

# Motion-based Spatial-Temporal Image Repairing

Wen-Yi Zhao  
Sarnoff Corporation  
201 Washington Road, Princeton, NJ 08540

## Abstract

In this paper, we present a motion-based method for image repairing where images with large missing regions can be restored. The key idea of this method is to reconstruct motion fields for regions of missing data by exploring both temporal and spatial information. After the full motion fields are reconstructed, missing regions can be repaired by the replacement of good corresponding regions from other images. To handle general scenes, we employ dense optical flow as our motion model. To compute flow without correspondence (due to missing pixels), we propose an effective multi-resolution, multi-frame flow method. We demonstrate the efficacy of our method using real images.

## 1 Introduction

This paper presents a method for repairing damaged images based on motion computation/reconstruction. This method addresses an important scenario when neither image inpainting [1, 8] nor image tweening [10] is effective. Figure 1(a) shows such an example when large portion of a damaged image is missing. This often occurs in practice due to the deterioration of storage media (photography films, CDs, files) or transmission error of digital video. In Fig. 1(b) the damaged image is repaired using an algorithm proposed in this paper. The key idea of this method is to reconstruct motion fields for regions with missing pixels. Using reconstructed motion fields, we can effectively repair the missing regions of the damaged image by warping corresponding good regions from other images.

If we can restrict motions among images to be parametric form, then it is trivial to compute motion fields using remaining valid pixels. Unfortunately, such restriction is not realistic. For example, the three images used in Fig. 1 undergo significant local motions especially in the missing region. This is clearly illustrated in Fig. 1(c) where the two warped images based on affine model are significantly different. In this paper, we adopt dense optical flow as our motion model. Hence, the key technical challenge is how to compute optical flow in regions of missing pixels. To solve the problem of *computing flow without correspondence* (due to missing pixels), we propose an effective multi-resolution, multi-frame flow method. If we view an



(a) Input image with significant missing data



(b) Repaired image based on motion reconstruction



(c) Overlaying two other frames based on affine motion

**Figure 1:** Motion-based image repairing/restoration: (a) Damaged image with significant missing data; (b) Repaired image using proposed algorithm with three images including the damaged one (the white rectangle indicates an area where local motions dominate); (c) To show that parametric motion model is not sufficient to describe complex motion, we warp the other two images based on affine motions and overlay them onto the coordinate system of the damage image. To save space, we only plot the missing region. Notice that small arrows indicate places where actual motions significantly deviate from the affine model due to independent object motions. Please note that image inpainting is not applicable here. For comparison with image tweening, please see Fig. 2.

image sequence in a 3D coordinate system  $(x, y, t)$ , then our method takes all spatial and temporal information to repair damaged images while image inpainting ignores the  $t$ -axis and image tweening ignores the  $(x, y)$  plane. Unlike existing space-time flow methods that enforce smoothness constraint of motion in the 3D coordinate system, the proposed method does not impose such constraint. As a result, we are able to handle general cases where the image frames are not equally spaced in terms of motion.

The remainder of this paper is organized as follows: After discussing related work in Section 2, we propose the flow-based image repairing method to restore damaged images by exploring both temporal and spatial information.

Experimental results are reported in Section 4. Finally, we conclude our paper in Section 5.

## 2 Related work

The task of image repairing/restoration is to restore damaged images based on available information. There exist three conditions for image repairing: I) Only one damaged image is available; II) The whole image to be restored is missing in a sequence of images; III) Regions of some or all images are missing in a sequence of images. The specific tasks under the first two conditions are referred to as image inpainting and image tweening. We will call the task under the third condition image repairing.

*Inpainting* or *retouching* is used by museum restoration artists to describe the process of reconstituting the missing or damaged portions of an artwork. It was borrowed in [1] to describe an image processing task that has applications ranging from restoration of damaged photographs to removal/replacement of selected objects. Inspired by an earlier work [7] on level line based disocclusion, the authors in [1] propose a PDE-based method that propagates the structure information (e.g., the change of 2D Laplacian) from surrounding areas into missing regions along the direction of smallest spatial change. The colors inside the missing regions are obtained via anisotropic image diffusion that preserves the structure information (e.g., contour lines). Though very good results are obtained, texture information was not used. To utilize texture information, authors in [8] propose a method that first performs texture-based segmentation and extrapolates the resulting partitioning curves into image holes, and then uses tensor voting to infer the most suitable pixel value in the image holes. As illustrated in [8], image inpainting methods have limitations in recovering large regions when low-level information such as contour, texture boundary is not sufficient enough to infer missing data. In this paper, we propose an algorithm to address this problem by exploring redundant information from a sequence of images.

