Deringing Cartoons by Image Analogies

GUANGYU WANG
TIEN-TSIN WONG
and
PHENG-ANN HENG
The Chinese University of Hong Kong

In this paper, we propose a novel method to reduce ringing artifacts in BDCT-encoded cartoon images using image analogies. The quantization procedure of BDCT compression (such as JPEG and MPEG) introduces annoying visual artifacts. Our main focus is on the removal of ringing artifacts that are seldom addressed by existing methods. In the proposed method, the contaminated image is modeled as a Markov random field (MRF). We 'learn' the behavior of contamination by extracting massive number of artifact patterns from a training set, and organizing them using tree-structured vector quantization (TSVQ). Instead of *post-filtering* the input contaminated image, we *synthesize* an artifact-reduced image. Our method is non-iterative and hence it can remove artifacts within a very short period of time. We show that substantial improvement is achieved using the proposed method in terms of visual quality and statistics.

Categories and Subject Descriptors: I.4.4 [**Image Processing and Computer Vision**]: Restoration; I.3.3 [**Computer Graphics**]: Picture/Image Generation; I.2.6 [**Artificial Intelligence**]: Learning—Analogies

General Terms: Algorithms

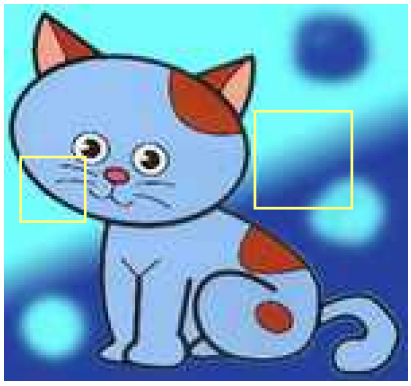
Additional Key Words and Phrases: Deringing, image analogies, texture synthesis, artifact removal.

1. INTRODUCTION

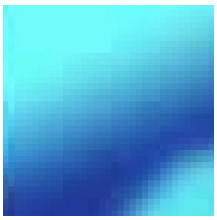
The block-based discrete cosine transform (BDCT) is the core of many current image and video compression standards, such as JPEG [Wallace 1991] and MPEG [Gall 1991]. In particular, JPEG has been a popular image standard since early 90's. Even though we now have wavelet-based JPEG2000, there are still many existing images already encoded using JPEG. MPEG family is the most widely used video encoding standard in digital TV broadcast and DVD format. Unfortunately, BDCT-based encoding has a drawback, which is the annoying visual artifact. Such artifact is especially apparent in *cartoon* compressed at low bit rates (Figure 1(a)).

In the JPEG (or MPEG) standard, an image (or a frame) is first divided into 8×8 pixel blocks and then each block is transformed from spatial domain to frequency domain using discrete cosine transform (DCT). The DCT coefficients are then quantized and run-length encoded. In principle, DCT does not introduce artifact. It merely transforms the original image to a domain where encoding can be effectively performed. However, the following quantization of DCT coefficients is lossy and introduces visual artifact. There are

This work is supported by the Research Grants Council of the Hong Kong Special Administrative Region under RGC Earmarked Grants (Project No. 417005 and 2050345) and CUHK Shun Hing Institute of Advanced Engineering. We would like to thank NYU Media Research Lab for their baseline image analogies software.
Authors' address: Department of Computer Science and Engineering, The Chinese University of Hong Kong, Shatin, N.T., Hong Kong; email: {gywang,ttwong,pheng}@cse.cuhk.edu.hk.



(a) a typical BDCT-encoded cartoon image



(b) blocking artifacts

(c) ringing artifacts

Fig. 1. BDCT compression artifacts. (a) A typical BDCT-encoded (contaminated) cartoon image, with (b) the blocking artifacts and (c) the ringing artifacts. Sub-figures (b) and (c) are the blowups of two boxes in (a).

mainly two types of artifacts, namely the *blocking artifact* and the *ringing artifact*. Excessive quantization of low-frequency coefficients introduces the blocking artifacts, which exhibits color discontinuity at the block boundary, as shown in Figure 1(b). Excessive quantization of high-frequency coefficients causes the ringing artifacts around strong edges, such as the outlines of the cartoon character (Figure 1(c)). Ringing artifacts may appear anywhere within the 8×8 block.

