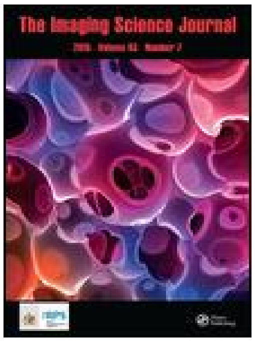



The Imaging Science Journal

ISSN: 1368-2199 (Print) 1743-131X (Online) Journal homepage: https://www.tandfonline.com/loi/yims20

# Efficient image inpainting using adaptive edge-preserving propagation

C Qin, F Cao & X P Zhang

**To cite this article:** C Qin, F Cao & X P Zhang (2011) Efficient image inpainting using adaptive edge-preserving propagation, The Imaging Science Journal, 59:4, 211-218, DOI: 10.1179/1743131X10Y.0000000010

**To link to this article:** https://doi.org/10.1179/1743131X10Y.0000000010

Published online: 12 Nov 2013.

# Efficient image inpainting using adaptive edge-preserving propagation

**C Qin*[a], F Cao[b,c] and X P Zhang[c]**
[a]School of Optical-Electrical and Computer Engineering, University of Shanghai for Science and Technology, Shanghai 200093, China
[b]College of Information Engineering, Shanghai Maritime University, Shanghai 200135, China
[c]School of Communication and Information Engineering, Shanghai University, Shanghai 200072, China

**Abstract:** We propose an image inpainting algorithm based on adaptive edge-preserving propagation for structure repairing. Neighbouring information is progressively propagated into damaged region. The optimal size and location of the window containing damaged pixel are adaptively chosen according to the intact degree and colour distribution. To preserve sharpness of edges, contributing weights of the pixels in neighbouring window are decided by their direction with isophote and distance with damaged pixels. Compared with typical partial differential equation (PDE)-based methods, the proposed approach is more concise and efficient, and can give satisfactory results for structural information repairing. Experiments are carried out to show effectiveness of the method.

**Keywords:** image inpainting, edge preserving, anisotropic propagation, efficiency

## 1 INTRODUCTION

The idea of image inpainting is inherited from the ancient technique of manually repairing valuable artworks in an indiscernible way.[1] Inpainting of digital images has found applications in such areas as restoration of historical photographs, filling in or removing chosen areas in images, and wiping out visible watermarks. Recently some researchers have applied inpainting techniques in de-interlacing,[2] image compression[3] and automatic image recovery.[4] Image inpainting is different from conventional image restoration in which the regions to be restored contain both noise and useful information. In image inpainting, however, the missing or damaged areas generally contain no useful information. Therefore, the task is to *generate* or *create* image regions that initially do not exist at all, based on the available information in the close neighbourhood. Since the late 1990s, image inpainting has been attracting much research attention, and various methods are proposed. There are three main types of the inpainting approaches:

1. *Interpolation-based methods*. Shih *et al.*, proposed a multi-resolution approach by pixel interpolation.[5] Image is divided into blocks, and the division is carried on until variance of sub-block is less than a given threshold. The damaged pixel is then replaced with the mean of the current sub-block or the sub-block at the previous level. Another similar inpainting algorithm based on interpolation mechanism is introduced in Ref. 6. The repairing procedure checks the surrounding information of a damaged pixel and determines the size of the reference window that can be used to compute an interpolated colour. But this kind of methods always causes edge blurring when the repaired pixels are close to edges.

