Intra Prediction via Edge-Based Inpainting

Dong Liu*
University of Science and Technology of China
liud@mail.ustc.edu.cn

Xiaoyan Sun, Feng Wu
Microsoft Research Asia
{xysun, fengwu}@microsoft.com

Abstract

We investigate the usage of edge-based inpainting as an intra prediction method in block-based image compression. The joint utilization of edge information and the well-known Laplace equation yields a simple and effective inpainting algorithm. As for intra prediction, the edge-based inpainting is a uniform solution, yet adaptive to local image features. During the integration of edge-based inpainting into a block-based coding scheme, edge extraction and coding are jointly considered to achieve the rate-distortion optimization. Our proposed schemes are compared with JPEG2000, and experimental results demonstrate that both PSNR gain and visible quality improvement are achieved.

1 Introduction

Intra prediction is an important technique in block-based image compression, which exploits the spatial correlation between neighboring image blocks. For example, JPEG adopts the well-known DC prediction [1]. When coding the DC coefficient of the current block, the coded DC coefficients in previous blocks are utilized as a prediction, and only the difference between the real coefficient and the prediction is recorded. The state-of-the-art video coding standard, MPEG-4 AVC/H.264, has successfully taken advantage of intra prediction in its intra frame coding [2]. Specifically, H.264 adopts spatial prediction of pixel values, rather than prediction of transformed coefficients. Given a prediction method, encoder and decoder can both estimate the current block based on the previous reconstructed blocks. Therefore, only the differences between real pixel values and the predicted values need to be transmitted, known as residues. Residues are then transformed, quantized and entropy coded. Moreover, H.264 predefines some prediction modes, from which encoder can choose one for each block and signal the choice to decoder, where the rate-distortion optimization routine is often adopted to select the mode.

The intra prediction in H.264 has two noticeable characteristics. Firstly, its defined prediction modes are all directional, except for DC prediction. Actually, each prediction mode is designed for a kind of blocks with the same dominant direction in their pixel values. It is observed that such directions are often related to distinct image singularities, i.e. edges. Secondly, its prediction method is to copy neighboring pixel values into the current block along a specified direction, followed by a low-pass filtering. It is arguable that such simple prediction method is not enough to exploit the correlation. Some works have been done to find better predictions, such as template matching [3].

Recognizing the directional feature of the H.264 intra prediction, we propose a uniform and adaptive prediction solution that specially emphasizes edges. *Uniform* here

* This work has been done during D. Liu's internship at Microsoft Research Asia.

1068-0314/08 $25.00 © 2008 IEEE
DOI 10.1109/DCC.2008.18

means that edges are represented in the same form – binary map consisting of curves with one-pixel width – and the prediction strategy is unique given different shapes of edges. *Adaptive* here means that such edges are not predefined, but rather extracted according to local image features. In this work, we choose edge-based inpainting realized by Laplace equation as the prediction method, yet other edge-based inpainting or restoration methods are also possible to be used in prediction.

Image inpainting, after introduced by Bertalmio *et al.* [4], has attracted a lot of interest in computer vision and graphics. In the wide sense, inpainting refers to the filling-in of missing image/video data in the specified regions. There are two approaches to inpainting, based on partial differential equation (PDE) and texture synthesis, respectively. In the former, high order PDEs are designed to restore smooth regions as well as thin structures [4–6]. In the latter, exemplar-based synthesis, i.e. copy-and-paste with designated guidance and post-processing, is believed to restore not only salient structures but also textural coarseness [7–10]. Moreover, the two approaches are combined to deal with complex images [11, 12]. In our work, the edge-based inpainting is realized by Laplace equation, which is similar to the PDEs but more simple and computational efficient.

Inpainting has already been applied into image compression. Rane *et al.* [11] proposed a scheme which adopts inpainting in compression in a straight-forward manner. Some blocks are removed at the encoder and are restored by inpainting at the decoder, so as to increase the compression ratio. In [13], we proposed to use assistant information to empower such coding schemes. For instance, the encoder keeps edge information when removing blocks and the edges are transmitted to decoder to guide the inpainting. Due to the transmitted edges, inpainting can restore more complex image blocks and the restoration is more precise compared with the original image. All these works target at the improvement of compression ratio at similar subjective quality levels. On the contrary, this work aims at the objective quality, i.e. PSNR. Meanwhile, the special design for edges can also improve subjective quality as demonstrated by experimental results.