Blocking artifacts are usually easier to reduce, and a lot of work [Reeve and Lim 1984; Chen et al. 2001; Qiu 2000; Chan et al. 1998] has been proposed in removing such kind of artifacts. However, they may not be effective in removing ringing artifacts, because ringing artifacts not only locate at the boundary of block, but also at the interior of block. There is not much work done on removing the ringing artifacts. Some methods apply spatial adaptive filter [Chen et al. 2001; Ramamurthi and Gersho 1986; Kuo and Hsieh 1995; Lee et al. 1998; Kong et al. 2004] for ringing artifact removal. But it is very hard to distinguish high-frequency details from noise. Excessive blurring of details and exaggeration of noise may appear. To remove ringing artifacts, we propose an analogy-based method that *synthesizes* images with fewer artifacts, rather than post-filters the BDCT-encoded image. Training examples provide *prior* knowledge of artifacts and help us to distinguish between artifacts and details.

Our approach stems from the image analogies [Hertzmann et al. 2001]. Given a pair of source images A and A', along with a contaminated input image B', we synthesize an artifact-reduced image B, such that B' relates to B in the same way as A' relates to A (Figure 2). The advantage of image analogies is that it provides a convenient and natural means of specifying image transformations. As ringing artifact is the result of reconstruction from the over-quantized DCT coefficients, there is no simple analytical equation to correct the artifact. By observation, we know that similar image contents will give similar ringing artifacts. This suggests us to use an analogy-based method for ringing artifact removal. We model the artifact as a Markov random field. To do so, we extract an enormous number of *artifact patterns* from the training set, which consists of the original and the contaminated image pairs. These artifact patterns are looked up during synthesis. To manage and query these patterns efficiently, the total number of patterns is reduced by making use of *luminance remapping* and *dominant implication* properties. The patterns left are then organized using the tree-structured vector quantization (TSVQ).

In Section 2, previous work is reviewed. Section 3 explains the rationale and the basic idea of the proposed example-based method. Section 4 describes the details of our method. Section 5 evaluates the proposed method experimentally. Finally, conclusions are drawn in Section 6.

2. PREVIOUS WORK

Our method is related to the recent example-based image synthesis techniques [Hertzmann et al. 2001; Efros and Leung 1999; Wei and Levoy 2000; Mese and Vaidyanathan 2001; Freeman et al. 2000; Freeman et al. 2002; Efros and Freeman 2001; Welsh et al. 2002] and BDCT compression artifacts removal techniques [Chen et al. 2001; Zakhor 1992; Triantafyllidis et al. 2002; Gunturk et al. 2002; Yang et al. 2001; Paek et al. 2000; Weerasinghe et al. 2002].

2.1 Image Analogies

Hertzmann *et al.* [2001] described a framework for synthesizing images by examples, called "image analogies." Given a pair of unfiltered and filtered images, this framework learns the behavior of such filter, and then the learned filter is applied to some new target image in order to create an analogous filtering effect. This framework can be applied to several applications, including traditional image filter, improved texture synthesis, super-resolution, texture transfer, artistic filter, texture-by-number, and color transfer. Later, Hertzmann *et al.* [2002] presented "curve analogies" using the same framework. Mese and Vaidyanathan [2001] proposed a look-up table (LUT) based method for inverse halftoning of images. The LUT is obtained from a few sample halftone images and corresponding grayscale images. Freeman *et al.* [2002] proposed another example-based method for super-resolution. This technique learns the relationship between high-frequency and low-frequency bands in an image, and then guesses the missing high-frequency band of a blurred image.

