IMAGE COMPRESSION WITH EDGE LOCAL VARIABLE BASED INPAINTING

[1]Sanjay Nayak, [2]Krishna Kant Nayak
Department of Electronics & Communication,
Bansal Institute of Science & Technology, Bhopal, India
Email: Sanjay1512nayak@gmail.com

Abstract— **In this thesis, image compression utilizing visual redundancy is investigated. Inspired by recent advancements in image inpainting techniques, we propose an image compression framework towards visual quality with pixel-wise fidelity. In this work, an original image is analyzed at the encoder side so that parts of the image are intentionally and automatically skipped. Instead of some information is extracted from these skipped regions and delivered to the decoder as assistant information in the compressed manner. The delivered assistant information plays a main role in the proposed work because it provides guide lines to image inpainting with restoration to restore these regions accurately at the decoder side. Moreover, to fully take advantage of the assistant information, a compression-oriented edge-based inpainting with restoration algorithm is proposed for image compression, 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. Our proposed method is a promising exploration towards image and video compression.**

Keywords — **Fuzzy Gibbs Random Field, local level processing, Multiresolution decomposition, Multispectral image fusion.**

I. INTRODUCTION

The objective of image compression is to reduce irrelevance and redundancy of the image data in order to be able to store or transmit data in an efficient form. 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.

The best image quality at a given bit-rate (or compression rate) is the main goal of image compression. Images require much storage space, large transmission bandwidth and long transmission time. Currently, there is only one way to improve resource requirements are to compress images, such that they can be transmitted quicker and then decompressed by the receiver. In Image compression, many applications want a representation of the image with minimal storage. In general, the representation of digital image requires a large memory. The greater the size of a particular image, the greater the memory it needs. On the other hand, most images contain duplicate data. There are two duplicated parts of data in the image. The first is the existence of a pixel that has the same intensity as its neighboring pixels. These duplicated pixels waste more storage space. The second is that the image contains many repeated sections (regions). These identical sections do not need to be encoded many times to avoid redundancies and, therefore, we need an image compression to minimize the memory requirement in representing a digital image.

Redundancy reduction is aimed at removing duplication in the image. According to Saha there are two different types of redundancy relevant to images: (i) Spatial Redundancy: Correlation between neighbouring pixels. (ii) Spectral Redundancy: Correlation between different colour planes and spectral bands. Where there is high correlation, there is also high redundancy, so it may not be necessary to record the data for every pixel. There are two parts to the compression, Find image data properties; grey-level histogram, image entropy, correlation functions etc. Find an appropriate compression technique for an image of those properties.

II. LITERATURE REVIEW

Dong Liu, Xiaoyan Sun Feng Wu, Shipeng Li, and Ya-Qin Zhang [36] said in their paper that 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 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 isotopic 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 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.

S. D. Rane, G. Sapiro, and M. Bertalmio, in their paper "Structure and texture filling-in of missing image blocks in wireless transmission and compression applications," and W. Zeng and B. Liu, in their paper "Geometric-structure-based error concealment with novel applications in block-based low-bit-rate coding," has given a framework of image compression scheme in which 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.

III IMAGE COMPRESSION WITH EDGE-BASED INPAINTING

The original image is not simply partitioned into two parts: one is coded by conventional transform-based approach [36], 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 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 [36].

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.

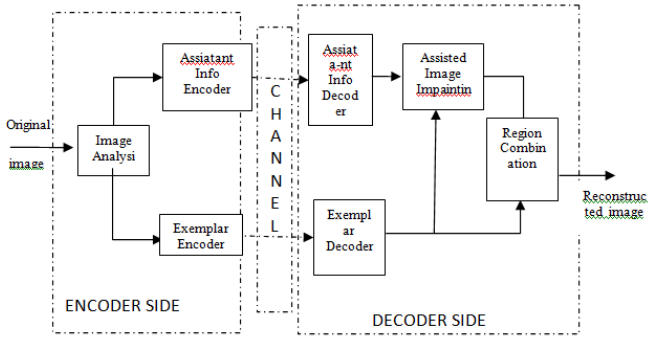


Figure: 1. Framework of the image compression scheme [36].

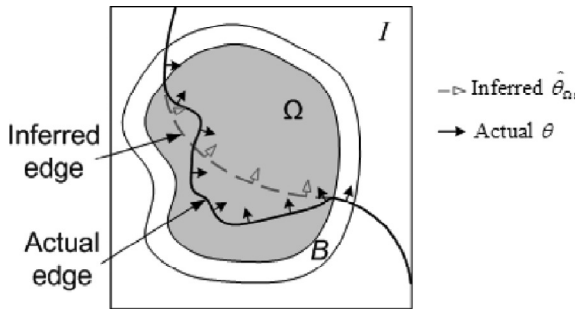


Figure: 2. Illustration of image inpainting, in which the gray region is to be, restored [36].

