Image Compression With Edge-Based Inpainting

Dong Liu, Xiaoyan Sun, *Member, IEEE*, Feng Wu, *Senior Member, IEEE*, Shipeng Li, *Member, IEEE*, and Ya-Qin Zhang, *Fellow, IEEE*

***Abstract*—In this paper, image compression utilizing visual redundancy is investigated. Inspired by recent advancements in image inpainting techniques, we propose an image compression framework towards visual quality rather than pixel-wise fidelity. In this framework, an original image is analyzed at the encoder side so that portions of the image are intentionally and automatically skipped. Instead, some information is extracted from these skipped regions and delivered to the decoder as assistant information in the compressed fashion. The delivered assistant information plays a key role in the proposed framework because it guides image inpainting to accurately restore these regions at the decoder side. Moreover, to fully take advantage of the assistant information, a compression-oriented edge-based inpainting algorithm is proposed for image restoration, integrating pixel-wise structure propagation and patch-wise texture synthesis. We also construct a practical system to verify the effectiveness of the compression approach in which edge map serves as assistant information and the edge extraction and region removal approaches are developed accordingly. Evaluations have been made in comparison with baseline JPEG and standard MPEG-4 AVC/H.264 intra-picture coding. Experimental results show that our system achieves up to 44% and 33% bits-savings, respectively, at similar visual quality levels. Our proposed framework is a promising exploration towards future image and video compression.**

***Index Terms*—Edge extraction, image compression, image inpainting, structure propagation, texture synthesis, visual redundancy.**

I. Introduction

OVER THE LAST two decades, great improvements have been made in image and video compression techniques driven by a growing demand for storage and transmission of visual information. State-of-the-art JPEG2000 and MPEG-4 AVC/H.264 are two examples that significantly outperform their previous rivals in terms of coding efficiency. However, these mainstream signal-processing-based compression schemes share a common architecture, namely transform followed by entropy coding, where only the statistical redundancy among pixels is considered as the adversary of coding. Through two decades of development, it has been becoming difficult to continuously improve coding performance under such architecture. Specifically, to achieve high compression performance, more and more modes are introduced to deal with regions of different properties in image and video coding. Consequently, intensive computational efforts are required to perform mode selection subject to the principle of rate-distortion optimization. At the same time, more and more memory-cost context models are utilized in entropy coding to adapt to different kinds of correlations. As a result, small improvements in coding efficiency are accomplished with great pain of increased complexity in both encoder and decoder.

Besides statistical redundancy, visual redundancy in videos and images has also been considered in several works. They are motivated by the generally accepted fact that minimizing overall pixel-wise distortion, such as mean square error (MSE), is not able to guarantee good perceptual quality of reconstructed visual objects, especially in low bit-rate scenarios. Thus, the human vision system (HVS) has been incorporated into compression schemes in [1] and [2], trying to remove some visual redundancy and to improve coding efficiency as well as visual quality. Moreover, attempts have been made to develop compression techniques by identifying and utilizing features within images to achieve high coding efficiency. These kinds of coding approaches are categorized as "second-generation" techniques in [3], and have raised a lot of interest due to the potential of high compression performance. Nevertheless, taking the segmentation-based coding method as an example, the development of these coding schemes is greatly influenced by the availability as well as effectiveness of appropriate image analysis algorithms, such as edge detection, segmentation, and texture modeling tools.

Recently, technologies in computer vision as well as computer graphics have shown remarkable progress in hallucinating pictures of good perceptual quality. Indeed, advancements in structure/texture analysis [4], [5] and synthesis are leading to promising efforts to exploit visual redundancy. So far, attractive results have been achieved by newly presented texture synthesis techniques to generate regions of homogeneous textures from their surroundings [6]–[14]. Furthermore, various image inpainting methods have been presented, aiming to fill-in missing data in more general regions of an image in a visually plausible way. In fact, the word *inpainting* was initially invented by museum or art restoration workers. It is first introduced into digital image processing by Bertalmio *et al.* [15], where a third order partial differential equation (PDE) model is used to recover missing regions by smoothly propagating information from the surrounding areas in isophote directions. Subsequently, more models are introduced and investigated in image inpainting, e.g., total variation (TV) model [16], coupled second order PDE model taking into account the gradient orientations [17], curvature-driven diffusion (CDD) model [18], and so on. All these approaches work at pixel level and are good at recovering small flaws and thin structures. Additionally, exemplar-based

Manuscript received August 25, 2006; revised December 11, 2006. This paper was recommended by Associate Editor T. Nguyen.

D. Liu was with Microsoft Research Asia, Beijing 100081, China. He is now with the University of Science and Technology of China, Hefei 230027, China (e-mail: liud@mail.ustc.edu.cn).

X. Sun, F. Wu, S. Li, and Y.-Q. Zhang are with Microsoft Research Asia, Beijing 100081, China (e-mail: xysun@microsoft.com; fengwu@microsoft.com; spli@microsoft.com; yzhang@microsoft.com).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TCSVT.2007.903663

1051-8215/$25.00 © 2007 IEEE

approaches have been proposed to generate textural coarseness; by augmenting texture synthesis with certain automatic guidance, edge sharpness and structure continuity can also be preserved [19]–[21]. Combining PDE diffusion and exemplar-based synthesis presents more encouraging inpainting results in [24]–[26]. Moreover, inpainting capability is further improved by simple human interactions when human knowledge is borrowed to imagine what unknown regions should be, so that the restoration results look natural to viewers [22], [23].

Due to its potential in image recovery, image inpainting likewise provides current transform-based coding schemes another way to utilize visual redundancy in addition to those that have been done in [1]–[3]. This inference has been successfully exemplified in error concealment when compressed visual data is transmitted over error-prone channels [26], [27]. Moreover, it has been reported that improvement is achieved by employing image inpainting techniques in image compression even though in a straightforward fashion [26]. Besides, image compression also brings new opportunities to image inpainting, as we have pointed out in [32]. Since the complete source images are available, many kinds of assistant information can be extracted to help inpainting deal with complex regions that contain structures or other features and which are unable to be properly inferred from the surroundings. Thus, inpainting here becomes a *guided* optimization for visual quality instead of a *blind* optimization for image restoration. Accordingly, new inpainting techniques may be developed to better serve image compression.

When image inpainting and image compression are jointly considered in an integrated coding system, two main problems need to be addressed. The first: What should be extracted from a source image as assistant information to represent important visual information? The second: How to reconstruct an image with this assistant information? On the one hand, it has been reported that using different image analyzers, various kinds of assistant information can be extracted, including edge, object, sketch [5], epitome [29], [30], and so on, to represent an image or portion of an image. Then, given a specific kind of assistant information, the corresponding restoration method should be developed to complete a desired reconstruction by making full use of it. On the other, from the compression point of view, the effectiveness of restoration methods as well as the efficiency of the compression of assistant information would also influence the choice of assistant information. Such dependency makes the problems more complicated.

In this paper, we propose an image coding framework in which currently developed vision techniques are incorporated with traditional transform-based coding methods to exploit visual redundancy in images. In this scheme, some regions are intentionally and automatically removed at the encoder and are restored naturally by image inpainting at the decoder. In addition, binary edge information consisting of lines of one-pixel width is extracted at the encoder and delivered to the decoder to help restoration. Techniques, including edge thinning and exemplar selection are proposed, and an edge-based inpainting method is presented in which distance-related structure propagation is proposed to recover salient structures, followed by texture synthesis. The basic idea of this paper has been discussed in our conference papers [31] and [32]. However, some problems have not been investigated carefully in those papers, including questions such as why the edges of image are selected as assistant information, or how to select the exemplar blocks automatically, and so on.

The remainder of this paper is organized as follows. In Section II, we introduce the framework of our proposed coding scheme. We also discuss on the necessity and importance of the assistant edge information via image inpainting models. The key techniques proposed for our coding approach are described in Sections III and IV. Specifically, Section III shows the edge extraction and exemplar selection methods, and the edge-based image inpainting is proposed in Section IV. Section V presents experimental results in terms of bit-rate and visual quality. In Section VI, we conclude this paper and discuss future work.

II. Framework of Our Proposed Image Compression Scheme

As the basic idea of "encoder removes whereas decoder restores" has been mentioned in literature for image compression [26], [28], we would like to point out the novelties of our proposed method here. First, in our approach, the original image is not simply partitioned into two parts: one is coded by conventional transform-based approach, and the other is skipped during encoding and restored during decoding. Instead, techniques for image partition, block removal, and restoration in our proposed scheme are carefully designed towards compression rather than straightforward adoption. Furthermore, skipped regions will not be completely dropped at the encoder side if they contain portion of information that is difficult to be properly recovered by conventional image inpainting methods. In fact, assistant information is extracted from the skipped regions to guide the restoration process and further induce new inpainting techniques.

The framework of our proposed compression scheme is depicted in Fig. 1. In this scheme, an original image is first analyzed at the encoder side. The "image analysis" module automatically preserves partial image regions as exemplars and sends them to the "exemplar encoder" module for compression using conventional approaches. Meanwhile, it extracts designated information from skipped regions as assistant information and sends it to the "assistant info encoder" module. Then, the coded exemplars and coded assistant information are banded together to form final compressed data of this image. Correspondingly, at the decoder side, exemplars and assistant information are first decoded and reconstructed. Then, the regions skipped at the encoder are restored by image inpainting based on the twofold information. At the end, the restored regions are combined with the decoded exemplar regions to present the entire reconstructed image.

