# Image Completion with Variable Scope Patch Sampling

Jing Liu
School of Software Engineering
Beijing University of Technology
dennis-liu@qq.com

Qing Zhu
School of Software Engineering
Beijing University of Technology
ccgszq@bjut.edu.cn

## ABSTRACT

This paper presents a simple but effective way of patch sampling for image completion which changes the assignment scope in different pixels when sampling patches during completion. In each pixel of missing regions, the scope expands from narrow to wide and stop expanding as soon as it randomly find a patch in searching region. In this way, all the sampled patches are almost around the missing regions and each pixel in the regions is assigned an offset value which is corresponds to one of the patches. Traditional methods simply assign the offsets from the whole images and then use algorithms (ANN, etc.) for further process. Our method reduces misleading results and is easier to acquire better patches than many other patch-based methods. Results show that with the same input images, the proposed method obtains better results than others.

## CCS Concepts

• **Computing methodologies → Image manipulation;** Computer vision;

## Keywords

Image completion, image inpainting, patch-based synthesis, approximate nearest neighbor

## 1. INTRODUCTION

Image completion has always being a challenging task. It attempts to remove unwanted objects from images and then fill the missing regions in a visually plausible way. It is also a significant issue in computer graphics, image processing and computer vision. While much progress has been made, the results of image completion are still not always satisfying.

To acquire proper pixels to fill missing regions, example-based methods exploit redundancy in natural images which are based on texture synthesis methods [1, 2]. These methods use structure-based priority [3], deterministic EM-like schemes [4-6], or MRF models with patches as labels, and solve them by using belief propagation [7] or graph cut [8]. Missing regions can also be filled with the help of external image datasets. Retrieving semantically similar images from a large dataset and copying a single large region to fill the missing pixels [9] or transferring the self-similarity field to guide the completion [10] are both scene matching strategies of these methods. In recent years, some patch-based methods are proposed to deal with structural images. They use mid-level structural analysis to guild the low-level synthesis algorithm [11] or integrate with structure-driven technique [12].

However, the increasingly high demand of GPU makes image completion an expensive task while complex procedures are not always effective. Besides, in some situations like painting restoration, it's hard to obtain external image datasets, the only way to fill the unknown regions is by exploiting pixels in the painting itself. As the reference information becomes limited, completion turns more difficult.

In this paper, we propose a simple but effective image completion method with variable scope patch sampling which is based on PatchMatch [6]. Traditional method initializes the offsets of patches in a random way, but we find out that the most useful patches are commonly around the missing regions. So, we use a variable scope for each assignment and narrow the sample scope from the whole image to a variable square, and then propagate the patches for final results. This method is effective for gap filling or large-scale hole-fix in less-texture areas. As the results using traditional methods are sometimes unpredictable (Figure 4 (b) and (c)) and misleading (Figure 5 Row 2), our method significantly reduces these problems and obtains better results.

## 2. COMPLETION

In this section, we present an image completion method using approximate nearest-neighbor algorithm. The core of the method is the algorithm which computes patch correspondences. Define a nearest-neighbor field (NNF) as,

