Journal of Zhejiang University-SCIENCE C (Computers & Electronics)
ISSN 1869-1951 (Print); ISSN 1869-196X (Online)
www.zju.edu.cn/jzus; www.springerlink.com
E-mail: jzus@zju.edu.cn

Image compression based on spatial redundancy removal and image inpainting

Vahid BASTANI, Mohammad Sadegh HELFROUSH[†‡], Keyvan KASIRI

(*Department of Electrical and Electronic Engineering, Shiraz University of Technology, Shiraz, Iran*)

[†]E-mail: ms_helfroush@sutech.ac.ir

Received Apr. 2, 2009; Revision accepted June 19, 2009; Crosschecked Dec. 8, 2009

Abstract: We present an algorithm for image compression based on an image inpainting method. First the image regions that can be accurately recovered are located. Then, to reduce the data, information of such regions is removed. The remaining data besides essential details for recovering the removed regions are encoded to produce output data. At the decoder, an inpainting method is applied to retrieve removed regions using information extracted at the encoder. The image inpainting technique utilizes partial differential equations (PDEs) for recovering information. It is designed to achieve high performance in terms of image compression criteria. This algorithm was examined for various images. A high compression ratio of 1:40 was achieved at an acceptable quality. Experimental results showed attainable visible quality improvement at a high compression ratio compared with JPEG.

Key words: Edge extraction, Image compression, Image inpainting, Spatial redundancy
doi:10.1631/jzus.C0910182 **Document code:** A **CLC number:** TN919.8

1 Introduction

Compression is acceptable for natural images as a large amount of redundancy is included in such images. A high compression ratio can be achieved by eliminating these redundancies, but at the cost of some information loss. Up to now, great achievements have been made in image compression. State-of-the-art methods such as JPEG (Pennebaker and Mitchell, 1992) and JPEG2000 (Taubman and Marcellin, 2002) efficiently exploit statistical redundancies among pixels and achieve high compression ratios. A common framework in almost all lossy image compression methods is image transformation followed by quantization and coding. In contrast, JPEG and JPEG2000 use the discrete cosine transform (DCT) and wavelet transform, respectively. Information loss causes decrease in the quality of reconstructed images, especially at high compression ratios. To improve perceptual visual quality, Höntsch and Karam (2000) and Malo *et al.* (2006) incorporated the human vision system (HVS) properties in compression schemes.

The dominant type of redundancy within images comes from their representation procedure. Each digital image is composed of discrete points, called pixels. The value relevant to each pixel is the result of sampling from light or color intensity in the original image domain. Natural images consist of separate areas indicating the object surfaces or sceneries. Because the light intensity and color in such areas are approximately constant, the relevant values for pixels are highly correlated. Every pixel in such areas is likely to be of the same or very close value compared with the adjacent pixels. In this case, images suffer from a high level of spatial correlation. Hence, representing the image by storing all pixel values results in a large amount of redundancy.

The most significant information within an image is located in the boundary regions or edges. In fact, the boundary of a region not only specifies its overall shape, but also shows how pixel values change from neighboring regions to the inner regions of interest. As a result, it is possible to retrieve the inner areas

[‡] Corresponding author

using pixels located on the boundaries. Therefore, boundaries or edges are all the required information for displaying an image. Fig. 1 clarifies this concept. Fig. 1a shows a synthetic image composed of three different regions. Fig. 1b shows the boundaries of regions in Fig. 1a. Certainly, by diffusing the information of boundaries shown in Fig. 1b into the corresponding region, the image in Fig. 1a will be recovered. In Fig. 1b, the boundary regions are preserved with 9 pixels width. Also, the information related to other regions is removed. It is evident that only 2 pixels rather than 9 are adequate for representing boundary regions.

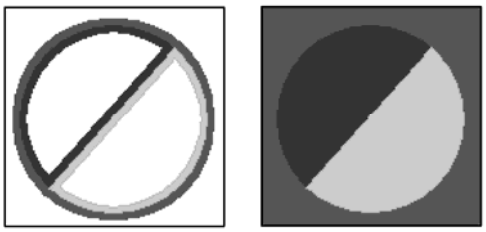

Fig. 1 (a) An image with three regions; (b) Extracted edges of the same image with 9 pixels width

The variation in values of pixels orthogonal to the edges is significant. Hence, areas in the neighborhood of edges may be considered as essential image information. On the other hand, while moving along the edge direction, no significant changes in pixel values will be observed. Moving further to the inner points of the boundaries will result in a considerable correlation for pixel values. Edges also represent some other necessary information including shapes. Redundancies related to the correlation along the edge direction may also be exploited via extracting shape information from pixel values in boundary regions.