Fig. 1 shows a general framework of the proposed compression scheme that does not constrain which kind of assistant information should be used there. Since source image is always available at the encoder side, there are many choices of assistant information extracted from the skipped regions, e.g., semantic object, visual pattern, complete structure, simple edges, and so on. Here we start from the mathematical models in image

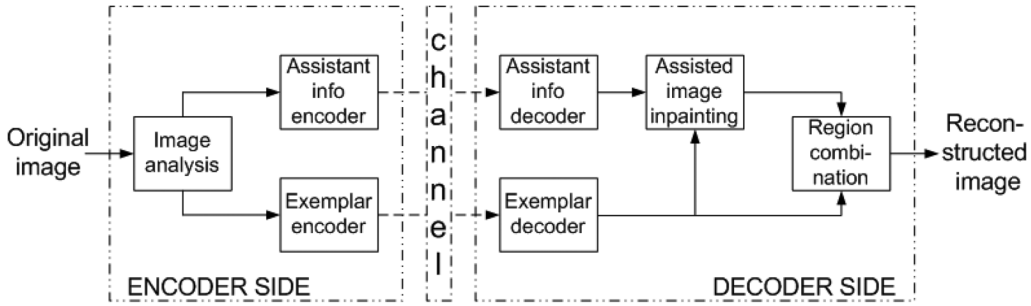


Fig. 1. The framework of our proposed image compression scheme.

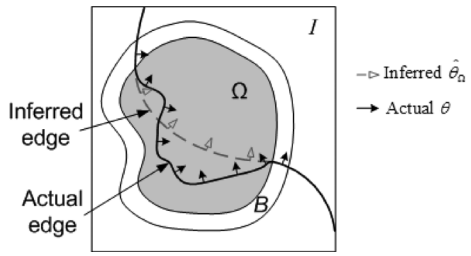


Fig. 2. Illustration of image inpainting, where the gray region is to be restored.

inpainting to discuss what on earth the assistant information should be.

As shown in Fig. 2, suppose that we are given an image function $f(x)$, $x \in I$, where I is a square region in $\mathbb{R}^2$. Ω, depicted as the gray region in Fig. 2, is an open bounded subset of I with Lipschitz continuous boundary. It is just the region to be restored by image compression, image inpainting, or a combination of them. This restoration problem can be generalized as

$$\arg\min\left(\int_{\Omega} D(f_\Omega(x) - \hat{f}_\Omega(x))dx + \lambda R\right). \tag{1}$$

Here, $f_\Omega(x)$ is the original image function in Ω, where it should satisfy $f_\Omega(x) = f(x)$ for any $x \in \Omega$. $\hat{f}_\Omega(x)$ is a reconstruction of $f_\Omega(x)$ at decoder. λ is a Lagrange factor. Clearly, (1) is to find the optimal function $\hat{f}_\Omega(x)$ by minimizing the joint cost consisting of reconstructed distortion $D()$ and coding bits R for Ω. Thus, image compression and image inpainting can be viewed as two extreme cases of (1). Specifically, in traditional image compression, $f_\Omega(x)$ is directly coded and sent to the decoder, where many bits may be needed to represent $f_\Omega(x)$; whereas in image inpainting, there is no bit to represent $f_\Omega(x)$ since $\hat{f}_\Omega(x)$ is inferred from $f_{I\backslash\Omega}(x)$. However, our proposed method, which is quite different from compression or inpainting, can be granted as a combination of them.

In typical inpainting scenarios, the restoration of $f_\Omega(x)$ is usually an ill-posed problem because information in Ω is totally unknown. Fortunately, an image is a 2-D projection of the 3-D real world. The lost region often has similar statistic, geometric and surface reflectivity regularities as those in the surroundings. It makes the above ill-posed problem possible to be solved. Therefore, some models are introduced in image inpainting to characterize statistic, geometric and surface regularities. These models should employ generic regularities, rather than rely on a specific class of images so that model-based inpainting can be applied in generic images.

One such model, TV model, is presented in [16] for image inpainting, in which the variation regularity is first introduced. Since local statistical correlation usually plays an more important role than the global one, as shown in Fig. 2, B instead of $I\backslash\Omega$ is used to infer the regularities in Ω, where B is a band around Ω. Then, the TV model is to find a function $\hat{f}_\Omega(x)$ on the extended inpainting region $B \bigcup \Omega$ such that it minimizes the following energy function:

$$\arg\min\left(\int_{B\bigcup\Omega} |\nabla \hat{f}_\Omega(x)|dx + \lambda \int_B |\hat{f}_\Omega(x) - f(x)|^2 dx\right). \tag{2}$$

The first term in (2) is to measure local homogeneity of image function in the region $B \bigcup \Omega$, and the second term, called as fidelity term, is the sum of squared difference (SSD) between the reconstructed B in $\hat{f}_\Omega(x)$ and the original B in $f(x)$. Equation (2) can be solved by the Euler–Lagrange method described in [16]. Accordingly, TV inpainting is good at restoring homogenous regions. But, if the lost region contains rich structures, it does not work well, especially when structures are separated far apart by the lost region.

To solve it, another parameter θ is introduced in the inpainting model [17]. Let θ be the vector field of normalized gradient of $f(x)$. $\hat{\theta}_\Omega$ is the corresponding parameter to be restored on Ω. With the new parameter of gradient directions, the inpainting problem is posed as extending the pair of functions (f, θ) on B to a pair of functions $(\hat{f}_\Omega, \hat{\theta}_\Omega)$ on Ω. It is completed by minimizing the following function:

$$\arg\min\left(\int_{B\bigcup\Omega} |\mathrm{div}(\hat{\theta}_\Omega(x))|^p (a + b|\nabla k * \hat{f}_\Omega(x)|)dx + \alpha \int_{B\bigcup\Omega} (|\nabla \hat{f}_\Omega(x)| - \hat{\theta}_\Omega(x) \cdot \nabla \hat{f}_\Omega(x))dx\right). \tag{3}$$

The first term presents smooth continuation demand on $\hat{\theta}_\Omega$, where a and b are positive constants, and k is a smoothing kernel. It is the integral of the divergence (in L^p function space) of the vector field $\hat{\theta}_\Omega$, with respect to the gradients of the smoothed $\hat{f}_\Omega(x)$. The second term is an constraint between $\hat{\theta}_\Omega$ and $\hat{f}_\Omega(x)$, where α is a positive weighing factor. $\hat{\theta}_\Omega$ should be related to $\hat{f}_\Omega(x)$ by trying to impose $\hat{\theta}_\Omega \cdot \nabla \hat{f}_\Omega = |\nabla \hat{f}_\Omega|$. The use of the vector field θ is the main point of the model given in (3). Thus, it enables image inpainting to restore missing regions

Fig. 3. Comparison with baseline JPEG on test image Lena. (a) Original image. (b) Edge map (blue curves). (c) Removed blocks (black blocks) and assistant edge information (blue curves), note that the assistant edge information is a subset of the entire edge map. (d) Reconstructed image after structure propagation. (e) Reconstructed image after texture synthesis. (f) Reconstructed image by baseline JPEG.

by continuing both the geometric and photometric regularities of images.

However, the model in (3) assumes that the parameter $\hat{\theta}_\Omega$ can be inferred from θ under a certain smooth constraint. But this assumption is not always true for nature images. Taking Fig. 2 as an example, the area to be restored consists of two homogenous regions divided by an edge denoted by the solid curve. The dashed curve is the inferred edge in Ω according to (3), which is quite different from the actual one. This problem is hard to be solved in conventional inpainting scenarios even using human intelligence as proposed in [23]. Therefore, in our proposed coding framework, assistant information should be used to correctly infer $\hat{\theta}_\Omega$ on Ω. As we have discussed, $\hat{\theta}_\Omega$ is the vector field of normalized gradient and is independent from the absolute magnitudes of gradients. It contains two parts of information: where $\hat{\theta}_\Omega$ exists and what its direction is. Commonly, it can be simply represented by binary edges of one-pixel width for the purpose of efficient compression.

Consequently, edge information is selected as assistant information for image inpainting in this paper. With assistant information, we could remove more regions in an image. Thus, it greatly enhances the compression power of our method. Since edges are low-level features in image, there are some mature tools available to automatically track them in an image. Moreover, edge information is concise and easy to describe in compressed fashion. Therefore, the employment of edge information can, on the one hand, help preserving good visual quality of the reconstructed image. On the other, it enables high compression performance by removing some structural regions and efficiently coding edge information.

Accordingly, an overview of our approach is exemplified in Fig. 3. In this figure, (a) is the input original image Lena. After image analysis, an edge map [denoted by blue curves in (b)] is generated, based on which the exemplars [denoted by the non-black blocks in (c)] are selected. Consequently, the exemplars and the needed edge information [shown as blue curves in (c)] will be coded into bit-stream. Then, at the decoder side, the edge information is utilized to guide the structure propagation for the recovery of edge-related regions. The corresponding result is given in (d). The remainder unknown regions will be restored by texture synthesis. The final reconstructed image after region combination is given in (e).

In the following two sections, we will explain the modules in our framework in detail, especially on the two most important modules, namely image analysis and assisted image inpainting. Here, we would like to emphasize that the introduction of assistant edge information raises different demands on both the encoder and decoder. We deal with them comprehensively in this paper.