The remainder of this paper is organized as follows. In Section 2, we introduce the edge-based inpainting as an intra prediction method. Section 3 devotes to the edge extraction and edge coding. Section 4 presents experimental results and comparisons. Section 5 concludes this paper.

2 Edge-Based Adaptive Intra Prediction

In traditional inpainting scenarios, actually no additional information is provided for the regions to be filled-in. Such “blind” inpainting problems are generally ill-posed and quite difficult to be solved. Fortunately, in the compression scenario, original images are always available to the encoder, thus different kinds of distinct features can be extracted to assist inpainting. Due to the assistant information, inpainting becomes constrained and new inpainting methods should be invented to utilize the assistant information. For example, simple edges have been demonstrated valuable in the inpainting to restore structural regions [13]. In that work, an edge-based inpainting algorithm is proposed, which combines pixel-wise and patch-wise texture synthesis methods and imposes the edges as constraints.

In this paper, we propose to utilize the edge-based inpainting as an intra prediction method in the context of block-based compression. To be specific, the current block to be

Algorithm 1 Block prediction by edge-based inpainting

Step 1 **Initialization.** Fill the block with initial values (commonly gray-level 0), and mark all pixels in the block as unknown. Note that the pixels on edges are also unknown now.

Step 2 **Block segmentation and region generation.**

2.1 Find an unknown and non-edge pixel. If no such pixel exists, go to step 3.

2.2 Set the pixel found in 2.1 as the seed. By flooding method, get a connected unknown region in 4- or 8-neighboring manner *bounded by the edges*.

2.3 If the region found in 2.2 is connective to the reconstructed image regions, use Laplace equation to generate it; otherwise, fill the region with default values (commonly gray-level 128). All pixels in this region is known now, go back to 2.1.

Step 3 **Edge generation.** Use Laplace equation to generate the unknown pixels on the edges. Halt.

predicted is divided by edges into some unknown regions, each of which is independently generated by the Laplace equation (see Algorithm 1 for details). Compared our proposed intra prediction with that in H.264, one can find two differences. Firstly, edges are used in our method to replace the predefined directional modes. It is generally believed that edges represent salient image singularities, as well as kinds of directional information in pixel values. Moreover, edges are extracted according to local image features, instead of predefined, which makes the prediction adaptive to different image structures. Secondly, the prediction is generated by the Laplace equation, which is inspired by the PDE-based inpainting techniques and is believed to improve the accuracy of prediction.

In the remainder of this section, we will discuss the Laplace equation in detail. Figure 1 illustrates the notations: let I be the image definition domain and Ω be the region to be generated, a closed subset of I. From 2.2 of Algorithm 1 we know that the geometry of Ω is arbitrary since the edges can be in arbitrary shapes. From 2.3 we also know that partial boundary of Ω is known. Denote the known boundary by $\partial\Omega$, and the known pixel values by f^*. Our objective is to find an f defined on Ω.

A well-known solution to this problem is to minimize the following functional,

$$\min_f \iint_\Omega |\nabla f|^2 \text{ with } f|_{\partial\Omega} = f^*|_{\partial\Omega} \tag{1}$$

where $\nabla = (\frac{\partial}{\partial x}, \frac{\partial}{\partial y})$ is the 2-D gradient operator. The Euler-Lagrange equation corresponding to (1) is,

$$\Delta f = 0 \text{ over } \Omega \text{ with } f|_{\partial\Omega} = f^*|_{\partial\Omega} \tag{2}$$

where $\Delta = \frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2}$ is known as the Laplacian operator. Equation (2) is the Laplace equation with Dirichlet boundary condition, in which we are interested. In practice, it is solved by an iterative method, i.e. to generate a series of functions $f^{(t)}$ according to,

$$f^{(t+1)} = f^{(t)} + c^{(t)}(\Delta f)^{(t)}, t = 0,1,2,\ldots \tag{3}$$

Until convergence, i.e. $\left\|f^{(\tau+1)} - f^{(\tau)}\right\|$ is less than a threshold, the function $f^{(\tau+1)}$ is regarded as the solution to (2). In (3), $c^{(t)}$ is the step size that can vary in iterations but

