvolution require very accurate PSFs that are usually hard to obtain.

Filling in missing image areas in a video is called *video completion*. In [6], mosaicing is used to fill up the missing image areas in the context of video stabilization. Unfortunately, the method does not address the problem of non-planar scenes and moving objects that may appear at the boundary of the video frames, which might cause significant artifacts. Wexler *et al.* [7] filled in the holes in a video by sampling spatio-temporal volume patches from different portions of the same video. This non-parametric sampling based approach

Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)
1063-6919/05 $20.00 © 2005 **IEEE**

produces a good result; however, it is highly computationally intensive. Also, it requires a long video sequence of a similar scene to increase the chance of finding correct matches, which is not often available in the context of video stabilization. Jia *et al.* [8] took a different approach to solve the same problem by segmenting the video into two layers, i.e., a moving object layer and a static background layer. One limitation of this approach is that the moving object needs to be observed for a long time, at least for a single period of its periodic motion; therefore, the method is not suitable for filling in the video boundaries where a sufficient amount of observation is not guaranteed.

1.2. Proposed approach

The limitations of the previous approaches and practical demands motivated us to develop effective completion and deblurring methods for generating full-frame stabilized videos. This paper has two primary contributions.

Video completion with motion inpainting: First, a new video completion method is proposed which is based on motion inpainting. The idea of motion inpainting is propagating local motion, instead of color/intensity as in image inpainting [9, 10], into the missing image areas. The propagated motion is then used to naturally fill up missing image areas even for scene regions that are non-planar and dynamic. Using the propagated local motion as a guide, image data from neighboring frames are locally warped to maintain spatial and temporal continuities of the stitched images. Image warping based on local motion was used in the de-ghosting algorithm for panoramic image construction by Shum and Szeliski [11]. Our method is different from theirs in that we propagate the local motion into an area where the local motion cannot be directly computed.

Practical motion deblurring method: Second, we address the problem of motion blur in the stabilized videos. While motion blur in original videos looks natural, it becomes an annoying noise in stabilized videos because it does not match the compensated camera motion. Furthermore, mosaicing without appropriate deblurring results in inconsistent stitching of blurry and sharp images. To solve this problem, we propose a practical deblurring method which does not require accurate point spread functions (PSFs) which are usually hard to obtain. Instead of estimating PSFs, we propose a method to transfer sharper pixels to corresponding blurry pixels to increase the sharpness and to generate a video of consistent sharpness. The proposed deblurring method is different from super-resolution methods such as [12] in that our method only transfers pixels from sharper frames and replaces pixels by weighted interpolation.

In the rest of this paper, Sec. 2 describes global and local motion estimation and smoothing methods which are used in our deblurring and completion methods. The video completion algorithm based on motion inpainting is described in Sec. 3. Sec. 4 presents the proposed image deblurring method. In Sec. 5, we show results of both stabilization and additional video enhancement applications. Conclusions are drawn in Sec. 6.

2. Motion Estimation and Smoothing

This section describes motion estimation methods which are used in the proposed deblurring and completion method. Sec. 2.1 describes the method to estimate interframe image transformation, or global motion. Local motion which deviates from the global motion is estimated separately as described in Sec. 2.2. The global motion is used for two purposes, stabilization and image deblurring, while the local motion is used for video completion. Sec. 2.3 describes the motion smoothing algorithm which is essential for stabilizing global motion.

2.1. Global motion estimation

We first explain the method of estimating global motion between consecutive images.

In the case that a geometric transformation between two images can be described by a homography (or 2D perspective transformation), the relationship between two overlapping images $I(\mathbf{p})$ and $I'(\mathbf{p}')$ can be written by $\mathbf{p}\sim\mathbf{T}\mathbf{p}'$. $\mathbf{p}=(x, y, 1)^T$ and $\mathbf{p}'=(x', y', 1)^T$ are pixel locations in projective coordinates, and $\sim$ indicates equality up to scale since the 3×3 matrix $\mathbf{T}$ is invariant to scaling.

Global motion estimation is done by aligning pair-wise adjacent frames assuming a geometric transformation. In our method, an affine model is assumed between consecutive images. We use the hierarchical motion estimation framework proposed by Bergen *et al.* [13]. By applying the parameter estimation for every pair of adjacent frames, a global transformation chain is obtained.

Throughout this paper, we denote the pixel location in the image coordinate I_t as $\mathbf{p}_t$. The subscript t indicates the index of the frame. We also denote the global transformation $\mathbf{T}_i^j$ to represent the coordinate transform from frame i to j. Therefore the transformation of image I_t to the I_{t-1} coordinate can be described as $I_t(\mathbf{T}_t^{t-1}\mathbf{p}_t)$. Note that transformation $\mathbf{T}$ only describes the coordinate transform, hence $I_{t-1}(\mathbf{T}_t^{t-1}\mathbf{p}_t)$ has the pixel values of frame $t-1$ in the coordinates of frame t.