III. Edge Extraction and Exemplar Selection

The image analysis module at the encoder side consists of two sub-modules: The first is to extract edge information from image and the second is to select exemplar and skipped regions at block level according to available edge information. They are discussed in the following two subsections.

A. Edge Extraction

As discussed in Section II, edge information plays an important role in the proposed coding scheme. It assists the encoder to select exemplar and skipped regions and the decoder to restore skipped regions with our proposed edge-based inpainting. Extracted edges do not need to represent complete and continuous topological properties of an image because our purpose is not to segment or restore an object. Discontinuous edges can likewise play the role of assistant information in the proposed scheme. But taking the topological properties into account in edge extraction will make edges more meaningful in terms of low-level vision.

Therefore, though there are many mature tools available to extract edges from images, the topology-based algorithm presented in [33] is adopted in our system to extract assistant

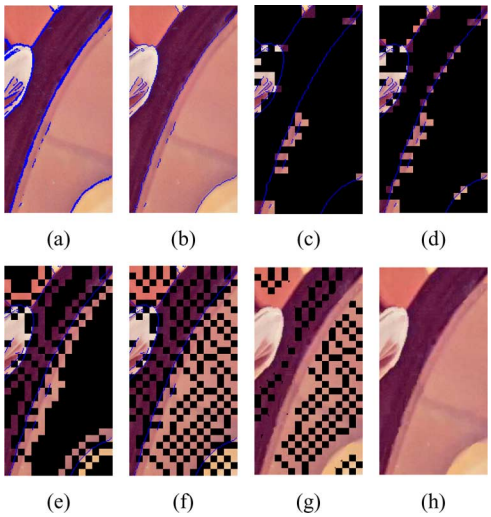


Fig. 4. Step-wise results of our scheme on test image Lena, zoomed-in partition. (a) Original image with detected edge pixels (blue curves). (b) Thinned edges (blue curves). (c) Chosen necessary structural blocks. (d) Chosen additional structural blocks. (e) Chosen necessary textural blocks. (f) Chosen additional textural blocks. (g) Structure propagation result. (h) Texture synthesis result.

information. The algorithm presents good results especially on extracting intersection edges. According to this method, an input image is first smoothed by a two-dimensional isotropic Gaussian filter so as to avoid noise. Second, $|\nabla f(x)|$ and θ are calculated on the filtered image for each pixel x. If $|\nabla f(x)|$ is the local maximum gradient along the direction θ and larger than a threshold, then pixel x belongs to an edge. At last, the pixels with non-maximum gradients are checked by spatially-adapted thresholds to prevent missing edges caused by the unreliable estimation of θ.

As shown in Fig. 4(a) by blue curves, edges extracted with the above algorithm (or most of the existing methods as well) are often of more than one-pixel width. This causes ambiguous directions in guiding the restoration at the decoder side and also increases the number of bits to code the edge information. Although [33] also proposes a thinning method, it does not satisfy the special requirement in our proposed edge-based inpainting. It is because that pixel values on edges are not coded but rather inferred from connected surrounding edges in our proposed scheme. Thus, a new thinning method is proposed here by taking into account the consistence of pixel values on edges as well as the smoothness of edges.

Here, we present the details of our proposed thinning method. Given the detected edge pixels, we first group them into eight connective links and each edge-link (also known as a connected component in the graph that is made up by edge pixels) is thinned independently. Complying with the terminologies defined in Section II, our goal is to find a one-pixel-width line which contains N pixels, i.e., $f(x_n)$ for $n = 1, 2, \ldots, N$, yielding the minimal energy

$$J = \alpha \sum_{n=1}^{N} |\Delta f(x_n)| + \beta \sum_{n=1}^{N} \sum_{k=1}^{N} d(f(x_n), f(x_k)) + \gamma \sum_{n=1}^{N} |\kappa(x_n)|^p \quad (4)$$

where α, β and γ are positive weighting factors. The energy function (4) consists of three terms. The first term is the Laplacian of each edge pixel. The second term is the constraint on pixel values of all edge pixels. After thinning, remaining edge pixels should have similar values. To make this constraint as simple as possible, only the difference among eight neighboring pixels are considered, and the function $d()$ is defined as

$$d(f(x_n), f(x_k)) = \begin{cases} |f(x_n) - f(x_k)|, & \text{if } x_k \in \mu_8(x_n) \\ 0, & \text{otherwise} \end{cases} \quad (5)$$

$\mu_8(x_n)$ denotes the 8-neighbor of x_n. The last term of (4) evaluates the curvature of the edge at each pixel. Similar to [17], [18], $\kappa(x_n)$ is defined as

$$\kappa(x_n) = \operatorname{div}\left(\frac{\nabla f(x_n)}{|\nabla f(x_n)|}\right). \quad (6)$$

In addition, we want to emphasize that the thinning process should not shorten the edge, thus only redundant pixels on the edge can be removed.

The optimal thinning solution for each edge-link is obtained through dynamic programming algorithm. Given a start point of each edge-link, the energies of all possible paths, linked in eight connective manner, are calculated according to (4). Referring to the width of the initial edge-link, several paths with smaller energies are recorded in the dynamic programming. Then, each recorded path is extended consequently by adding one neighbor pixel which results in the minimal energy. Note that the thinning algorithm can be performed in parallel manner for all edge-links in an image, because they are independent in terms of thinning process. Fig. 4(b) presents the corresponding thinning results using our proposed method.

B. Exemplar Selection

After edges are extracted, exemplar selection is performed based on these available edges. Here, for simplicity, the exemplar selection process is performed at block level. Specifically, an input image is first partitioned into non-overlapped 8×8 blocks, and each block is classified as *structural* or *textural* according to its distance from edges. In detail, if more than one-fourth of pixels in a block are within a short distance (e.g., five-pixel) from edges, it is regarded as a structural block, otherwise a textural one. Then, different mechanisms are used to select the exemplars for textural blocks and structural blocks. Blocks that are not selected as exemplars will be skipped during encoding. Moreover, exemplar blocks are further classified into two types, the *necessary* ones and the *additional* ones, based on their impacts on inpainting as well as on visual fidelity. Generally, one image can not be properly restored without necessary exemplar blocks, whereas additional blocks help to further improve visual quality.

1) Textural Exemplar Selection: Fig. 5(a) illustrates the process of exemplar selection for textural blocks. In this figure, edge information is denoted by thickened lines, based on which the image is separated into structural regions (indicated by gray blocks) and textural regions (indicated by white and black blocks).

It is generally accepted that pure textures can be satisfactorily generated even given a small sample. However, in practice,

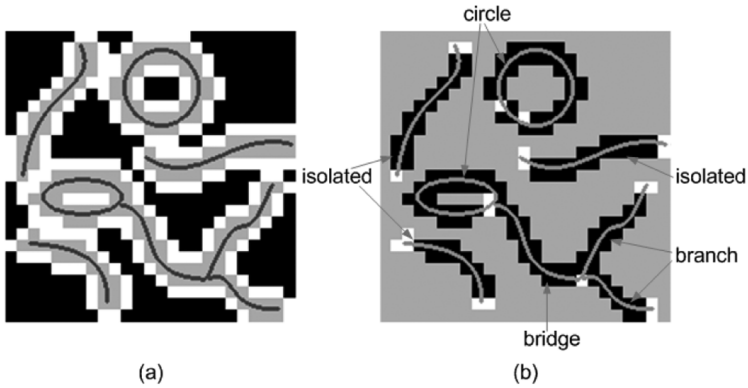


Fig. 5. An example of necessary exemplar selection in which curves denote edges and black blocks denote skipped regions. (a) Textural exemplar selection in which white blocks are necessary textural exemplars; (b) structural exemplar selection in which white blocks are necessary structural exemplars, four types of edges are also distinguished in (b). Also see Fig. 4 for a practical example.

image regions are often not pure textures, but rather contain kinds of local variations, such as lighting, shading, and gradual changing. Furthermore, exemplar-based texture synthesis is sensitive to the chosen samples. In image inpainting, a common solution to unknown textural regions is to synthesize them from samples in their neighborhood.

In our scheme, the necessary textural exemplars are selected in the border of textural regions. That is, as shown in Fig. 5(a), denoted by white blocks, if a textural block is next to a structural one, along either horizontal or vertical direction, it is considered as necessary. Such blocks are selected because they contain the information of transitions between different textural regions, which are hard to be restored by inner samples. Besides, propagation of these blocks, from outer to inner, can reconstruct the related textural regions.

To further improve visual quality of reconstructed images, additional blocks can be progressively selected to enrich exemplars. In this process, we consider additional blocks as representatives of local variations. On the one hand, if a block contains obvious variation, it should be preserved in advance. On the other, because the variation is a local feature, removing large-scale regions should be avoided in exemplar selection. Thus, each non-necessary textural block B_i is related to a variation parameter V_i defined as

$$V_i = w_1 Var(B_i) + w_2 \sum_{B_j \in \mu_4(B_i)} |E(B_i) - E(B_j)|. \quad (7)$$

Here w_1 and w_2 are positive weighting factors. $\mu_4()$ indicates 4-neighbor of a certain block. The functions $Var()$ and $E()$ are the variance and mean value of the pixel values in a block, respectively. In our system, according to an input ratio, the blocks with higher variation parameters will be selected, during which we also check the connective degree of each block so that the removed blocks do not constitute a large region.