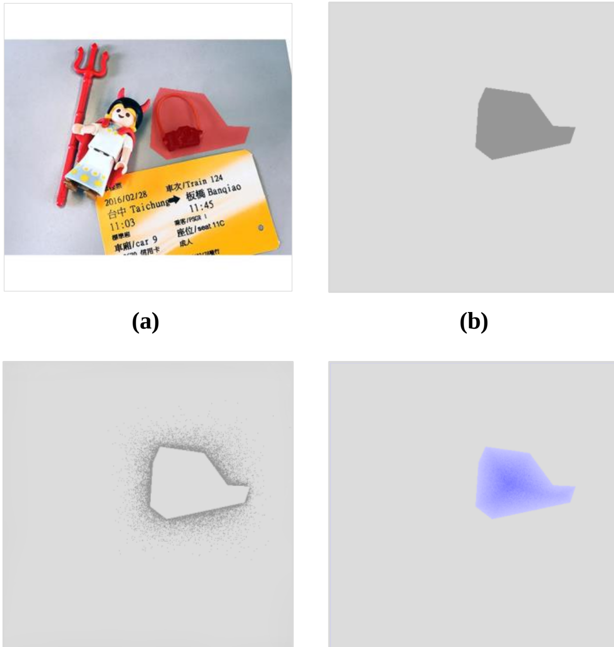
$$f : A \to R^2.$$

The offsets of NNF are defined over all possible patch coordinates in image A (source image). Given patch coordinate $a$ in image A and its corresponding nearest neighbor $b$ in image B (target image), the offset values are simply $b - a$. In this paper, we consider the situation of no extra source image, and the source images we use are all same with the target images.

The algorithm contains three stages. First is initialization. Traditionally, the nearest-neighbor fields are assigned random offsets or prior information. Here, we use our variable scope sampling method to generate some meaningful patches. Next, these offsets are propagated to replace with better results. Traditional methods are always followed by random search which is the third stage in the end and an iterative update process is applied to the algorithm. With the integration of our method, random search becomes optional and one or two iterations would be enough to obtain a good result.


*ICAIR and CACRE '16,* July 13-15, 2016, Kitakyushu, Japan

DOI: http://dx.doi.org/10.1145/2952744.2952746



**(a)** **(b)**

**(c)** **(d)**

**Figure 1: (a) Original image with missing region marked; (b) Image mask; (c) Distribution of sampled patches; (d) The squared L2 distance map.** ***Image credits: Flickr users Jessie.yang.***

## 2.1 Preparation

Before completion, we need to mark two types of regions according to user inputs: hole (missing region) and search (searching region). Here, we use a prescribed color as mask color to highlight user inputs. The user inputs are marked as missing region and the rest of the image are marked as searching region. Figure 1 (b) demonstrates the look of image mask in different gray-scales. Pixels in dark region are marked as hole while the others are marked as search.

## 2.2 Variable Scope Assignment

For initialization, a set of offset values should be assigned in each pixel of missing regions. Instead of initializing from the whole image, we change the assignment scope in different pixels in order to sample patches near the missing regions which are meaningful for filling.

For each pixel in missing regions, we first give it a searching radius $r$ to randomly generate an offset that is in searching region. As it's in a random way, we cannot immediately notice if the assignment scope is not included in the searching region. So we define an assignment time threshold $T$. If assignment time exceeds the threshold, we consider that the current scope is not included in searching region. Then we extend the radius linearly and loop the assignment step until we generate an offset corresponded to searching region.

The method can be illustrated in Figure 2. We define $(x, y)$ as the coordinate of the pixel in missing regions, $(i, j)$ as patch coordinate and $R$ as search region. The initial radius is $r$, and every time the radius extends in a fixed length $d$.

The process goes through following steps.

1) Randomly generate a coordinate $(i, j)$ of pixel $v$ within scope (radius = $r$). If $(i, j)$ satisfy the condition,

$$(-r < i - x < r) \cap (-r < j - y < r) \cap ((i, j) \in S).$$

Go to step 4. Else, go to step 2.

2) Generate a coordinate again and add *1* to counter. If satisfy the condition (described in step 1), go to step 4. Else, redo step 2. When counter reaches $T$, go to step 3.

3) Extend radius $r$ in a fixed length $d$,

$$r = r + d.$$

Set the counter to *0* then go to step 2.

4) Calculate offset value $f(v)$,

$$f(v) = f(i - x, j - y).$$

Save the offset to offset map.

We store the squared L2 distance of each pixel of missing region in a bitmap (Figure 1 (d)). The squared L2 distances are calculated from offsets. From the image we can see that in the edge of the missing region, the color are light as the squared L2 distances are short, which means the corresponded patches are near the edge pixels. As the region goes inside, the distances get longer. Figure 1 (c) shows the distribution of the sampled patch coordinates. Here, we can clearly find out that the sample patches are typically close to the missing region.