Image tweening automatically creates in-between frames given two end frames. As a result, it has been widely used in industry for efficiently creating animation [10]. To create in-between frames, the motion field between the two end frames is first computed and then divided into segments to represent motion fields from one end frame to the first in-between frame, the second in-between frame etc. Finally, in-between frames are created by warping the two end frames and blending warped pixels. To distinguish this technique from image morphing (shape tweening), it is also called motion tweening. One fundamental assumption of image tweening is that the whole in-between frames are missing while in our case we assume that we have partial information of every frame. While image tweening works well for image sequences with motion

fields that are linearly distributed across frames, accurate motion reconstruction is generally not possible without exploring the partial information at each frame.

The closest work to our method is the interpolation of missing data in image sequences [4, 5]. In their work, frame motions are first computed as in [6, 2]. Then a method [4] is applied to detect the regions of missing data based on the assumption that they can only happen when both forward and backward image discontinuities are present. On the contrary, occlusion/disocclusion introduces either forward or backward discontinuity, but not both. Motions in the missing regions (called blotches) can be inferred by a Markov Random Field model. Finally, a 3D (space-time) autoregressive model is used to model the image sequence and generate the missing pixels. Though the generative model is very flexible (both motion prior and image prior can be incorporated), the actual pixel synthesis is computationally very expensive. In addition, it is not clear how to generate accurate motion vectors and image pixels for large missing regions. On the contrary, our method explores both temporal and spatial redundancy to directly reconstruct motion fields for missing regions. Nevertheless, it is interesting to see if we can combine the strengths of these two complementary approaches in future.

## 3 Flow-based Image Repairing

### 3.1 Multi-frame consistent flow

Observing that flow consistency should be fully enforced to achieve accurate flow estimate, authors in [9] proposed algorithm to compute consistent flow. A flow estimate is consistent if any pixel warped from frame A to frame B, and then from frame B to frame A goes back to its original location. For computing flow among two frames, traditional flow algorithms [2] seek to minimize the following one-directional least square errors:  $Err_i = (I_i(\mathbf{p}_i) - I_j(\mathbf{p}_i + \mathbf{u}_i[\mathbf{p}_i]))^2$  ( $i = 1, 2$ ), where  $\mathbf{p}_1$  and  $\mathbf{p}_2$  are the coordinates of frame 1 and 2 respectively. The consistent flow algorithm [9] seeks to minimize the consistent least-square error:  $Err_{cons} = [I_1(\mathbf{p} - \alpha\mathbf{u}[\mathbf{p}]) - I_2(\mathbf{p} + (1 - \alpha)\mathbf{u}[\mathbf{p}])]^2$ , where  $\mathbf{p}$  is the coordinate of a virtual frame and  $\alpha$  is a control parameter that is in the range of [0,1]. The choice of the exact value for  $\alpha$  depends on the noise ratio of the two frames and is set to be 0.5 for typically similar noises.

The consistency concept can be extended into multiple frames, resulting in a multi-frame algorithm that minimizes the consistent least square error [9]

$$\begin{aligned}
 Err_{cons} &= \sum Err_{f2r} + Err_{f2f} \\
 &= \sum_{i \neq r} [(I_i(\mathbf{p} - \mathbf{u}_i[\mathbf{p}]) - I_r(\mathbf{p}))^2 \\
 &\quad + \sum_{i \neq j} (I_i(\mathbf{p} - \mathbf{u}_i[\mathbf{p}]) - I_j(\mathbf{p} - \mathbf{u}_j[\mathbf{p}]))^2,
 \end{aligned} \tag{1}$$

where  $Err_{f2r}$  are the errors between each image and the reference image  $I_r$  and  $Err_{f2f}$  are the errors between a

pairs of images  $(I_i, I_j)$  other than  $I_r$ . However, in this multi-frame formulation, all the images are assumed to be of good quality. To handle the case where some images have missing blocks is the topic of the next section.