2) Structural Exemplar Selection: Fig. 5(b) shows the exemplar selection method for structural blocks. In this figure, edges are represented by lines with indicated different types, and structural regions are indicated in white and black blocks, whereas all textural regions in gray.

As we have discussed, besides many textural blocks, some structural blocks are also skipped at the encoder side and restored at the decoder side by the guidance of edge information. Therefore, necessary and additional structural exemplars are also selected based on available edges. To better introduce the method, edges are categorized into four types according to their topological properties, as indicated in Fig. 5(b): "isolated" edge traces from a free end (i.e., an edge pixel connected with only one other edge pixel) to another free end; "branch" edge traces from a free end to a conjunction (i.e., an edge pixel connected with more than three other edge pixels); "bridge" edge connects two conjunctions; and, "circle" edge gives a loop trace.

Commonly, edge acts as the boundary of different region partitions. For the sake of visual quality, in image inpainting, two textural partitions along both sides of an edge should be restored independently. The tough problem here is how to restore the transition between two partitions. We may use a model to interpolate the transition from textures of two partitions, but usually the results look very artificial and unnatural. Therefore, the blocks containing the neighborhood of free ends should be selected as exemplar so that the transitions of textural partitions can be restored by propagating information in these blocks along the edges. Conjunction blocks of edges are also selected as exemplar for similar reason because there are transitions among more than three textural regions. For circle edges, a circle completely divides the image into two partitions—inner part and outer part—so we choose two blocks as necessary exemplars, which contain the most pixels belonging to inner region and outer region of a circle edge, respectively. In a few words, by necessary exemplars, we provide not only samples for different textures separated by an edge, but also the information of the transitions between these textures, and thus the decoder is able to restore the structural regions.

Additional structural blocks can also be selected as exemplars to further improve visual quality. Given an input ratio, the process is quite similar to that for textural blocks. Each non-necessary structural block is also related to a variation parameter, which can be calculated by (7). Here, the different partitions separated by the edges are independently considered in calculating the mean value as well as the variance, and resulting parameters of different partitions are summed up to get the total variation parameter of a block.

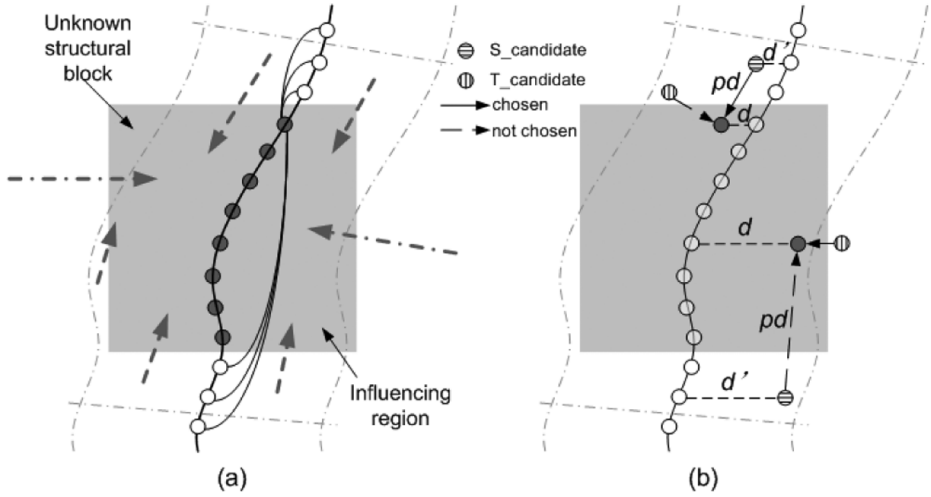


Fig. 6. Pixel-wise structure propagation. (a) A piece of edge and its influencing region, with arrowed dash-dot lines and dash lines showing the propagation directions. (b) Restoration of influencing region in which each generated pixel is copied from one of two candidate pixels.

In Fig. 4, we present the step-wise results of the exemplar selection method. Based on the edge information shown in (b), the selected necessary and additional structural exemplars are shown in (c) and (d) gradually. Similarly, in (e) and (f) we add the necessary and additional textural exemplars, as well.

IV. Edge-Based Image Inpainting

Based on the received edges and exemplars, we propose an edge-based image inpainting method to recover the non-exemplar regions at the decoder side. Different from the encoder, the inpainting algorithm is not block-wise but rather designed to deal with arbitrary-shaped regions. Still, the non-exemplar regions are classified into structures and textures according to their distances to the edge as the encoder. Generally, structures are propagated first, followed by texture synthesis [as shown in Fig. 3(d) and (e)]. A confidence map, similar to that in [20], [21], is constructed to guide the order of structure propagation as well as texture synthesis. Specifically, at the very beginning, known pixels (pixels in decoded exemplars) are marked with confidence 1 and unknown pixels (pixels in removed blocks) are marked with confidence 0. Afterwards, each generated pixel is related with a confidence value between 0 and 1 during the inpainting process. Besides, known pixels as well as generated pixels are all called "available" ones in this section.

In the literature, exemplar-based inpainting methods can be roughly classified into two types, i.e., pixel-wise schemes and patch-wise schemes. Pixel-wise methods are suitable for restoration of small gaps, but may introduce blurring effects or ruin texture pattern while dealing with large areas. Patch-wise methods, on the contrary, are good at keeping texture pattern, but may introduce seams between different patches, which are quite annoying. In our scheme, these two strategies are adapted for different circumstances.

A. Structure Propagation

A sketch map of structure propagation is shown in Fig. 6. The gray block in Fig. 6 indicates an unknown structural block; the black curve with circle points represents an edge piece and related pixels; and four dash-dot lines restrict a region, namely *influencing region*, including unknown pixels within a short distance (e.g., ten-pixel) from the edge. Notice that it is the edge piece together with the influencing region, rather than a structural block, is treated as a basic unit in the structure propagation. Since the free ends and conjunctions of edges are all selected as exemplars, the textural regions along an edge can be readily divided and independently generated in inpainting process.

To recover a basic unit, the unknown pixels belonging to the edge piece are firstly generated. As shown in Fig. 6(a), the unknown pixels (denoted by black points) are generated from the known pixels (indicated by white points) using linear interpolation, i.e.,

$$\hat{f}(x_n) = \frac{\sum_{k=1}^{N} \lambda_{nk} \hat{f}(x_k)}{\sum_{k=1}^{N} \lambda_{nk}} \tag{8a}$$

$$\text{where } \lambda_{nk} = \begin{cases} |n-k|^{-2}, & \text{if } x_k \text{ is known} \\ 0, & \text{otherwise} \end{cases} \tag{8b}$$

where, similar to (4), N gives the number of pixels in this edge piece and n and k index different pixels.

After the edge restoration, neighboring structure as well as texture within the influencing region will be filled-in with regard to the recovered edge. The inpainting method for completion of influencing region is designed concerning the following facts. First, pixel-wise approach is preferred since narrow regions along edge pieces are to be handled. Second, edges are expressed by one-pixel-width curves, which can be quite different in geometric shapes among exemplar and non-exemplar regions, so we have to wrap the edges to reconstruct the unknown structure. Finally, the widths of structures are local variant, which means that it is hard to tell the exact boundary between structure and texture in an influencing region. Therefore, in our scheme, each pixel in the influencing region will have two candidates: one is treated as a structural pixel to be propagated parallel along the edge; the other is regarded as a textural pixel to be generated from the neighboring available pixels. Then, the one that makes a smooth transition from structure to texture will be selected to fill-in the unknown pixel. Moreover, as the decision making on candidate pixels is highly relevant to its available

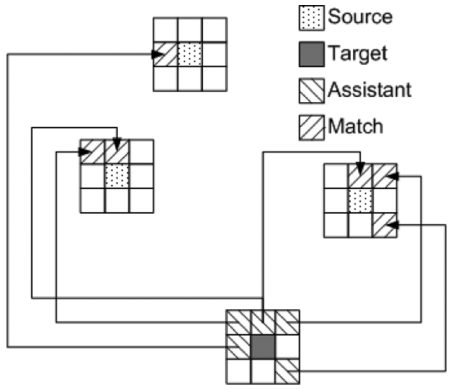


Fig. 7. Pair matching in our structure propagation algorithm.

neighbors, the order for pixel completion is another important issue that should be considered. Thus, we also construct a confidence map, as mentioned at the beginning of this section, to control the generation order. For the unknown pixel, the higher the neighboring confidence is, the earlier it will be generated.

Accordingly, the recovery of influencing region is performed as follows. Here, unknown pixels to be recovered in the influencing region are called target pixels. They are denoted by black points in Fig. 6(b). For each target pixel, two candidate pixels are searched out from the surrounding available pixels. The structural candidate (S-candidate) of the target pixel, which lies within the influencing region, is indicated by horizontal striped point in Fig. 6(b); whereas the textural candidate (T-candidate) of the target pixel is denoted by vertical striped point, which locates within a short distance from the target pixel despite whether it is within the influencing region or not.

A pair matching method, similar to that in [8], is utilized to generate both the S-candidate and the T-candidate. As illustrated in Fig. 7, for each assistant pixel, also known as any available pixel belonging to the 8-adjacent neighborhood of the target pixel, we will search for its match pixel(s) with the most similar value to it. Then, complying with the spatial relation between the assistant pixel and the target one, a pixel adjacent to a match pixel in the same relative spatial position is selected as a source pixel. As indicated in Fig. 7, an assistant pixel may correspond to several match pixels and gives several source pixels; meanwhile, several assistant pixels in 8-adjacent neighborhood may generate the same source pixel, as well.