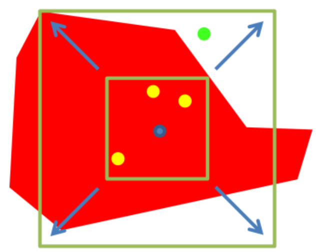


**Figure 2: Illustration on variable scope patch sampling. The missing region is in red, one of the pixels (blue point) is searching for a patch. Yellow points represent the failed patches sampled in the small square scope. Green point represents the successful patch sampled in expanded scope.**

## 2.3 Propagation

After variable scope assignment, we obtain initial offsets of pixels in missing regions which correspond to patches over the image. Using these offsets, we then need to propagate better patches. For each pixel $f\ (x, y)$, assume that $f\ (x - 1, y)$ and $f\ (x, y - 1)$ are nearly same with $f\ (x, y)$. Then, we calculate the two patch distance of both pixels respectively. If one of the two path distances is smaller than the distance of $f\ (x, y)$, we replace the offset of $f\ (x, y)$ with new value. For example, if $f\ (x - 1, y)$ has smaller patch distance, $f\ (x, y)$ is now assigned the offset value of $f\ (x - 1, y)$. Define $d(v)$ the patch distance of pixel $v$, the final distance of the pixel $v$ should be,

$$min\ \{d(f(x, y), d(f(x - 1, y), d(f(x, y - 1)\}.$$

We always iterate for more than one time to obtain better patches. On even iterations we propagate from up and left, and on odd iterations we use down and left.

## 2.4 Additional Processes

In order to obtain better completion results, we need to use random search to further generate an offset that is likely to be a good guess for patches over the whole image. For details of random search, see [6] in references.

Another way to improve completion quality is iteration. Iteration is used for both propagation and random search. In most cases, more than 5 iterations would be enough. Some situations like gap filling or small hole-fix, there is no difference after one time iteration.

# 3. EXPERIMENTS

We test our method in a large amount of images and compare with traditional methods involving no variable scope patch sampling. Here, we choose PatchMatch [6] as traditional method. All the input images are from Flickr users and the missing regions are marked in red. First we test the stability of our method compared with traditional method. The results are shown in Figure 4, as we can see, with same input images, the results are different using traditional method twice (Figure 4 Column 2 & 3) while the results stay almost same using our method (Figure 4 Column 4 & 5).

Figure 3 presents comparison results on watermark removal. We compare our method with traditional method. We use parameters $r = 1$ and $d = 1$ in our method. For fair consideration, we use 2 iterations in propagation stage and no random search in both methods. As we can see in the second row, the traditional method performs not quite well on watermark removal, some letters are still remain clear with a close look, this is because that the sampled patches are from everywhere of the whole image which may generate misleading results. With the integration of our sampling method (Figure 3 Row 3), the watermarks are removed with no imprint remains. Our method sample the right patches with low cost of computing. In other words, compare with traditional method, our method shows the effectiveness on watermark removal.

In Figure 5, we test our method in different shapes of missing regions marked in different images compared with traditional method. With the same inputs (Figure 5 Column 1), we use 2 iterations in propagation and random search stages in both methods. The second column shows the result of traditional method and the results of our method can be seen in Column 4. The results of traditional method are always filled with misleading patches which come from the other part of the images. For example in Row 1, we want to remove the person in front of the horse and replace with the ground, but the traditional method fills the missing region using not only ground patches but also horse patches (Figure 5 (b)). It makes the result not satisfying. On the contrary, our method samples only ground patches and obtains better result (Figure 5 (d)). The other results (Figure 5(f), (j) and (n)) with traditional method also emerge similar problems, especially in Figure 5(f), the windows in the building could not be completely removed while our method performs well in this case and another two images (Figure 5(h), (l) and (p)).

We also compare our method with Photoshop® Content Aware Fill. As shown in the third column of Figure 5, the results are much better than using traditional method, but in images like Figure 5(g) and (o), some unwanted patches show up again. However, our method performs stably in these cases.