Pixel values at the endpoints of an edge will be used for recovering the entire edge and boundary region. Of course, it holds true if the location of the edge points is known. In this paper, the edge endpoint is called the 'source point'. In order to recover boundary regions and pixels located perpendicular to the edge direction, samples of source points should be provided. These samples should come from different areas at each side of the edge. Fig. 2 indicates edge locations and source points for the image shown in Fig. 1a.

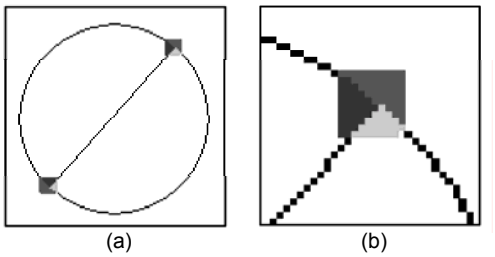

Fig. 2 (a) Source points and boundaries for Fig. 1a (the black lines stand for edges); (b) Zoomed version of (a)

The method proposed in this paper is based on eliminating the information of correlated regions and filling in the missing areas using sample pixels. In this method, some regions are intentionally removed at the encoder and recovered using an inpainting or interpolation technique at the decoder. Image inpainting is a method for recovering regions in images whose pixels are distorted or removed in some way. Inpainting methods are commonly based on partial differential equations (PDEs) (Bertalmio *et al.*, 2000; 2001; Bertalmio, 2006; Tai *et al.*, 2006), total variation (TV) (Shen and Chan, 2002; Chan and Shen, 2005b), and exemplar-based texture synthesis (Criminisi *et al.*, 2004). PDE and TV approaches work satisfactorily for homogenous inpainting regions with linear structures. However, in dealing with textural regions, the texture synthesis approach is more suitable. Some studies (Rane *et al.*, 2003; Grossauer, 2004) combined image inpainting and texture synthesis approaches to take advantage of both techniques. In PDE techniques, pixel values around the region to be inpainted are considered to be the boundary condition for a boundary value problem. Then, a proper equation for interpolating in that area will be solved. Image inpainting has a variety of applications such as text and object removal (Criminisi *et al.*, 2004; Patwardhan *et al.*, 2005), denoising, super resolution, digital zooming (Chan and Shen, 2005a), filling-in (Rane *et al.*, 2003), and compression.

Image inpainting for compression has been utilized in some previous studies (Rane *et al.*, 2003; Liu *et al.*, 2007; Galić *et al.*, 2008). Rane *et al.* (2003) proposed a method for recovering the removed blocks during JPEG transmission. The authors mentioned that this filling-in technique may be used as a preprocessing step for increasing JPEG performance. In this work, the original image is first partitioned into blocks, and each block is classified as a textural or

structural type. At the receiver, depending on the type of received block, two different approaches are selected for decoding of the blocks. A PDE-based inpainting method for structural blocks and a texture synthesis algorithm for textural blocks are applied. Liu *et al.* (2007) offered an algorithm similar to that in Rane *et al.* (2003). In this case, the inpainting method takes advantage of exemplar-based texture synthesis. In this method, edge information is stored as the ancillary information required for the recovering process.

Galić *et al.* (2008) used an inpainting technique directly for compression while Rane *et al.* (2003) and Liu *et al.* (2007) treated image inpainting only as a preprocessing step to increase other existing image compression standards. Here, the original image is analyzed into triangular regions in such a way that the triangle area is interpolated via triangle vertexes. For the interpolation phase, an anisotropic diffusion equation is used. Using this equation the edges can be recovered in an appropriate manner.

The method of this paper is involved in an independent compression with an inpainting technique. In our scheme, feature extraction is based on image structures. In the decoding phase, boundary region information is used for the recovering process. As a result, at high compression ratios, distortions caused by loss of information will occur mainly in pixel values rather than in image structures. Note that, for the HVS, distortions in image structures are more noticeable than distortions in pixel values. Furthermore, our method uses a high quality inpainting, which is specially designed to recover the removed information using edge direction and source points.

2 Image inpainting

Image inpainting is aimed to fill in missing regions or to modify damaged regions in a visually plausible and non-detectable way. In order to well clarify the inpainting problem, assume u_0 as the intensity function of the image defined in domain D. As indicated in Fig. 3, there is a hole $\Omega \subset D$ with unknown information. The objective is to find the recovered version of u_0, namely u, in such a way that the intensity function in the area $D-\Omega$ remains equal to u_0 while meaningfully filling in other regions. In this way, information on the boundary ε is diffused into Ω via a 2D interpolation technique.