After obtaining several source pixels, we propose to use a weighted-SSD (sum of squared difference) criterion to choose the S-candidate, as given in

$$D_S = \sum_i \left(\left|d\left(x_S^i\right) - d\left(x_t^i\right)\right| + 1\right) \times \left|\hat{f}\left(x_S^i\right) - \hat{f}\left(x_t^i\right)\right|^2 \tag{9}$$

where x_S^i and x_t^i are corresponding, the ith pixel in the neighborhood of the S-candidate and the target pixel, respectively, and $d()$ indicates the distance from each pixel to the edge, $\hat{f}()$, as used before, is the reconstructed image. By minimizing (9), we can find the S-candidate from the obtained source pixels, which is situated in a similar relative position to that of the target pixel with respect to the edge, thus ensure the parallel diffusion of structural information.

Differently, since no direction information involved in textural region, only the ordinary SSD between the neighborhood of source pixels and target pixel is considered as the criterion to choose the T-candidate,

$$D_T = \sum_i \left|\hat{f}\left(x_T^i\right) - \hat{f}\left(x_t^i\right)\right|^2. \tag{10}$$

Similar to that in (9), x_T^i here represents the ith pixel in the neighborhood of the T-candidate. Thus, the source pixel that has the most similar neighboring values to the target one will be selected as the T-candidate.

In fact, the two diffusions, or S-candidate selection and T-candidate selection, are simultaneous and competitive. These two candidates have to compete with each other and only one of them will be chosen to fill-in the target pixel. Normally, if target pixel nears edge, the choice will bias to the S-candidate. In addition, it can be observed that long-distant parallel diffusion of structural information often leads to blurring artifacts. Thus, the determination is made by comparing D_T and D'_S which are defined in (10) and (11), respectively

$$D'_S = \left(\frac{d}{d_0} + \frac{pd}{pd_0}\right) \times \sum_i \left|\hat{f}\left(x_S^i\right) - \hat{f}\left(x_t^i\right)\right|^2. \tag{11}$$

Here d_0 and pd_0 are constants and $d = d(x_t)$ stands for the distance from the target pixel to the edge and pd indicates the distance from the target pixel to the S-candidate, as shown in Fig. 6(b). If D_T is less than D'_S, then the T-candidate is chosen to fill-in the target pixel, otherwise the S-candidate is selected. In this way, all unknown pixels within the influencing region of an edge are generated.

B. Texture Synthesis

The edges as well as their influencing regions are readily restored by structure propagation. Then, in this subsection, the remainder unknown regions are treated as textural regions, so texture synthesis is employed to fill-in these holes.

For textural regions, we prefer patch-wise algorithms because they are good at preserving large-scale texture pattern. We choose square patches as the fundamental elements while a confidence map is introduced to guide the order of synthesis. Unknown textural regions are progressively restored during texture synthesis by first reconstructing the prior patches and then the others that remain. The priority of a patch is determined by calculation of confidence and the distance from the edge. As shown in Fig. 8, for each patch centered at a marginal pixel of unknown regions (denoted by target patch), we calculate the average confidence value of all pixels in this patch, as well as the average distance of all pixels from the edge. Then the patch with the highest confidence rating and the greatest distance from the edge will be synthesized first.

Afterwards, a source patch, which is most similar to the target patch, will be searched out from the neighborhood of the target patch. Here, the similarity of two patches is measured by the SSD of pixel values between overlapped available pixels of two patches. A patch that results in the least SSD will be chosen as the source patch. Notice that the filling-in process is not as

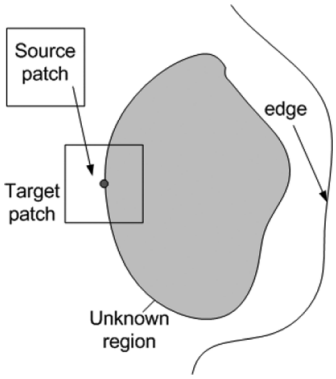


Fig. 8. Patch-wise texture synthesis in our scheme.

simple as copy-paste work, we have to deal with overlapped regions as well as seams. In our algorithm, the graph-cut method proposed in [12] is used to merge the source patch into the existing image, and the Poisson editing [34] is utilized to erase the seams. After one patch is restored, the confidence map is updated. All newly recovered pixels are treated as available pixels in the following synthesis steps. Then, the next target patch is searched and processed until no unknown pixel exists.

V. Experimental Results

A. Implementation

Our presented approach can be integrated with the state-of-the-art coding schemes to enhance compression performance. In our experiments, two compression standards, JPEG and MPEG-4 AVC/H.264 (referred to simply as H.264 hereafter), are adopted. Thus, two fully automatic image coding systems, based on JPEG and H.264 respectively, have been constructed to evaluate the effectiveness of our proposed compression approach. In this subsection, we would like to clarify several implementation details of the systems.

First, in both systems, the one-pixel-width edge information is coded using JBIG method. Note that the edge information coded into final bit-stream is only a subset of the entire edge map. In other words, the edges that are fully covered by exemplar regions will not be coded [it can be observed by comparing the edges in Fig. 4(b) and (f)]. Second, in the JPEG-based system, the exemplar locations are denoted at block level by a binary map, in which 1 stands for a removed block and 0 for an exemplar block, and the map is coded by an arithmetic coder. The original image is then coded by JPEG coding method, during which the removed blocks will be skipped in encoding but to be filled with the DC values copied from previous blocks, so that the DC prediction in JPEG can still be performed in exemplar block compression. Third, in the H.264-based system, since the basic coding unit is macro-block 16×16, we consider two instances: if a macro-block is totally removed, then a new macro-block type I_SKIP is coded; otherwise, the macro-block has a new element called block removal pattern (BRP) for indicating which of the four 8×8 blocks is removed, and the BRP is later coded by the arithmetic coder, too. Similar to the JPEG-based method, the exemplar blocks are coded using H.264 scheme and DC values from previous blocks are filled to the removed blocks to enable the intra prediction of H.264 scheme.

In addition, there are some predefined parameters in both encoder and decoder. To test the robustness of our system, we fix these parameters as follows for all test images. At the encoder side, the weighting factors are defined as $\alpha = 1.0, \beta = \gamma = 0.2$, and $p = 1$ (suggested by [17]) for (4) in edge thinning, while $w_1 = w_2 = 1.0$ for (7) in exemplar selection. At the decoder side, structure propagation works on pixels that have less than ten-pixel distances from edges. The search range for T-candidate is 9×9, and the S-candidate is found in the entire influencing region. The search range and patch size for texture synthesis are 11×11 and 7×7, respectively. The parameters d_0 and pd_0 in (11) are set to 5. We would like to remark that the weighting factors for edge thinning have been carefully tuned using our test images, while the other parameters are just empirically selected with consulting the existing systems (e.g., [23]). However, it should be noticed that the parameters can greatly influence the computational complexity of both encoder and decoder, which will be further analyzed in the following. At last, the only two flexible parameters in our experiments are the additional block ratios for structural exemplar and textural exemplar; actually, they act as quality control parameters in our system.

B. Test Results

We test our compression systems on a number of standard color images from the USC-SIPI image database[1] and the Kodak image library.[2] Some results are presented here to evaluate the compression ratio as well as reconstructed quality of our scheme. In all tests, the quality parameter (QP) of JPEG coding method is set to 75, while the QP of H.264 intra coding is set to 24. Bit-rate savings are listed in Table I.

Fig. 3 shows test image Lena and corresponding results of our JPEG-based system. As mentioned before, the coded exemplars and the edge information are denoted in (c). In this test, 10% additional structural blocks as well as 50% additional textural blocks are preserved. Based on the preserved blocks and assistant edge information, our presented structure propagation gives inpainting results in (d), and the final reconstructed image after texture synthesis is shown in (e). Compared with the reconstructed image shown in (f) by baseline JPEG, our scheme saves 20% bits (as given in Table I) but presents similar visual quality. More comparison results in visual quality concerning standard images can be found in Fig. 9. It shows that up to 44% bits-saving (shown in Table I) is achieved by our scheme at the similar visual quality levels, compared to baseline JPEG.

