Pacific Graphics (2014)
J. Keyser, Y. J. Kim, and P. Wonka (Editors)

PatchMove: Patch-based fast image interpolation with greedy bidirectional correspondence

S. Saito[†1], R. Sakamoto[2] and S. Morishima[1,3]

[1]Department of Applied Physics, Waseda University, Japan
[2]Yahoo Japan Corp., Japan
[3]Waseda Research Institute for Science and Engineering, Japan

Abstract

In this paper, we present a method for the plausible interpolation of images. This method has several applications, such as for smooth view interpolation, low frame-rate video upsampling, and animation. The central idea is to quickly form dense correspondences using a patch-based nearest-neighbor search method called PatchMatch. However, the conventional PatchMatch method does not always find an accurate correspondence. This means that some patches do not find appropriate counterparts. Our method employs a greedy algorithm and an occlusion handling technique to correct inaccurate correspondences. Furthermore, our texture reconstruction method successfully reduces blurring effects. We demonstrate that our method significantly reduces the computation time required for interpolation, and show that the quality of reconstructed images is almost identical to that of those generated using state-of-the-art methods.

Categories and Subject Descriptors (according to ACM CCS): I.3.m [Computer Graphics]: Image Processing—Image Interpolation

1. Introduction

Plausible image interpolation methods have been proposed for many applications, such as smooth view interpolation [GGSC96, ZKU*04, CSD11, KLS*13], the upsampling of low frame-rate videos [MHM*09], and for animation [Wol90, LWCS96, SRAIS10]. However, generating an intermediate frame from two given images is still a challenging task, because it requires a dense but accurate correspondence between two images. Incorrect correspondences cause undesirable results such as blurring and ghost effects. To prevent visual artifacts, optical flow [LYT11] and texture reconstruction techniques [SCSI08] have been used [MHM*09, SRAIS10, DSB*12]. However, these techniques involve time-consuming iterative steps. Therefore, these state-of-the-art methods are not suitable for interactive applications.

In this paper, we propose a method called PatchMove that efficiently and robustly produces intermediate frames from two given images. The underlying idea is to form correspondences using patches, which are sets of pixels extracted from the images. A comparison of the similarity between the patches from two images enables us to form a plausible correspondence between them, because these patches contain valuable image features. In fact, patch-based synthesis methods have been widely used for image inpainting [SCSI08, MWDH14], texture synthesis [WL00, RSK13], morphing [SRAIS10], and interactive image editing [BSFG09, HZW*13]. To rapidly find nearest-neighbor fields (NNFs), we use PatchMatch [BSFG09, BSGF10]. This is an efficient method that reconstructs an image from several patches in $O(mM\log M)$ time for m pixels on each patch and M pixels on an image, and requires $O(M)$ memory. We then obtain plausible interpolated images by linearly moving the patches according to their correspondence.

However, the PatchMatch framework does not always pro-

† shun-1616@moegi.waseda.jp

DOI: 10.2312/pgs.20141261

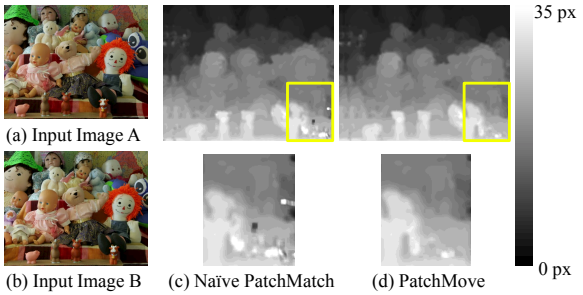


Figure 1: *(a) and (b) show input stereo images from Middlebury's stereo dataset [SP07]. (c) shows the transition distance at each pixel based on the nearest-neighbor mapping using PatchMatch. Without greedy optimization, PatchMatch suffers from distortion caused by incorrect correspondences. However, (d) shows that our greedy optimization corrects the correspondences by reducing the bidirectional error.*

duce accurate correspondences. We introduce a correction process for the nearest-neighbor map obtained by PatchMatch so that the position of inaccurate correspondences can be corrected. A greedy approach successfully reduces the bidirectional error in the NNFs computed between two images and rapidly generates accurate correspondences. The efficiency of this greedy algorithm means that PatchMove can be used in interactive applications.