*The MS was accepted for publication on 10 October 2010.*
* *Correspondence author: Chuan Qin, School of Optical-Electrical and Computer Engineering, University of Shanghai for Science and Technology, Shanghai 200093, China; email: qin@usst.edu.cn*

2. *PDE-based methods*. This typical technique is motivated by the intensive work on the use of partial differential equations (PDE) and variational methods in image processing. Bertalmio *et al.*[1,7] established a mathematical model of image inpainting by borrowing ideas from classic fluid dynamics. By iteratively solving the numerical representation of a PDE, they managed to smoothly propagate information of grey values from surrounding areas into the region $\Omega$ to be inpainted along isophotes. Guided by the connectivity principle of human visual perception, Chan *et al.*[8] proposed a non-texture inpainting method using the third-order PDE based on a total variation model.[9] This method in fact represents an anisotropic diffusion process. To satisfy human visual requirements, the intensity of diffusion is related to curvature. When the disconnected remaining object parts are separated far apart by inpainting domain, this method still gives good results. But PDE-based methods often have high-order mathematic model and introduce higher computation complexity.
3. *Example/patch-based methods*. This kind of methods employs the texture synthesis technique to repair large image region from sample images. Criminisi *et al.*[10] proposed an inpainting method for region filling and object removal. The method performs the synthesis task through a best-first filling strategy that depends entirely on the priority values assigned to each patch. After finding the patch with the maximum priority, the most similar patch is chosen from the intact region to replace it, then the priority values are updated to continue the above steps repeatedly, and the whole process is terminated until all damaged patches are inpainted. This example-based method can deal with relatively larger region than PDE-based methods, but will lead to some artificial seams between patches.

The proposed method incorporates the strengths of above kinds of approaches into an efficient algorithm. The key of this inpainting method is based on the adaptive edge-preserving propagation. In propagating process, the size and location of neighbouring region used for repairing damaged pixel are adaptively changed with the intact degree and distribution of the region, and all pixels in the neighbouring region are considered differently according to the direction and distance between with the damaged pixel. By iteratively propagating the information into damaged region, the method can acquire satisfactory inpainting result and solves the edge blurring problem and artificial seams simultaneously. Compared with high order PDE-based method, this method is relatively concise in mathematic expression, and costs lower computation complexity.

In the following, Section 2 describes our fast image inpainting method based on adaptive edge-preserving propagation, Section 3 presents experimental results and discussion, and Section 4 concludes the paper.

## 2 IMAGE INPAINTING-BASED ON ADAPTIVE EDGE-PRESERVING PROPAGATION

A primitive way of obtaining the guessed value of a damaged pixel is to utilise its surrounding pixel colours. In interpolation-based inpainting approaches,[5,6] one inpainted pixel is only processed for one time during interpolating calculation. The information of neighbouring region cannot be utilised appropriately, especially when the inpainting region is relatively larger. If the neighbouring window for interpolation contains edges, the reconstructed area will be blurred due to the low-pass nature of the interpolating operator. In this paper, we borrow the idea of PDE-based method to improve the interpolation-based method and present a new inpainting method. The proposed method processes every damaged pixel iteratively and propagates the surrounding information progressively, while the adaptive edge-preserving technique can restore the sharpness of edges.

### 2.1 Strategy of image inpainting

We express our algorithm in a discrete form and solve it by iterations. Suppose that the damaged image is $u$, and the region needed inpainting is represented as $\Omega$. The recursion formula of our algorithm is shown in equation (1), where the superscript ($i$) is a time index, viz. number of iteration steps, $T$ is the total iteration steps, $u^{(i)}(x,y)$ represents the pixel value after $i$ iteration steps, $u_t^{(i)}(x,y)$ is the updating increment of each step and $\alpha$ denotes the updating speed.

$$u^{(i+1)}(x,y) = u^{(i)}(x,y) + \alpha\, u_t^{(i)}(x,y),$$
$$i = 1, 2, \ldots, T, \ \forall (x,y) \in \Omega \tag{1}$$

When applying equation (1) to the pixels in the border $\partial\Omega$ of the region $\Omega$ to be inpainted, known pixels from outside this region are used. That is,

