

Fast Digital Image Inpainting

Manuel M. Oliveira Brian Bowen Richard McKenna Yu-Sung Chang

Department of Computer Science
State University of New York at Stony Brook
Stony Brook, NY 11794-4400
{oliveira|bbowen|richard|yusung}@cs.sunysb.edu}

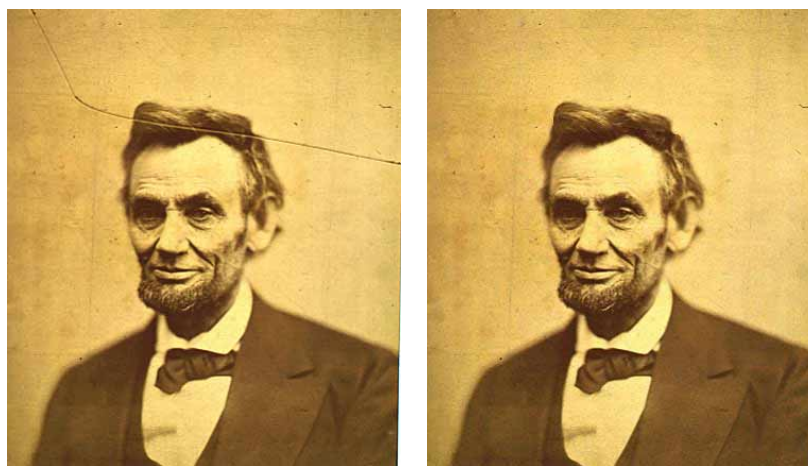


Fig. 1. Left: An 1865 Photograph of Abraham Lincoln taken by Alexander Gardner (courtesy of Wing Yung and Ajeet Shankar from Harvard University). Right: Image restored with our algorithm. The inpainting time took about half of a second.

ABSTRACT

We present a very simple inpainting algorithm for reconstruction of small missing and damaged portions of images that is two to three orders of magnitude faster than current methods while producing comparable results.

KEY WORDS: image inpainting, image restoration.

1. INTRODUCTION

Reconstruction of missing or damaged portions of images is an ancient practice used extensively in artwork restoration. Also known as *inpainting* or *retouching*, this activity consists of filling in the missing areas or modifying the damaged ones in a non-detectable way by an observer not familiar with the original images [2]. Applications of image inpainting range from restoration of photographs, films and paintings, to removal of occlusions, such as text, subtitles, stamps and publicity from images. In addition, inpainting can also be used to produce special effects.

Traditionally, skilled artists have performed image inpainting manually. But given its range of applications, it would be desirable to have image inpainting as a standard feature of popular image tools such as PhotoShop. Recently, Bertalmio et al [2] have introduced a technique for digital inpainting of still images that produces very

impressive results. Their algorithm, however, usually requires several minutes on current personal computers for the inpainting of relatively small areas. Such a time is unacceptable for interactive sessions and motivated us to design a simpler and faster algorithm capable of producing similar results in just a few seconds.

The results produced by our algorithm are comparable to those found in the literature [2, 4, 5], but two to three orders of magnitude faster. We illustrate the effectiveness of our approach with examples of restoration of photographs, vandalized images, and text removal. Figure 1 (left) shows a famous cracked photograph of Abraham Lincoln taken in 1865. The image to its right shows the result obtained with our algorithm in 0.61 seconds on a 450 MHz Pentium III PC.

2. PREVIOUS AND RELATED WORK

Bertalmio et al [2] pioneered a digital image-inpainting algorithm based on partial differential equations (PDEs). A user-provided mask specifies the portions of the input image to be retouched and the algorithm treats the input image as three separate channels (R, G and B). For each channel, it fills in the areas to be inpainted by propagating information from the outside of the masked region along level lines (*isophotes*). Isophote directions are obtained by computing at each pixel along the inpainting contour a

discretized gradient vector (it gives the direction of largest spatial change) and by rotating the resulting vector by 90 degrees. This intends to propagate information while preserving edges. A 2-D Laplacian [8] is used to locally estimate the variation in color smoothness and such variation is propagated along the isophote direction [2]. After every few step of the inpainting process, the algorithm runs a few diffusion iterations to smooth the inpainted region. Anisotropic diffusion [13] is used in order to preserve boundaries across the inpainted region.

