

# A Scale Consistent Approach to Image Completion\*

Michal Holtzman-Gazit<sup>†</sup> and Irad Yavneh<sup>‡</sup>

November 13, 2008

## Abstract

Most patch based algorithms for completing missing parts of images fill in the absent regions by copying patches from the known part of the image into the unknown part, somewhat like plastic surgery. The criterion for deciding which patch to copy is compatibility of the copied patch with the vicinity of the region being completed. In this paper we propose introducing a new dimension to this compatibility criterion, namely, scale. The patch is thus chosen by evaluating its consistency with respect to a hierarchy of smoothed (less detailed) versions of the image, as well as its surroundings in the current version. Applied recursively, this approach results in a multi-scale framework that is shown to yield a dramatic improvement in the robustness of patch based image completion.

**Keywords:** image completion, image inpainting, scale-consistency, multi-scale

## 1 Introduction

The new age of digital images allows us to take a picture and then alter it by removing an undesired object it contains. The question is then how to complete the missing information so that the image still looks natural. In recent years many new algorithms for this problem have been proposed, and yet the problem remains difficult.

The problem is difficult largely because the term “looks natural” is hard to define mathematically. This difficulty is closely tied with the multi-scale nature of images, especially images containing complex textures. The aim of this work is to develop a systematic approach for addressing this multi-scale structure. The main idea is that the completed image must look natural at *all* (suitably defined) scales of the image. This implies a certain consistency between a completed image, and similarly completed “smoother” versions of the image containing less and less texture details. In effect, an additional dimension—scale—is thus incorporated into the completion process. Employing a given image completion method within a suitable multi-scale framework, that

---

\*This work was supported in part by a research grant from Applied Materials, Inc. and by the Israeli Ministry of Science Culture and Sports in the program of scholarships for women in science

<sup>†</sup>M. Holtzman-Gazit is with the CS department, Technion - Israeli Institute of Technology, Haifa, Israel 32000. Tel: +972-4-8294931. Fax: +972-4-8294898 (email: mikih@cs.technion.ac.il)

<sup>‡</sup>I. Yavneh is with the CS department, Technion - Israeli Institute of Technology, Haifa, Israel 32000. Tel: +972-4-8294945. Fax: +972-4-8293900 (email: irad@cs.technion.ac.il)

imposes this consistency, should therefore be expected to improve its robustness substantially. This is demonstrated in a set of experiments with synthetic and natural images.

Recent studies on image completion may be divided into a few main categories. Inpainting methods are designed to repair images by removing small artifacts such as scratches, small “holes” or overlaid text. These include PDE (partial differential equations) based methods (Ballester *et al.* , 2001; Bertalmio *et al.* , 2000; Chan & Shen, 2001), fast marching (Telea, 2004), diffusion by convolution (Oliveira *et al.* , 2001), and other more complicated methods such as (Elad *et al.* , 2005; Rares *et al.* , 2005; Shih *et al.* , 2003). The main drawback of all such methods is that, in large holes, the data inside the hole is smoothed out. Therefore, they are suitable mostly for small artifacts.

A second approach is using texture synthesis to fill large holes with pure texture, by sampling the texture and generating a large area with the same texture. In (Bonet, 1997; Efros & Leung, 1999; Harrison, 2001; Wei & Levoy, 2000; Hertzmann *et al.* , 2001) a new texture is synthesized pixel by pixel, by looking for similar neighborhoods in the example texture. Other methods (Ashikhmin, 2001; Efros & Freeman, 2001; Kwatra *et al.* , 2003) copy full patches (also called “blocks”) of different sizes and shapes from the source image to the target.

Recently, more complex methods have been designed, mainly for object removal and completion of the complex textures behind it. These variants include techniques such as different orders of filling (Harrison, 2001; Zhang *et al.* , 2004), segmentation (Jia & Tang, 2004), image decomposition (Bertalmio *et al.* , 2003), rotation and scaling (Drori *et al.* , 2003), Gaussian pyramids (Cant & Langensiepen, 2003), global consistency measures (Wexler *et al.* , 2004), and user guidance (Sun *et al.* , 2005).