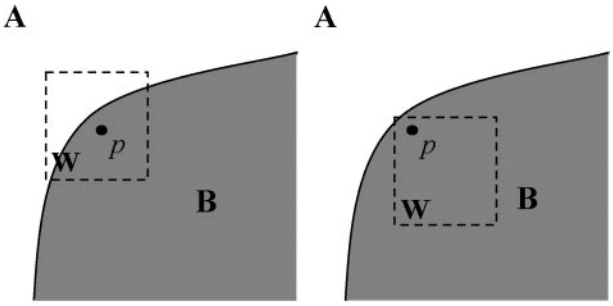
```
mask: mask(x, y) will be non-zero, when image pixel u(x, y) is damaged.
ind: larger ind corresponds to bigger size of neighboring window.
Get_minvarwin: return the window which has minimum variance.
Vaildpixelrate: return the percentage of valid pixels in current window.
Inpaint_EPP: return the value obtained by edge-preserving propagation.

For damaged pixel u^(i)(x, y) in the image u after i iteration steps:
Let ind = 1;
W = Get_minvarwin(x, y, ind, mask);
vpr = Vaildpixelrate(W, mask);
while (true)
{
    if (ind ≥ INDMAX)
      mean = Inpaint_EPP(W, x, y); break;
    else if (vpr ≥ VALIDRATE)
          mean = Inpaint_EPP(W, x, y); break;
       else
       {
          ind = ind + 1;
          W = Get_minvarwin(x, y, ind, mask);
          vpr = Vaildpixelrate(W, mask);
       }
}
u^(i+1)(x, y) = u^(i)(x, y) + α (mean - u^(i)(x, y));
```

**1** Overview of proposed inpainting algorithm

conceptually, we compute equation (1) in the region $\Omega^{\varepsilon}$ (an $\varepsilon$ dilation of $\Omega$),[1] although we update the values only inside $\Omega$ (that is, equation (1) is applied only inside). The information in the narrow band $\Omega^{\varepsilon}-\Omega$ is propagated inside $\Omega$. Propagation of the information of both grey-values and structural edges is fundamental for the success of our algorithm.

All damaged pixels are updated simultaneously after every iteration step. Figure 1 gives the detailed inpainting procedure for one iteration step. In Fig. 1, the *mask* is a location matrix which has the same size as the repairing image $u$. $mask(x,y)$ is non-zero corresponding to the damaged region $\Omega$ in $u$, while $mask(x,y)$ is equal to zero for other intact region. For every damaged pixel localised by *mask*, such as $u(x,y)$, the proposed algorithm first uses the function **Get_minvarwin** to obtain its neighbouring window W. The window size is decided by the parameter *ind*, and larger *ind* corresponds to bigger size of W. If the percentage of valid pixels in W calculated by the function **Vaildpixelrate** is also satisfied the given condition, the inpainting algorithm will utilise the current neighbouring window W to update the current damaged pixel $u(x,y)$, which is conducted by the function **Inpaint_EPP**. Otherwise, the size of the neighbouring window should be increased to do the above procedure continually until the window size




**2** Different neighbouring windows with different variances

reaches its preset maximum. By implementing this process, the algorithm can acquire the detailed recursion formula for equation (1), which is shown in the last line in Fig. 1.

This method repairs the damaged pixels progressively by propagating the information of neighbouring window. There are two key issues in the method: the one is how to choose the neighbouring window which is decided by the two functions **Get_minvarwin** and **Vaildpixelrate**, and the other is how to propagate the information of the neighbouring window which is conducted by the function **Inpaint_EPP**. The following two subsections will give the detailed description.

## 2.2 Neighbouring window choosing

Unlike other methods proposed in Refs. 6 and 10, the damaged pixel in our method may not be at the centre of the neighbouring window, and the size and location of the window are not fixed. How to choose the neighbouring window depends on two factors. One is the distribution of the pixels in the window. When the damaged pixel nears edge region, in which the variance and gradient is larger, and if the damaged pixel is just at the centre of the window for repairing, the inpainted colour using mean value of the window will be blurring and cause unsatisfactory result. So the neighbouring window containing damaged pixel which has the least variance should be chosen for inpainting (Fig. 2). When damaged pixel $p$ is located near edge (regions A and B have significant different colours), we should use the window W in the right of Fig. 2 rather than the left. We invoke the function **Get_minvarwin** in Fig. 1 to implement this feature. Finding the neighbouring window W with the least variance can be formulated as equation (2), where $\overline{W}$ denotes as the mean value of all valid pixels in window W. Note that the summation is also done over all valid pixels in W. So the window W obtained by equation (2) is the one that has the least variance and includes the current repairing pixel $p$ simultaneously:

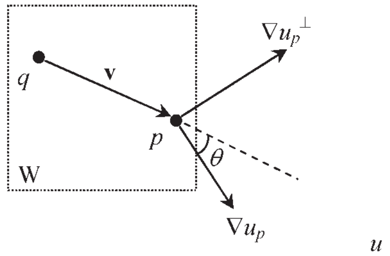
$$W = \arg\min_{W}\left\{\sum_{(x,y)\in W}\left[u(x,y)-\overline{W}\right]^2\right\}, \quad \text{subject to } p\in W \tag{2}$$

The other factor about how to choose neighbouring window is the valid degree of pixels in the window, which is calculated by the function **Vaildpixelrate** in Fig. 1. This is done with the help of the *mask* that locates the damaged areas. If the valid degree of the window W with initial size obtained by the function **Get_minvarwin** is high enough, this window can be regarded as the best neighbouring window for repairing the current damaged pixel. To avoid that there is no enough surrounding information for inpainting, if the percentage of valid pixels in the window W is below the preset threshold VALIDRATE, the size of W should be enlarged to find the window again by the function **Get_minvarwin**. However, if the window is extended to a certain large level, e.g. the size index of neighbouring window exceeds desired maximum INDMAX, in this situation, the colour information of the window with maximum size is directly used to inpaint the damaged pixel.

### 2.3 Adaptive edge-preserving propagation

After choosing the best neighbouring window, we present an adaptive edge-preserving propagation method to transmit the colour information into damaged pixels. The key point of the method is to assign different contributing weights to the pixels in neighbouring window during inpainting. This edge-preserving propagation method considers two important matters: the distance between neighbouring pixels with the damaged pixel, and the direction of the vector connecting neighbouring pixels and the damaged pixel.

Obviously, the pixels in neighbouring window which are nearer to the damaged pixel should contribute more information and have larger weights in repairing. To acquire better structural inpainting quality and preserve sharp edges, the main direction of information propagation should be along the isophote.[1] The direction of isophote is always normal to gradient direction (Fig. 3). The point $p$ in the image $u$ denotes the damaged pixel to be inpainted,




**3** The principle of edge-preserving propagation

and the point $q$ denotes the neighbouring pixel in the neighbouring window W of $p$. The vector $\mathbf{v}$ connects these two pixels from $q$ to $p$. We assume that the gradient direction at point $p$ is $\nabla u_p$, so the isophote direction is $\nabla u_p{}^{\perp}$. Let $\theta$ be the angle between the vector $\mathbf{v}$ and $\nabla u_p$. So in our method, the pixel in neighbouring window with larger angle $\theta$ implies that $\mathbf{v}$ is more adjacent to the direction of the isophote $\nabla u_p{}^{\perp}$, and will have larger contributing weights to repair the damaged pixel.

To express the steps of edge-preserving propagation, we denote the pixels in neighbouring window W as $q_1$, $q_2$, ..., $q_n$ not including the current repairing pixel $p$. The distance between every neighbouring pixel $q_i$ with $p$ is:

$$d_i = \|p-q_i\|_2,\ \forall q_i\in W,\ i=1,2,\ \ldots,n \tag{3}$$

where $\|\bullet\|_2$ denotes the $L_2$-norm and $d_i$ can be normalised to $D_i$ by:

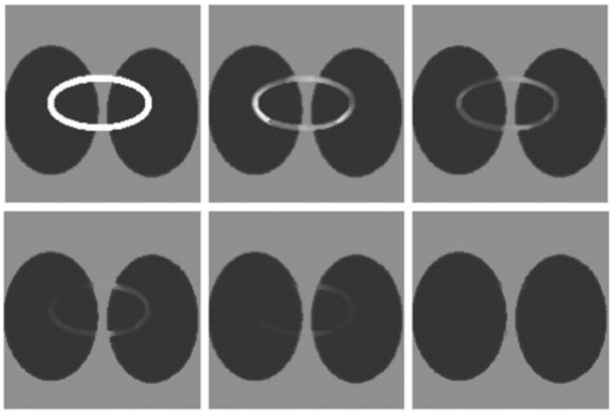
$$D_i = \frac{d_i}{\sum_{j=1}^{n} d_j} \tag{4}$$

The directional relation $C_i$ between the unit vectors $\mathbf{v}_i$ and the isophote at pixel $p$ is:

$$C_i = \frac{\nabla u_p \cdot \mathbf{v}_i}{K} \tag{5}$$

$$\mathbf{v}_i = \frac{p-q_i}{\|p-q_i\|_2} \tag{6}$$

where $\nabla$ is the gradient operator, and $\bullet$ is the dot product for vectors. We let $K$ be a predetermined factor for normalisation (e.g. $K$=255 for a typical grey-level image).[10] The $C_i$ can express the cosine value of the angle $\theta$ in Fig. 3, and $C_i$ will be smaller when $\theta$ is bigger. Based on the above analysis, we can conclude that the neighbouring pixels with smaller $D_i$ and $C_i$ should have larger weights when to inpaint the damaged pixel.



4 Progressive nature of the algorithm. The leftmost of the first row is the damaged image, and the rightmost of the second row is the final inpainting result with PSNR=45.8 dB. Several intermediate steps of the repairing are also shown

We rearrange all the neighbouring pixels $q_i$ ($i$=1, 2, …, $n$) in descending order of $D_i+C_i$ to form $q'_i$, and use the exponential function to compute the contributing weights $w_i$ of $q'_i$:

$$w_i=\frac{\exp(i)-\exp(1)}{\exp(1)\{[\exp(n)-1]/[\exp(1)-1]-n\}} \quad (i=1, 2, \ldots, n) \tag{7}$$

Note that $\sum w_i \equiv 1$, for all $i$=1, 2, …, $n$. So the neighbouring pixels $q'_i$ with smaller $D_i+C_i$ are assigned with higher contributing weights $w_i$ in repairing. The function of edge-preserving propagation named **Inpaint_EPP** in Fig. 1 returns the weighted-sum of $w_i$ and $q'_i$ to update the damaged pixel $p$.

Both interpolation-[6] and example-based[10] methods dispose all damaged pixels or patches only once, but our inpainting technique progressively restores damaged pixels by iterations. Because the distribution and direction information of neighbouring window are considered during the process of propagation, the method can preserve sharpness of edges and acquire better result. Compared with PDE-based methods,[1,7–9] this method is more concise without introducing complex high-order mathematic model in every iteration step.

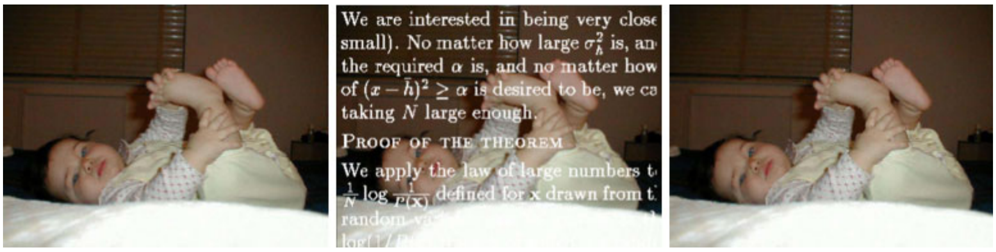
## 3 RESULTS AND COMPARISONS

Experiments were carried out on a group of colour and grey images with different sizes. Damages in the test images include block impairment, scattered spots, random scratches and superimposed watermark. For colour images, inpainting is done on the R, G and B channels, respectively. The obtained components are then combined to give the final results. The parameters used in experiments are: the threshold VALIDRATE of the valid percentage in neighbouring window is set to 40%, the updating speed parameter $\alpha$=1.0 and the size sequence of neighbouring windows is [$3\times3$, $5\times5$, $7\times7$, $9\times9$, $11\times11$, $13\times13$, $15\times15$].