Most of the experimental results produced by our method do not contain visual artifacts (i.e., blurring or ghost effects). However, undesirable results can be obtained when the transition of two patches occurs in an interpolated frame. In particular, large patches can blend in with other patches. To handle this problem, we introduce a texture reconstruction technique. This enables intermediate frames to recover the resolution of the two original images. Even if visual artifacts occur, our texture reconstruction method removes undesirable effects and generates high-quality images.

In summary, PatchMove makes the following contributions:

- **Patch-based Linear Interpolation**: Patch-based linear interpolation allows us to quickly and robustly obtain intermediate frames from two images using the PatchMatch framework.
- **Greedy Modification Algorithm**: Our greedy algorithm rapidly reduces the number of incorrect correspondences produced by PatchMatch, thus reducing annoying visual artifacts.
- **Occlusion Handling**: By taking account of bidirectional errors, our modification algorithm detects occluded regions around the boundaries of input images, and assigns them appropriate transition vectors.
- **Texture Reconstruction**: The texture reconstruction algorithm gives intermediate images of the same resolution as the original images.

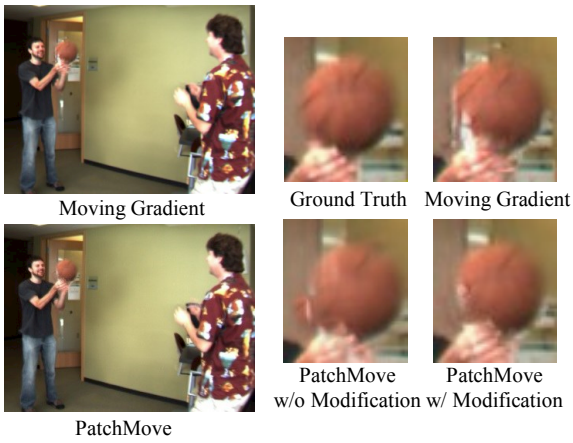


Figure 2: *Middlebury's basketball dataset, available at http://vision.middlebury.edu/flow/. Compared to the Moving Gradient method, PatchMove obtains more plausible intermediate frames. Moreover, whereas PatchMove without greedy optimization produces minor artifacts on the left-hand side of the basketball, PatchMove with greedy optimization successfully reduces these artifacts.*

2. PatchMove

The PatchMove algorithm is divided into three parts. First, the patch-based correspondence between two images, as computed by PatchMatch [BSFG09], generates intermediate frames by linear interpolation. Second, a greedy approach reduces the occurrence of visual artifacts caused by incorrect correspondences, and the bidirectional error metric handles occluded regions along the image boundaries. Finally, our texture reconstruction method removes any blurring effects, and recovers an image of the same resolution as the originals. Note that, in our experiments, only the motion parallax morphing requires texture reconstruction. The PatchMove pseudo-code is described in Algorithm 1.

Algorithm 1 PatchMove( )

```
Initialize NNF;
for e = 0 ⇒ n do
    Propagation based on SSD;
    Random Search based on SSD;
end for
for e = 0 ⇒ n do
    Propagation based on Bidirectional Error;
    Random Search based on Bidirectional Error;
end for
Occlusion Handling;
Linear Interpolation using NNFs;
Texture Reconstruction; //Optional
```