2.2. Local motion estimation

Local motion describes the motion which deviates from the global motion model, e.g., motion of moving objects or image motion due to non-planar scenes. Local motion is estimated by computing optical flow between frames after applying a global transformation, using only the common coverage areas between the frames. A pyramidal version of Lucas-Kanade optical flow computation [14] is applied to obtain the optical flow field $\mathbf{F}_t^{t'}(\mathbf{p}_t) = [u(\mathbf{p}_t)\ \ v(\mathbf{p}_t)]^t$. $\mathbf{F}_t^{t'}(\mathbf{p}_t)$ represents an optical flow from frame $I_t(\mathbf{p}_t)$ to $I_{t'}(\mathbf{T}_{t'}^t\mathbf{p}_t')$, and u and v represent the flow vector along the x- and y-direction respectively in $\mathbf{p}_t$ coordinates.

2.3. Removal of undesired motion

A stabilized motion path is obtained by removing undesired motion fluctuation. As assumed in [6], the intentional motion in the video is usually slow and smooth, so we define

Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)
1063-6919/05 $20.00 © 2005 **IEEE**

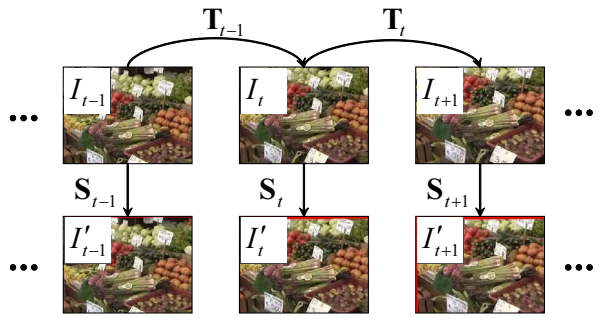


Figure 2: *Illustration of the global transformation chain* $\mathbf{T}$ *defined over the original video frames* I_i*, and the transformation from the original path to the smoothed path* $\mathbf{S}$*. The bottom frame sequence is the motion compensated sequence.*

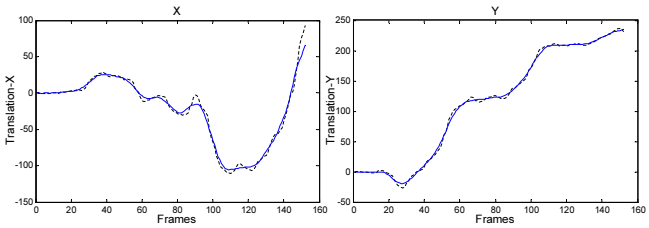


Figure 3: *Original motion path (dotted line) and smoothed motion path (solid line) with our local displacement smoothing. Translations along X and Y direction are displayed.*

the high frequency component in the global motion chain as the unintentional motion.

Previous motion smoothing methods smooth out the transformation chain itself or the cumulative transformation chain with an anchoring frame. Our method, on the other hand, smoothes *local displacement* in order to accomplish motion smoothing.

When smoothing is applied to the original transformation chain $\mathbf{T}_0^1 \dots \mathbf{T}_{i-1}^i$ as it is done in prior works, the smoothed transformation chain $\tilde{\mathbf{T}}_0^1 \dots \tilde{\mathbf{T}}_{i-1}^i$ is obtained. In this case, a motion compensated frame I_i' is obtained by transforming I_i with $\prod_{n=0}^{i} \mathbf{T}_{n+1}^n \tilde{\mathbf{T}}_n^{n+1}$. This cascade of the original and smoothed transformation chain often generates accumulation error. In contrast, our method is free from accumulative error because our method locally smoothes displacement from the current frame to the neighboring frames.

Instead of smoothing out the transformation chain along the video, we directly compute the transformation $\mathbf{S}$ from a frame to the corresponding motion-compensated frame using only the neighouring transformation matrices. We denote the indices of neighboring frames as $\mathcal{N}_t = \{j | t-k <= j <= t+k\}$. Let's assume that frame I_t is located at the origin, aligned with the major axes. We can calculate the position of each neighboring frame I_s, relative to frame I_t, by the local displacement $\mathbf{T}_t^s$. We seek the correcting transformation $\mathbf{S}$ from the original frame I_t to the motion-compensated frame I_t' according to