The method we shall be using in our experiments is the algorithm proposed in (Criminisi *et al.* , 2004). This is an effective yet computationally efficient approach for patch based completion. The method uses exemplar based synthesis, where the order in which the missing part of the image is completed is determined by the direction and sharpness of gradients impinging on the boundary of the missing part. Thus, linear structures in the image tend to be continued into the missing region. Although it is relatively simple, it performs well for many examples.

Our aim is to show how to improve the robustness of image completion procedures by incorporating them in a novel scale consistent framework. The basic ideas were introduced by the authors in (Holtzman-Gazit & Yavneh, 2006). However, the realization of these ideas suffered from several drawbacks, which are overcome in this paper as described in detail below. One problem was a high computational cost. Presently, the algorithm costs only a fraction more than the algorithm of Criminisi *et al.* that it is based on, and yet its results are consistently and substantially better. Another drawback in (Holtzman-Gazit & Yavneh, 2006) is sensitivity to inadequate completions of the smoothest version of the image. This is overcome by the new algorithm introduced here, which promotes flow of information from fine to coarse scales.

In section 2 we present the main ideas and underlying assumptions in an abstract form and propose an approach for implementing them. Section 3 describes a specific implementation, along with a detailed description of a completion algorithm. Section 4 presents experimental results, and section 5 is a summary and conclusion.

## 2 Scale-Consistency

### 2.1 The Main Ideas and Notation

Let  $I = I(\Omega) : \Omega \rightarrow [0, 1]^{d \times |\Omega|}$  denote an image on a set of pixels,  $\Omega$ . Here, e.g.,  $d = 1$  for grey-level images and  $d = 3$  for color images. The fact that grey levels and colors are quantized in practice is unimportant for this discussion. Assume that  $\Omega$  is partitioned into two regions:  $\Omega = \Omega_k \cup \Omega_m$ , where  $\Omega_k$  is the subset of pixels where  $I$  is known, while  $\Omega_m$  is the region where  $I$  is missing; see illustration in Fig. 1. Throughout this paper, missing pixels will be set to  $0^d$  (black). An image completion algorithm is a function,  $C : [0, 1]^{d \times |\Omega|} \rightarrow [0, 1]^{d \times |\Omega|}$ , such that  $\mathbf{J} = C(I(\Omega))$  satisfies  $\mathbf{J}(\Omega_k) = I(\Omega_k)$ . That is,  $C$  returns an image that is identical to  $I$  wherever the latter is known. Here and below, we use boldface capital letters to denote completed images. The purpose of  $C$  is to introduce values in  $\mathbf{J}(\Omega_m)$  such that  $\mathbf{J}(\Omega)$  “looks natural”. Evidently, this is a subjective term, which is, to a large extent, the reason why this problem is difficult.

Next, we introduce the notion of *smoothing*. A smoothing algorithm is a function,  $S : [0, 1]^{d \times |\Omega|} \rightarrow [0, 1]^{d \times |\Omega|}$ , such that  $I_S = S(I)$  is a less-detailed version of  $I$  retaining the same set of missing values,  $\Omega_m$ , which may or may not be empty. A simple example is a convolution with a Gaussian (modified such that missing values remain black), though we prefer smoothing algorithms that preserve important features, particularly so-called edges, where the image undergoes sharp changes. Below we shall be using nonlinear diffusion smoothers for this, but other methods may also be used, such as the hierarchical segmentation of (Galun *et al.*, 2003).

The key idea underlying our approach is that a high-quality completion can only be achieved if the smoothing,  $S$ , and completion,  $C$ , approximately commute:

$$C(S(I)) \approx S(C(I)) . \quad (1)$$

This means that if we perform the completion algorithm on the smooth version of the original image, or, alternatively, perform the completion algorithm on the original image and then apply smoothing, we should get a similar result. This similarity is measured by an appropriate norm of the difference between the resulting images. This assumption is motivated by the fact that both sides of (1) represent images that are smooth versions of a naturally completed  $I$ . It compares between the completions of the fine-detailed image,  $I$ , and that of its smooth version,  $I_S$ . If (1) is satisfied in a sense that we define later, we say that the completion is *scale-consistent*. On this basis, we next describe a general scale-consistent patch-based image completion approach.