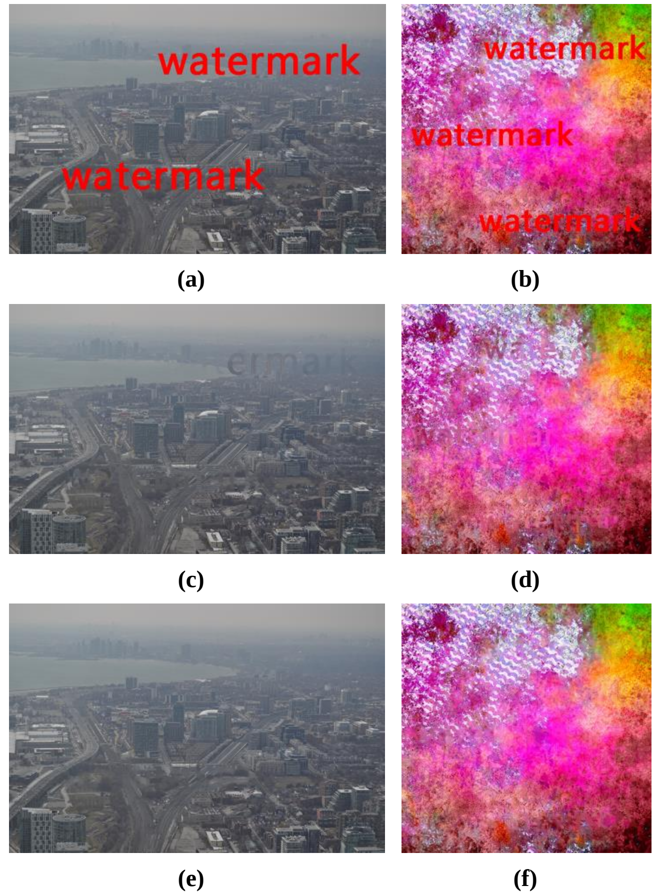



**Figure 3: Input images and results on watermark removal compared with PatchMatch. Row 1: original images with watermarks; Row 2: results of PatchMatch; Row 3: results of our method.** ***Image credits: Flickr users Matt Clare and Phoebe Baker.***

# 4. CONCLUSION

This paper presents an image completion method which integrates variable scope patch sampling. Pixels in missing regions are assigned meaningful patch offsets instead of random ones. This method helps reduce the occurrence of misleading results and sometimes shortens the process time without high demand of GPU. Experimental results demonstrate the effectiveness of the completion method especially for gap filling and object removal. Our method can be well used in quick painting restoration and watermark removal. However, to deal with more complex images or structural images, our method does not perform very well without user guidance.