### 3.2 Flow-based image repairing

Our problem can be stated as follows: *given a sequence of images and assuming some or all images have good and missing regions, we want to repair all damaged images.* To solve this problem, we propose the following two-step approach:

- Motion reconstruction for missing regions
- Interpolating pixels in missing regions based on recovered motion fields

Of these two steps, the reconstruction of motion fields for missing regions is the key and will be the topic of the next section.

#### 3.2.1 Motion reconstruction

In the following discussions, we assume that the information about the missing regions is known. For compressed video, such information is readily available. For detection of such regions, we refer readers to [4] and similar papers for details. Without loss of generality, we further assume that we have three frames,  $I_1$ ,  $I_2$  and  $I_3$ . After  $I_2$  is chosen as the reference frame, the two flow fields  $\mathbf{u}_1$ ,  $\mathbf{u}_2$  are from  $I_1$  to  $I_2$  and  $I_3$  to  $I_2$  respectively. Now the problem becomes *computing* flow without correspondence in the missing regions. Before we propose our algorithm, we would like to review the existing multi-frame flow algorithms (Eq. 1) in detail. Recall that the computation of flow fields are based on iteratively solving the following linear matrix equation [9]

$$\begin{bmatrix} 2 \sum (\nabla I'_1)(\nabla I'_1)^T & - \sum (\nabla I'_1)(\nabla I'_3)^T \\ - \sum (\nabla I'_3)(\nabla I'_1)^T & 2 \sum (\nabla I'_3)(\nabla I'_3)^T \end{bmatrix} \begin{bmatrix} \delta \mathbf{u}_1 \\ \delta \mathbf{u}_3 \end{bmatrix} = \begin{bmatrix} I_{t1} \nabla I'_1 + I_{t13} \nabla I'_1 \\ I_{t3} \nabla I'_3 + I_{t31} \nabla I'_3 \end{bmatrix} \quad (2)$$

where  $I'_i$  are the warped version  $I_i^w$  of  $I_i$  using motion from previous iteration,  $\delta \mathbf{u}_1$  and  $\delta \mathbf{u}_3$  are the incremental flows computed at each iteration.  $I_{tij} \stackrel{\text{def}}{=} I'_i - I'_j$  is equivalent to  $-I_{tji}$  and  $I_{tj} \stackrel{\text{def}}{=} I'_j - I_r$ .

Notice that in missing regions, both temporal gradients  $I_{ti}$  and spatial gradients  $\nabla I'_i$  can not be computed. We can either set them to be zero or fill-in these missing regions using relaxation techniques. However, setting the spatial gradients to be zero yields a degenerated equation (Eq. 2) and filling-in missing regions will likely produce an inaccurate one. To overcome this, we propose methods to reconstruct these gradients.

For spatial gradients  $\nabla I'_i$ , the original formula is  $\nabla I_i^w$  and it will not work in the missing regions. One way

around this is the idea of ‘‘borrowing information’’ from close-by reference image  $I_r$ . Hence we can have the following modified formula

$$\nabla I'_i = (m_i^w \nabla I_i^w + m_r \nabla I_r) / (2[m_i^w + m_r]), \quad (3)$$

where  $m_r$  and  $m_i^w$  are (warped) image masks indicating missing pixels for the reference image and warped image  $I_i^w$  respectively. However, this scheme only works if either of images  $I_i$  and  $I_r$  has valid pixels in missing regions. To handle more general case, we extend the idea of borrowing information from neighboring images, this time, to image  $I_j$ . Since  $I_j$  is temporarily further away from  $I_i$  and needs to be warped to the coordinate system of  $I_r$ , we treat it as a secondary information. In summary, a modified formula that can handle general case of missing regions across multiple frames is as follows

$$\nabla I'_i = \begin{cases} m_j^w \nabla I_j^w, & m_i^w = 0 \parallel m_r = 0 \\ 0.5(m_i^w \nabla I_i^w + m_r \nabla I_r) / (m_i^w + m_r), & \text{else} \end{cases} \quad (4)$$