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 R^2 .Ω, depicted as the gray region in Fig. 3.2, 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

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

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 $f_{\frac{I}{\Omega}}(x)$.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 $f_\Omega(x)$ of 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 $\frac{I}{\Omega}$ 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 ∪ Ω such that it minimizes the following energy function:

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

The first term in (2) is to measure local homogeneity of image function in the region B ∪ Ω, 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 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:

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

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 $\hat{\theta}_\Omega$. $\nabla\hat{f}_\Omega = \left|\nabla\hat{f}_\Omega\right|$.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 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. 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.

III.I 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 previous Section, 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 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, 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. 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…,N, yielding the minimal energy,

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

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

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

$\mu_8(x_n)$ 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], $k(x_n)$ is defined as:

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

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 neighbour 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.

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 cannot be properly restored without necessary exemplar blocks, whereas additional blocks help to further improve visual quality.

C. 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, 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.

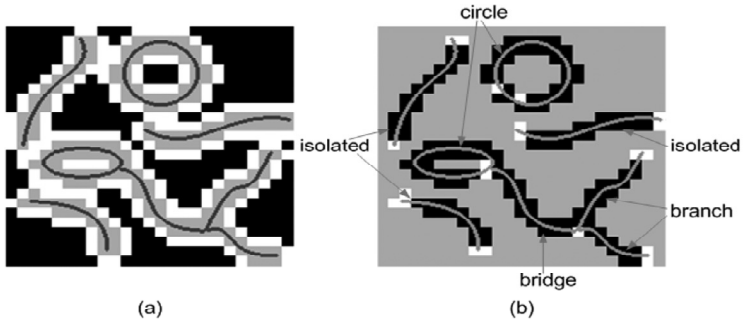


Fig.5. An example of necessary exemplar selection in which curves denote edges and black blocks denote skipped regions. (a) Textural exemplar selection in which white blocks are necessary textural exemplars; (b) structural exemplar selection in which white blocks are necessary structural exemplars, four types of edges are also distinguished in (b) [36].

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 *Bi* is related to a variation parameter *Vi* defined as

$$V_i = \omega_1 Var(B_i) + \omega_2 \sum_{B_j \in \mu_4(B_i)} \big|E(B_i) - E(B_j)\big| \quad (7)$$

Here and are positive weighting factors indicates 4-neighbor of a certain block. The functions and 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.

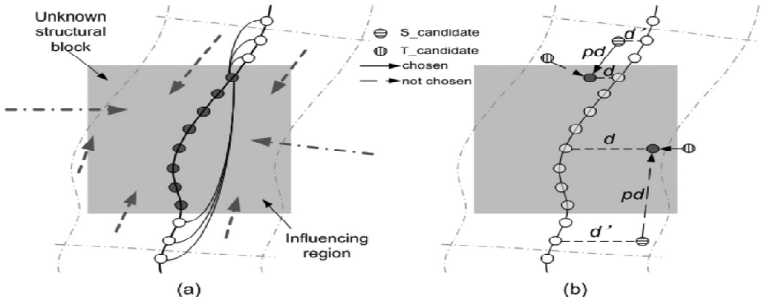


Fig. 6. Pixel-wise structure propagation. (a) A piece of edge and its influencing region, with arrowed dash-dot lines and dash lines showing the propagation directions. (b) Restoration of influencing region in which each generated pixel is copied from one of two candidate pixels [36].

III.II 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 [36]. 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 [36].

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.,

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

$$where, \lambda_{nk} = \begin{cases} |n-k|^{-2}, if\, x_k is\ known \\ 0, \qquad\qquad otherwise \end{cases} \quad (8b)$$

Where, similar to (4), 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

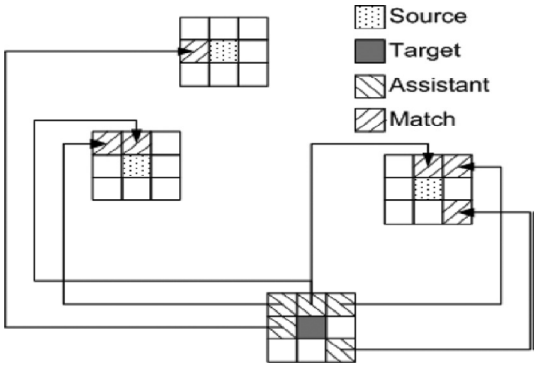


Figure: 7. Pair matching in our structure propagation algorithm [36].