### 2.2 Patch-Based Scale-Consistent Image Completion – SCIC

We restrict our discussion to patch-based completion methods of the following type. The input is an image,  $I(\Omega)$ , with values missing (blacked out) in  $\Omega_m \subset \Omega$ , but known in the complement,  $\Omega_k$ . The output image,  $\mathbf{J}(\Omega)$ , is initialized by setting  $\mathbf{J} = I$ . At each step of the algorithm, we consider a subset of pixels called a *target patch*,  $p \subset \Omega$ , which overlaps with both the known (or already completed) and missing parts of the image:  $p_m \equiv p \cap \Omega_m \neq \emptyset$ , and  $p_k \equiv p \setminus p_m \neq \emptyset$ . Usually, the shape and size of  $p$  are pre-defined, and the shape is simple, e.g., a square or a disk of pixels. Next, a *source patch*,  $T(p) \subset \Omega_k$  is selected, where  $T$  is one of a family of simple transformations that preserve shape and size—usually just translations, but possibly also some rotations. Then, the

missing portion of the image corresponding to  $p_m$  is completed by setting:  $\mathbf{J}(p_m) \leftarrow \mathbf{J}(T(p_m))$ . Finally, the missing part is reduced by redefining  $\Omega_m \leftarrow \Omega_m \setminus p_m$ . This completes one step of the algorithm. The process is repeated for a new target patch, etc., until  $\Omega_m = \emptyset$ .

Two key decisions are the choices of  $p$  and the corresponding  $T(p)$  at each step. Given a target patch,  $p$ , the source patch,  $T(p)$ , is generally chosen such that  $\mathbf{J}(T(p_k))$  is “as similar as possible” to  $\mathbf{J}(p_k)$ , employing some suitable measure of similarity. We write this criterion as:

$$\mathbf{J}(T(p_k)) \approx \mathbf{J}(p_k). \quad (2)$$

Typically, similarity is measured by some norm of the difference between the two sides of (2). Different algorithms use different measures of similarity, different allowable sets of transformations, and/or additional considerations such as preferring “common” patches over more “exotic” ones, even if the match is not as good. The point here is that, very often, there are several possible target patches that are of comparable quality by a given measure, and the fact that one of these happens to be slightly better than the others is not very compelling compared to other considerations.

The choice of  $p$  at each step is also important. Most algorithms choose a patch whose intersection with  $\Omega_m$  is relatively small (so that  $p_k$  is sufficiently rich for estimating the suitability of  $T(p)$  via Eq.(2).) Another important consideration is the details of the image in the vicinity of  $\Omega_m$ . For example, preference may be given to a region with a strong edge impinging on the boundary of  $\Omega_m$ .

Next we describe the scale-consistent approach, using just two levels of detail for simplicity. Later, we generalize to a multi-scale framework. Given a patch-based completion method,  $C$ , and a smoothing function  $S$ , denote  $I_S = S(I)$ , and  $\mathbf{J}_S = C(I_S)$ . For the latter we use, of course, criterion (2), applied to  $\mathbf{J}_S$ , i.e.,

$$\mathbf{J}_S(T_S(p_k)) \approx \mathbf{J}_S(p_k), \quad (3)$$

where  $T_S(p_k)$  is the source patch selected in the smoothed image.

We would like to complete  $I$  while respecting our key underlying assumption, (1). Since we do not yet know the completed image at this stage, and therefore cannot evaluate  $S(\mathbf{J}(p))$ , which is required for the right-hand side of (1), we approximate it by  $\mathbf{J}_S(T(p))$ . This leads to the scale-consistency criterion,

$$\mathbf{J}_S(p) \approx \mathbf{J}_S(T(p)). \quad (4)$$

Equations (2,3,4) represent three criteria that we would like to satisfy simultaneously. To this end, we first complete a single patch of the smoothed image,  $I_S$ , using the usual completion algorithm,  $C$ . Next, we complete the same target patch in the detailed image,  $I$ , taking both (2) and (4) into account with approximately equal weight as described in the next section. Note that giving much greater weight to (2) would yield the usual completion,  $C(I)$ . On the other hand, giving much greater weight to (4) would mean setting  $T = T_S$ , which completely ignores the detailed image. By employing both criteria equally, we obtain a completion that is scale consistent, while still complying with the rules of the completion algorithm,  $C$ . Finally, we go back and correct  $I_S$  by replacing the existing completion with one using the source patch found in the finer level. Thus, at this point the image segments at both levels of detail have been completed using the same source patch, and this has been done in a scale-consistent manner. Note that in this approach the information flows both upscale and downscale, which turns out to be crucial for achieving robustness.