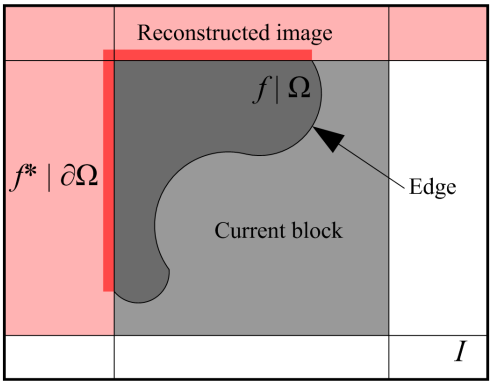


Figure 1. Laplace equation notations.

should be set to ensure the convergence. $f^{(0)}$ as the initial state should be explicitly assigned, such as step 1 of Algorithm 1 does.

To present the discrete form of (3), the Laplacian can be estimated by,

$$\mathcal{L}f(x,y) = \sum_{(i,j)\in\mu_4(x,y)} f(i,j) - 4f(x,y) \tag{4}$$

where $\mu_4(x,y)$ contains the four neighbors of the pixel (x,y). However, the four neighboring pixels are not always available in our scenario: they may be at the edges or inside the subsequent blocks. For simplicity, the unavailable pixels are assumed to have the same pixel value to $f(x,y)$. Equivalently, Equation (4) is replaced by,

$$\mathcal{L}f(x,y) = \sum_{(i,j)\in\mu_4(x,y)} \chi(i,j)f(i,j) - N_{xy}f(x,y) \tag{5}$$

where χ is an indicator function that evaluates 1 for available pixels and 0 for unavailable ones, N_{xy} is the number of available 4-neighboring pixels of the pixel (x,y). Then, the step size is set to,

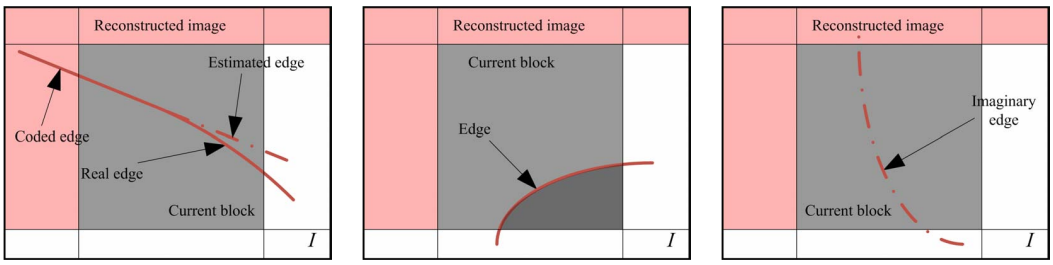
$$c^{(t)}(x,y) = c(x,y) = \frac{1}{N_{xy}} \tag{6}$$

And the iterative equation is now,

$$f^{(t+1)}(x,y) = \frac{1}{N_{xy}} \sum_{(i,j)\in\mu_4(x,y)} \chi(i,j)f^{(t)}(i,j) \tag{7}$$

which is virtually the Jacobi iteration. If the pixels are processed in scan-line order, $f^{(t)}(x-1,y)$ and $f^{(t)}(x,y-1)$ can be replaced by $f^{(t+1)}(x-1,y)$ and $f^{(t+1)}(x,y-1)$, respectively. This is known as the Gauss-Seidel iteration which is faster.

The Laplace equation (2) provides a simple estimation of the unknown region by assuming it to be smooth, or the Laplacians are zero. It seems trivial and indeed it is not able to recover image singularities. High order PDEs have been proposed to replace (2) in image inpainting. However, in our scheme, the salient edges have already been extracted and each unknown region divided by edges is independently generated. It is reasonable to



(a) (b) (c)

Figure 2. Several cases in edge-based inpainting: (a) real and estimated edges are quite similar – edge can be modified; (b) failure of the Laplace equation – edge can be erased; (c) inpainting with imaginary edge – edge can be added.

assume that each such unknown region is smooth enough to contain no singularity, thus (2) can be readily adopted. The experimental results also show satisfactory prediction with the Laplace equation. Moreover, PDE-based inpainting cannot restore the textural oscillations and so does the Laplace equation. Actually, these textures are put into the residue and compressed by transforms accordingly.