The same idea of borrowing information can be applied to reconstruct temporal gradients. Again, we distinguish the borrowed information based on how close it is to the missing true information. We can discuss the modifications of how to compute  $I_{ti}$  and  $I_{tij}$  separately. But it is more convenient to discuss the combined temporal gradient  $I_{ti} + I_{ij}$  that is actually used in the formula (Eq. 2). We discuss the following scenarios for reconstructing the combined gradient. Assuming that  $I_i$  has missing pixels, we can compute  $I_{tj}$  and approximate  $I_{ti} + I_{ij}$  as follows

$$I_{ti} + I_{tij} \approx -I_{tj} - 2I_{tj}. \quad (5)$$

For justification of such approximation, please see the appendix. Similarly, if  $I_r$  has missing pixels, we can compute  $I_{tij}$  and approximate  $I_{ti} + I_{ij}$  as

$$I_{ti} + I_{tij} \approx I_{tij} / 2 + I_{tij}. \quad (6)$$

In summary, we have the following modified formula to compute the combined temporal gradient

$$I_{ti} + I_{tij} = \begin{cases} I_i^w - I_r + I_i^w - I_j^w, & (m_i^w m_r m_j^w) > 0 \\ 1.5(I_i^w - I_j^w), & m_r = 0 \\ -3(I_j^w - I_r), & m_i^w = 0 \\ 3(I_i^w - I_r), & m_j^w = 0 \end{cases} \quad (7)$$

The above formula is a full extrapolation and only makes sense when the motion estimate is close to the ground-truth. To have a good initialization, we apply a multi-resolution scheme [3] where good initialization can be first achieved at the lowest-resolution and then gradually propagated to higher-resolution. To be more conservative, especially during iterations, we can have approximation values between zero and fully-extrapolated.

## 4 Experiments

We have tested the proposed algorithm on real images under different scenarios. In the first experiment (Fig. 2),

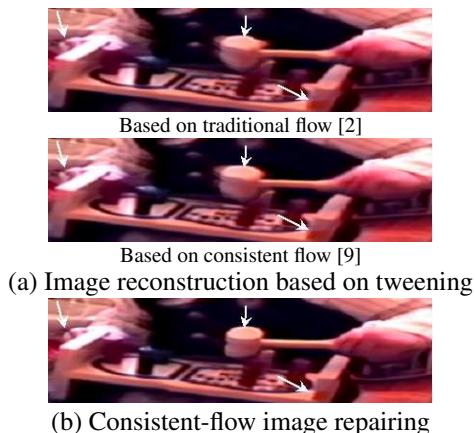


Figure 2: Comparison of image repairing/restoration: (a) Results based on image tweening (notice that consistent flow yields a slightly better results than traditional flow); (b) Results based on flow-based image repairing. Notice that small arrows indicate places where significant differences exist.

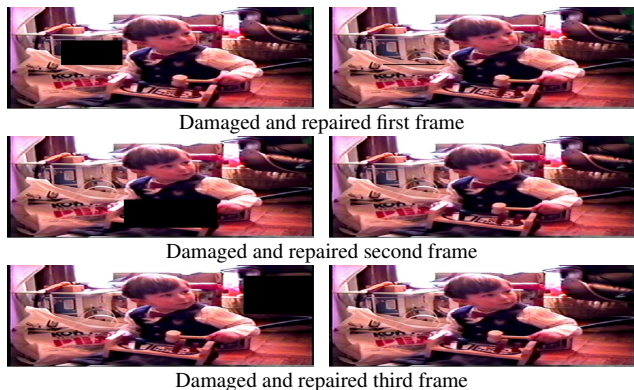


Figure 3: Image repairing result for a sequence of three frames where each frame has missing regions of pixels. In all three row of plots, the left image is the damaged one and the right one is the repaired one.

we demonstrate the efficacy of applying flow-based image repairing for a sequence of three frames where only the middle/reference image has a large missing region (Fig. 1). As mentioned earlier, we must adopt optical flow as our motion model in order to handle this sequence (Fig. 1(c)). For the purpose of comparison, we also applied the image tweening method to infer the reference image assuming it is entirely not available. For easier comparison, we select the middle portion within the missing regions (refer to the white rectangle in Fig. 1(b)) where the local motions dominate; hence, the largest difference in image reconstruction. It should not be surprising that image repairing method (Fig. 2(b)) yields the best result since it has more spatial information available.