2.1. Patch-based Interpolation

Our patch-based interpolation framework is inspired by PatchMatch [BSFG09]. Given images A and B, PatchMatch

computes the NNF, which is a mapping of every patch in image A to the nearest-neighbor patch in B in terms of some patch distance metric, such as the sum of squared differences (SSD). Intermediate frames are linearly interpolated using the bidirectional nearest-neighbor offsets, i.e., from one image to the other and vice-versa. When several patches overlap at a pixel, that pixel is assigned the average value of all votes from the overlapping patches. Bidirectional interpolation prevents regions being neglected, and enables intermediate frames to be obtained for occluded regions. Although patch-based interpolation can handle most occluded regions, visual artifacts may occur in stereo-view interpolation if an image is partly cut-off. In Section 2.2.2, we present an occlusion handling technique for such situations. Although optimized NNF search methods based on PatchMatch, such as PatchMatch Belief Propagation [BRFK13] and Propagation-Assisted KD-Trees [HS12], could improve the accuracy and speed of image interpolation, we employ the original PatchMatch because of its simplicity.

2.2. Offset Modification

2.2.1. Greedy Optimization

PatchMatch does not always find a perfect correspondence, because patch-based metrics do not consider relative coordinates. That is, if we find the nearest-neighbor patch in a different region, there is no way to tell whether it is matched in terms of the relative position. As shown in the bottom-middle panel of Fig. 2, incorrect correspondences cause visual artifacts. To improve the correspondence, we assume that the error between the mappings (i.e., from one image to the other and vice versa) will be small if the correspondence is correct.

The bidirectional error e of the offset at point p between image A and image B is defined as

$$e^p = v^p_{AB} + v^{p+v^p_{AB}}_{BA}. \quad (1)$$

If $\|e\|$ is sufficiently large, at least one direction has an incorrect correspondence. A greedy algorithm reduces the bidirectional error in entire sets of correspondences, and is similar to the propagation and random search processes in PatchMatch.

The greedy optimization process also follows the PatchMatch framework because of its well-documented efficiency. First, we compare the bidirectional error of each patch and its similarity to adjacent patches in the scan order (from left to right, top to bottom). Let $p' = (x-1, y)$, $p'' = (x, y-1)$ be the patches to the left of and above $p = (x, y)$ in image A. We choose an offset that minimizes $\{\|e^p\|, \|e^{p'}\|, \|e^{p''}\|\}$ and ensures that the distance between the patch at p in image A and its counterpart in image B is smaller than the current patch distance. Taking into account the patch similarity, the greedy method prevents the correspondence between patches from worsening. A random search ensures the NNF can escape from local minima. As shown in Fig. 1, PatchMove with greedy optimization successfully reduces the occurrence of small artifacts.

2.2.2. Occlusion Handling

Although greedy optimization reduces the number of incorrect correspondences, our patch-based method cannot find appropriate counterparts for occluded patches, because the nearest-neighbor mapping is based on a one-to-one correspondence. An occluded region causes undesirable visual artifacts, such as unnatural deformation and blurring. In particular, the sides of images are likely to be occluded in stereo-view interpolation. As shown in Fig. 3, the left- and right-hand sides are occluded because input image A moves to the right toward image B. Hence, it is impossible to find the correct correspondence in these regions. Fig. 3(c) exhibits unwanted ghost effects caused by incorrect correspondences at the right-hand corner of the image. This difficulty in using patch-based interpolation methods for stereo-view interpolation is also a limitation of other patch-based morphing methods [SRAIS10, DSB*12]. Therefore, we focus on occluded regions partly cut-off by stereo movement.

We introduce a simple yet robust method to handle occlusion around the boundaries of input images. Fig. 4 shows an overview of our occlusion handling technique. Although PatchMove allows patches to overlap, we do not show overlapping patches in this overview for simplicity of presentation. First, Eq. (1) is used to detect an occluded region. Following greedy optimization, the value of $\|e\|$ at each pixel should be (sufficiently close to) zero. However, if there is no appropriate counterpart in the second image, $\|e\|$ will be non-zero, which makes occlusion detection easy. To reduce distortion, the offset vector of an occluded patch is set to the negative offset vector of the corresponding position in the other image. As shown in Fig. 3(d), our occlusion handling successfully reduces visual artifacts.