Inspired by the work of Bertalmio et al., Chan and Shen proposed two image-inpainting algorithms [4, 5]. The *Total Variational* (TV) inpainting model [4] uses an Euler-Lagrange equation and inside the inpainting domain the model simply employs anisotropic diffusion [13] based on the contrast of the isophotes. This model was designed for inpainting small regions and while it does a good job in removing noise, it does not connect broken edges (single lines embedded in a uniform background) [4]. The *Curvature-Driven Diffusion* (CDD) model [5] extended the TV algorithm to also take into account geometric information of isophotes when defining the “strength” of the diffusion process, thus allowing the inpainting to proceed over larger areas. CDD can connect some broken edges, but the resulting interpolated segments usually look blurry.

While nonlinear PDE-based image restoration methods have the potential to systematically preserve edges, the inpainting problem is very ill posed in general and fast numerical implementations are difficult to achieve [5]. It is equally hard to find appropriate mathematical models for inpainting [5]. Despite their high quality, a careful examination of the results presented in [2] (not reproduced here) reveals that sharp edges are not always preserved. For instance, the reconstructed region where the mask crosses the VW Beetle near the windshield appears blurred with broken edges (Figure 6 (top) in [2]).

Hirani and Totsuke [11] combine global frequency and local spatial information for noise removal and use it for post-production of special effects shots. Such a technique can produce very nice results, but requires the existence of sample sub-images whose contents are approximately translated versions of the regions to be repaired.

Digital techniques have also been used for automatic restoration of scratched films [10], and commercial products are available for scratch removal of digitized films [1], photo retouching [7] and wire-and-rig removal [6, 14].

3. THE INPAINTING ALGORITHM

Images may contain textures with arbitrary spatial discontinuities, but the sampling theorem [8] constraints the spatial frequency content that can be automatically restored. Thus, for the case of missing or damaged areas, one can only hope to produce a plausible rather than an exact reconstruction. Therefore, in order for an inpainting

model to be reasonably successful for a large class of images *the regions to be inpainted must be locally small*. As the regions become smaller, simpler models can be used to locally approximate the results produced by more sophisticated ones. Another important observation used in the design of our algorithm is that the human visual system can tolerate some amount of blurring in areas not associated to high contrast edges [9].

Thus, let Ω be a small area to be inpainted and let $\partial\Omega$ be its boundary. Since Ω is small, the inpainting procedure can be approximated by an isotropic diffusion process that propagates information from $\partial\Omega$ into Ω . A slightly improved algorithm reconnects edges reaching $\partial\Omega$ (for instance, using an approach similar to the one described in [12]), removes the new edge pixels from Ω (thus splitting Ω into a number of smaller sub-regions), and then performs the diffusion process as before.

The simplest version of the algorithm consists of initializing Ω by clearing its color information and repeatedly convolving the region to be inpainted with a diffusion kernel. $\partial\Omega$ is a one-pixel thick boundary and the number of iterations is independently controlled for each inpainting domain by checking if none of the pixels belonging to the domain had their values changed by more than a certain threshold during the previous iteration. Alternatively, the user can specify the number of iterations. As the diffusion process is iterated, the inpainting progresses from $\partial\Omega$ into Ω .

Convolving an image with a Gaussian kernel (*i.e.*, computing weighted averages of pixels’ neighborhoods) is equivalent to isotropic diffusion (linear heat equation). Our algorithm uses a weighted average kernel that only considers contributions from the neighbor pixels (*i.e.*, it has a zero weight at the center of the kernel). Figure 2 shows the pseudocode of the algorithm and two diffusion kernels. All reconstructed images shown in this paper were obtained with this algorithm or with a minor variation of it explained in section 3.1.