In the second experiment (Fig. 3), we show that the proposed method can repair all three images in a sequence of three frames where all three frames have missing regions (Fig. 3).

## 5 Discussion and Conclusions

We have presented a motion-based spatial-temporal image repairing algorithm for image restoration. This algorithm effectively addresses one scenario that has not been addressed by image inpainting or image tweening. The efficacy of this algorithm has been demonstrated with experiments using real images.

For future research directions, we plan to seek a unified image restoration framework that includes image inpainting, tweening and repairing as special cases. It is also of our interest to investigate the benefit of combining the generative modeling approach [5] with the algorithm proposed in this paper.

### A Approximation of temporal gradients

As in the derivation of the basic motion equation [6], we assume the constant brightness constraint and a first-order Taylor expansion

$$I_i^w(\mathbf{x}) \approx I_r(\mathbf{x} + \nabla \mathbf{x}) \approx I_r(\mathbf{x}) + \nabla I_r(\mathbf{x}) \nabla \mathbf{x}^T. \quad (8)$$

In addition, for warped images that are becoming closer and closer, we have the following assumption

$$I_i^w(\mathbf{x} - \nabla \mathbf{x}) \approx I_r(\mathbf{x}) \approx I_j^w(\mathbf{x} + \nabla \mathbf{x}). \quad (9)$$

Based on these two assumptions, we can easily derive the following formulae

$$I_i^w(\mathbf{x}) - I_r(\mathbf{x}) \approx \nabla I_r(\mathbf{x}) \nabla \mathbf{x}^T, \quad (10)$$

$$I_j^w(\mathbf{x}) - I_r(\mathbf{x}) \approx -\nabla I_r(\mathbf{x}) \nabla \mathbf{x}^T, \quad (11)$$

and

$$I_i^w(\mathbf{x}) - I_j^w(\mathbf{x}) \approx I_r(\mathbf{x} + \nabla \mathbf{x}) - I_r(\mathbf{x} - \nabla \mathbf{x}) = 2\nabla I_r(\mathbf{x}) \nabla \mathbf{x}^T. \quad (12)$$

## References

- [1] M. Bertalmio, G. Sapiro, C. Ballester, and V. Caselles, "Image Inpainting", *ACM Siggraph 2000*, pp. 417-424.
- [2] J. Bergen, P. Anandan, K. Hanna, and R. Hingorani, "Hierarchical Molde-Based Motion Estimation," In *Proc. European Conf. Comp. Vision*, pp. 237-252, 1992.
- [3] P. Burt and E. Adelson, "The Laplacian Pyramid as A Compact Image Code," *IEEE Trans. on Communications*, Vol 31, pp.583-540, 1983.
- [4] A. Kokaram, R. Morris, W. Fitzgerald, and P. Rayner, "Detection of Missing data in image sequences," *IEEE Trans. Image Processing*, 11(4), pp. 1496-1508, 1995.
- [5] A. Kokaram, R. Morris, W. Fitzgerald, and P. Rayner, "Interpolation of Missing data in image sequences," *IEEE Trans. Image Processing*, 11(4), pp. 1509-1519, 1995.
- [6] B. D. Lucas and T. Kanade. 1981. An iterative image registration technique with an application to stereo vision. In *Proc. 7th Int. Joint Conf. on Art. Intell.*.
- [7] S. Masnou and J.-M. Morel, "Level lines based disocclusion," In *Proc. Int. Conf. Image Processing*, pp. 259-263, 1998. "A Variational Model for Filling-In Gray Level and Color Images,"
- [8] J. Jia and C.-K. Tang, "Image Repairing: Robust Image Synthesis by Adaptive ND Tensor Voting," In *Proc. Int. Conf. Comp. Vision*, 2003.
- [9] W. Zhao and H. Sawhney, "Is super-resolution with optical flow possible?" In *Proc. European Conf. Comp. Vision*, 2002.
- [10] <http://www.virtual-fx.net/tutorials/html/tweening.4.html>