$$\mathbf{S}_t = \sum_{i \in \mathcal{N}_t} \mathbf{T}_t^i \star G(k), \tag{1}$$

where $G(k) = \frac{1}{\sqrt{2\pi}\sigma} e^{-k^2/2\sigma^2}$ is a Gaussian kernel, and the $\star$ operator represents convolution, and $\sigma = \sqrt{k}$ is used. Using the obtained matrices $\mathbf{S}_0, \dots, \mathbf{S}_t$, the original video frames can be warped to the motion-compensated video frames by

$$I_t'(\mathbf{p}_t') \leftarrow I_t(\mathbf{S}_t \mathbf{p}_t). \tag{2}$$

Fig. 3 shows the result of our motion smoothing method with k=6 in Eq. (1). In the figure, x- and y-translation elements of the camera motion path are displayed. As we can see in the figure, abrupt displacements which are considered to be unwanted camera motion are well reduced by our motion smoothing. The smoothness of the new camera motion path can be controlled by changing k, with a larger k yielding a smoother result. We found that annoying high frequency motion is well removed by setting k=6, i.e., about 0.5 *sec* with NTSC. k can be increased when a smoother video is preferred.

3. Video Completion with Motion Inpainting

Our video completion method locally adjusts image mosaics using the local motion field in order to obtain seamless stitching of the mosaics in the missing image areas. At the heart of our algorithm, *motion inpainting* is proposed to propagate the motion field into the missing image areas where local motion cannot be directly computed. The underlying assumption is that the local motion in the missing image areas is similar to that of adjoining image areas. The flow chart of the algorithm is illustrated in Fig. 4. First, the local motion from the neighboring frame is estimated over the common coverage image area. The local motion field is then propagated into missing image areas. Note that unlike prior image inpainting works, we do not propagate color but propagate local motion. Finally, the propagated local motion is used as a guide to locally warp image mosaics to achieve smooth stitching of the mosaics.

Let $\mathcal{M}_t$ be the missing pixels, or undefined image pixels, in the frame I_t. We wish to complete $\mathcal{M}_t$ for every frame t while maintaining visually plausible video quality.

0. Mosaicing with consistency constraint As a first step of video completion, we attempt to cover the static and planar part of the missing image area by mosaicing with an evaluation of its validity. When the global transformation is correct and the scene in the missing image area is static and planar, mosaics generated by warping from different neighboring frames should be consistent with each other in the missing area. Therefore, it is possible to evaluate the validity of the mosaic by testing the consistency of the multiple mosaics which cover the same pixels. We use the variance of the mosaic pixel values to measure the consistency; when the variance is high, the mosaic is less reliable at the pixel. For each

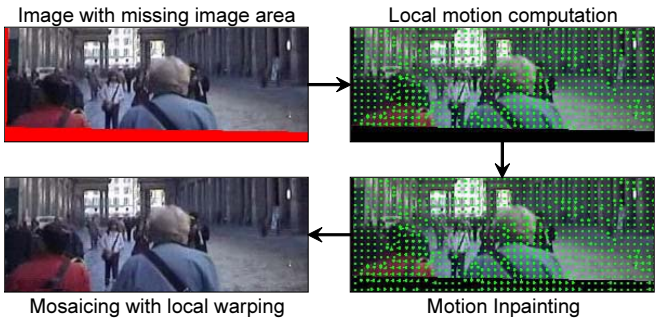


Figure 4: *Video completion. Local motion is first computed between the current frame and a neighboring frame. Computed local motion is then propagated with motion inpainting method. The propagated motion is finally used to locally adjust mosaics.*

pixel $\mathbf{p}_t$ in the missing image area $\mathcal{M}_t$, the variance of the mosaic pixel values is evaluated by

$$v_t(\mathbf{p}_t) = \frac{1}{n-1} \sum_{t' \in \mathcal{N}_t} \left[I_{t'}(\mathbf{T}_t^{t'} \mathbf{p}_t) - \overline{I}_{t'}(\mathbf{T}_t^{t'} \mathbf{p}_t) \right]^2, \qquad (3)$$

where

$$\overline{I}_{t'}(\mathbf{T}_t^{t'} \mathbf{p}_t) = \frac{1}{n} \sum_{t' \in \mathcal{N}_t} I_{t'}(\mathbf{T}_t^{t'} \mathbf{p}_t), \qquad (4)$$

and n is the number of neighboring frames. For color images, we use the intensity value of the pixel which is computed by $0.30R+0.59G+0.11B$ [17]. A pixel $\mathbf{p}_t$ is filled in by the median of the warped pixels only when the computed variance is lower than a predefined threshold T:

$$I_t(\mathbf{p}_t) = \begin{cases} \text{median}_{t'} \left(I_{t'}(\mathbf{T}_t^{t'} \mathbf{p}_t) \right) & \text{if} \quad v_t < T \\ \text{keep it as missing} & \text{otherwise.} \end{cases} \qquad (5)$$

If all missing pixels $\mathcal{M}_t$ are filled with this mosaicing step, we can skip the following steps and move to the next frame.