3 Rate-Distortion Optimized Edge Extraction and Coding

With the edge-based inpainting method available, the problem now is how to extract the required edges as well as how to code them efficiently. Traditionally, edges are extracted according to some mature methods provided by the computer vision researches [14]. Edge coding has also been extensively studied, such as chain coding [15], or lossless/lossy binary image coding known as JBIG/JBIG2 [16, 17]. It is possible to directly employ these tools for edge extraction and coding. However, such separate processes are not optimal from the compression point of view.

Consider the three cases shown in Figure 2. In (a), edge detectors present the real edge (shown as the solid line), partial of which has been coded with the previous blocks. According to the coded edge, encoder and decoder can both estimate an edge with for example Algorithm 2 (shown as the dash-dot line). The estimated edge is often quite similar to the real one. Therefore, on the one hand, the estimated edge can be directly used in inpainting. It may introduce a worse prediction compared to that using the real edge, but will cost no bits. On the other, encoder can still use the real edge for inpainting, but only need to record the difference between the estimated and the real edges. It often costs less bits compared to the direct coding of the real edge. In (b), there is an edge extracted by edge detectors crossing the bottom-right corner of a block. According to the inpainting algorithm, in such case the Laplace equation cannot generate the unknown region at the bottom-right corner of the block (colored by dark-gray in (b)). The edge may be erased to provide another prediction, which can be even better than that with the edge. Moreover, in (c), edge detectors have not found any edge in the block. However, since the inpainting method uses edges to divide the block into regions, it is possible that an imaginary edge can be imposed in order to derive a better prediction.

The examples shown in Figure 2 illustrate that edge can be modified, erased, or even added, in order to help inpainting give a better prediction, as well as to cost less bits to code. In other words, edge can be revised to achieve better rate-distortion performance. We can draw an analogy between the edges in our scheme and the motion vectors in

Table 1. Four modes of edge estimation

Mode	Description	Remark
0	Not use edge	
1	Use and code the real edge	Real edge coded by JBIG
2	Use the estimated edge	See Algorithm 2
3	Use the real edge, and code the difference between estimated and real edges	See Algorithm 2 Difference coded by JBIG

Algorithm 2 Estimate an edge

Step 1 **Identify the case.** Find the coded edges that stop at the boundary of the current block. If no such edge exists, go to step 3.

Step 2 **Case 1: Elongate a coded edge.** If more than one coded edges stop at the boundary, choose the one whose pixels have the maximum gradients. Elongate it in straight direction. Halt.

Step 3 **Case 2: Imagine an edge.**

3.1 From the boundary of the current block, choose the pixel that has the maximum gradient.

3.2 For the pixel found in 3.1, find in its 8-neighborhood one pixel that has the minimum difference from it.

3.3 Draw a straight line across the two pixels found in 3.1 and 3.2, respectively. Halt.

video coding. And similarly, we develop a rate-distortion optimized edge estimation algorithm, which jointly performs edge extraction and coding.

To be specific, before the input image is coded block by block, an edge detector is applied on the entire image [18], followed by a thinning process which makes edges to be curves with one-pixel width [13]. Then, for each block, the edge estimation will try the four modes listed in Table 1, and choose the one that has the minimum joint cost,

$$J_i = D_i + \lambda_{\text{edge}} R_i, i = 0,1,2,3 \quad (8)$$

where D_i is the sum of squared difference (SSD) between the prediction and the original image block, R_i is the bits to code the edge, and λ_{edge} is the Lagrange multiplier. The chosen mode will be coded into the bitstream for the decoder to perform edge estimation. Edges corresponding to the chosen mode are cached in a binary map at both encoder and decoder, in order to facilitate the edge estimation of the following blocks.

4 Experimental Results

Our proposed intra prediction method can be integrated into any existing block-based image or video compression systems. In this work, we have realized two image compression schemes, based on 8×8 block size and 8×8 DCT transform (similar to JPEG) and 16×16 block size and 4×4 DCT transform (similar to H.264), respectively. The two schemes are denoted as "EBI8×8" and "EBI16×16", respectively. In either scheme, the image is divided into non-overlapped blocks; each block is predicted by edge-based inpainting with the edge estimation method; the residues are DCT transformed, scalar quantized and entropy coded by arithmetic codec. The realized schemes are tested on a number of gray-scale images in comparison with the state-of-the-art image compression

standard JPEG2000[†]. Note that JPEG2000 is not block-based but rather adopts global wavelet transform, obviously different from our schemes.