### 3 A Specific SCIC Algorithm

To implement the ideas proposed in the previous section, we first generate a set of images  $I_0, \dots, I_n$ , with varying levels of detail, using the smoothing function described in section 3.1. Here,  $I_0 = I$ , the input image. Then, we begin with the image containing the least detail,  $I_n$ , and complete a single patch in the missing region using a patch based completion algorithm,  $C$ , which is a modification of the algorithm of (Criminisi *et al.*, 2004) (see subsection 3.2). Next, we progressively complete the same patch in the finer versions, ending with  $I_0$ . For each version but the smoothest— $I_i$  for all  $i < n$ —we construct a completion which is consistent with that of the previous level of detail,  $I_{i+1}$  (see subsection 3.3). Once we reach the finest level, the final source patch is applied at all scales, replacing the ones used previously. The algorithm can be written as follows, with the details described later.

---

**Algorithm 1** SCIC

---

**Input:**  $I_n, \dots, I_0 = I$  images with varying levels of detail, and the missing region,  $\Omega_m$   
**while**  $\Omega_m \neq \emptyset$  **do**  
    Select target patch,  $p$ , at detail level  $i = n$  (see subsection 3.2).  
    In detail level  $i = n$   
    Find the best matching source patch,  $T_n(p)$ , according to subsection 3.2  
    Set  $\mathbf{J}_n(p_m) \leftarrow \mathbf{J}_n(T_n(p_m))$ .  
    **for** detail level  $i = n - 1, \dots, 0$ , **do**  
        Find the best matching source patch,  $T_i(p)$ , according to subsections 3.2 and 3.3.  
        Set  $\mathbf{J}_i(p_m) \leftarrow \mathbf{J}_i(T_i(p_m))$ .  
    **end for**  
    **for** detail level  $i = 0 \dots n - 1$ , **do**  
        Set  $\mathbf{J}_i(p_m) \leftarrow \mathbf{J}_i(T_n(p_m))$   
    **end for**  
    Set  $\Omega_m \leftarrow \Omega_m \setminus p_m$ .  
**end while**

---

#### 3.1 The smoothing function, $S$

We generate the progressively smoother images using Perona-Malik flow (Perona & Malik, 1990) for grey-level images, and Beltrami flow (Spira *et al.*, 2003) for color images. These procedures apply an adaptive filter that preserves the strong edges in the image while progressively smoothing out texture. All the images generated are of the same size.

#### 3.2 The basic completion algorithm, $C$

Our basic completion algorithm is a modification of the algorithm of (Criminisi *et al.*, 2004). The modifications are aimed at improving the basic algorithm, and are unrelated to the scale consistency concept, which is the main contribution of this paper. As described in section 2.2, the method is characterized by the choice of target patch, which determines the order in which the hole is filled,

and the similarity measure, which determines the choice of source patch. We choose the target patch as in (Criminisi *et al.*, 2004), by maximizing the product of two factors: the component of the image gradient tangential to the hole boundary, and the confidence in the patch, which is a function of the relative size of the known region within the target patch,  $|p_k|/|p|$ .

Next, we describe the modifications we introduce in order to improve the performance while also reducing, or even eliminating, user intervention.

**Region Compatible Patch** In (Criminisi *et al.*, 2004) the size of  $p$  is fixed and predetermined by the user. This implicitly assumes that there is some single size that is suitable for every one of the completion steps, and this size must be determined by the user. We prefer to avoid this assumption and reduce user intervention. In order to prevent large-scale errors, we would like to choose a patch which blends in with the surrounding region of our target patch. Therefore, we would like to consider not only the pixels in the patch but also the neighborhood surrounding the patch. However, the surrounding pixels should have a reduced effect on the selection of the patch compared to the values of the patch itself. Therefore, we first search for the best matching source using only the pixels within the patch. Next, we examine the surroundings of the  $K$  best matching source patches thus discovered. For each of them we perform the similarity measure described in 3.2 on the patch together with its surrounding neighborhood. Motivated by (Simoncelli & Olshausen, 2001), we apply Gaussian weighting to the similarity measure between our target patch with its surrounding and the source patch with its surrounding, such that pixels that are distant from the patch have a much smaller effect on our choice.