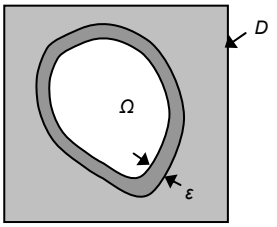

Fig. 3 A general case of the image inpainting problem

Note that in a general form, no information is available about the regions to be inpainted. It is obvious that the resulting inpainted image is not necessarily similar to the original one. However, for the application of compression, it is necessary for the inpainted image to be similar to the original one with a sufficient degree of accuracy. As the original image is in hand, it is possible to extract all of the information required for compression with an acceptable quality. In the method presented in this study, the only essential information for retrieving an image includes source point pixels and edges. In this particular case, one should design an inpainting method compatible with this application.

As an example, assume the region Ω in Fig. 3 as one of the closed regions in Fig. 1b. Here ε denotes the boundary of this region. In this case it is desirable to recover Ω using previously stored information available in ε. The resulting information should produce a high degree of correlation and smoothing. Hence, the problem can be expressed as the following boundary value problem in Ω:

$$\begin{cases} \Delta u = 0 \ \text{ in } \ \Omega, \\ u = u_0\big|_{\varepsilon}, \end{cases} \tag{1}$$

where Δ, u_0 and u stand for the Laplacian operator, the original image and the inpainted image, respectively. This is the Laplace equation for two dimensions. Comparing this equation with traditional inpainting equations, like those used in Bertalmio *et al.* (2000) and Galić *et al.* (2008), we can find that analytical and numerical theory for solving the Laplace equation has been completely extended, and that the existence and uniqueness of its response have already been proven.

It can be deduced from the above explanation that information on boundary ε is quite helpful in inpainting the regions inside that boundary. Here the boundary regions are specified by the source pixels

located on the endpoints of each edge and the shape of the corresponding edges. Hence, it is essential to deal with the recovering process by first considering the boundaries. There are no considerable variations along with the boundary or edge. Hence, the recovering process for boundary regions is similar to that for inner areas of the boundaries. Here the boundary value problem is addressed as a 1D problem along with the edges. A generic form of this problem is illustrated in Fig. 4, where μ_1 and μ_2 are the endpoints or the source points. μ_1 and μ_2 are neighboring as the boundary is a closed path. The curve Γ is the boundary indicating the edge, which is necessary for the retrieving process.

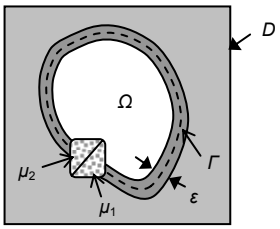

Fig. 4 The inpainting problem description for the application of image compression

In this stage the objective is to reconstruct the information in boundary region ε using the information available at endpoints μ_1 and μ_2 and using the curve Γ as an edge path. One can express Γ in a parametric form as follows:

$$\Gamma=\{(x_t,y_t)|x_t=f(t),y_t=g(t)\}, \tag{2}$$

where f and g are functions of t describing the curve Γ. Now, the problem of recovering u in the boundary region ε may be converted to the following boundary value problem:

$$\begin{cases}\dfrac{\mathrm{d}^2u}{\mathrm{d}l^2}=0,\\ u\big|_{\mu_1}=u_0\big|_{\mu_1},\ \ u\big|_{\mu_2}=u_0\big|_{\mu_2},\end{cases} \tag{3}$$

where dl is the differential element along with the curve. This equation arises from the minimum variation property along the path. Let $\boldsymbol{n}$ be a tangent vector to the curve:

$$\boldsymbol{n}=\left(\frac{\mathrm{d}x_t}{\mathrm{d}t},\frac{\mathrm{d}y_t}{\mathrm{d}t}\right)\Bigg/\sqrt{\left(\frac{\mathrm{d}x_t}{\mathrm{d}t}\right)^2+\left(\frac{\mathrm{d}y_t}{\mathrm{d}t}\right)^2}. \tag{4}$$

The first derivative along with the curve Γ in Eq. (3) can be defined as

$$\frac{\mathrm{d}u}{\mathrm{d}l}=\left(\frac{\partial u}{\partial x}\frac{\mathrm{d}x_t}{\mathrm{d}t}+\frac{\partial u}{\partial y}\frac{\mathrm{d}y_t}{\mathrm{d}t}\right)\Bigg/\sqrt{\left(\frac{\mathrm{d}x_t}{\mathrm{d}t}\right)^2+\left(\frac{\mathrm{d}y_t}{\mathrm{d}t}\right)^2}. \tag{5}$$