Figure 3 shows the results with the EBI16×16 scheme on the image Cameraman (256×256). In (b) the detected and thinned edges are shown in a binary map. The used edges, predicted blocks and reconstructed images at different quality levels are shown in

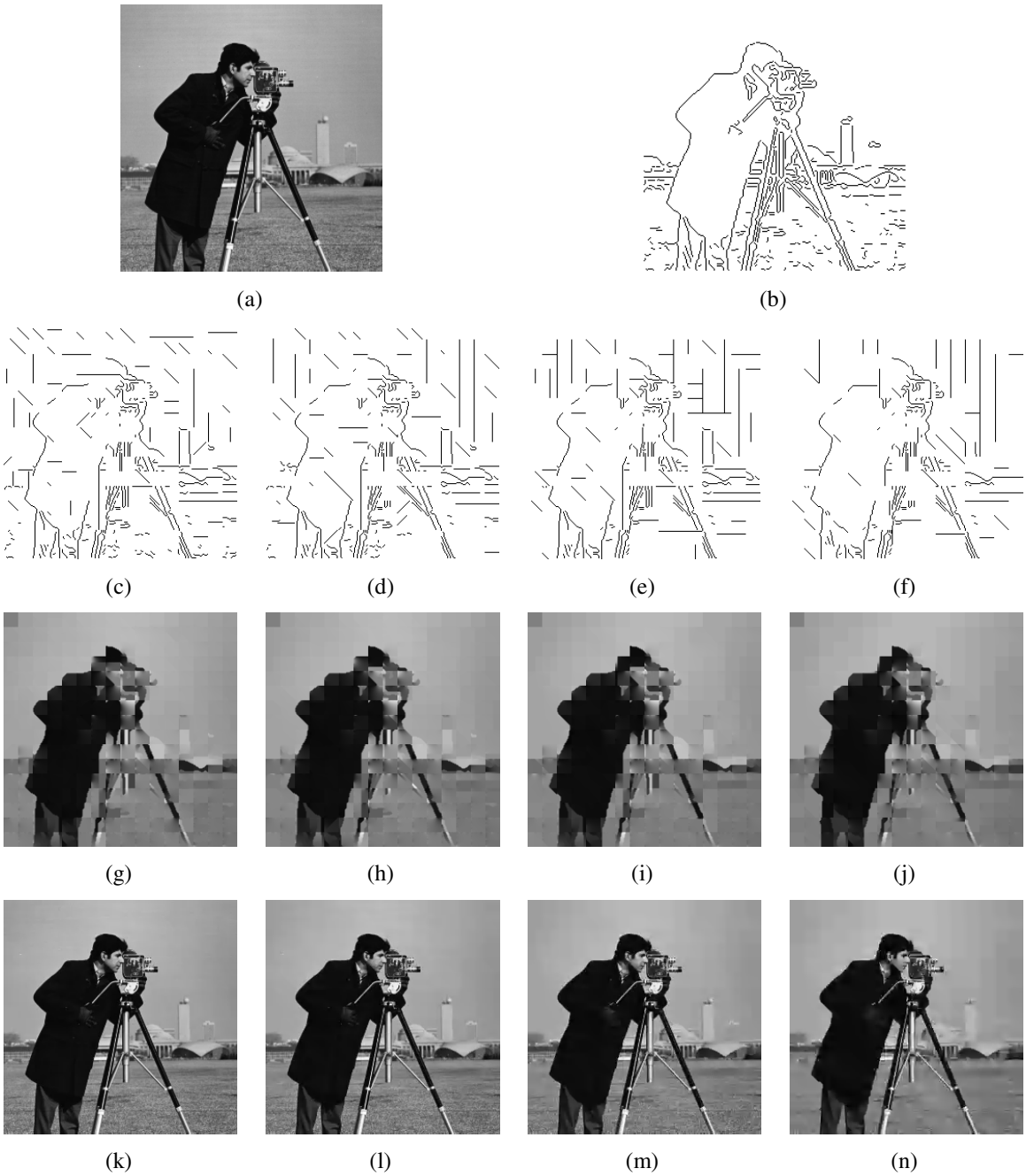


Figure 3. Results with the EBI16×16 scheme on the image Cameraman. (a) The original image; (b) detected and thinning edges; (c) – (n) from top to bottom: used edges, predicted blocks, and reconstructed images; from left to right: at different quality levels: 1.37bpp, 0.86bpp, 0.47bpp and 0.24bpp.