With this approach, the patch size is still fixed in advance, but independently of the image and of the scale, as the compatibility with the surrounding region thus incorporated takes into account a wide range of scales. In our tests this approach yielded significantly superior results at a very low additional cost.

**Similarity Measure** The choice of source patch,  $T(p)$ , is determined in (Criminisi *et al.*, 2004) by the mean square of  $(\mathbf{J}(p_k) - \mathbf{J}(T(p_k)))$ , which we denote by  $\|\Delta(\mathbf{J}, p_k)\|^2$ . We modify the algorithm slightly by attaching greater weight to the differences in regions that are close to the border of  $p_m$ . The reason for this is that  $p_m$  will be replaced by  $T(p_m)$ , and a large difference along the border may create a visible “seam”. The weight is efficiently calculated using a simple convolution with a  $7 \times 7$  kernel that varies with the distance from the border of  $p_m$ , which is relatively large at the seam but becomes constant a few pixels away from the seam.

### 3.3 The Scale Consistency Measure

We wish to satisfy Equations (2) and (4) with approximately equal weight. Suppose that the target patch  $p$  in image  $I_{i+1}$  has already been completed. For the same target patch in image  $I_i$ , we search for the source patch,  $T(p)$ , that minimizes

$$C_1^{-1} \|\Delta(\mathbf{J}_i, p_k)\|^2 + C_2^{-1} \|\Delta(\mathbf{J}_{i+1}, p)\|^2, \quad (5)$$

using the weighted norms described above. Here,  $p_k$  is the known part of the patch  $p$  and  $C_1$  and  $C_2$  are normalization constants, given by

$$C_1 = \sum \|\Delta(\mathbf{J}_i, p_k)\|^2, \quad C_2 = \sum \|\Delta(\mathbf{J}_{i+1}, p)\|^2, \quad (6)$$

where the sum is taken only over source patches for which  $\|\Delta(\mathbf{J}_i, p_k)\|$  and  $\|\Delta(\mathbf{J}_{i-1}, p)\|$  are not both dominating (hence, clearly inferior). This means that all source patches for which the  $\Delta$  is larger than some other source patch in both images  $I_i$  and  $I_{i-1}$  are ignored when computing the constants  $C_1$  and  $C_2$ , leaving only candidates that may turn out to be optimal. Including all the source patches in the normalization instead would yield irrelevant normalization constants.

### 3.4 Computational Complexity

Although it may seem that using  $n$  versions of detail of the image multiplies the cost of the entire completion by  $n$ , our implementation of the scale consistent algorithm costs only a fraction more than the original algorithm of (Criminisi *et al.*, 2004). The reason for this is that the lion’s share of work invested in the basic completion algorithm is spent in searching for the best source patches, which is carried out over a very large number of different candidate patches in the known part of the image (exhaustive search). The vast majority of these turn out to be poor choices. These hopeless candidates are already determined in the first stage, when we complete  $I_n$ . There is no reason to test them again when we complete images  $I_{n-1}, \dots, I_0$ . Instead, we store the locations of the small percentage of “reasonable” candidates (candidates that provide a good match according to the similarity measure given in section 3.2) identified in image  $I_n$ , and use only them in the search for a best match in all other levels. Due to the scale consistency requirement, a patch that provides a substantially inferior match for the smooth image is extremely unlikely to be the best choice at a more detailed version of the image, and can therefore be ignored.

Additionally, the region compatibility described in section 3.2 is tested for only the  $K$  best matching patches. Thus, even though the computations involve a patch which is significantly larger than the original patch for this part of the search, we find in practice that  $K$  can be less than 1% of all the candidates in the entire search area, and therefore it only has a small impact on the overall computational complexity. To the overall cost one should add the expense of producing the different levels of detail, which depends on the method used.

To summarize, when using, as we do in our tests,  $N = 2.5\%$  best matches for levels  $i = n - 1 \dots 0$ , and using  $K = 0.06\%$  best matches for the region compatibility criterion, the complexity per each level is only about 7% of the complexity of the single-scale completion of one level. Thus, the entire computational complexity is about  $(1 + 0.07n) \times C(I_n)$ , where  $n + 1$  is the number of levels. This adds up to just a fraction more than cost of producing  $C(I_n)$ —the usual completion of a single image. In particular, this implementation is far more efficient than that used in (Holtzman-Gazit & Yavneh, 2006), while yielding improved results.