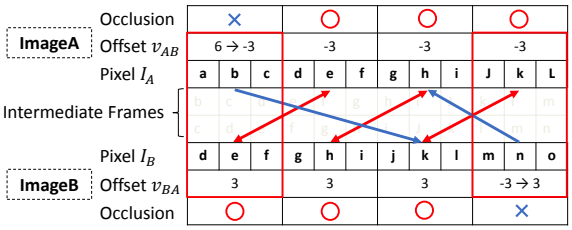
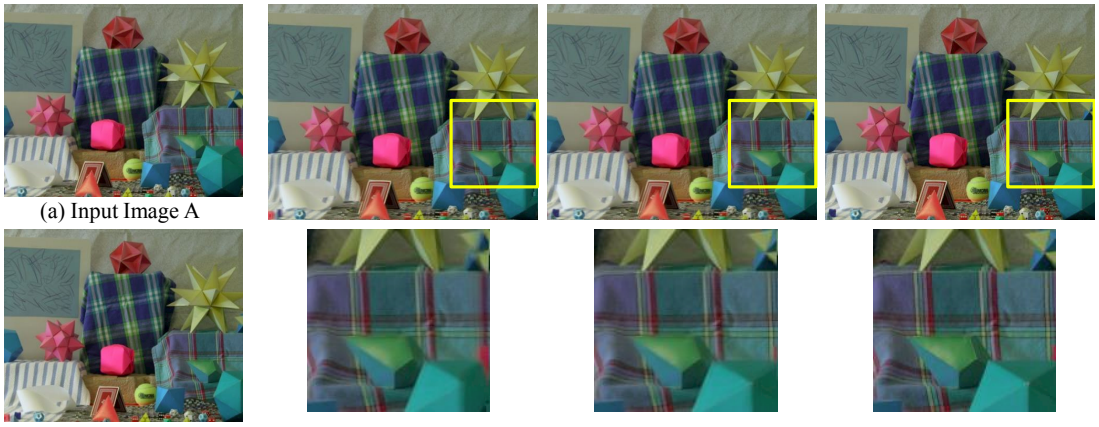


Figure 4: *Overview of occlusion handling. Letters represent the values of pixels. This example shows that the patches at the top-left corner in image A and the bottom-right corner in image B do not have appropriate corresponding pairs, because image A moves to the left toward image B. By taking account of the bidirectional error $\|e\|$, our method quickly detects occluded patches. By negating the offset vector at the same position in the other image, PatchMove successfully obtains appropriate offset vectors for occluded patches.*



(a) Input Image A (b) Input Image B (c) w/o Occlusion Handling (d) w/ Occlusion Handling (e) Ground Truth

Figure 3: *(a) and (b) show the input stereo images from Middlebury's stereo dataset [SP07]. Because each patch in an image requires a counterpart in the other image in a nearest-neighbor search, stereo images have undesirable squeezing effects on the right-hand side of the image (c). On the contrary, our occlusion handling technique obtains plausible interpolated images (d) that are similar to the ground truth (e).*

2.3. Texture Reconstruction

The greedy modification and occlusion handling algorithms successfully remove incorrect transitions in intermediate frames. Blurring effects are negligible in most results. However, with regard to the generation of still images, some intermediate images generated by PatchMove are unsatisfactory. When overlapping patches have very different offset vectors, the RGB values assigned to each pixel are likely to be different. Therefore, the resulting images may lose some fine features of the originals. PatchMove employs bidirectional similarity (BDS) [SCSI08] with two sources to ensure the intermediate frames have as fine a resolution as the input images. Known as α-blended BDS [SRAIS10], this takes into account the *relative similarity* α between two images. Our PatchMove system discards the *Temporal Coherence* term used in the original PatchMatch framework, because patch-based interpolation with our bidirectional greedy modification successfully produces continuous temporal images without using the BDS method. In contrast to Regenerative Morphing [SRAIS10], our texture reconstruction technique does not require a pyramid to be built or an iterative update process, because the greedy modification ensures rapid convergence to a plausible result. Therefore, our texture reconstruction is much faster than the original BDS method with multiple sources.