3.1. Preserving Edges

The simplest version of the algorithm can introduce artifacts (noticeable blurring) when Ω crosses the boundaries of high contrast edges (Figure 3 (front left)). In practice, intersections between Ω and high contrast edges are the only places where anisotropic diffusion is

```

initialize  $\Omega$ ;
for (iter =0; iter < num_iteration; iter++)
    convolve masked regions with kernel;

```

a	b	a
b	0	b
a	b	a

c	c	c
c	0	c
c	c	c

Fig 2. (top) Pseudocode for the fast inpainting algorithm. (bottom) Two diffusion kernels used with the algorithm. $a = 0.073235$, $b = 0.176765$, $c = 0.125$.

required and such regions usually account for a small percentage of the total area.

Creating the mask used to specify the regions to be inpainted is the most time-consuming step of the inpainting process, requiring user intervention. Since our algorithm can inpaint an image in just a few seconds, it can be used for interactive construction of tight masks. We exploit this interactivity to implement edge reconnection by defining *diffusion barriers*, which are boundaries for the diffusion process inside Ω . This accomplishes a result similar to boundary reconstruction and anisotropic diffusion, but without the associated overheads. In practice, a diffusion barrier is a two-pixel wide line segment. As the diffusion process reaches a barrier, the reached pixel has its color set, but the process stops. Figure 3 illustrates the idea, with the clear crossing lines in Figure 3 (back left) representing the inpainting domain. The simple diffusion-based inpainting algorithm produces blurred spots at the intersections between Ω and high contrast edges (see the small circles in Figure 3 (front left)). By appropriately adding diffusion barriers (line segments across the mask in Figure 3 (back right)), the user stops the diffusion process from mixing information from both sides of the mask. The resulting straight lines are shown in Figure 3 (front right).

4. RESULTS

We have implemented the algorithm described in Figure 2 in C++ and tried the two different diffusion kernels. In both cases the results were similar. All images shown in the paper were generated using a 450 MHz Pentium III PC with 128 MB of memory running Windows98 and using the leftmost kernel shown in Figure 2. Results shown in Figures 5, 8, 9 and 10 were produced with the simplest version of the algorithm without diffusion barriers. For Figure 1, a mask and two diffusion barriers were used (Figure 4). For the three girls example, four diffusion barriers associated with regions of the mask crossing high contrast edges were used (Figure 6 (right)). In all cases, 100 diffusion iterations were used.

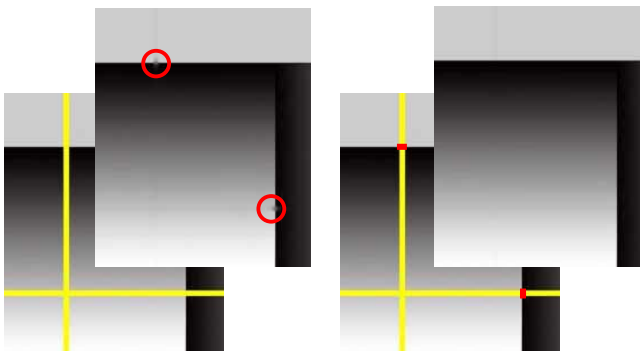


Fig. 3. The crossing lines define the inpainting domain (back left). Result of the isotropic diffusion introduces some blurring along high contrast edges (top left). User-added diffusion barriers (back right). Result produced with diffusion barriers (front right).

All inpainting and wire-and-rig removal systems require a step of manual masking. Given a painting system with a set of features, the time required to create a mask only depends on the available features and is independent of the inpainting algorithm used. For interactive applications, it is desirable to have the masking capabilities and the inpainting algorithm in the same system to avoid switching between different environments. In our current prototype, we have implemented a simple painting system and the ability to import and export JPEG files.