## 4 Experimental Results

We test the efficient Scale Consistent Image Completion (SCIC) algorithm on both synthetic and natural images, and compare it to our implementation of the original algorithm of (Criminisi *et al.*, 2004). Fig. 2 demonstrates the efficacy of the SCIC approach for two images: a synthetic image comprised of two textures, and a natural color image with a curved fuzzy border between the textures. In the top image, the gradient of the straw is strong, and therefore the Criminisi *et al.* algorithm, applied to the original image, tends to extend such edges rather than the border

between the two textures, which only dominates at smoother versions of the image. In SCIC this is prevented because it clashes with the completions of the smoother images. In the bottom example, the Criminisi et al. algorithm does not extend correctly the curved border between the road and the grass. Using SCIC, this error is prevented.

Before proceeding to further examples of natural images, we show an experiment that attempts to neutralize to a large extent the arbitrariness in the choice of example images. A systematic comparison is performed for a set of experiments on a synthetic image comprised of two different textures separated by a diagonal border. The unknown region is centered on the border; see example in Fig. 3a. We compare three algorithms for 50 equally spaced locations of the hole along the border. For the SCIC algorithm we used 8 levels of detail generated by the Perona-Malik flow (Perona & Malik, 1990). Then, we repeat the experiment smoothing instead by a convolution with a Gaussian, to test the importance of edge-preserving smoothing. Finally, we also apply the original algorithm of Criminisi et al. for the same 50 tests. A trinary quality measure,  $Q$ , is defined, with  $Q = 3$  corresponding to a high-quality completion (essentially perfect),  $Q = 2$  for a completion that is slightly flawed but still good, and  $Q = 1$  for completions with visible defects. The grading is subjective since there is not automatic algorithm which can measure the quality of the completion. Figs. 2b, 2c, and 2d show typical examples of each of these cases.

The results of the  $50 \times 3$  experiments are summarized in table 1. The SCIC algorithm scores a 1 only in 18% of the experiments, and a perfect 3 in 64% of the experiments. The average score for SCIC is 2.46. Our implementation of the algorithm of Criminisi et al., in contrast, scores a 1 in 56% of the experiments, and a 3 in only 8% of the experiments, with an average score of 1.52. We thus find that SCIC exhibits a dramatic improvement in robustness for this set of experiments. Finally, when the Perona-Malik smoother is replaced by a simple isotropic Gaussian filter, the average score is 2.22—a substantial degradation in performance, though still significantly better than the single-level results.

We next test natural images, obtained from (Schaefer & Stich, 2004). The efficient SCIC algorithm employs 6 levels of detail, where the basic patch size is  $11^2$ , and the region compatible patch size is  $41^2$ . Here and in the previous examples, the patch size for the Criminisi et al. algorithm is also  $11^2$ .

In Fig. 4 we show two examples with color images. SCIC performs far better than the single-scale approach, which leaves a “bite” in the squirrel’s tail and what appears like traces of vandalism in the phone booth. In Figure 5 we show a difficult image containing many different patterns at various scales and a large hole. Again we see that, by using our scale consistent algorithm, all patterns and the borders between them are reconstructed well. The algorithm of Criminisi et al. yields a completion that is also mostly quite good, but it is flawed in many locations, leaving interrupted water curves in the fountain, mixing patterns, and introducing a part of the building where there should only be trees. In Fig. 6 we show an image in which we have removed the car parked in front of the building. In the completion result of our algorithm we see that most of the linear edges in the image were completed well, as opposed to the single scale algorithm of Criminisi et al., where the sidewalk line was not completed, and the white marking on the road is completed in the wrong place. Still, there are of course problems that SCIC cannot overcome. We do not use any high level information or user guidance, therefore we are unable to tell which line should be above the other when two lines intersect. Thus, the intersection between the fence wall and the building line is not completed very well by either algorithm.

## 5 Summary

A new concept—scale consistency—is introduced for the solution of image completion problems. Scale consistency is manifest in an approximate commutativity between the processes of smoothing and completion of images. The resulting framework introduces an additional dimension—scale—into the patch-based image completion formalism.