Figures 4–7 show some repairing results of proposed method. Evaluation of the quality of repaired images is often done with subjective assessment because no original undamaged version is available in reality, while in experiments we can generate

5 Scattered spots inpainting. From left to right: original image, damaged image and inpainting result of the present method with PSNR=37.6 dB

**6** Random scratches inpainting. From left to right: original image, damaged image and inpainting result of the present method with PSNR=34.2 dB




**7** Superimposed watermark inpainting. From left to right: original image, damaged image and inpainting result of the present method with PSNR=39.1 dB

damaged images from some known originals, and compare the repaired version with the original ones. By both subjective observation and peak signal-to-noise ratio (PSNR) evaluations with respect to the originals, we conclude that the proposed method gives satisfactory output for structural inpainting.

We mainly compare our method with several reported inpainting methods,[1,6–9] which are mostly used for structural information inpainting (Fig. 8). Compared with the interpolation-based method,[6] the present method produces better visual quality and better performance in terms of PSNR. Sharp edges are restored and preserved after inpainting with our method, while the method in Ref. 6 causes much blurring effect near edges. We also compare the performance of our method with typical PDE-based methods, such as BSCB method,[1,7] total variation method[9] and CDD method[8] listed in Table 1. From the second row of Fig. 8, we can find that result of proposed method has higher PSNR than BSCB method, and has almost the same visual quality and PSNR value with the methods in Refs. 8 and 9. But because our method has more concise mathematical model and lower computation complexity, much less iteration steps of our method are required to reach stability in numerical implementation than these PDE-based methods, as illustrated in Fig. 9.

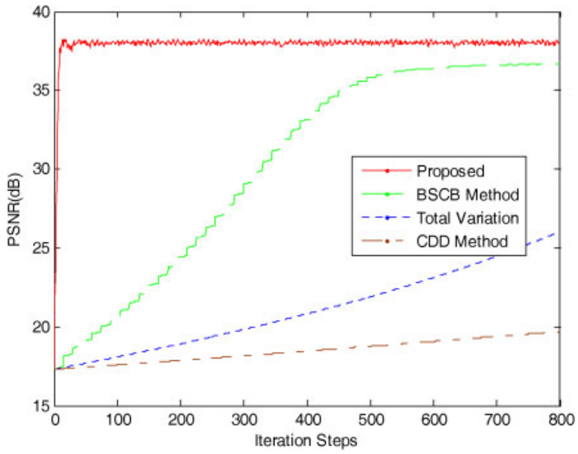
**Table 1** Typical PDE-based inpainting methods

| Method | Mathematic expression | Order of PDE |
|---|---|---|
| BSCB[1,7] | $\frac{\partial u}{\partial t}=\nabla\Delta u\cdot\nabla^{\perp}u$ | 3 |
| Total variation (TV)[9] | $\frac{\partial u}{\partial t}=\nabla\cdot\left[\frac{\nabla u}{\lvert\nabla u\rvert}\right]$ | 2 |
| Curvature-driven diffusion (CDD)[8] | $\frac{\partial u}{\partial t}=\nabla\cdot\left[\frac{g(\lvert\omega\rvert)}{\lvert\nabla u\rvert}\nabla u\right]$ | 3 |

## 4 CONCLUSION

How to preserve sharpness of image edges is very important for structural inpainting. We have proposed an efficient image inpainting approach, which progressively propagates neighbouring information into damaged region and can restore sharp edge successfully. This is achieved by adaptive edge-preserving propagation technique. In the inpainting process, the size and location of neighbouring window can be adaptively decided according to intact degree and colour distribution. The window with the higher intact degree and lower variance is preferable. The contributing weights of the pixels in neighbouring window are considered differently by calculating their direction and distance with damaged pixel. The pixel which is close to damaged pixel and whose corresponding vector is near to the isophote will be given larger weight in repairing. Different from interpolation-based method, our method disposes every damaged pixel iteratively, and acquire better visual quality with