[†] JasPer codec. Available online: http://www.ece.uvic.ca/~mdadams/jasper/.

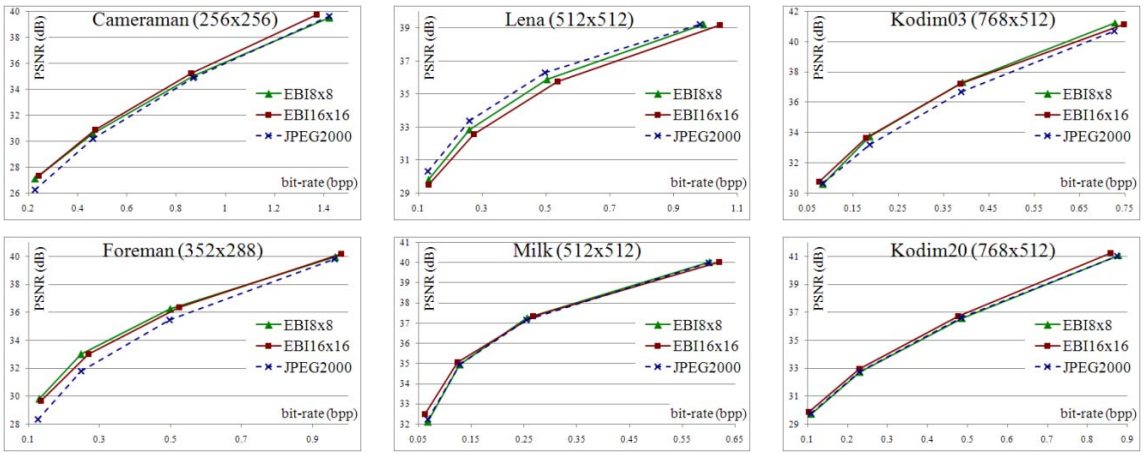


Figure 4. PSNR versus bit-rate comparisons with JPEG2000.

(c) – (f), (g) – (j) and (k) – (n), respectively. The clear difference between (b) and (c) – (f) is due to our edge estimation method. For example, the background looks smooth but actually contains gradations, therefore the edge estimation extracts some imaginary edges to help inpainting give better predictions. These edges are all straight lines because of our algorithm design. Likewise, some short pieces of edges are useless in inpainting and the edge estimation can automatically erase them. Moreover, the used edges are less when quality level is lower, which can be interpreted by two reasons. Firstly, due to the low quality of reconstructed image, inpainting does not work well even given the precise edges. Secondly, due to the rate-distortion optimized edge estimation, the Lagrange multiplier in (8) is larger at low quality level, which further punishes the bits for edge coding.

Figure 4 presents the coding efficiency comparisons for typical images: Cameraman (256×256), Foreman (352×288), Lena (512×512), Milk (512×512), and two images (768×512) from the Kodak image library[‡]. Generally, our two schemes are comparable to JPEG2000 in terms of the objective quality. For the Foreman image, as high as 1.4dB gain is achieved over JPEG2000 at low bit-rate. Only for the Lena image, our schemes have a little loss (less than 0.5dB in the case of EBI8×8). It is noticed that the Lena image contains rich textures, which are not well predicted by our designed edge-based inpainting. Moreover, the global wavelet transform in JPEG2000 is also more efficient in dealing with textures, compared to the block-based DCT transform in our schemes.

Figure 5 presents visual quality comparisons for the Cameraman, Foreman, Milk, and Kodim03 images at low bit-rates. The JPEG2000 reconstructed images generally suffer from ringing artifacts around the edges, which have been greatly eliminated in our reconstructed images (EBI16×16). This is not surprising, given the edge preserving nature of the intra prediction. Therefore, besides the PSNR values, our scheme can achieve better visual quality compared to JPEG2000. We would like to remind that textures are not well preserved by our scheme, see for example the Kodim03 image; and block-based coding always suffers from blockiness at low bit-rate, see for example the Milk image.

[‡] Available online: http://r0k.us/graphics/kodak/.