These analogy methods are inspired by texture synthesis [Efros and Leung 1999; Wei and Levoy 2000; Praun et al. 2000; Efros and Freeman 2001]. Efros and Leung [1999] modeled the sample input texture as MRF and synthesized larger texture seamlessly. The key idea is to synthesize a pixel value by looking up its nearest neighborhood in the sample texture. Wei and Levoy [2000] presented an algorithm for realistic texture synthesis and accelerated this synthesis process using tree-structured vector quantization (TSVQ). Efros and Freeman [2001] first incorporated seam finding by dynamic programming. A new image is synthesized by stitching together small patches of existing images. Cohen *et al.* [2003] described a simple stochastic system for image and texture synthesis with a small set of Wang tiles [Wang 1965]. Kwatra *et al.* [2003] introduced an algorithm for image and video texture synthesis based on graph cut. This approach is ideal for computing seams of patches and determining placement of patches to generate visually smooth images and video. Liu *et al.* [2004] presented algorithms for efficient synthesis of bidirectional texture functions on arbitrary surfaces. Bar-Joseph *et al.* [2001], gave an algorithm based on statistical learning for synthesizing static and time-varying textures matching the appearance of an input texture.

2.2 BDCT Artifact Removal

As the visual artifact in the contaminated image is usually high-frequency in nature, a straightforward solution for reducing such artifact is to apply lowpass filtering. Reeve and Lim [1984] first proposed a space-invariant lowpass filtering method. However, such filtering often causes the loss of high-frequency details, such as edges. Therefore, a number of adaptive spatial filtering techniques [Chen et al. 2001; Ramamurthi and Gersho 1986; Kuo and Hsieh 1995; Lee et al. 1998; Jarske et al. 1994; Hsu and Chen 1993; Liu et al. 1998; Minami and Zakhor 1995; Tomasi and Manduchi 1998] have been proposed to overcome this problem. Kuo and Hsieh [1995] proposed an adaptive filter, which is a space-variant lowpass filter. It smoothes pixel values at the block boundary, while pixels close to edges are adaptively applied with regional or directional lowpass filter. Lee *et al.* [1998] proposed a method based on signal adaptive filter, which is controlled by

edge information. The filter consists of a one-dimensional directional smoothing filter for edge areas and a two-dimensional adaptive average filter for monotone areas. Bilateral filtering [Tomasi and Manduchi 1998] averages the gray values based on both geometric closeness and photometric similarity. Although it is originally not tailormade for deringing, we found that it is indeed quite effective. The main difficulty in post-filtering techniques is how to distinguish high-frequency details from BDCT compression artifacts. Incorrect filtering usually introduces excessive blurring of details and/or exaggeration of visual artifacts.

Projection onto convex sets (POCS) [Combettes 1993] is a theory widely used for artifact removal [Weerasinghe et al. 2002; Zakhor 1992; Jeong et al. 2000; Paek et al. 1998; Yang and Galatsanos 1997; Yang et al. 1995; Luo et al. 1996]. This method updates the BDCT coefficients by iteratively projecting onto several constraints, which are formulated by the *priori* knowledge of contaminated images. Zakhor [1992] proposed an iterative technique based on this theory. The basic idea is to impose a number of constraints on the contaminated image in order to restore its original artifact-free form. Since methods based on POCS are usually iterative, they are computationally expensive and seldom used in interactive applications.

Chan *et al.* [1998] proposed to classify small local boundary regions according to their pixel value distribution. Appropriate linear predictor is selected to estimate the corresponding pixels at the block boundary. One difficulty is that linear predictor is not accurate enough to deblock the contaminated image. Qiu [2000] proposed an example-based method to postprocess block-coded images. The technique is based on the multilayer perceptron (MLP) neural network. Just like [Chan et al. 1998], this method only processes pixels at the block boundary and does not handle artifacts within the block. Note that, most previous artifact-removal methods are designed for removing blocking artifact, and are usually not effective to reduce ringing artifact.