3. Results

The advantage of our proposed technique over other state-of-the-art methods (i.e., Moving Gradients [MHM*09], Regenerative Morphing [SRAIS10]) is its computational efficiency. Other methods require a large number of iterations to attain convergence. In contrast, our method only needs to computes PatchMatch a few times, which enables intermediate frames to be produced much more quickly. It takes 1.31 s to produce 10 intermediate frames from 320×240 pixel images without texture reconstruction, and texture reconstruction takes 63.98 s using a single-core Intel Core i7 2.40 GHz processor with 16 GB RAM. In our implementation, Moving Gradients and Regenerative Morphing take 115.74 s and 1374.53 s, respectively, for images of the same size.

From the above, it is evident that PatchMove is much more efficient than state-of-the-art methods, even if we include texture reconstruction. If texture reconstruction is not applied, PatchMove is an order of magnitude faster. Note that, in our method, only motion parallax morphing requires texture reconstruction. Further, because our current C++ implementation of PatchMatch does not utilize any CPU or GPU parallelization, there is a strong likelihood that this computation time can be improved.

As shown in Fig. 6 and the supplemental movies, our patch-based interpolation with greedy optimization successfully generates intermediate frames without blurring or ghost effects, similar to state-of-the-art approaches. Moreover, while moving gradients are not applicable to complex movements in which more than two regions are occluded, PatchMove obtains plausible results without blurring (see Fig. 2).

4. Application

PatchMove can be used for a wide range of applications, such as view interpolation, temporal interpolation including complex motion, and non-rigid morphing. In terms of usability, the advantage of PatchMove is that there is no need to adjust its parameters for different types of applications.

4.1. View Interpolation

One of the main applications of PatchMove is view interpolation using two images, such as stereo images. Fig. 3 shows

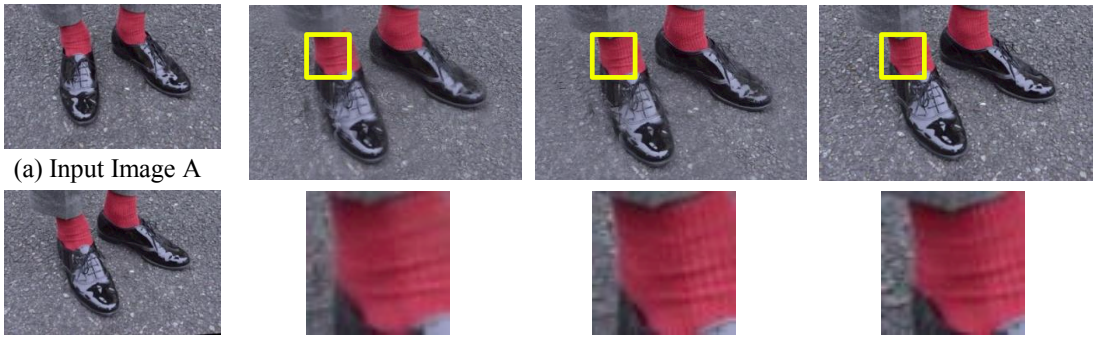


Figure 5: *(a) and (b) show input images with motion parallax. Simple patch-based interpolation causes unwanted blurring and loss of detailed texture, such as in (c). In such a challenging situation, the texture reconstruction algorithm renders an intermediate frame (d) that is as fine as the ground truth (e).*

interpolated frames between two images taken from different points of view. Even if the entire image moves, PatchMove can compute an accurate transition using its greedy optimization and occlusion handling techniques. Comparing the intermediate frame produced using occlusion handling with the ground truth, we see that PatchMove can handle a large stereo transition without blurring or ghost effects. In Fig. 3, the largest transition corresponds to 35 pixels, which is greater than that of [MHM*09] (30 pixels). We used the *Middlebury stereo dataset* [SP07], and took stereo images that were one frame apart as input images to compare with the ground truth, which was the middle frame between the input images.