1. Local motion computation From this step, each neighboring frame $I_{t'}$ is assigned a priority to be processed based on its alignment error. Usually, it is observed that the nearer frame shows a smaller alignment error, and thus has a higher processing priority. The alignment error is computed using the common coverage area of $I_t(\mathbf{p}_t)$ and $I_{t'}(\mathbf{T}_t^{t'} \mathbf{p}_t)$ by

$$e_{t'}^t = \sum_{\mathbf{p}_t} |I_t(\mathbf{p}_t) - I_{t'}(\mathbf{T}_t^{t'} \mathbf{p}_t)|. \qquad (6)$$

Local motion is estimated by the method described in Sec. 2.2.

2. Motion Inpainting In this step, the local motion data in the known image areas is propagated into the missing image areas. The propagation starts at pixels on the boundary of the missing image area. Using motion values of neighboring known pixels, motion values on the boundary are defined,

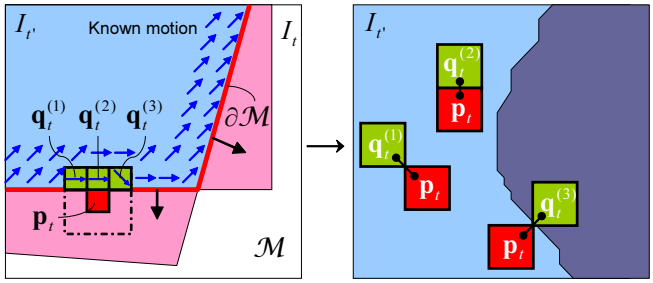


Figure 5: *Motion inpainting. Motion field is propagated on the advancing front $\partial\mathcal{M}$ into $\mathcal{M}$. The color similarities between $\mathbf{p}_t$ and its neighbors $\mathbf{q}_t$ are measured in the neighboring frame $I_{t'}$ after warped by local motion of $\mathbf{q}_t$, and they are used as weight factors for the motion interpolation.*

and the boundary gradually advances into the missing area $\mathcal{M}$ until it is completely filled as illustrated in Fig. 5.

Suppose $\mathbf{p}_t$ is a pixel in a missing area $\mathcal{M}$. Let $\mathcal{H}(\mathbf{p}_t)$ be the pixels of the neighborhood of $\mathbf{p}_t$, that already has a defined motion value by either the initial local motion computation or prior extrapolation of motion data. The motion value for pixel $\mathbf{p}_t$ is generated by a weighted average of the motion vectors of the pixels $\mathcal{H}(\mathbf{p}_t)$:

$$\mathbf{F}_t^{t'}(\mathbf{p}_t) = \frac{\sum_{\mathbf{q}_t \in \mathcal{H}(\mathbf{p}_t)} w(\mathbf{p}_t, \mathbf{q}_t) \mathbf{F}_t^{t'}(\mathbf{q}_t)}{\sum_{\mathbf{q}_t \in \mathcal{H}(\mathbf{p}_t)} w(\mathbf{p}_t, \mathbf{q}_t)}, \qquad (7)$$

where $w(\mathbf{p}_t, \mathbf{q}_t)$ determines the contribution of the motion value of $\mathbf{q}_t \in \mathcal{H}(\mathbf{p}_t)$ to pixel $\mathbf{p}_t$. We use color similarity (or intensity similarity in the case of grayscale videos) as a measurement for motion similarity, assuming that neighboring pixels of similar colors belong to the same object in the scene, and thus they will likely move in a similar motion. Since the color of pixel $\mathbf{p}_t$ is unknown in frame I_t, we use the neighboring frame $I_{t'}$ for the estimation of $w(\mathbf{p}_t, \mathbf{q}_t)$. As illustrated in Fig. 5, $\mathbf{q}_{t'}$ are first located in the neighboring image $I_{t'}$ using $\mathbf{q}_t$ and their local motion. Using the geometric relationship between $\mathbf{q}_t$ and $\mathbf{p}_t$, $\mathbf{p}_{t'}$ are tentatively determined in $I_{t'}$. Using $\mathbf{p}_{t'}$ and $\mathbf{q}_{t'}$, we measure the color similarity by $w(\mathbf{p}_t, \mathbf{q}_t) = 1/\{\text{ColorDistance}(I_{t'}(\mathbf{p}_{t'}), I_{t'}(\mathbf{q}_{t'})) + \epsilon\}$, where ϵ is a small value for avoiding division by zero. In this way, the weight factor is computed using the color similarity, and the motion value computed by Eq. (7) is assigned to $\mathbf{p}_t$. We are currently using the l^2-norm for the color difference in RGB space for the sake of computation speed, but a different measure could alternatively be used.