In Fig. 10, our proposed structure propagation method is evaluated. In this test, we remove only structural blocks and use different approaches to recover them. The details of partial images together with the assistant edge information are given in the first column. Then, results generated by the PDE-based diffusion [35], which is the traditional solution to structural regions, are shown in the second column. This method works well only

[1]http://sipi.usc.edu/services/database/

[2]http://r0k.us/graphics/kodak/

TABLE I
BIT-RATE SAVINGS OF OUR SCHEME COMPARED TO JPEG (QP IS SET TO 75) AND H.264 (QP IS SET TO 24)

Original image	Additional block ratio		JPEG-based comparison			H.264-based comparison		
	Structural	Textural	JPEG	Our scheme	Bit-rate saving	H.264	Our scheme	Bit-rate saving
Jet	*0.1*	*0.3*	1.156	0.919	**20.6%**	0.985	0.880	**10.7%**
Lena	*0.1*	*0.5*	1.112	0.888	**20.1%**	0.993	0.869	**12.5%**
Milk	*0.1*	*0.3*	0.913	0.512	**44.0%**	0.616	0.415	**32.7%**
Peppers	*0.1*	*0.3*	1.217	0.965	**20.8%**	1.311	1.080	**17.6%**
Kodim02	*0.1*	*0.2*	1.058	0.709	**33.0%**	0.948	0.701	**26.0%**
Kodim03	*0.2*	*0.3*	0.895	0.608	**32.1%**	0.710	0.562	**20.9%**
Kodim05	*0.1*	*0.3*	2.027	1.719	**15.2%**	2.086	1.845	**11.6%**
Kodim07	*0.1*	*0.5*	1.079	0.802	**25.7%**	0.876	0.751	**14.3%**
Kodim11	*0.2*	*0.3*	1.368	1.047	**23.5%**	1.354	1.098	**18.9%**
Kodim19	*0.2*	*0.5*	1.276	0.915	**28.4%**	1.246	0.956	**23.3%**
Kodim20	*0.1*	*0.4*	0.897	0.638	**28.9%**	0.823	0.636	**22.6%**
Kodim23	*0.2*	*0.3*	0.821	0.567	**30.9%**	0.622	0.504	**18.9%**

Fig. 9. Comparisons with baseline JPEG on test images Jet, Milk, and Peppers. From left to right: removed blocks (black blocks) and assistant edge information (blue curves); reconstructed image by our scheme; reconstructed image by baseline JPEG.

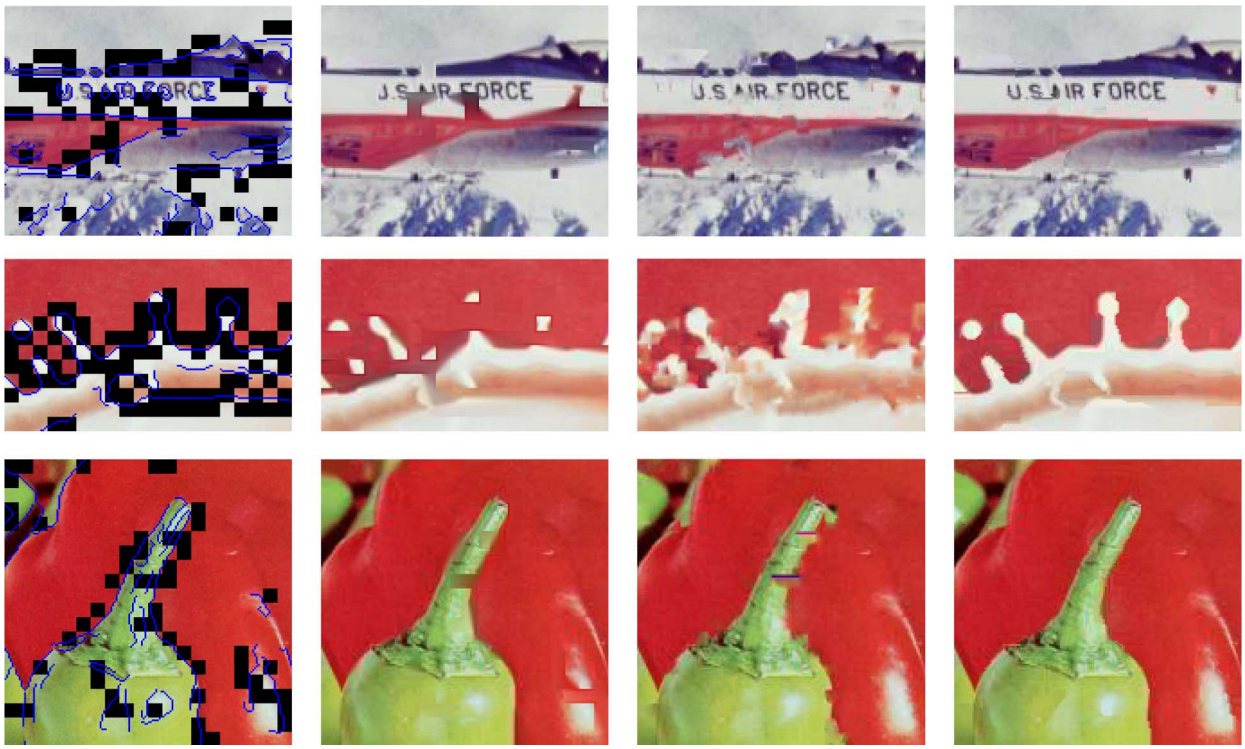


Fig. 10. Comparisons of different structure propagation approaches, zoomed-in partitions. From left to right: removed blocks (black blocks) with assistant edge information (blue curves); inpainting result by PDE-diffusion [35]; inpainting result by patch-wise synthesis [23]; inpainting result by our scheme. Note that only our scheme takes advantage of the edges.

for smooth regions and certain simple structures. In addition, image completion method presented in [23] is also tested in the third column, only the user interaction is omitted. Since no edge information is utilized, these two methods result in annoying distortion in most of the structural regions. In the last column, we give our results that are accomplished by the help of edge information. It is clearly demonstrated that the assistant edges help greatly in structure restoration, and thus empower the encoder to remove more blocks.

Furthermore, our JPEG-based system is also tested using the images in the Kodak image library, which contains photographs of natural scenes at high resolution. The comparison results on visual quality are shown in Fig. 11 in which the top row shows our results whereas the bottom row presents baseline JPEG results. It can be observed that the visual quality of our resulting image is very similar to that of JPEG. The bits-saving of our JPEG-based system is indicated in Table I. Our method averagely saves 27% bits for the five images shown in Fig. 11 at the similar visual quality levels.

To investigate the detailed bit-rate cost in our scheme, we list the percentage of different coded elements in Table II, from which we notice that even different images will lead to different allocations of coded elements, the exemplar location information as well as the edge information still costs only a little overhead. Commonly, the bits used to code the exemplar blocks occupy more than 90% of total bits cost. However, it is still possible to further reduce the bits cost on edge information, taking into account the exemplar locations, or skipping those edges that can be inferred from the exemplar blocks.

In Fig. 12 we show the reconstructed images by our H.264-based system in comparison with standard H.264 intra coding. Both results show similar visual quality to each other, as in the JPEG comparisons. The bit-rate saving is also noticeable, shown in Table I, but not as much as the comparison with JPEG. This is caused by two reasons. On the one hand, the H.264 intra coding is more efficient than JPEG in coding performance, so the non-exemplar blocks, especially the textural ones, will cost fewer bits in standard H.264 than in baseline JPEG, but in our scheme the edge information still cost the same bits in either realization. On the other, due to the complicated spatial predictions performed in the H.264 intra coding, the filling of only DC values for removed blocks is proved not good enough, since it breaks the original spatial relations between neighboring blocks, but for JPEG this filling of DC values seems enough since JPEG only conduct DC prediction. Nevertheless, our scheme can still acquire up to 33% bit-rate saving compared to the state-of-the-art H.264 intra coding.

C. Discussion

It can be observed that the ratio of additional textural exemplar has a big effect on visual quality of the reconstructed images. As given in Table I, for homogeneous textural regions, such as the red wooden door in kodim02 [Fig. 11(a)], low exemplar ratio is used to pursue high compression ratio; whereas for complex and irregular textural regions, e.g., flowers and leaves in kodim07 [Fig. 11(d)], high ratio is preferred to ensure good visual quality. However, thanks to the given edge information,

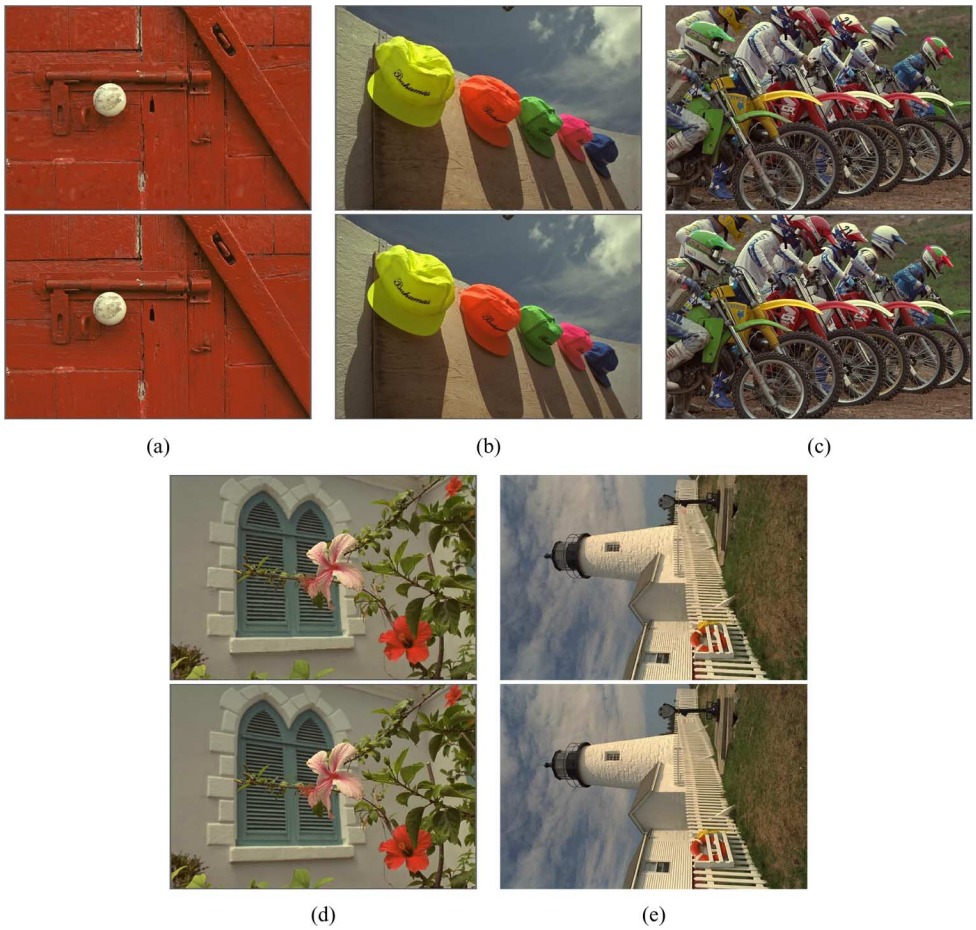


Fig. 11. Comparisons with baseline JPEG on the Kodak Image Library. (a) kodim02; (b) kodim03; (c) kodim05; (d) kodim07; (e) kodim19. The top row shows the reconstructed images by our scheme and the bottom row shows the reconstructed images by baseline JPEG.