The masks used for restoring Lincoln's portrait and the picture of the three girls (Figures 4 and 6 (right), respectively), were created with our painting system. The mask used in the New Orleans example (Figure 5) was instantly obtained by selecting "red" using the *select color range* feature of Photoshop. The resulting image was saved and imported into our system. The masks used with Figures 8, 9 and 10 were JPEG images containing the corresponding text and scribble shown in these images.

The cost of inpainting is linear on the size of the inpainted region and algorithms are cache intensive. For the example of Lincoln's portrait, the inpainting time of our algorithm was 0.61 seconds. Figures 5 and 6 (left) were used in [2] and obtained from Bertalmio's web site [3]. For the example shown in Figure 6, Bertalmio et al. reported an inpainting time (for one color channel) of approximately 7 minutes, or 2 minutes when a two-level multiresolution approach is used [2]. These times were measured on a 300 MHz Pentium II PC (128 MB of memory running Linux). The image shown in Figure 8 (left) was produced with our algorithm in 1.21 seconds.

Figures 8, 9 and 10 illustrate different kinds of features found in actual photographs. Figure 8 shows a 640x480-pixel photograph exhibiting uncorrelated high frequencies represented by the leaves of the trees. It was superimposed with a textual mask (18 pt font size) covering 18.77% of its original area. The restored image, obtained in 6.37 seconds, essentially recovers all details of the original picture. Notice, for instance, the children playing in the back. Figure 9 shows a 640x480-pixel image containing very few high contrast edges, but with 14.54% of its area scratched. The image shown on its right was recovered in 5.87 seconds. Finally, Figure 10 shows an underwater scene (512x384 pixels) containing a large number of high contrast edges and superimposed with a mask covering 16.19% of its area. Figure 10 (right) was reconstructed in 4.06 seconds. Notice that such a reconstruction is mostly fine, except for some disconnected branches on the top right. Due to the relatively small scale of some of the masked branches, other inpainting techniques are also likely to fail to connect these edges. Table 1 summarizes the inpainting times obtained on two different systems.

The quality of an inpainting is a subjective issue. Error measurements should take into account a perceptual metric, such as the S-CIELAB metric [16]. Unfortunately,

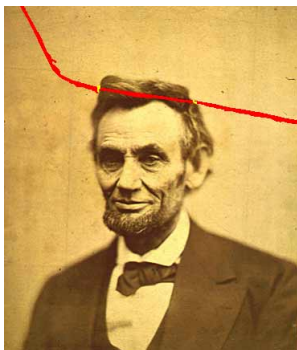


Fig. 4: Lincoln portrait showing mask and two diffusion barriers (at the boundaries of Lincoln's hair).

we were unable to use S-CIELAB this time, and, instead, we used the mean-square error (MSE) of the reconstructed region computed for the R, G and B channels as a measure of the quality of the reconstruction. MSE is frequently used in image processing to assess error. For the case of Figures 5 and 6, the MSE was computed against images restored

by Bertalmio et al. and available at their web site [3]. The errors associated with the reconstruction of the images shown in Figures 8, 9 and 10 were computed using the original photographs as reference. The results are summarized in Table 2, sorted by increasing error. Notice that for the case of images not containing sharp color or intensity discontinuity (e.g. *three girls* and *baby Lu*) the error is small. In particular, for the case of the three girls, our result is virtually indistinguishable from the Bertalmio's.

As expected, images containing large amounts of high frequencies (*yard* and *underwater*), present larger reconstruction errors. Despite the error values, the reconstructed images still look good (Figures 8 (right) and 10 (right)). For the yard, most high frequency regions correspond to tree leaves, which due to its stochastic nature help to mask the error. In the case of the underwater image, the error is again distributed across all high frequency regions. However, it only seems to be noticeable in areas containing predictable high contrast edges, such as the branches on the top right.