The actual scanning and composition in the missing area $\mathcal{M}$ is done using the Fast Marching Method (FMM) [18] as described by [19] in the context of image inpainting. Let $\partial\mathcal{M}$ be the group of all boundary pixels of missing image area M (pixels which have a defined neighbor). Using FMM, we are able to visit each undefined pixel only once, starting with pixels of $\partial\mathcal{M}$, and advancing the boundary inside $\mathcal{M}$ until all undefined pixels are assigned motion values as shown in

Fig. 5. The pixels are processed in ascending distance order from the initial boundary $\partial\mathcal{M}$, such that pixels close to the known area are filled first. The result of this process is a smooth extrapolation of the local motion flow to the undefined area in a manner that preserves object boundaries with color similarity measure.

3. Mosaicing with local warping Once the optical flow field in the missing image area $\mathcal{M}_t$ is obtained, we use it as a guide to locally warp $I_{t'}$ in order to generate a smooth mosaic even including moving objects.

$$I_t(\mathbf{p}_t) \leftarrow I_{t'}(\mathbf{T}_t^{t'}(\mathbf{F}_t^{t'}\mathbf{p}_t)). \tag{8}$$

If some missing pixels still exist in I_t, the algorithm goes back to Step 1 and uses the next neighboring frame.

After the loop of Steps 1~3, usually all missing pixels are filled; however, it is possible that there still remain missing image pixels which are not covered by warped mosaics. Such image areas are considered to be small; therefore, we simply apply a blur filter to fill up the areas. Richer methods such as non-parametric sampling [20, 7] or diffusion methods can also be used to produce higher quality completion than blurring, with additional computational cost.

4. Image Deblurring

After stabilization, motion blur which is not associated to the new motion of the video becomes a noticeable noise that needs to be removed. As mentioned in Sec. 1, it is usually difficult to obtain accurate PSFs from a free-motion camera; therefore, image deblurring using deconvolution is unsuitable for our case. In order to sharpen blurry frames without using PSFs, we developed a new interpolation-based deblurring method. The key idea of our method is transferring sharper image pixels from neighboring frames to corresponding blurry image pixels.

Our method first evaluates the "relative blurriness" of the image which represents how much of the high frequency component has been removed from the frame in comparison to the neighboring frames. Image sharpness, which is the inverse of blurriness, has been long studied in the field of microscopic imaging where accurate focus is essential [15, 16]. We use the inverse of the *sum of squared gradient measure* to evaluate the relative blurriness because of its robustness to image alignment error and computational efficiency. By denoting two derivative filters along the x- and y-directions by f_x and f_y respectively, the blurriness measure is defined by

$$b_t = \frac{1}{\sum_{\mathbf{p}_t}\left\{\left((f_x \star I_t)(\mathbf{p}_t)\right)^2 + \left((f_y \star I_t)(\mathbf{p}_t)\right)^2\right\}}. \tag{9}$$

This blurriness measure does not give an absolute evaluation of image blurriness, but yields relative image blurriness among similar images when compared to the blurriness of other images. Therefore, we restrict the measure to be used in a limited number of neighboring frames where significant scene change is not observed. Also, the blurriness is computed using a common coverage area which is observed in all neighboring frames. Relatively blurry frames are determined by examining $b_t/b_{t'}, t' \in \mathcal{N}_t$, e.g., when $b_t/b_{t'}$ is larger than 1, frame $I_{t'}$ is considered to be sharper than frame I_t.

Figure 6: *The result of image deblurring. Top of the image pairs: original blurry images, bottom: deblurred images with our method.*

Once relative blurriness is determined, blurry frames are sharpened by transferring and interpolating corresponding pixels from sharper frames. To reduce reliance on pixels where a moving object is observed, a weight factor which is computed by a pixel-wise alignment error $E_{t'}^t$ from $I_{t'}$ to I_t is used:

$$E_{t'}^t(\mathbf{p}_t) = |I_{t'}(\mathbf{T}_t^{t'}\mathbf{p}_t) - I_t(\mathbf{p}_t))|. \tag{10}$$