4.2. Temporal Interpolation

There is considerable demand for temporal interpolation in the mobile industry, because it is an important method for reducing the size of movies. Although the original frame rate must be low to ensure a small file size, viewers require smooth sequences to watch videos without visual stress. The problem is that movie sequences contain complex movement. For example, in Fig. 2 each body part moves differently. Computationally expensive methods are not suitable for interactive applications, even if they produce plausible interpolated frames.

Figs. 2 and 6 show temporally interpolated frames from images including complex movement. Even though PatchMove considers only two images, it obtains a smooth transition without causing undesirable blurring or ghost effects. Moreover, because this temporal interpolation does not require texture reconstruction, PatchMove can be used as a temporal interpolation method for mobile interactive applications.

4.3. Motion Parallax Morphing

Motion parallax morphing is challenging because of its wide transitions and occluded regions. For motion parallax, the focus is not on producing an accurate interpolation, but on plausible morphing. Regenerative Morphing [SRAIS10] requires image features and manual annotation to handle this situation, but Fig. 5 shows that our greedy optimization is able to produce plausible intermediate frames without any manual annotation or other image features. Further, comparing the intermediate frame in Fig. 5(d) with the ground truth in Fig.5(e), we can see that our texture reconstruction technique recovers the fine detail of the input images. Note that Fig. 5 uses the intermediate image as the ground truth. Therefore, even if two images involve significant movement, including motion parallax, PatchMove can produce a smooth transition from one to the other.

5. Conclusions and Future Work

This paper presented a framework for the rapid generation of intermediate frames from two images. The dense correspondence given by PatchMatch enables the successful interpolation of patch transitions. Our proposed offset modification, including greedy optimization and occlusion handling, prevents undesirable transitions that cause blurring and ghost effects. Greedy optimization reduces the bidirectional error caused by incorrect correspondence, and our occlusion detection technique modifies the offset vectors of occluded regions. Even if unwanted blurring occurs, our texture reconstruction method generates intermediate images that are as fine as the input images.

We have not yet implemented our framework on a GPU or introduced parallelization on CPUs. If we can optimize the implementation, it should be possible to further reduce the computation time. For reasons of computational efficiency, we used the original PatchMatch [BSFG09] rather than Generalized PatchMatch [BSGF10]. However, the generalized version has various benefits, such as robustness to rotation and scale changes, which would allow PatchMove to handle challenging interpolations, including images involving

Figure 6: *Comparison of interpolation between PatchMove (red) and Moving Gradient [MHM*09] (green). PatchMove obtains plausible intermediate frames, as given by the state-of-the-art method, without the occurrence of blurring or ghost effects.*

complicated movement. Therefore, in future, we plan to introduce Generalized PatchMatch into our framework.

References

[BRFK13] BESSE F., ROTHER C., FITZGIBBON A., KAUTZ J.: Pmbp: Patchmatch belief propagation for correspondence field estimation. *International Journal of Computer Vision* (2013), 1–12. 3

[BSFG09] BARNES C., SHECHTMAN E., FINKELSTEIN A., GOLDMAN D.: Patchmatch: a randomized correspondence algorithm for structural image editing. *ACM Transactions on Graphics-TOG 28*, 3 (2009), 24. 1, 2, 5

[BSGF10] BARNES C., SHECHTMAN E., GOLDMAN D. B., FINKELSTEIN A.: The generalized patchmatch correspondence algorithm. In *Computer Vision–ECCV 2010*. Springer, 2010, pp. 29–43. 1, 5

[CSD11] CHAURASIA G., SORKINE O., DRETTAKIS G.: Silhouette-aware warping for image-based rendering. In *Computer Graphics Forum* (2011), vol. 30, Wiley Online Library, pp. 1223–1232. 1