**8** Comparison results. The first row show original image, damaged images and result of the method in Ref. 6 with PSNR=35.4 dB, respectively. The second row from left to right are the results of BSCB method[1,7] with PSNR=36.7 dB, total variation inpainting[9] with PSNR=38.0 dB, CDD-based method[8] with PSNR=38.3 dB and proposed method with PSNR=38.2 dB




**9** Performance comparisons between proposed method and typical PDE-based methods in Refs. 1 and 7–9

higher PSNR values. Because our method has concise mathematical expression, lower computational complexity is introduced and less iteration steps are needed than typical PDE-based methods.

Now our method cannot repair the texture information of images. The future work will consider the factor of texture periodicity and incorporate it into the edge-preserving propagation model. We also assume that, as in other works reported thus far, the locations of damaged region are known. But for real applications, detecting and locating the damaged region is important, therefore deserving in-depth investigations.

## ACKNOWLEDGEMENTS

This work was supported by the Natural Science Foundation of China (nos. 60872116, 60832010 and 60773079), the Shanghai Rising-Star Program (no. 10QH14011), the Key Scientific Research Project of Shanghai Education Committee (no. 10ZZ59), the

Shanghai Specialized Research Foundation for Excellent Young Teacher in University (no. slg09005) and the OECE Innovation Foundation of USST (no. GDCX-Y-103).

## REFERENCES


1 Bertalmio, M., Sapiro, G., Caselles, V. and Ballester, C. Image inpainting, Proc. 27th Annual Conf. on *Computer graphics: SIGGRAPH 2000*, New Orleans, LA, USA, July 2000, ACM, pp. 417–424.

2 Ballester, C., Bertalmío, M., Caselles, V., Garrido, L., Marques, A. and Ranchin, F. An inpainting-based deinterlacing method. *IEEE Trans. Image Process.*, 2007, **16**, 2476–2491.

3 Liu, D., Sun, X., Wu, F., Li, S. and Zhang, Y. Q. Image compression with edge-based inpainting. *IEEE Trans. Circuits Syst. Video Technol.*, 2007, **17**, 1273–1287.

4 Wang, S. S. and Tsai, S. L. Automatic image authentication and recovery using fractal code embedding and image inpainting. *Patt. Recogn.*, 2008, **41**, 701–712.

5 Shih, T. K., Lu, L. C., Wang, Y. H. and Chang, R. C. Multi-resolution image inpainting, Proc. IEEE Int. Conf. on *Multimedia & Expo: ICME 2003*, Baltimore, MD, USA, December 2003, Vol. 1, IEEE Computer Society, pp. 485–488.

6 Shih, T. K. Adaptive digital image inpainting, Proc. 18th Int. Conf. on *Advanced information networking and applications* (AINA 2004), Fukuoka, Japan, March 2004, FIT, Vol. 1, pp. 71–76.

7 Bertalmio, M., Bertozzi, A. L. and Sapiro, G. Navier-stokes, fluid dynamics, and image and video inpainting, Proc. IEEE Int. Conf. on *Computer vision and pattern recognition: CVPR 2001*, Kauai, HI, USA, December 2001, IEEE Computer Society, Vol. I, pp. 355–362.

8 Chan, T. F. and Shen, J. Nontexture inpainting by curvature-driven diffusions. *J. Visual Commun. Image Represent.*, 2001, **12**, 436–449.

9 Chan T. F. and Shen, J. Mathematical models for local non-texture inpaintings, *SIAM J. Appl. Math.*, 2001, **62**, 1019–1043.

10 Criminisi, A., Pérez, P. and Toyama, K. Region filling and object removal by exemplar-based image inpainting. *IEEE Trans. Image Process.*, 2004, **13**, 1200–1212.