TABLE II
PERCENTAGE OF DIFFERENT CODED ELEMENTS IN OUR JPEG-BASED SYSTEM (QP IS SET TO 75)

Original image	Exemplar locations (arithmetic coder)	Assistant edge information (JBIG)	Exemplar blocks (JPEG)
Jet	1.4%	6.3%	92.3%
Lena	1.4%	6.6%	92.0%
Milk	2.7%	5.2%	92.1%
Peppers	1.4%	6.7%	91.9%

the reconstructed quality of structural regions is less sensitive to the additional structural exemplar ratio.

In addition, the improvement of our scheme in terms of compression ratio is various for different images. Commonly, the more complicated the image is, the less gain we can provide. It is because that when coding images with lots of details [such as kodim05, Fig. 11(c)], the extracted edge map usually contains miscellaneous edges which makes many blocks as necessary exemplars. Thus, only a limited number of regions can be removed at encoder. However, in this case, 15% bits-saving is still provided by our JPEG-based system without noticeable visual loss, as shown in Table I.

The computational complexity of our scheme is relatively higher than that of the traditional coding schemes, since at the encoder side we perform extra edge extraction and exemplar selection, and at the decoder side we add the inpainting process. In particular, the computation of the decoder is greatly related with the parameters used in the inpainting, such as search range and patch size, which determine the necessary number of SSD calculations. There are several previous work proposed to reduce the computations of SSD, so as to accelerate the image synthesis [7], [11], [20], and those methods can be adopted in our system as well.

The visual quality assessment is highly related to our work, that is, if we have a good metric used to measure visual quality, we are able to not only better evaluate our scheme, but also further improve the performance by rate-"distortion" optimization, where "distortion" measures the perceptual quality in addition to the statistical fidelity. Unfortunately, we have not yet found such a good metric for our purposes. Thus, for visual quality comparisons in our experiments, we always set the same quality parameters for both the standard compression scheme and our

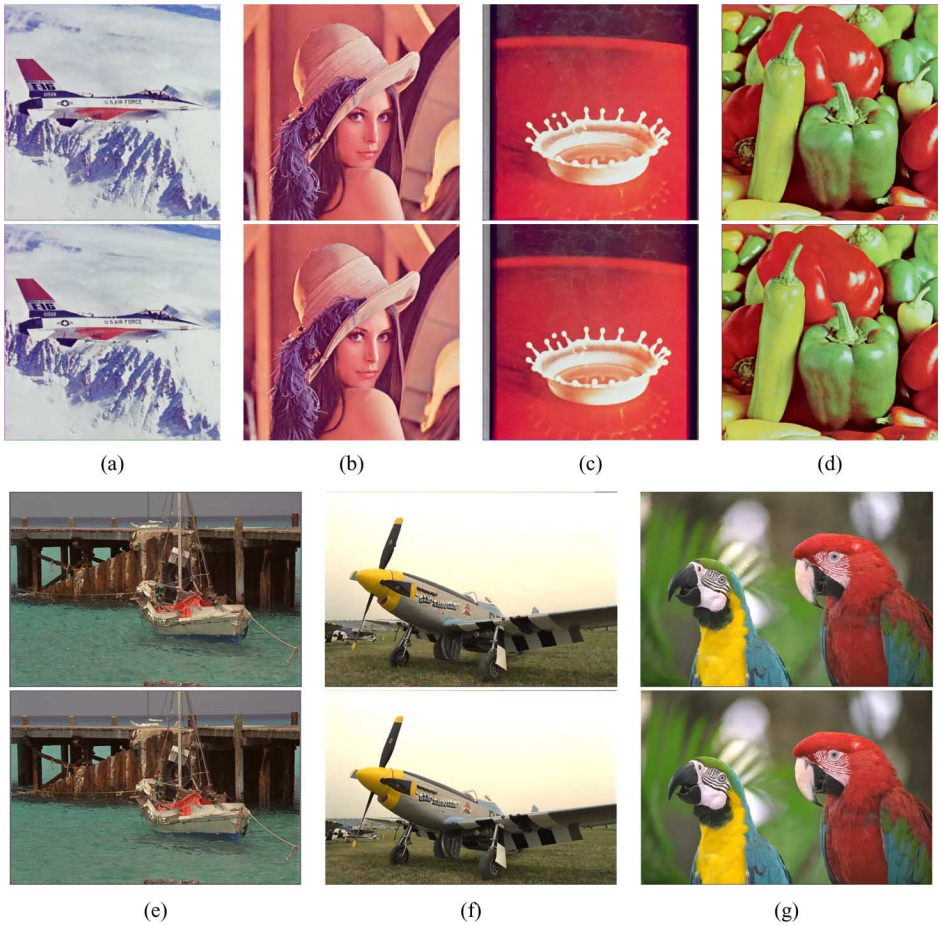


Fig. 12. Comparisons with standard H.264 intra picture coding. (a) Jet. (b) Lena. (c) Milk. (d) Peppers. (e) kodim11. (f) kodim20. (g) kodim23. The top row shows the reconstructed images by our scheme and the bottom row shows the reconstructed images by standard H.264 intra coding.

inpainting-based scheme. Thus, the exemplar regions will have the same quality (both subjectively and objectively). Additionally, the restored regions still have acceptable visual quality. Accordingly, in this paper, the "comparable quality" or "similar visual quality levels" indicates visually similar qualities, which are examined by human observations.

VI. Conclusion and Future Work

In this paper, we present an image compression framework that adopts inpainting techniques to remove visual redundancy inherent in natural images. In this framework, some kinds of distinctive features are extracted from images at the encoder side. Based on the obtained features, some regions of an image are skipped during encoding, only to be recovered by the assisted inpainting method at the decoder side. Due to the delivered assistant information, our presented framework is able to remove enough regions so that the compression ratio can be greatly increased. Our presented inpainting method is capable in effectively restoring the removed regions for good visual quality, as well.

Moreover, we present an automatic image compression system, in which edge information is selected as the assistant information because of its importance in preserving good visual quality. The main techniques we proposed for this compression system, i.e., edge thinning, exemplar selection and edge-based inpainting, are also addressed in this paper. Experimental results using many standard color images validate the ability of our proposed scheme in achieving higher compression ratio while preserving good visual quality. Compared to JPEG and H.264, at the similar visual quality levels, up to 44% and 33% bits-saving can be acquired by our approach, respectively.

Further improvements of current scheme are still promising. First, the assistant information as well as the selected exemplars can be described and compressed into bit-stream in more compact fashion. Second, extraction of the distinctive features can be more flexible and adaptable. Besides edge information, there are other candidates, such as sketch [5] and epitome [29], [30], which could be derived from source images to assist the vision technologies and the compression methods as well. Furthermore, image inpainting is still a challenging problem when some kinds of assistant information are provided, into which we need to put more effort in the future.

Acknowledgment

The authors acknowledge all anonymous reviewers for their constructive opinions that contribute to the improvement of this paper. Special thanks go to H.-Y. Shum and J. Sun for their helpful discussions, and to D. Daniels for his editing. Also, thanks go to C. Wang for his previous work that has been integrated into our current system.

REFERENCES

[1] N. Jayant, J. Johnston, and R. Safranek, "Signal compression based on models of human perception," *Proc. IEEE*, vol. 81, no. 10, pp. 1385–1422, Oct. 1993.

[2] I. Höntsch and L. J. Karam, "Locally adaptive perceptual image coding," *IEEE Trans. Image Process.*, vol. 9, no. 9, pp. 1472–1483, Sep. 2000.

[3] M. M. Reid, R. J. Millar, and N. D. Black, "Second-generation image coding: An overview," *ACM Comput. Surveys*, vol. 29, no. 1, pp. 3–29, Mar. 1997.

[4] M. Tuceryan and A. K. Jain, "Texture analysis," in *The Handbook of Pattern Recognition and Computer Vision*, C. H. Chen, L. F. Pau, and P. S. P. Wang, Eds., 2nd ed. Singapore: World Scientific, 1998, pp. 207–248.

[5] C. en Guo, S.-C. Zhu, and Y. N. Wu, "Towards a mathematical theory of primal sketch and sketchability," in *Proc. IEEE Int. Conf. Computer Vision (ICCV'03)*, 2003, pp. 1228–1235.