[DSB*12] DARABI S., SHECHTMAN E., BARNES C., GOLDMAN D. B., SEN P.: Image Melding: Combining inconsistent images using patch-based synthesis. *ACM Transactions on Graphics (TOG) (Proceedings of SIGGRAPH 2012) 31*, 4 (2012), 82:1–82:10. 1, 3

[GGSC96] GORTLER S. J., GRZESZCZUK R., SZELISKI R., COHEN M. F.: The lumigraph. In *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques* (1996), ACM, pp. 43–54. 1

[HS12] HE K., SUN J.: Computing nearest-neighbor fields via propagation-assisted kd-trees. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on* (2012), IEEE, pp. 111–118. 3

[HZW*13] HU S.-M., ZHANG F.-L., WANG M., MARTIN R. R., WANG J.: Patchnet: a patch-based image representation for interactive library-driven image editing. *ACM Transactions on Graphics (TOG) 32*, 6 (2013), 196. 1

[KLS*13] KOPF J., LANGGUTH F., SCHARSTEIN D., SZELISKI R., GOESELE M.: Image-based rendering in the gradient domain. *ACM Transactions on Graphics (Proceedings of SIGGRAPH Asia 2013) 32*, 6 (2013), to appear. 1

[LWCS96] LEE S., WOLBERG G., CHWA K.-Y., SHIN S. Y.: Image metamorphosis with scattered feature constraints. *Visualization and Computer Graphics, IEEE Transactions on 2*, 4 (1996), 337–354. 1

[LYT11] LIU C., YUEN J., TORRALBA A.: Sift flow: Dense correspondence across scenes and its applications. *Pattern Analysis and Machine Intelligence, IEEE Transactions on 33*, 5 (2011), 978–994. 1

[MHM*09] MAHAJAN D., HUANG F.-C., MATUSIK W., RAMAMOORTHI R., BELHUMEUR P.: Moving gradients: a path-based method for plausible image interpolation. *ACM Transactions on Graphics (TOG) 28*, 3 (2009), 42. 1, 4, 5, 6

[MWDH14] MU T.-J., WANG J.-H., DU S.-P., HU S.-M.: Stereoscopic image completion and depth recovery. *The Visual Computer 30*, 6-8 (2014), 833–843. 1

[RSK13] RUITERS R., SCHWARTZ C., KLEIN R.: Example-based interpolation and synthesis of bidirectional texture functions. In *Computer Graphics Forum* (2013), vol. 32, Wiley Online Library, pp. 361–370. 1

[SCSI08] SIMAKOV D., CASPI Y., SHECHTMAN E., IRANI M.: Summarizing visual data using bidirectional similarity. In *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on* (2008), IEEE, pp. 1–8. 1, 4

[SP07] SCHARSTEIN D., PAL C.: Learning conditional random fields for stereo. In *Computer Vision and Pattern Recognition, 2007. CVPR'07. IEEE Conference on* (2007), IEEE, pp. 1–8. 2, 4, 5

[SRAIS10] SHECHTMAN E., RAV-ACHA A., IRANI M., SEITZ S.: Regenerative morphing. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on* (2010), IEEE, pp. 615–622. 1, 3, 4, 5

[WL00] WEI L.-Y., LEVOY M.: Fast texture synthesis using tree-structured vector quantization. In *Proceedings of the 27th annual conference on Computer graphics and interactive techniques* (2000), ACM Press/Addison-Wesley Publishing Co., pp. 479–488. 1

[Wol90] WOLBERG G.: *Digital image warping*, vol. 10662. IEEE computer society press Los Alamitos, CA, 1990. 1

[ZKU*04] ZITNICK C. L., KANG S. B., UYTTENDAELE M., WINDER S., SZELISKI R.: High-quality video view interpolation using a layered representation. In *ACM Transactions on Graphics (TOG)* (2004), vol. 23, ACM, pp. 600–608. 1