According to Eq. (1), using boundary information ε, the region Ω is reconstructed. The boundary ε can be retrieved by solving Eq. (3) using the source points μ_1, μ_2 and the path Γ. In order to model the overall problem into a single equation, the function $\lambda(x, y)$ is defined in the region $\Omega\cup\varepsilon$ as follows:

$$\lambda(x,y)=\begin{cases}1, & (x,y)\in\varepsilon,\\ 0, & (x,y)\in\Omega,\end{cases} \tag{6}$$

Combining Eqs. (1), (3) and (6), a generic form of the problem for the entire region is given as

$$\begin{cases}\lambda\dfrac{\mathrm{d}^2u}{\mathrm{d}l^2}+(1-\lambda)\Delta u=0,\\ u\big|_{\mu_1}=u_0\big|_{\mu_1},\quad u\big|_{\mu_2}=u_0\big|_{\mu_2}.\end{cases} \tag{7}$$

This equation describes the inpainting problem in recovering the removed spatial redundancy. Here the boundary conditions are those mentioned in Eq. (3). Solving Eq. (3) will result in ε which by itself is the boundary condition of Eq. (1). Hence, solving Eq. (7) analytically consists of two stages: first solving Eq. (1) to find values in ε and then solving Eq. (3) to retrieve values in Ω using information available in ε. One may combine these two steps into one by some brief modifications in the numerical approach, as described in the following.

Based on the idea used in Perona and Malik (1990) for numerical approximation, we use the mask shown in Fig. 5. The central difference of the Laplace equation will be

$$\begin{cases}4u_{\mathrm{C}}-u_{\mathrm{N}}-u_{\mathrm{E}}-u_{\mathrm{S}}-u_{\mathrm{W}}=0,\\ u_{\mathrm{C}}=u_{i,j},\ \ 1\le i\le N,\ 1\le j\le M.\end{cases} \tag{8}$$

NW	N	NE
W	C	E
SW	S	SE

Fig. 5 Pixels in the neighborhood of central pixel C in a 3×3 mask

For an image with $M\times N$ dimensions, each pixel should be solved. Therefore, the numerical approach for solving the Laplace equation can be changed into a system of $M\times N$ equations and unknowns, which is solvable through any iterative method. In order to solve Eq. (7), we extend Eq. (8) to eight pixels and add coefficients as

$$u_{\mathrm{C}}=\frac{1}{8}\left(c_{\mathrm{N}}\cdot u_{\mathrm{N}}+c_{\mathrm{E}}\cdot u_{\mathrm{E}}+c_{\mathrm{S}}\cdot u_{\mathrm{S}}+c_{\mathrm{W}}\cdot u_{\mathrm{W}}+c_{\mathrm{NW}}\cdot u_{\mathrm{NW}}+c_{\mathrm{NE}}\cdot u_{\mathrm{NE}}+c_{\mathrm{SW}}\cdot u_{\mathrm{SW}}+c_{\mathrm{SE}}\cdot u_{\mathrm{SE}}\right). \tag{9}$$

The coefficients c_i's are defined via Γ and λ as follows:

$$\lambda=0\Rightarrow\begin{cases}c_{\mathrm{N}}=c_{\mathrm{E}}=c_{\mathrm{S}}=c_{\mathrm{W}}=2,\\ c_{\mathrm{NW}}=c_{\mathrm{NE}}=c_{\mathrm{SW}}=c_{\mathrm{SE}}=0,\end{cases} \tag{10a}$$

$$\lambda=1\Rightarrow\begin{cases}c_{t-1}=c_{t+1}=4,\\ c_{\mathrm{else}}=0,\end{cases} \tag{10b}$$

where c_{t-1} and c_{t+1} are the coefficients of adjacent points along Γ in the boundary region ε. This means that for λ=1, i.e., along the edges, only the coefficients of previous and next points along Γ are nonzero. Therefore, Eq. (9) will be converted into a 1D equation in the Γ direction. For λ=0 it may be considered as a 2D equation as stated in Eq. (8).

3 Proposed image compression scheme

The method proposed in this section is based on removing the spatial redundancy at the encoder and restoring the removed information using an inpainting method at the decoder. In this algorithm spatial redundancy removal is performed through detecting edges and dividing the image into homogenous regions. The overall system is depicted in Fig. 6. In the following subsections, encoder and decoder blocks are discussed separately.

3.1 Encoder

At the encoder the input image needs to be denoised (Fig. 6). Noise removal is carried out through the following anisotropic diffusion equation (Perona and Malik, 1990):

$$\frac{\partial u}{\partial t}=\mathrm{div}(g(\|\nabla u\|)\cdot\nabla u), \tag{11}$$

where g is an exponential function of $\|\nabla u\|$. This function ranges from zero to one: near the edges it vanishes and away from the edges it monotonically increases to one. Therefore, the input image will be smoothed without missing sharpness of edges. The smoothing process may help to avoid recognizing the wrong edges caused by noise. As a result, this step improves the overall efficiency.