3. CONTAMINATION AS MARKOV RANDOM FIELD

To reduce the visual artifacts due to BDCT-encoding, we need to understand its source. During BDCT-encoding, an 8×8 block of pixel values, $f(u, v)$, is transformed to frequency domain by the following discrete cosine transform:

$$F(u,v) = \frac{1}{4}C(u)C(v)\left[\sum_{i=0}^{7}\sum_{j=0}^{7} f(i,j)\cos\frac{(2i+1)u\pi}{16}\cos\frac{(2j+1)v\pi}{16}\right], \tag{1}$$

where

$$C(u),\ C(v) = \begin{cases} \frac{1}{\sqrt{2}} & \text{if} \quad u, v = 0 \\ 1 & \text{otherwise,} \end{cases}$$

and $F(u, v)$'s are 64 DCT coefficients to be stored. Each DCT coefficient is then quantized according to a fine-tuned quantization table in JPEG standard. DCT itself does not introduce error, but the following quantization does.

During DCT, the block of pixel values is transformed in a deterministic manner. Moreover, since DCT coefficients are quantized by predefined quantization table in JPEG standard, the introduced error is also deterministic. In other words, given the same input image and compression ratio, the same visual artifacts will be obtained.

After inverse discrete cosine transform (IDCT), a quantization error in one DCT coefficient may affect every pixel in the 8×8 block. In other words, once a pixel is error-contaminated, its local neighborhood is very likely to be contaminated as well. This observation suggests us to model the *contaminated image* as Markov random field (MRF) [Li 1995]. MRF is the 2D expansion of Markov random chain. It has been widely used in image processing and analysis. In MRF, the probability that a pixel takes certain value is determined by the values of its local neighbors. This local property is known as Markovianity. As contaminated image is

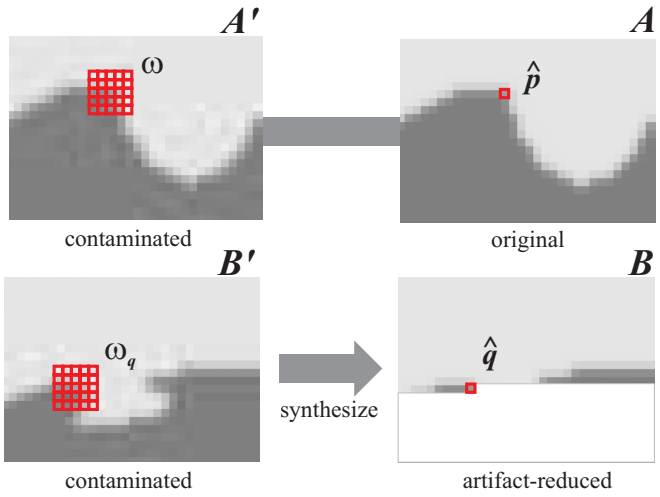


Fig. 2. The basic idea. Instead of post-filtering the contaminated image B', we try to synthesize an artifact-reduced image B based on the knowledge learned from the training pairs of the original error-free (A) and the contaminated images (A').

subdivided into 8×8 pixel blocks and each block is encoded independently, the visual artifacts must also be localized. Hence it conforms to the spirit of MRF.

Intuitively speaking, the basic idea of our method is to learn the relationship between the original error-free (A) and the error-contaminated images (A') from a training set (Figure 2). The training set contains pairs of the original and contaminated images. With the learned knowledge, we try to *synthesize* an artifact-reduced image (B) instead of post-filtering the artifacts in the contaminated image (B'). Note that the prime in our notation indicates the "artifact". This is different from the original notation in [Hertzmann et al. 2001]. The synthesis is performed in a pixel-by-pixel manner. When synthesizing a pixel $\hat{q}$ in image B based on the contaminated image B', the corresponding local neighborhood ω_q in B' is used to search within the training set. When the best matching block ω is found in the contaminated image A', its corresponding center pixel $\hat{p}$ in the original image A is copied to the synthesized image B. This process is applied to every pixel in the edge region where the ringing artifacts usually appear. Since we learn how artifact is introduced in the examples from the training set, our method is an *example-based artifact-removal* algorithm.