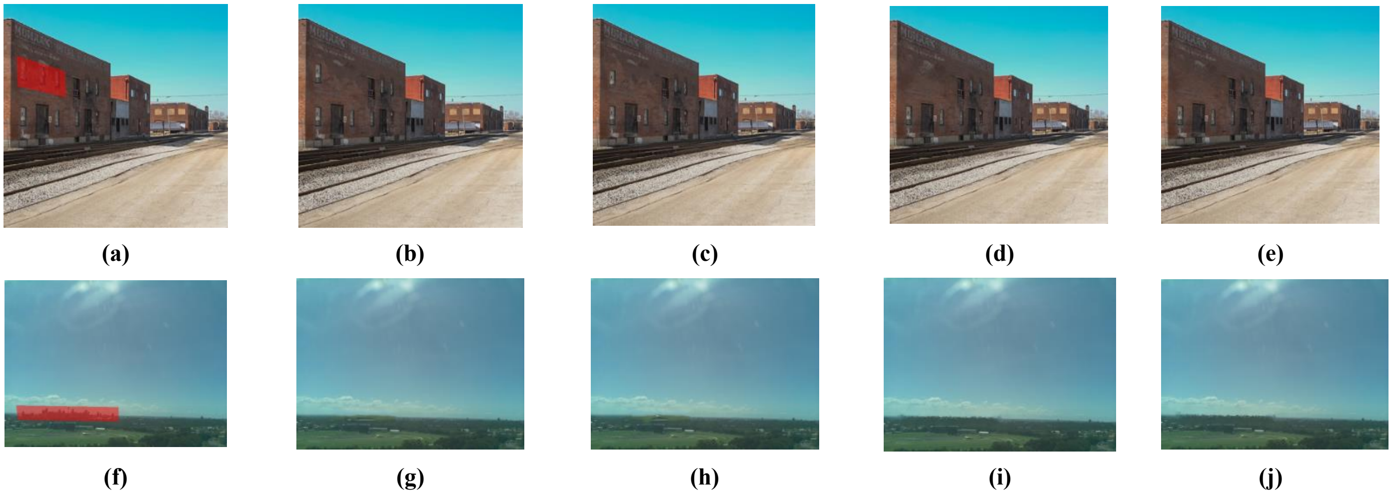



**Figure 4: Stability test. Column 1: original images with missing regions marked; Column 2 & 3: results using PatchMatch twice. Column 4 & 5: results using our method twice. *Image credits: Flickr users Paul Sableman and Climate Change Research Centre.***

## 5. ACKNOWLEDGMENTS

We would like to thank the Flickr users who put their images under Creative Commons license and allowed us to use them. This work was supported by the National Natural Science Foundation of China (Grant No. 4152008).

## 6. REFERENCES

[1] Efros, Alexei A., and T. K. Leung. Texture syntesis by non-parametric sampling. 1999. *The Proceedings of the Seventh IEEE International Conference on* IEEE, 1999:1033.

[2] Alexei A. Efros and William T. Freeman. 2001. Image quilting for texture synthesis and transfer. In *Proceedings of the 28th annual conference on Computer graphics and interactive techniques* (SIGGRAPH '01). ACM, New York, NY, USA, 341-346.

[3] Criminisi, A., Pérez, P., and Toyama, K. 2004. Region filling and object removal by exemplar-based image inpainting. *IEEE TIP* 13.9(2004):1200 - 1212.

[4] Vivek Kwatra, Irfan Essa, Aaron Bobick, and Nipun Kwatra. 2005. Texture optimization for example-based synthesis. *ACM Trans. Graph.* 24, 3 (July 2005), 795-802.

[5] Wexler, Yonatan, E. Shechtman, and M. Irani. 2007. Space-Time Completion of Video. *IEEE Transactions on Pattern Analysis & Machine Intelligence* 29.3(2007):463-476.

[6] Connelly Barnes, Eli Shechtman, Adam Finkelstein, and Dan B Goldman. 2009. PatchMatch: a randomized correspondence algorithm for structural image editing. *ACM Trans. Graph.* 28, 3, Article 24 (July 2009), 11 pages.

[7] Komodakis, N., and G. Tziritas. 2007. Image Completion Using Efficient Belief Propagation via Priority Scheduling and Dynamic Pruning. *IEEE Transactions on Image Processing* 16.11(2007):2649-61.