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:

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

Where x_S^i and x_t^i are corresponding, the *i*th 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,

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

Similar to that in (9), x_T^i here represents the i^{th} 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

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

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 thanD'_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 [36]. 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.

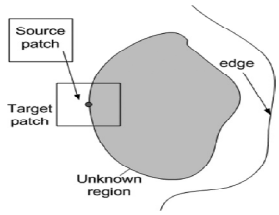


Fig. 8. Patch-wise texture synthesis in our scheme [36].

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 [39] 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. 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.

IV. PROPOSED WORK: IMAGE COMPRESSION WITH EDGE-BASED INPAINTING AND IMAGE RESTORATION

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

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

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 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$f_{\frac{I}{\Omega}(x)}$.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 is usually an ill-posed problem because information in is totally unknown. Then, the TV model is to find a function on the extended inpainting region such that it minimizes the following energy function:

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

The first term in (2) is to measure local homogeneity of image function in the region $B \cup \Omega$, 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].

To recover a basic unit, the unknown pixels belonging to the edge piece are firstly generated. As shown in Fig. 6(a),

$$\hat{f}(x_n) = \frac{\sum_{k=1}^N \lambda_{nk}\hat{f}(x_k)}{\sum_{k=1}^N \lambda_{nk}} \quad (3)$$

The unknown pixels (denoted by black points) are generated from the known pixels (indicated by white points) using linear interpolation, i.e.

$$where\lambda_{nk} = \begin{cases} |n-k|^{-2}, if\, x_k is\ known \\ 0, \qquad\qquad otherwise \end{cases} \quad (4)$$

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. 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

$$D_s = \sum_i \left(\left|d\left(x_S^i\right) - d\left(x_t^i\right)\right| + 1\right) \times \left|\hat{f}\left(x_S^i\right) - \hat{f}\left(x_t^i\right)\right|^2 \quad (5)$$

Where x_S^i and x_t^i are corresponding, the i^{th} pixel in the neighborhood of the S-candidate and the target pixel, respectively, and indicates the distance from each pixel to the edge, as used before, is the reconstructed image.

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.

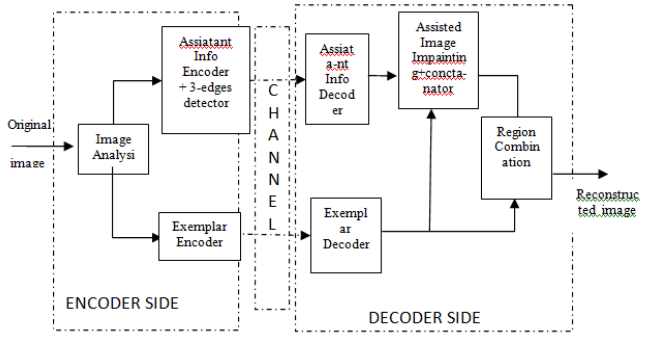


Figure: 1. The framework of the image compression scheme [36].

Assistant Encoder Information

The Assistant Information is finding out by the edges of the image in different directions this the added term to increase the compressed image more close to original one.

Performance Parameters

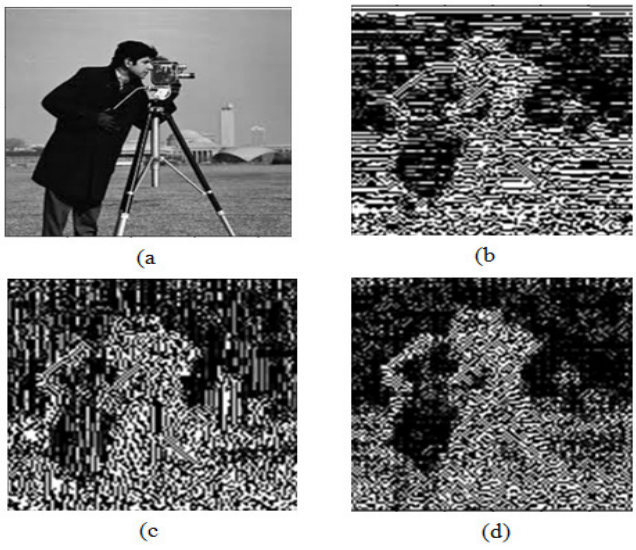
Performance of the image compression coding, it is necessary to define a measurement that can estimate the difference between the original image and the decoded image. Two common used measurements are the Mean Square Error (MSE) and the Peak Signal to Noise Ratio (PSNR), which are defined in Equations below. f(x, y) is the pixel value of the original image, and f'(x, y)is the pixel value of the decoded image [36]. Most image compression systems are designed to minimize the MSE and maximize the PSNR.