The scale consistent image completion (SCIC) algorithm employs patch based image completion for multiple versions of the image under consideration, with varying amount of detail. Each step of the completion algorithm takes into account the entire hierarchy of images. Eventually, all versions of the image are completed using the same set of patches, yet all satisfy the basic requirements of the underlying patch-based completion algorithm. This approach is implemented, employing an image completion algorithm based on (Criminisi *et al.*, 2004), and the edge-preserving smoothing algorithms of Perona-Malik for grey-level images and Beltrami flow for color images. Results of experiments—both systematic tests with synthetic images and experiments with natural images—demonstrate a very substantial improvement in the robustness of the completions.

The paper also contains technical improvements to the basic patch-based completion algorithm, in particular, so-called region compatibility. This approach allows us to use relatively small patches of fixed size—which tend to allow a better match of details than larger patches—while still taking into account an extended region around the patch, which reduces the chances of large-scale errors.

Although we use several versions of detail of the image, the computational complexity does not grow very significantly. Employing an efficient implementation, we find that the algorithm ends up increasing the computational complexity by just a fraction when compared to a single-image completion. For this moderate cost, we gain very significantly in robustness.

The scale consistent approach can be employed with other patch-based completion algorithms as well. Although the algorithm requires choosing some parameters for efficient implementation, the results have not been sensitive to these parameters. In particular, all the parameters—including patch size—were fixed and remained the same throughout all the tests.

We speculate that the notion of scale consistency may be significant in other applications of image analysis and synthesis. Several research directions testing this hypothesis are currently under way, in particular, multi-modal image registration and fusion.

## References

- Ashikhmin, M. 2001. Synthesizing Natural Textures. *Pages 217–226 of: Proceedings of the 2001 symposium on Interactive 3D graphics*. New York, NY, USA: ACM Press.
- Ballester, C., Bertalmio, M., Caselles, V., Sapiro, G., & Verdera, J. 2001. Texture Mixing and Texture Movie Synthesis Using Statistical Learning. *IEEE Trans. Image Processing*, **10**(8), 1200–1211.
- Bertalmio, M., Sapiro, G., Caselles, V., & Ballester, C. 2000. Image Inpainting. *Pages 417–424 of: SIGGRAPH 2000, Computer Graphics Proceedings*. ACM SIGGRAPH.
- Bertalmio, M., Vese, L., Sapiro, G., & Osher, S. 2003. Simultaneous Structure And Texture Image Inpainting. *In Proc. of Computer Vision and Pattern Recognition*, **2**, 707–712.

- Bonet, J. S. De. 1997. Multiresolution Sampling Procedure for Analysis and Synthesis of Texture Images. *Pages 361–368 of: Computer Graphics Proceedings. ACM SIGGRAPH.*
- Cant, R.J., & Langensiepen, C.S. 2003 (June). A Multiscale Method For Automated Inpainting. *Pages 148–153 of: Proceeding of ESM2003.*
- Chan, T., & Shen, J. 2001. Non-Texture Inpaintings By Curvature-Driven Diffusions. *J. Visual Communication and Image Representation*, **12(4)**, 436–449.
- Criminisi, A., Perez, P., & Toyama, K. 2004. Region Filling And Object Removal By Exemplar-Based Inpainting. *IEEE Trans. Image Processing*, **13(9)**, 1200–1212.
- Drori, I., Cohen-Or, D., & Yeshurun, H. 2003. Fragment-Based Image Completion. *ACM Trans. Graph.*, **22(3)**, 303–312.
- Efros, A., & Freeman, W. 2001. Image Quilting for Texture Synthesis and Transfer. *Pages 341–346 of: SIGGRAPH 2001, Computer Graphics Proceedings. ACM SIGGRAPH.*
- Efros, A., & Leung, T. 1999 (September). Texture Synthesis by Non-parametric Sampling. *Pages 1033–1038 of: ICCV proceedings.*
- Elad, M., Starck, J-L., Querre, P., , & Donoho, D.L. 2005. Simultaneous Cartoon and Texture Image Inpainting Using Morphological Component Analysis (MCA). *Journal on Applied and Computational Harmonic Analysis*, **19(November)**, 340–358.
- Galun, M., Sharon, E., Basri, R., & Brandt, A. 2003. Texture Segmentation by Multiscale Aggregation of Filter Responses and Shape Elements. *Pages 716–723 of: ICCV.*
- Harrison, P. 2001. A Non-Hierarchical Procedure for Re-Synthesis of Complex Textures. *Pages 190–197 of: WSCG.*
- Hertzmann, A., Jacobs, C. E., Oliver, N., Curless, B., & Salesin, D. H. 2001. Image Analogies. *Pages 327–340 of: SIGGRAPH 2001, Computer Graphics Proceedings. ACM SIGGRAPH.*
- Holtzman-Gazit, Michal, & Yavneh, Irad. 2006. Scale Consistent Image Completion. *Pages 648–659 of: ISVC (1). Lecture Notes in Computer Science, vol. 4291. Springer.*
- Jia, J., & Tang, C. 2004. Inference of Segmented Color and Texture Description by Tensor Voting. *IEEE Trans. Pattern Anal. Mach. Intell.*, **26(6)**, 771–786.
- Kwatra, V., Schödl, A., Essa, I., Turk, G., & Bobick, A. 2003. Graphcut Textures: Image and Video Synthesis Using Graph Cuts. *ACM Trans. on Graph.*, *SIGGRAPH 2003*, **22(3)**, 277–286.
- Oliveira, M. M., Bowen, B., McKenna, R., & Chang, Y.-S. 2001. Fast Digital Image Inpainting. *Pages 261–266 of: Proceeding of International Conference on VIIP.*
- Perona, P., & Malik, J. 1990. Scale-Space and Edge Detection Using Anisotropic Diffusion. *IEEE Trans. Pattern Anal. Mach. Intell.*, **12(7)**, 629–639.