High alignment error is caused by either moving objects or error in the global transformation. Using the inverse of pixel-wise alignment error E as a weight factor for the interpolation, blurry pixels are replaced by interpolating sharper pixels. The deblurring can be described by

$$\hat{I}_t(\mathbf{p}_t) = \frac{I_t(\mathbf{p}_t) + \sum_{t'\in\mathcal{N}} w_{t'}^t(\mathbf{p}_t) I_{t'}(\mathbf{T}_t^{t'}\mathbf{p}_t)}{1 + \sum_{t'\in\mathcal{N}} w_{t'}^t(\mathbf{p}_t)} \tag{11}$$

where w is the weight factor which consists of the pixel-wise alignment error $E_{t'}^t$ and relative blurriness $b_t/b_{t'}$, expressed as

$$w_{t'}^t(\mathbf{p}_t) = \begin{cases} 0 & \text{if } \frac{b_t}{b_{t'}} < 1 \\ \frac{b_t}{b_{t'}} \frac{\alpha}{E_{t'}^t(\mathbf{p}_t)+\alpha} & \text{otherwise.} \end{cases} \tag{12}$$

$\alpha \in [0, \infty]$ controls the sensitivity on the alignment error, e.g., by increasing α, the alignment error contributes less to the weight. As it is seen in the weighting factor defined in Eq. (12), the interpolation uses only frames which are sharper than the current frame.

Fig. 6 shows the result of our deblurring method. As we can see in the figure, blurry frames in the top row are well

Figure 7: *Result of video stabilization. Top row: Original input sequence, middle row: stabilized sequence which still has missing image areas, and bottom row: stabilized and completed sequence. The grid is overlaid for better visualization.*

sharpened in the bottom row. Note that since our method considers the pixel-wise alignment error, moving objects are well preserved without yielding ghost effects, which are often observed with simple frame interpolation methods.

5. Results

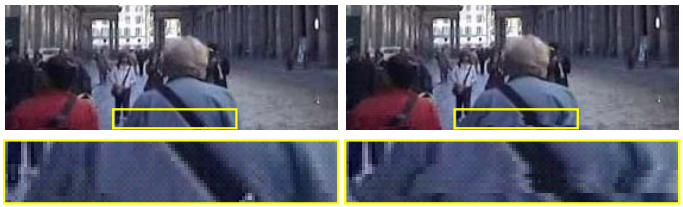
To evaluate the performance of the proposed method, we have conducted extensive experiments on 30 video clips (about 80 minutes in total) to cover different type of scenes. We set the number of neighboring frames to be $2k$=12 throughout the motion smoothing, deblurring, and completion. The computation speed of our current research implementation is about 2 frames per second for a video of size 720×480 with a Pentium4 2.8 GHz CPU without any hardware acceleration. We show the result of video completion in Sec. 5.1. We also show practical applications of our video completion method in Sec. 5.2.

5.1. Video completion results

In our experiment, a 5×5 size filter h is used to perform motion inpainting. In Fig. 7, the result of video stabilization and completion is shown. The top row shows the original input images, and the stabilized result is in the middle row which contains a significant amount of missing image areas. The missing image areas are naturally filled in with our video completion method as shown in the bottom row.

Fig. 8 shows a comparison result. (a) shows the result of our method, and the result of direct mosaicing is shown in (b). As we can see clearly in (b), the mosaicing result looks jaggy on the moving object (since multiple mosaics are used), while our result (a) looks more natural and smoother.

In a synthetic example (Fig. 9), a missing image area is created in one of the input images as shown in the top-middle image, and the video completion method is applied. The bottom row, from left to right, shows (a) the result of our method, (b) the ground truth and (c) the result of direct mosaicing. Our result looks quite similar to the ground truth while the mosaicing method yields noticeable artifacts on the moving sphere.



(a) (b)

Figure 8: *Comparison of completion results. (a) Our method and (b) Mosaicing.*

Fig. 10 shows our video completion results over different scenes. The top-left pair of images shows a successful result in a scene containing a fast moving plane. The top-right pair shows the case of a non-planar scene, and in the left-bottom pair, an ocean wave is naturally composited with our method. Similar to Fig. 8(a), our method accurately handles local motion caused by either moving objects or non-planar scenes.

5.2. Other video enhancement applications

In addition to video stabilization, the video completion and deblurring algorithms we developed in this paper can also be used in a range of other video enhancement applications. We show two interesting ones here: sensor dust removal from a video, caused by dirt spots on the video lens or broken CCD, and overlaid text/logo removal. They can be considered as a problem of filling up specific image areas which are marked as missing. This can be naturally applied to time-stamp removal from a video. In particular, when a stabilizing process is applied to a video, it is essential to remove these artifacts from the video since they become shaky in the final stabilized video. In this experiment we manually marked artifacts as missing image areas. The missing image areas are then filled up by our video completion method.