$$MSE = \sqrt{\frac{\sum_{x=0}^{W-1}\sum_{y=0}^{H-1}[f(x,y)-f'(x,y)]^2}{WH}}$$

$$PSNR = 20\log_{10}\frac{255}{MSE}$$

IV. RESULTS AND ANALYSIS

The algorithm proposed here will compress an image without prior knowledge of the image geometry and any specific parameter. The figure below shows the results: The proposed method is based on determining the image edges in all possible directions such as horizontal, vertical and diagonal (principal Diagonal) profile of the image. In this proposed method we sued the Run Length encoding method for encoding that increases the compression ratio and we use the edge information as the to restore the image. It is implemented using MATLAB 7.9.0 (R2009b) on i-5 processor with 4-GB RAM. The simulations have been tested on aerial images in figure 5.1. Figure 5.1 (a) shows the Original Image of Cameraman, Image of Cameraman along with edge maps of that Image and its respective Compressed Image.



(e (f

Figure: 5.1(a) Original Image of Cameraman (b) Horizontal edge map (c) Vertical edge map (d) Diagonal edge map (e) Decomposition of image at level-1(f) Compressed Image by proposed method.

Figure 5.2 shows the result obtained from the method based on image compression with edge based inpainting.
Now Figure 5.3 shows the comparison between Edge-based inpainting method and our proposed method. The proposed method gives better result as compared to the previous method of image compression with edge based inpainting.

(a) (b)

Figure: 5.3(a) Compressed Image by Edge based inpainting method (b) Compressed Image by proposed method.

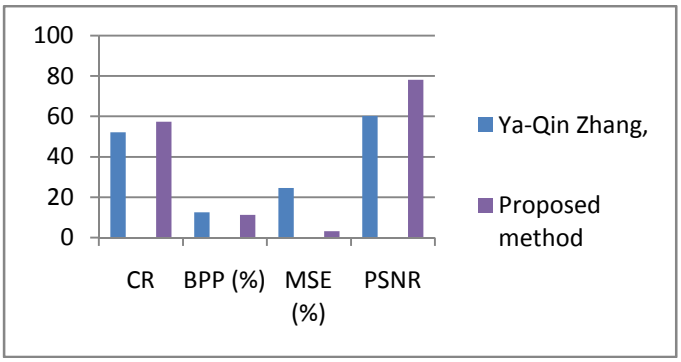


Figure 5.10 Graphs for the Cameraman Image

V CONCLUSION

Here our method of image compression provides the wavelet decomposition method run length encoding to encode the data and as assistant information is used from the edges in three directions. The new image compression algorithm called Shape- Adaptive Image Compression, which is proposed by Huang [5], takes advantage of the local characteristics for image compaction The DCT-based image compression such as JPEG performs very well at moderate bit rates; however, at higher compression ratio, the quality of the image degrades because of the artifacts resulting from the block-based DCT scheme. Wavelet-based coding such as JPEG 2000 on the other hand provides substantial improvement in picture quality at low bit rates because of overlapping basis functions and better energy compaction property of wavelet transforms. Because of the inherent multi-resolution nature, wavelet-based coders facilitate progressive transmission of images thereby allowing variable bit rates. We also briefly introduce the technique that utilizes the statistical characteristics for image compression. The new image compression algorithm called Shape- Adaptive Image Compression, which is proposed by Huang [5], takes advantage of the local characteristics for image compaction. The SAIC compensates for the shortcoming of JPEG that regards the whole image as a single object and do not take advantage of the characteristics of image segments. However, the current data compression methods might be far away from the ultimate limits. Interesting issues like obtaining accurate models of images, optimal representations of such models, and rapidly computing such optimal representations are the grand challenges facing the data compression community. Image coding based on models of human perception, scalability, robustness, error resilience, and complexity are a few of the many challenges in image coding to be fully resolved and may affect image data compression performance in the years to come.

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.

[36] Dong Liu, Xiaoyan Sun Feng Wu, Shipeng Li, and Ya-Qin Zhang "Image compression with edge-based inpainting" *IEEE Transactions* On Circuits And Systems For Video Technology, Vol. 17, No. 10, October 2007.

[37] Jian-Jiun Ding and Jiun-De Huang, "Image Compression by Segmentation and Boundary Description", Master's Thesis, National Taiwan University, Taipei, 2007.

[38] W.K.Pratt, "Digital Image Processing, 2nd ed. New York: Wiley.

[39] R. C. Gonzalez R.E. Woods, Digital Image Processing, 2nd ed. reading, MA: Addison-Wesley, 2002.