After noise removal, edge detection is carried out to specify the boundary of different regions. Here, edges are extracted by a conventional method (i.e., Sobel). In this application, edges must give boundaries of regions, where a significant transition presents. Thus, the corresponding edge detector should be able to detect real transitions. As methods such as Sobel, Prewitt and Roberts are based on a local gradient, they can detect local transitions. Experimental results show

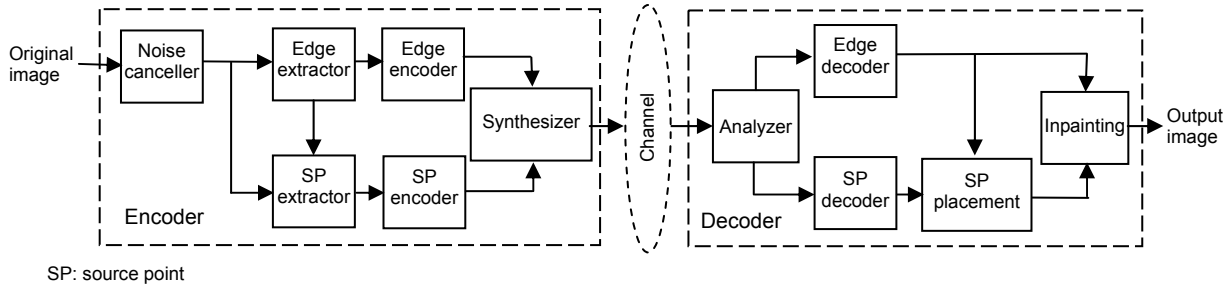

Fig. 6 The overall block diagram of the proposed system

that Sobel and Prewitt techniques produce nearly the same results. However, Roberts' method is of weaker response because it detects more false edges. Methods that perform edge linking, such as Canny, are not suitable for this application as the edges detected using this method may have some displacements compared to the actual position of transitions. These edges are then preserved as a binary image and encoded through a lossless encoder. They are used to assist in the recovering process later. While the size of the resultant binary image compared to the original one is 1:8, using an appropriate encoder, this size will be reduced further.

We aim to extract source points and eliminate others using edge information. For each edge, the source points are the endpoints by which the edges may be recovered. Each source point includes at least two pixels located on both sides of the related edge and indicates the variations in the direction perpendicular to the edge direction. Then, the source point information is gathered row by row and stored in a row array for being encoded and appended to the edge encoder output in a specified template and ultimately the final encoded image is formed. Fig. 7 illustrates the encoding process for an instant image of 256×256 and 8 bpp (bits per pixel). In this example the number of source points is 6% of the total number of pixels in the image. The size of the output encoded image is 7.5% of the original image size with 0.6 bpp.

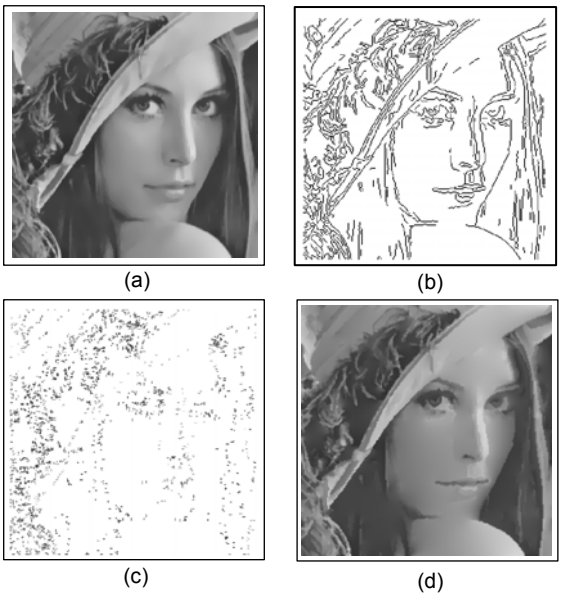

Fig. 7 Encoding and decoding processes
(a) Original image (256×256, 8 bpp); (b) Edges; (c) Source points; (d) Recovered image (0.6 bpp)

3.2 Decoder

At the decoder side, the codes related to the source points and edges are decoded independently. Similar to what is performed at the encoder, the location of the source points is found through edge information, and therefore the location of source points whose values are stored in a row array will be identified. The inverse of the encoder operation is done in the decoder and the source point values will be located on the appropriate coordinates. Finally, using the resulting source points and edge information, the eliminated parts of the original image will be recovered via the mentioned inpainting technique. Fig. 7d shows the output of the entire process.