Figure 10: *Result of video completion over different types of scenes. In each pair, left image is the stabilized image with missing image area (filled in by red), and right image is the completion result.*

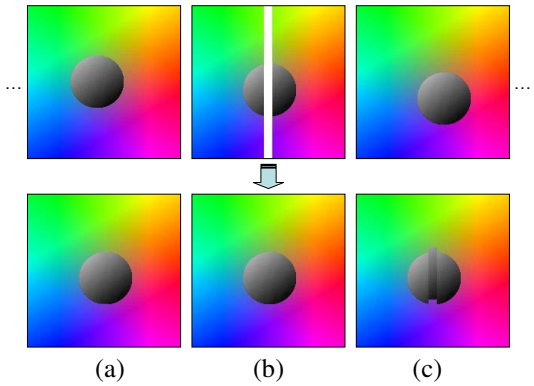


Figure 9: *Result of video completion over a synthetic scene. Top row: input image sequence. White area in the middle image shows the missing image area. Bottom row: (a) result of our method, (b) ground truth and (c) the result of mosaicing.*

Figure 11: *Sensor dust removal. Left: Spots on the camera lens are visible in the original video. Right: The spots are removed from the entire sequence by masking out the spot areas and applying our video completion method.*

Fig. 11 shows the result of sensor dust removal. The left image is a frame from the original sequence, and circles indicate the spots on the lens. The resulting video frames are free from these dirt spots as they are filled up naturally as shown in the right image. Fig. 12 shows the result of text removal from a video. The first row shows the original sequence, and some text is overlaid in the second row. Marking the text areas as missing image areas, our video completion method is applied. The bottom row shows the result of the completion. The result looks almost identical to the original images since the missing image areas are naturally filled up. The absolute intensity difference of the original and result images is taken in Fig. 13(d). The result image is not identical to the original image; however, the difference is small, and more importantly, visual appearance is well preserved.

6. Discussion

We have proposed video completion and deblurring algorithms for generating full-frame stabilized videos. A new efficient completion algorithm based on motion inpainting is proposed. Motion inpainting propagates motion into missing image areas, and the propagated motion field is then used to seamlessly stitch image mosaics. We have also proposed a practical deblurring algorithm which transfers and interpolates sharper pixels of neighboring frames instead of estimating PSFs. The proposed completion method implicitly enforces spatial and temporal consistency supported by motion inpainting. Spatial smoothness of the constructed mosaics is indirectly guaranteed by the smoothness of the extrapolated optical flow. Also, temporal consistency on both static and dynamic areas is given by optical flow from the neighboring frames. These properties make the resulting videos look natural and coherent.

Our method strongly relies on the result of global motion estimation which may become unstable, e.g., when a moving object covers large amount of image area. We are using a robust technique to eliminate outliers; however, it fails when more than half the area of the image is occluded by a moving object. Local motion estimation also has limitations, and may generate wrong results for very fast moving objects. As a result, mosaics might not be warped correctly.

The proposed method has been tested on a wide variety of video clips to verify its effectiveness. In addition, we have demonstrated the applicability of the proposed method for practical video enhancement by showing sensor dust removal and text removal results.

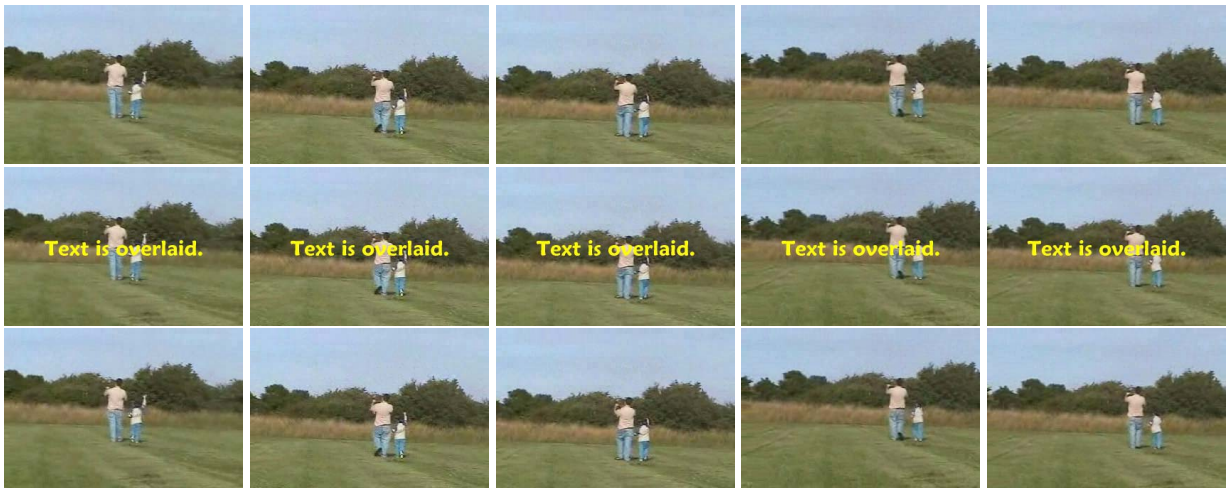


Figure 12: *Result of overlaid text removal from the entire sequence of a video. Top row: original image sequence, middle row: the input with overlaid text, and bottom row: result of overlaid text removal.*