Table 1: The quality of the completions of our SCIC algorithm are compared with those obtained using the algorithm of Criminisi et al. on the image of Fig. 3, with the unknown part of the image centered at fifty equally spaced locations along the diagonal. For the SCIC algorithm we test both Perona-Malik (edge-preserving) smoothing and isotropic Gaussian filtering

Q	SCIC (Perona-Malik)	SCIC (Gaussian)	Criminisi
<b>3</b>	64%	40%	8%
<b>2</b>	18%	42%	36%
<b>1</b>	18%	18%	56%
<b>Avg.</b>	2.46	2.22	1.52

- Rares, A., Reinders, M., & Biemond, J. 2005. Edge-Based Image Restoration. *IEEE Trans. Image Processing*, **14**(10), 1454–1468.
- Schaefer, G., & Stich, M. 2004. UCID - An Uncompressed Colour Image Database. *Proc. SPIE, Storage and Retrieval Methods and Applications for Multimedia 2004*, 472–480.
- Shih, T.K., Lu, L.-C., Wang, Y.-H., & Chang, R.-C. 2003. Multi-Resolution Image Inpainting. *Pages 485–8 of: Proceedings of the 2003 International Conference on Multimedia and Expo*, vol. 1.
- Simoncelli, E P, & Olshausen, B. 2001. Natural Image Statistics and Neural Representation. *Annual Review of Neuroscience*, **24**(May), 1193–1216.
- Spira, A., Kimmel, R., & Sochen, N. 2003 (June). Efficient Beltrami Flow using a Short Time Kernel. *Pages 511–522 of: Proc. of Scale Space 2003, Lecture Notes in Computer Science (vol. 2695)*.
- Sun, J., Yuan, L., Jia, J., & Shum, H.-Y. 2005. Image completion with structure propagation. *ACM Trans. Graph.*, **24**(3), 861–868.
- Telea, A. 2004. An Image Inpainting Technique Based on the Fast Marching Method. *Journal of Graphics Tools: JGT*, **9**(1), 23–34.
- Wei, L., & Levoy, M. 2000. Fast Texture Synthesis Using Tree-Structured Vector Quantization. *Pages 479–488 of: SIGGRAPH 2000, Computer Graphics Proceedings*. ACM SIGGRAPH.
- Wexler, Y., Shechtman, E., & Irani, M. 2004. Space-Time Video Completion. *In Proc. of Computer Vision and Pattern Recognition*, **1**, 120–127.
- Zhang, Y., Xiao, J., & Shah, M. 2004. Region Completion in a Single Image. *EUROGRAPHICS*, Aug.