4 Experimental results and discussion

Throughout the experiment, the following implementation details were considered. In accordance with Eq. (11), the noise removal block was accomplished using the Jacobi iterative method during 30 iterations. Sobel was chosen as the edge detection method and the threshold value was set manually. The edge detection block constructed a binary image, which was encoded through the JBIG algorithm. The source point extractor block took the image of edges as its input and identified the location of source points and labeled their two adjacent pixels as source points. Values of these points were then fed to the source point encoder in which the entropy coding was applied. As the edge image is required for finding the source points, choosing the edge threshold plays an important role in attaining high quality output. The less the threshold value is, the more edges will be extracted and the more source points and finally the better output will be obtained. However, small thresholds may produce spurious edges.

The source point placement block operation was similar to what has been done in the source point extractor at the encoder. In order to implement Eq. (9), the successive over relaxation (SOR) iterative algorithm was applied because of the sparse boundary condition in the inpainting problem.

For the given grayscale truncation error of 0.001 and compression ratio of 1:20 in a 256×256 image, the steady state mode was obtained after about 600 successive iterations. The number of iterations

depends on the number of eliminated points. The processing time during the simulation via Matlab R2007b, a Pentium Celeron 1.8 GHz processor and 512 MB RAM was about 4 s for the encoder and 40 s for the decoder. Note that the reported time was obtained from an implementation using Matlab codes, which are not optimum codes when the processing speed is of concern. Furthermore, in the decoding process, faster methods such as the Multigirid (Briggs *et al.*, 2000) can be implemented to accelerate the decoding process.

Test images were chosen from the USC-SIPI Standard Images Database (http://sipi.usc.edu/database/). We have tested our algorithm using gray level images. Hence, all color images in the database were converted to grayscale ones. Note that the presented method is applicable for RGB image compression as well, because the source point coordination for all channels remains the same while the image structures in color channels are similar to those in grayscale channels.

Fig. 8 shows the results obtained using the proposed algorithm for four typical samples: Splash, House, Airplane, and Peppers. The larger leftmost images in Figs. 8a–8d were the original images after noise removal, which acted as the input of each process. The first and second rows of each sample corresponded to the proposed algorithm and the JPEG compression technique with the same compression ratio, respectively. The first column was related to the compression ratio of 1:10 or 0.8 bpp. As was observed, there was no considerable distortion visible in all of the resulting images. The proposed algorithm worked properly for images with simple structures, like Splash, at high compression ratios such as 1:40, unlike JPEG which suffered from an undesirable blocking effect.

A conventional image quality index is the peak signal-to-noise ratio (PSNR), which is the ratio of the squared image intensity dynamic range to the mean squared difference of the original and distorted images. It is widely used for the estimation of quality in lossy image compression algorithms. This index is popular for its simplicity; however, it loses its advantages compared with natural human perception (Wang and Bovik, 2009). A better index for image quality measurement is the structural similarity (SSIM), calculated as

$$\mathrm{SSIM}=\frac{4\sigma_{XY}\mu_X\mu_Y}{(\sigma_X^2+\sigma_Y^2)(\mu_X^2+\mu_Y^2)}, \tag{12}$$