(a) (b) (c) (d)

Figure 13: *Comparison of the ground truth and the text removal result. (a) a frame from the original video, (b) a text is overlaid on the original video, (c) result of text removal, and (d) absolute intensity difference between the original and result frame.*

References

[1] P.A. Jansson, *Deconvolution of Image and Spectra*, Academic Press, 2nd edition, 1997.

[2] R. Fabian and D. Malah, "Robust identification of motion and out-of-focus blur parameters from blurred and noisy images," *CVGIP: Graphical Models and Image Processing*, 53(5):403–412, 1991.

[3] Y. Yitzhaky, G. Boshusha, Y. Levy, and N.S. Kopeika, "Restoration of an image degraded by vibrations using only a single frame," *Optical Engineering*, 39(8):2083–2091, 2000.

[4] B. Bascle, A. Blake, and A. Zisserman, "Motion deblurring and super-resolution from an image sequence," in *Proc. of 4th European Conf. on Computer Vision*, 1996, 2:573–582.

[5] A. Rav-Acha and S. Peleg, "Restoration of multiple images with motion blur in different directions," in *Proc. of 5th IEEE Workshop on Application of Computer Vision*, 2000, pp. 22–28.

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

[7] Y. Wexler, E. Shechtman, and M. Irani, "Space-time video completion," in *Proc. of IEEE Conf. on Computer Vision and Pattern Recognition*, 2004, vol. 1, pp. 120–127.

[8] J. Jia, T.P. Wu, Y.W. Tai, and C.K. Tang, "Video repairing: Inference of foreground and background under severe occlusion," in *Proc. of IEEE Conf. on Computer Vision and Pattern Recognition*, 2004, vol. 1, pp. 364–371.

[9] M. Bertalmio, G. Sapiro, V. Caselles, and C. Ballester, "Image inpainting," in *Proc. of SIGGRAPH*, 2000, pp. 417–424.

[10] A. Criminisi, P. Perez, and K. Toyama, "Object Removal by Exemplar-Based Inpainting," in *IEEE Conf. on Computer Vision and Pattern Recognition*, 2003, vol. 2, pp. 721–728.

[11] H.-Y. Shum and R. Szeliski, "Construction of panoramic mosaics with global and local alignment," *Int'l Journal of Computer Vision*, 36(2):101–130, 2000.

[12] M. Irani and S. Peleg, "Improving resolution by image registration," *CVGIP: Graphical models and Image Processing*, pp. 231–239, 1991.

[13] J.R. Bergen, P. Anandan, K.J. Hanna, and R. Hingorani, "Hierarchical model-based motion estimation," in *Proc. of 2nd European Conf. on Computer Vision*, 1992, pp. 237–252.

[14] J.Y. Bouguet, "Pyramidal implementation of the lucas kanade feature tracker : description of the algorithm," *OpenCV Document, Intel, Microprocessor Research Labs*, 2000.

[15] E.P. Krotkov, "Focusing," *Int'l Journal of Computer Vision*, 1(3):223–237, 1987.

[16] N.F. Zhang, M.T. Postek, R.D. Larrabee, A.E. Vladar, W.J. Keery, and S.N. Jones, "Image sharpness measurement in the scanning electron microscope," *The Journal of Scanning Microscopies*, vol. 21, pp. 246–252, 1999.

[17] J.D. Foley, A. van Dam, S.K. Feiner, and J.F. Hughes, *Computer Graphics: Principles and Practice*, Addison-Wesley, 1996.

[18] J.A. Sethian, *Level Set Methods: Evolving Interfaces in Geometry, Fluid Mechanics, Computer Vision and Materials Sciences.*, Cambridge Univ. Press, 1996.

[19] A. Telea, "An image inpainting technique based on the fast marching method,"*Journal of Graphics Tools*, 9(1):23–34, 2004.

[20] A.A. Efros and T. Leung, "Texture synthesis by non-parametric sampling," in *Proc. of Int'l Conf. on Computer Vision*, 1999, pp. 1033–1038.