5. CONCLUSION AND FUTURE WORK

We have presented a simple and fast inpainting algorithm based on an isotropic diffusion model extended with the notion of user-provided diffusion barriers. The results produced by this simple model are, in many cases, comparable to previously known non-linear inpainting models, but two to three orders of magnitude faster, thus making inpainting practical for interactive applications.

Ideally, the mask Ω should include exactly the region to be retouched. If smaller, $\partial\Omega$ will contain spurious information, which will be carried into the restored area. If bigger, some possibly important information might be discarded. Being able to create and refine Ω interactively can greatly improve the quality of the reconstruction.

The presented algorithm is intended for filling in locally small areas. For larger inpainting domains, a scale-space approach [15] can be used to preserve the algorithm's speed at the expense of reconstruction quality.

Although diffusion barriers could be used to reconnect edges in Figures 5 and 11, an automatic procedure similar to the one described by Nitzberg et al [12] is preferable. Finally, we intend to evaluate the quality of the restored images using the S-CIELAB metric for perceptual color fidelity [16].

Table 1 Inpainting time measured using two systems

Image	Time PIII 450 MHz	Time Athlon 1 GHz
<i>Lincoln</i>	0.61 sec.	0.30 sec.
<i>New Orleans</i>	2.53 sec.	0.71 sec.
<i>Three girls</i>	1.21 sec.	0.49 sec.
<i>Yard</i>	6.37 sec.	1.90 sec.
<i>Baby Lu</i>	5.87 sec.	1.70 sec.
<i>Underwater</i>	4.06 sec.	1.11 sec.

Table 2 MSE for the RGB channels of the restored images

Image	MSE r	MSE g	MSE b	# Masked pixels
<i>Three Girls</i>	33.88	33.88	33.88	9,264
<i>Baby Lu</i>	61.32	66.9	72.40	42,061
<i>New Orleans</i>	347.53	269.57	290.59	20,795
<i>Yard</i>	729.76	725.73	732.90	57,688
<i>Underwater</i>	802.10	589.26	510.06	31,831

REFERENCES

- [1] Applied Science Fiction. DIGITAL ICE Technology (<http://www.appliedsciencefiction.com>).
- [2] Bertalmio, M, Sapiro, G., Caselles, V., Ballester, C. Image Inpainting. *SIGGRAPH 2000*, pages 417-424.
- [3] Bertalmio, M. Marcelo Bertalmio's web site: <http://www.ece.umn.edu/users/marcelo/restoration.html>
- [4] Chan, T., Shen, J. Mathematical Models for Local Deterministic Inpaintings. UCLA CAM TR 00-11, March 2000.
- [5] Chan, T., Shen, J. Non-Texture Inpainting by Curvature-Driven Diffusions (CCD). UCLA CAM TR 00-35, Sept. 2000.
- [6] Discreet. Revival. (<http://www.discreet.com>)
- [7] Eismann, K. and Simmons, S. Photoshop Restoration and Retouching. Que. ISBN: 0789723182.
- [8] Gomes, J., Velho, L. *Image processing for computer graphics* (New York, NY, Springer-Verlag, 1997. ISBN: 0387948546)
- [9] Kanizsa, G. *Organization in vision: essays on gestalt perception*. Praeger Publishers, 1979.
- [10] Kokaram, A. Morris, R., Fitzgerald, W., Rayner, P. Interpolation of Missing Data in Image Sequences. IEEE Transactions on Image Processing, 11(4), pages 1509-1519, 1995.
- [11] Hirani, A., Totsuka, T. Combining Frequency and Spatial Domain Information for Fast Interactive Image Noise Removal. *SIGGRAPH 1996*, pages 269-276.
- [12] Nitzberg, M., Mumford, D., Shiota, T. Filtering, Segmentation and Depth. Lecture Notes in Computer Science Number 62, Springer-Verlag, 1993. ISBN: 0387564845.
- [13] Perona, P. Malik, J. Scale-space and edge detection using anisotropic diffusion. IEEE-PAMI 12, pp. 629-639, 1990.
- [14] PuffinDesigns. Commotion (<http://www.puffindesigns.com/products/commotion.html>)
- [15] Witkin, A. P. Scale-space filtering. Proceedings International Joint Conference on Artificial Intelligence, pages 1019 - 1022, 1983.
- [16] Zhang, X. M. and Wandell, B. A. A spatial extension to CIELAB for digital color image reproduction. Journal of the Society for Information Display, 5, No. 1, pp. 61-63, 1997.