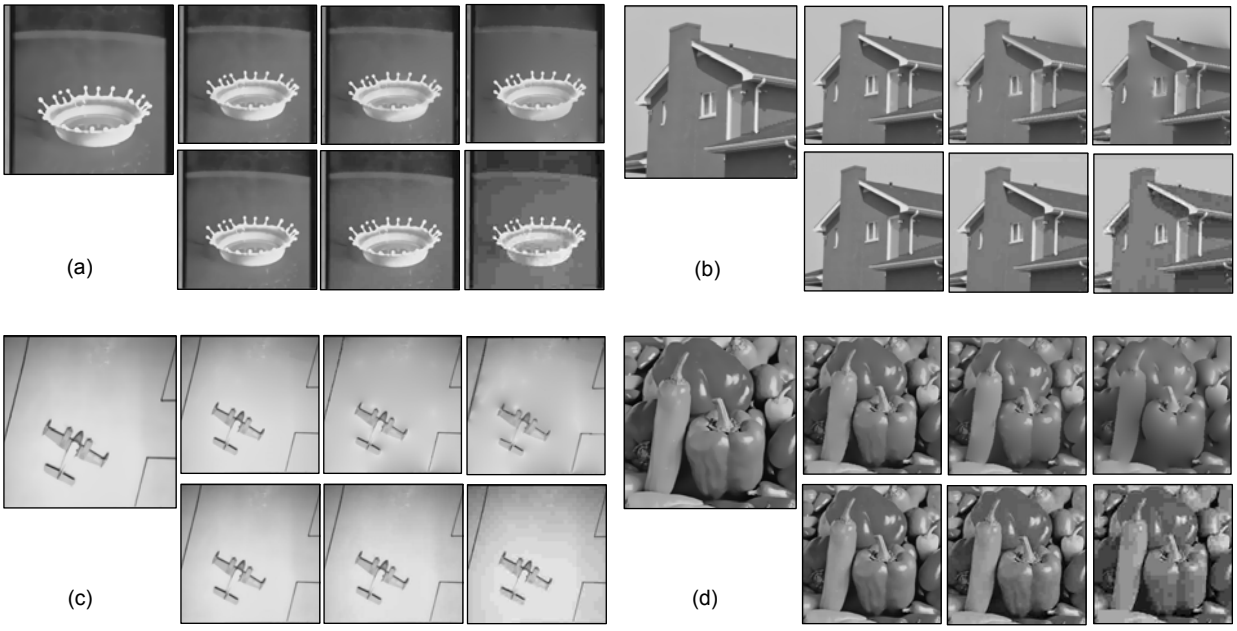

Fig. 8 Compression at different ratios through the proposed algorithm and JPEG

(a) Splash; (b) House; (c) Airplane; (d) Peppers. The larger leftmost image in each set is the original image at 8 bpp. In each set, the first and second rows show results of the proposed algorithm and JPEG respectively, and from left to right the compression ratios are 0.8, 0.4 and 0.2 bpp, respectively

where σ_X^2, σ_Y^2, μ_X and μ_Y are variances and means of the first and second images respectively, and σ_{XY} is the covariance between the images (see Wang *et al.* (2004) for details). The SSIM dynamic range is between −1 and 1, and the more SSIM, the more fidelity of images in comparison. SSIM is used for comparing structures of the distorted and the reference images. It is based on the fact that the HVS is highly sensitive to the structural information of the scene. Table 1 gives the PSNR and SSIM of the sample images in Fig. 8, showing that our proposed method outperformed JPEG at high compression ratios such as 1:40 (0.2 bpp) and were more outstanding in low structured and low textured images such as Splash.

Table 1 Comparison of PSNR and SSIM between JPEG and the proposed algorithm at different compression ratios for samples shown in Fig. 8

Image	Method	PSNR (dB)			SSIM		
		0.8 bpp	0.4 bpp	0.2 bpp	0.8 bpp	0.4 bpp	0.2 bpp
Splash	Proposed	32.83	30.07	28.48	0.9748	0.9631	0.9509
	JPEG	41.87	36.00	30.16	0.9859	0.9579	0.8780
House	Proposed	41.13	27.95	26.47	0.9916	0.9462	0.9220
	JPEG	41.15	35.31	29.53	0.9841	0.9532	0.8717
Airplane	Proposed	36.45	29.43	23.59	0.9940	0.9661	0.9471
	JPEG	46.48	38.94	32.67	0.9935	0.9763	0.9136
Peppers	Proposed	28.30	23.90	20.61	0.9266	0.8513	0.7832
	JPEG	35.40	31.00	36.31	0.9544	0.8890	0.7593

5 Conclusion

We propose a new method for compressing images, which selects image inpainting to remove spatial redundancy. In this framework, some kinds of distinctive features are extracted from images at the encoder side, and regions with high correlation values are intentionally skipped during encoding. The remaining areas along with information associated with the edges are encoded to form the compressed output data. Removed information is to be recovered with the assistance of information sent to the decoder side. At the decoder, using a PDE-based inpainting algorithm, the removed areas are recovered after the remaining regions and edges are decoded.

The results obtained from the implementations of the proposed algorithm show the possibility of reaching a compression ratio of 1:40 while keeping the output at an acceptable quality. Furthermore, resulting images in case of high compression ratios and inputs of simple structure are visually more acceptable compared with the JPEG algorithm. The JPEG compression method suffers from blocking drawback at high ratios; in contrast, our algorithm can obtain smoother output for correlated regions.

Further improvements of the current scheme are still promising. Edge extraction can be more flexible and adaptable for compression. Finding the regions that can be eliminated is considered to be an open problem and it seems that solving this problem will lead to further increase in compression ratios and output quality.