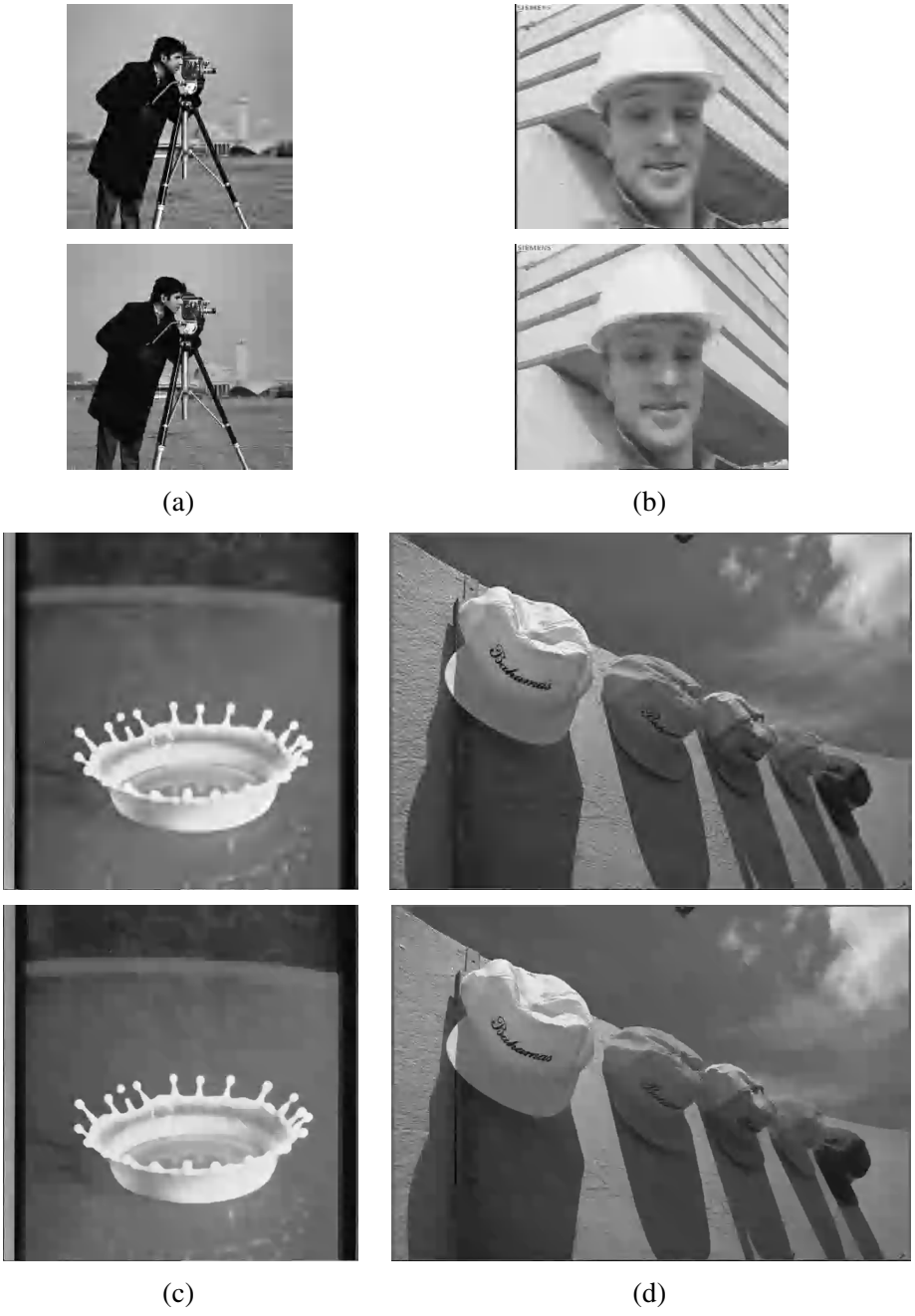


Figure 5. Visual quality comparisons between JPEG2000 (the top one) and our scheme EBI16×16 (the bottom one). (a) Cameraman at 0.24bpp; (b) Foreman at 0.13bpp; (c) Milk at 0.06bpp; (d) Kodim03 at 0.08bpp.