[6] A. A. Efros and T. K. Leung, "Texture synthesis by non-parametric sampling," in *Proc. IEEE Int. Conf. Computer Vision (ICCV'99)*, 1999, pp. 1033–1038.

[7] L.-Y Wei and M. Levoy, "Fast texture synthesis using tree-structured vector quantization," in *Proc. ACM SIGGRAPH*, 2000, pp. 479–488.

[8] M. Ashikhmin, "Synthesizing natural textures," in *Proc. ACM Symp. Interactive 3D Graphics (SI3D)*, 2001, pp. 217–226.

[9] A. Hertzmann, C. E. Jacobs, N. Oliver, B. Curless, and D. H. Salesin, "Image analogies," in *Proc. ACM SIGGRAPH*, 2001, pp. 327–340.

[10] A. A. Efros and W. T. Freeman, "Image quilting for texture synthesis and transfer," in *Proc. ACM SIGGRAPH*, 2001, pp. 341–346.

[11] L. Liang, C. Liu, Y.-Q. Xu, B. Guo, and H.-Y. Shum, "Real-time texture synthesis by patch-based sampling," *ACM Trans. Graphics*, vol. 20, no. 3, pp. 127–150, Jul. 2001.

[12] V. Kwatra, A. Schödl, I. Essa, G. Turk, and A. Bobick, "Graphcut textures: Image and video synthesis using graph cuts," in *Proc. ACM SIGGRAPH*, 2003, pp. 277–286.

[13] S. Lefebvre and H. Hoppe, "Parallel controllable texture synthesis," in *Proc. ACM SIGGRAPH*, 2005, pp. 777–786.

[14] V. Kwatra, I. Essa, A. Bobick, and N. Kwatra, "Texture optimization for example-based synthesis," in *Proc. ACM SIGGRAPH*, 2005, pp. 795–802.

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

[16] T. F. Chan and J. Shen, "Mathematical models for local nontexture inpaintings," *SIAM J. Appl. Math.*, vol. 62, no. 3, pp. 1019–1043, Feb. 2002.

[17] 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 Process.*, vol. 10, no. 8, pp. 1200–1211, Aug. 2001.

[18] T. F. Chan and J. Shen, "Non-texture inpainting by curvature-driven diffusions (CDD)," *J. Visual Commun. Image Represen.*, vol. 12, no. 4, pp. 436–449, Dec. 2001.

[19] J. Jia and C.-K. Tang, "Image repairing: Robust image synthesis by adaptive ND tensor voting," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR'03)*, 2003, pp. 643–650.

[20] I. Drori, D. Cohen-Or, and H. Yeshurun, "Fragment-based image completion," in *Proc. ACM SIGGRAPH*, 2003, pp. 303–312.

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

[22] P. Pérez, M. Gangnet, and A. Blake, "PatchWorks: Example-based region tiling for image editing," Microsoft Research, Redmond, WA, Tech. Rep. MSR-TR-2004-04, 2004.

[23] J. Sun, L. Yuan, J. Jia, and H.-Y. Shum, "Image completion with structure propagation," in *Proc. ACM SIGGRAPH*, 2005, pp. 861–868.

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

[25] H. Grossauer, "A combined PDE and texture synthesis approach to inpainting," in *Proc. Eur. Conf. Comput. Vis. (ECCV'04)*, 2004, pp. 214–224.

[26] 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 Process.*, vol. 12, no. 3, pp. 296–303, Mar. 2003.

[27] L. Atzori and F. G. B. De Natale, "Error concealment in video transmission over packet networks by a sketch-based approach," *Signal Process.: Image Commun.*, vol. 15, no. 1-2, pp. 57–76, Sep. 1999.

[28] W. Zeng and B. Liu, "Geometric-structure-based error concealment with novel applications in block-based low-bit-rate coding," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 9, no. 6, pp. 648–665, Jun. 1999.

[29] N. Jojic, B. J. Frey, and A. Kannan, "Epitomic analysis of appearance and shape," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV'03)*, 2003, pp. 34–41.

[30] V. Cheung, B. J. Frey, and N. Jojic, "Video epitomes," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR'05)*, 2005, pp. 42–49.

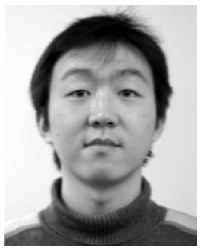
[31] C. Wang, X. Sun, F. Wu, and H. Xiong, "Image compression with structure-aware inpainting," in *Proc. IEEE Int. Symp. Circuits Syst (ISCAS'06)*, 2006, pp. 1816–1819.

[32] X. Sun, F. Wu, and S. Li, "Compression with vision technologies," presented at the Picture Coding Symp. (PCS), Beijing, China, Apr. 2006.

[33] C. A. Rothwell, J. L. Mundy, W. Hoffman, and V.-D. Nguyen, "Driving vision by topology," in *Proc. Int. Symp. Comput. Vis.*, 1995, pp. 395–400.

[34] P. Pérez, M. Gangnet, and A. Blake, "Poisson image editing," in *Proc. ACM SIGGRAPH*, 2003, pp. 313–318.

[35] D. Tschumperlé and R. Deriche, "Vector-valued image regularization with PDEs: A common framework for different applications," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 27, no. 4, pp. 506–517, Apr. 2005.



Dong Liu received the B.S. degree in electrical engineering from the University of Science and Technology of China (USTC), Hefei, China, in 2004. He is now pursuing the Ph.D. degree at USTC.

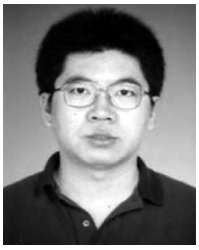
He has been a visiting student in Microsoft Research Asia since 2005, where his research concentrates on image/video compression and compression-oriented vision technologies. He is also interested in image segmentation and image representation.

Xiaoyan Sun (M'04) received the B.S., M.S., and Ph.D. degrees in computer science from Harbin Institute of Technology, Harbin, China, in 1997, 1999, and 2003, respectively.

She joined Microsoft Research Asia, Beijing, China, as an Associate Researcher in 2003 and has been a Researcher since 2006. She has authored or co-authored over 30 conference and journal papers and submitted several proposals and contributed techniques to MPEG-4 and H.264. Her research interests include video/image compression, video streaming, and multimedia processing.

Feng Wu (M'99–SM'06) received the B.S. degree in electrical engineering from the University of Xi'an Electrical Science and Technology, Xi'an, China, in 1992, and the M.S. and Ph.D. degrees in computer science from Harbin Institute of Technology, Harbin, China, in 1996 and in 1999, respectively.

He joined Microsoft Research Asia, Beijing, China, as an Associate Researcher in 1999 and was promoted to Researcher in 2001. He has played a major role in Internet Media Group to develop scalable video coding and streaming technologies. He has authored or co-authored over 60 papers in video compression and contributed some technologies to MPEG-4 and H.264. His research interests include video and audio compression, multimedia transmission, and video segmentation.



Shipeng Li (M'97) received the B.S. and M.S. degrees from the University of Science and Technology of China (USTC), Hefei, China, in 1988 and 1991, respectively, and the Ph.D. degree from Lehigh University, Bethlehem, PA, in 1996, all in electrical engineering.

He was with the Electrical Engineering Department, USTC, during 1991–1992. He was a Member of Technical Staff with Sarnoff Corporation, Princeton, NJ, during 1996–1999. He has been a Researcher with Microsoft Research China, Beijing, since May 1999. He has contributed some technologies in MPEG-4 and H.264. His research interests include image/video compression and communications, digital television, multimedia, and wireless communication.

Ya-Qin Zhang (S'87–M'90–SM'93–F'98) received the B.S. and M.S. degrees in electrical engineering from the University of Science and Technology of China (USTC), Hefei, Anhui, China, in 1983 and 1985, and the Ph.D. degree in electrical engineering from George Washington University, Washington, DC, in 1989.

He is currently the Corporate Vice President of Microsoft, responsible for mobility and embedded system products. He was the Managing Director, Microsoft Research Asia, Beijing, China, in 1999. Previously, he was the Director of the Multimedia Technology Laboratory, Sarnoff Corporation, Princeton, NJ (formerly David Sarnoff Research Center and RCA Laboratories). Prior to that, he was with GTE Laboratories, Inc., Waltham, MA, from 1989 to 1994. He has been engaged in research and commercialization of MPEG2/DTV, MPEG4/VLBR, and multimedia information technologies. He has authored or co-authored over 200 refereed papers in leading international conferences and journals, and has been granted over 40 U.S. patents in digital video, Internet, multimedia, wireless, and satellite communications. He serves on the Board of Directors of five high-tech IT companies and has been a key contributor to the ISO/MPEG and ITU standardization efforts in digital video and multimedia.

Dr. Zhang served as the Editor-In-Chief for the IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS FOR VIDEO TECHNOLOGY from July 1997 to July 1999. He was the Chairman of the Visual Signal Processing and Communications Technical Committee of the IEEE Circuits and Systems (CAS) Society. He serves on the editorial boards of seven other professional journals and over a dozen conference committees. He has received numerous awards, including several industry technical achievement awards and IEEE awards, such as the CAS Jubilee Golden Medal. He was named "Research Engineer of the Year" in 1998 by the Central Jersey Engineering Council. He received The Outstanding Young Electrical Engineer of 1998 Award.