Fig. 5: New Orleans: (left) Picture with superimposed text (courtesy of Marcelo Bertalmio [4]). (right) Restored image obtained with our algorithm.

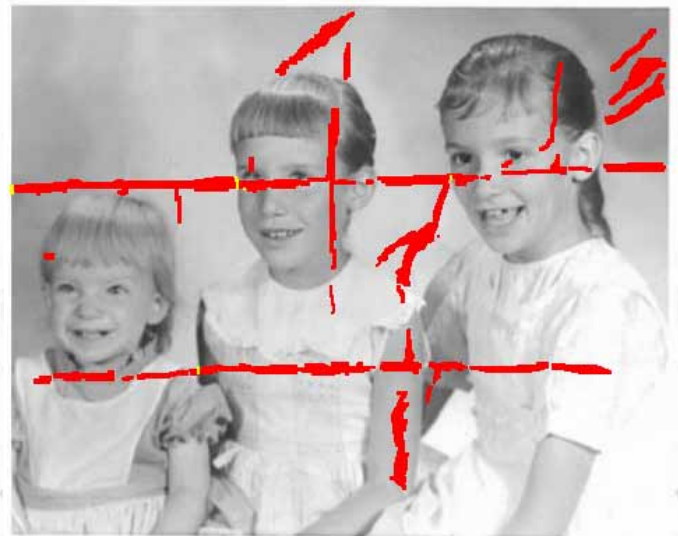


Fig. 6. Left: Old photograph (courtesy of Marcelo Bertalmio [4]). Right: Mask and diffusion barriers superimposed. Diffusion barriers: one between the left white border and the gray background, one between the background and the left side of each of the two bigger girls' faces, and one between the "left" arm of the girl in the center and the background.



Fig. 7. Three girls. Left: Restored image obtained with our algorithm. Right: Result produced with Bertalmio's algorithm (courtesy of Marcelo Bertalmio [4]). Different masks were used for the two images.