5 Conclusion

We have proposed an intra prediction method for block-based image compression. This prediction utilizes edge-based inpainting, which is realized by Laplace equation and thus simple and computational efficient. In order to drive the edge-based inpainting towards better rate-distortion performance, we have presented an edge estimation method which is capable to revise the traditionally extracted edges. In general, the intra predic-

tion is uniformly defined and it is adaptive to local image features, i.e. edges in this work. Experimental results show that block-based image compression with our proposed intra prediction is comparable to JPEG2000 in terms of PSNR. The preservation of edges also enhances the visual quality of the reconstructed images by our schemes. We remark that our proposed inpainting cannot handle textures now, which restricts the performance of the entire compression scheme when coding images with rich textures. We will investigate the texture synthesis methods to approach this problem in our future work.

References

[1] G. K. Wallace, "The JPEG still picture compression standard," *Communications of the ACM*, vol. 34, no. 4, pp. 30-44. Apr. 1991.

[2] T. Wiegand, G. J. Sullivan, G. Bjøntegaard, and A. Luthra, "Overview of the H.264/AVC video coding standard," *IEEE Trans. Circuits and Systems for Video Technology*, vol. 13, no. 7, pp. 560-576. Jul. 2003.

[3] T. K. Tan, C. S. Boon, and Y. Suzuki, "Intra prediction by template matching," in *Proc. IEEE International Conference on Image Processing (ICIP'06)*, pp. 1693-1696. Oct. 2006.

[4] M. Bertalmio, G. Sapiro, V. Caselles, and C. Ballester, "Image inpainting," in *Proc. ACM SIGGRAPH 2000*, pp. 417-424.

[5] C. Ballester, M. Bertalmio, V. Caselles, G. Sapiro, and J. Verdera, "Filling-in by joint interpolation of vector fields and gray levels," *IEEE Trans. Image Processing*, vol. 10, no. 8, pp. 1200-1211. Aug. 2001.

[6] T. F. Chan and J. Shen, "Mathematical models for local nontexture inpaintings," *SIAM Journal on Applied Mathematics*, vol. 62, no. 3, pp. 1019-1043. Feb. 2002.

[7] J. Jia and C.-K. Tang, "Image repairing: robust image synthesis by adaptive ND tensor voting," in *Proc. IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'03)*, pp. 643-650. Jun. 2003.

[8] I. Drori, D. Cohen-Or, and H. Yeshurun, "Fragment-based image completion," *ACM Trans. Graphics (SIGGRAPH 2003)*, vol. 22, no. 3, pp. 303-312. Jul. 2003.

[9] A. Criminisi, P. Pérez, and K. Toyama, "Region filling and object removal by exemplar-based image inpainting," *IEEE Trans. Image Processing*, vol. 13, no. 9, pp. 1200-1212. Sep. 2004.

[10] J. Sun, L. Yuan, J. Jia, and H.-Y. Shum, "Image completion with structure propagation," *ACM Trans. Graphics (SIGGRAPH 2005)*, vol. 24, no. 3, pp. 861-868. Jul. 2005.

[11] S. D. Rane, G. Sapiro, and M. Bertalmio, "Structure and texture filling-in of missing image blocks in wireless transmission and compression applications," *IEEE Trans. Image Processing*, vol. 12, no. 3, pp. 296-303. Mar. 2003.

[12] M. Bertalmio, L. Vese, G. Sapiro, and S. Osher, "Simultaneous structure and texture image inpainting," *IEEE Trans. Image Processing*, vol. 12, no. 8, pp. 882-889. Aug. 2003.

[13] D. Liu, X. Sun, F. Wu, S. Li, and Y.-Q. Zhang, "Image compression with edge-based inpainting," *IEEE Trans. Circuits and Systems for Video Technology*, vol. 17, no. 10, pp. 1273-1287. Oct. 2007.

[14] J. Canny, "A computational approach to edge detection," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 8, no. 6, pp. 679-698. Nov. 1986.

[15] H. Freeman, "Computer processing of line-drawing images," *ACM Computing Surveys*, vol. 6, no. 1, pp. 57-97. Mar. 1974.

[16] ISO/IEC 11544/ITU-T T.82. Mar. 1993.

[17] ISO/IEC 14492/ITU-T T.88. Feb. 2000.

[18] C. A. Rothwell, J. L. Mundy, W. Hoffman, and V.-D. Nguyen, "Driving vision by topology," in *Proc. International Symposium on Computer Vision*, pp. 395–400. Nov. 1995.