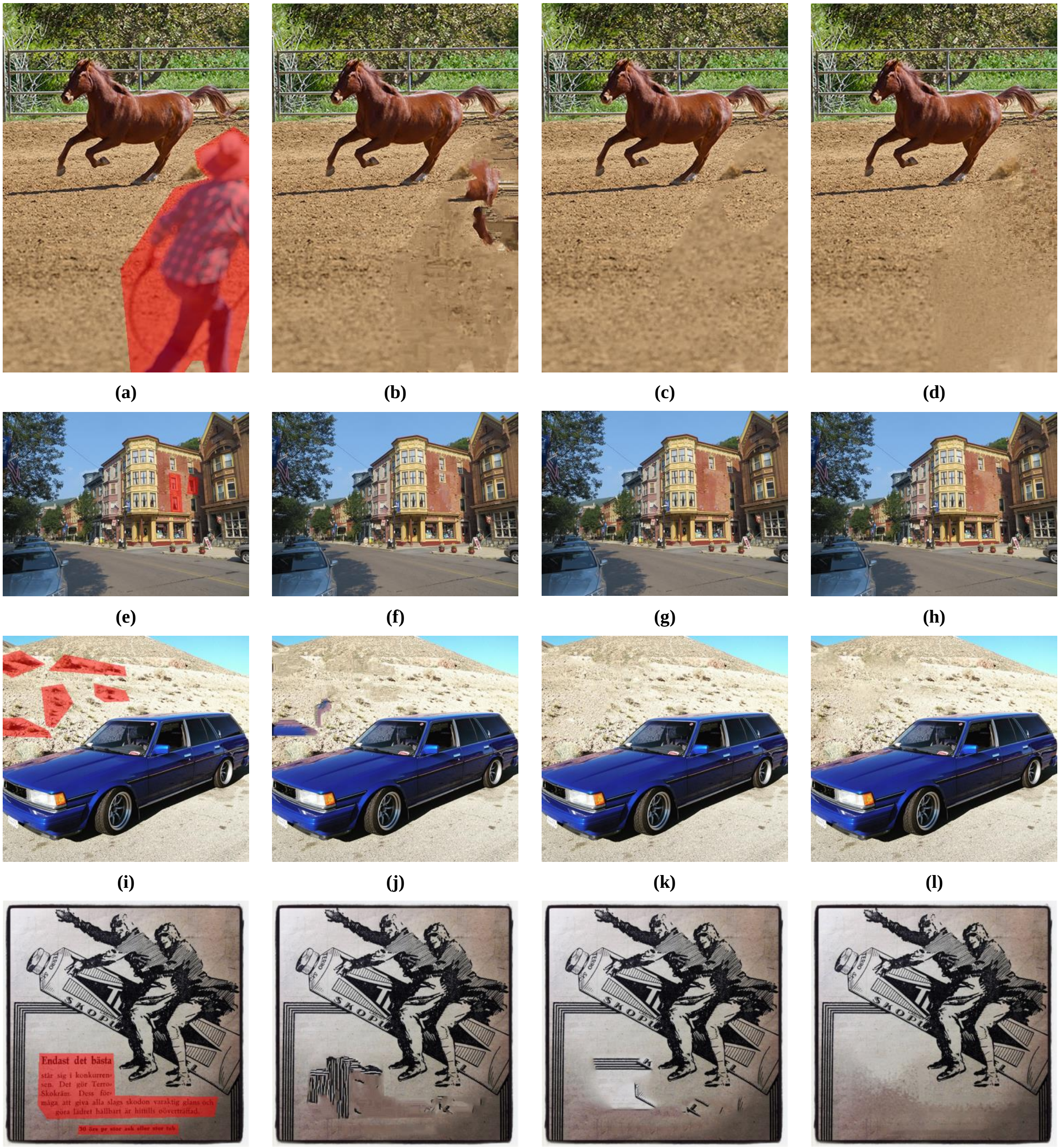
[8] Pritch, Y., E. Kav-Venaki, and S. Peleg. 2009. Shift-map image editing. 30.2(2009):151-158.

[9] James Hays and Alexei A. Efros. 2008. Scene completion using millions of photographs. *Commun. ACM* 51, 10 (October 2008), 87-94.

[10] Zhang, Yinda, et al. 2013. FrameBreak: Dramatic Image Extrapolation by Guided Shift-Maps. *IEEE Conference on Computer Vision & Pattern Recognition* 2013:1171-1178.

[11] Jia-Bin Huang, Sing Bing Kang, Narendra Ahuja, and Johannes Kopf. 2014. Image completion using planar structure guidance. *ACM Trans. Graph.* 33, 4, Article 129 (July 2014), 10 pages.

[12] Hui Huang, Kangxue Yin, Minglun Gong, Dani Lischinski, Daniel Cohen-Or, Uri Ascher, and Baoquan Chen. 2013. "Mind the gap": tele-registration for structure-driven image completion. *ACM Trans. Graph.* 32, 6, Article 174 (November 2013), 10 pages.



(a) (b) (c) (d)
(e) (f) (g) (h)
(i) (j) (k) (l)
(m) (n) (o) (p)

**Figure 5: Comparisons with different methods. Column 1: original images with missing regions marked; Column 2: results of PatchMatch; Column 3: results of Photoshop® Content Aware Fill; Column 4: results of our method.** ***Image credits: Flickr users BorrowedLightPhoto, David Wilson, Moto Club4AG Miwa and Ulf K.***