References

Bertalmio, M., 2006. Strong-continuation, contrast-invariant inpainting with a third-order optimal PDE. *IEEE Trans. Image Process.*, **15**(7):1934-1938. [doi:10.1109/TIP.2006.877067]

Bertalmio, M., Sapiro, G., Caselles, V., Ballester, C., 2000. Image Inpainting. Proc. Int. Conf. on Computer Graphics and Interactive Techniques, p.417-424. [doi:10.1145/344779.344972]

Bertalmio, M., Bertozzi, A.L., Sapiro, G., 2001. Navier-Stokes, Fluid Dynamics, and Image and Video Inpainting. Proc. IEEE Computer Society Conf. on Computer Vision and Pattern Recognition, **1**:355-362. [doi:10.1109/CVPR.2001.990497]

Briggs, W.L., Henson, V.E., McCormick, S.F., 2000. A Multi-Grid Tutorial (2nd Ed.). Society for Industrial and Applied Mathematics (SIAM), Philadelphia, p.1-85.

Chan, T.F., Shen, J., 2005a. Image Processing and Analysis. Society for Industrial and Applied Mathematics (SIAM), Philadelphia, p.277-279.

Chan, T.F., Shen, J., 2005b. Variational image inpainting. *Commun. Pure Appl. Math.*, **58**(5):579-619. [doi:10.1002/cpa.20075]

Criminisi, A., Perez, P., Toyama, K., 2004. Region filling and object removal by exemplar-based image inpainting. *IEEE Trans. Image Process.*, **13**(9):1200-1212. [doi:10.1109/TIP.2004.833105]

Galić, I., Weickert, J., Welk, M., Bruhn, A., Belyaey, A., 2008. Image compression with anisotropic diffusion. *J. Math. Imag. Vis.*, **31**(2-3):255-269. [doi:10.1007/s10851-008-0087-0]

Grossauer, H., 2004. A combined PDE and texture synthesis approach to inpainting. *LNCS*, **3022**:214-224. [doi:10.1007/b97866]

Höntsch, I., Karam, L.J., 2000. Locally adaptive perceptual image coding. *IEEE Trans. Image Process.*, **9**(9):1472-1483. [doi:10.1109/83.862622]

Liu, D., Sun, X.Y., Wu, F., Li, S.P., Zhang, Y.Q., 2007. Image compression with edge-based inpainting. *IEEE Trans. Circ. Syst. Video Technol.*, **17**(10):1273-1287. [doi:10.1109/TCSVT.2007.903663]

Malo, J., Epifanio, I., Navarro, R., Simoncelli, E.P., 2006. Nonlinear image representation for efficient perceptual coding. *IEEE Tran. Image Process.*, **15**(1):68-80. [doi:10.1109/TIP.2005.860325]

Patwardhan, K.A., Sapiro, G., Bertalmio, M., 2005. Video Inpainting of Occluding and Occluded Objects. IEEE Int. Conf. on Image Processing, **2**:69-72. [doi:10.1109/ICIP.2005.1529993]

Pennebaker, W.B., Mitchell, J.L., 1992. JPEG: Still Image Data Compression Standard. Kluwer Academic Publishers, Norwell, MA, USA.

Perona, P., Malik, J., 1990. Scale-space and edge detection using anisotropic diffusion. *IEEE Trans. Pattern Anal. Mach. Intell.*, **12**(7):629-639. [doi:10.1109/34.56205]

Rane, S.D., Sapiro, G., Bertalmio, M., 2003. Structure and texture filling-in of missing image blocks in wireless transmission and compression applications. *IEEE Trans. Image Process.*, **12**(3):296-303. [doi:10.1109/TIP.2002.804264]

Shen, J., Chan, T.F., 2002. Mathematical models for local nontexture inpaintings. *SIAM J. Appl. Math*, **62**(3):1019-1043. [doi:10.1137/S0036139900368844]

Tai, X.C., Osher, S., Holm, R., 2006. Image Inpainting Using a TV-Stokes Equation. Image Processing Based on Partial Differential Equations, p.3-22. [doi:10.1007/978-3-540-33267-1_1]

Taubman, D.S., Marcellin, M.W., 2002. JPEG 2000: Image Compression Fundamentals, Standards and Practice. Kluwer Academic Publishers, Norwell, MA, USA.

Wang, Z., Bovik, A.C., 2009. Mean squared error: love it or leave it? A new look at signal fidelity measures. *IEEE Signal Process. Mag.*, **26**(1):98-117. [doi:10.1109/MSP.2008.930649]

Wang, Z., Bovik, A.C., Sheikh, H.R., Simoncelli, E.P., 2004. Image quality assessment: from error visibility to structural similarity. *IEEE Trans. Image Process.*, **13**(4):600-612. [doi:10.1109/TIP.2003.819861]