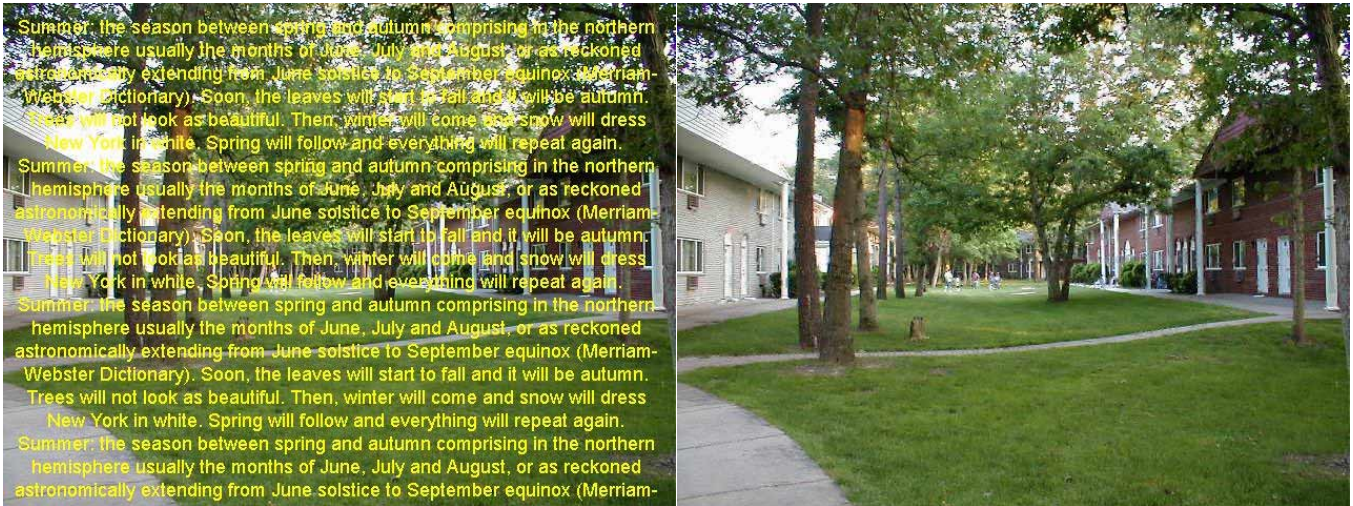


Fig. 8. Yard: Image containing uncorrelated high frequency with text covering 18.77% of its area. Right: restored image obtained with our algorithm. Notice the children playing in the back, and the details of the doors, windows and columns.



Fig. 9. Baby Lu: Image containing few high contrast edges. Mask covers 14.54% of its area. Right: restored image.

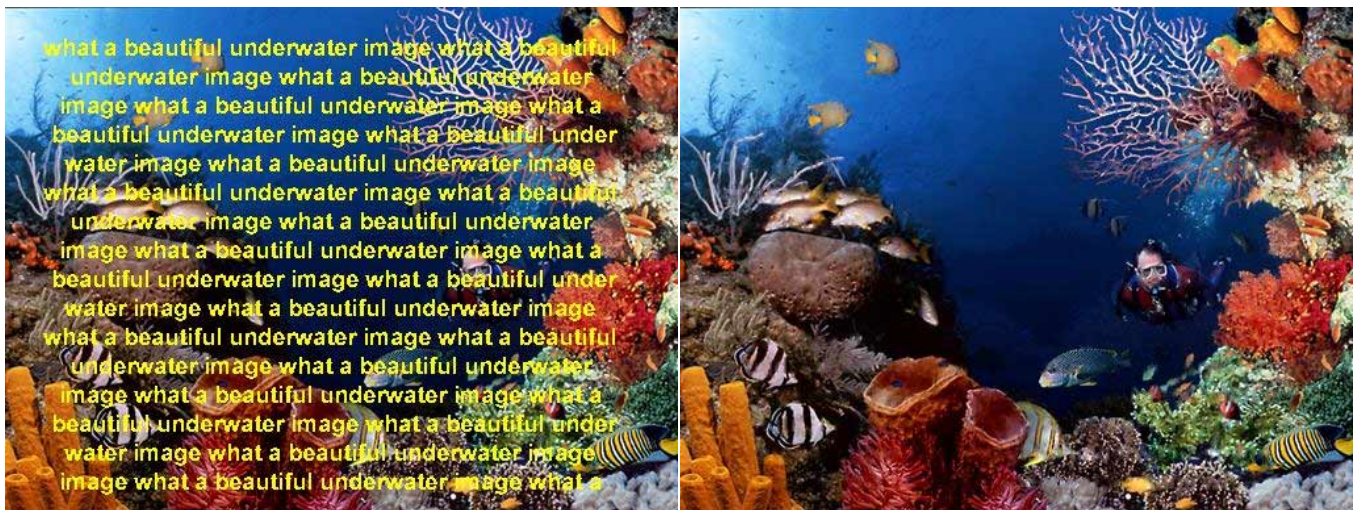


Fig. 10. Underwater. Left: Image containing many high contrast edges, with text covering 16.19% of its area. Right: restored image. Although the error is distributed across all high frequency regions, it is noticeable at the broken and blurred white edges on the top right.