4. AN EXAMPLE-BASED ARTIFACT-REMOVAL ALGORITHM

4.1 Formation of Artifact Vectors

The first step is to setup a training set. We use several pairs of original images and their corresponding BDCT-encoded images (contaminated images) to form the training set. One way to obtain the training set is to randomly select images from an arbitrary image library. However, a huge number of images are needed in order to acquire sufficient amount of distinct artifact patterns.

Instead, we generate the training set to tailor for our purpose. As ringing artifacts appear near the sharp edges, we prepare training set by generating images containing two basic shapes, *circle* and *corner* (Figure 3). Although circle looks simple, it captures most edge information. Almost any curve can be formed by connecting pieces of circular edges. The corner type images are mainly used to tackle sharp changing outlines like the star shape. Multiple instances of these two basic images are generated, each with different intensity, background intensity, line width, and resolutions (scales).

To facilitate the searching during artifact removal, we extract the distinct *artifact patterns* from training

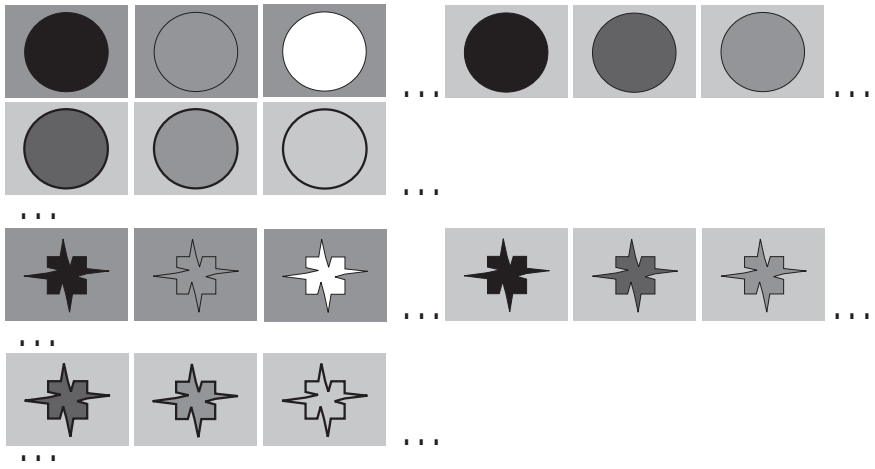
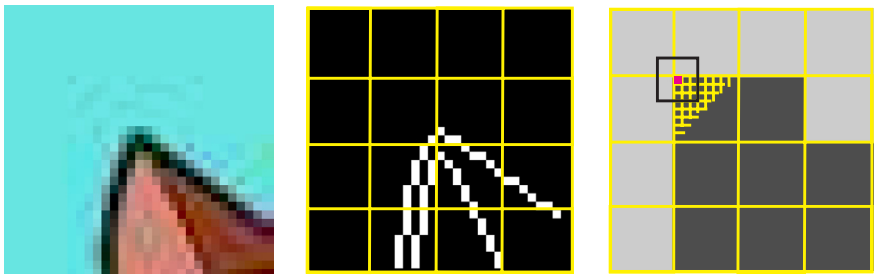


Fig. 3. Our training set. The upper half shows the "circle" type images while the lower half shows the "corner" type images.



(a) (b) (c)

Fig. 4. Extraction of artifact vectors from edge blocks. (a) Contaminated image. (b) Edge detection. (c) The tagged edge blocks. The 5×5 neighborhood of each pixel in the edge block is used to form an artifact pattern in the training set.

set. Such approach saves the space due to duplicate patterns. An artifact pattern is a block of pixels (neighborhood) ω in the contaminated image A'. Each artifact pattern *implies* a corresponding center pixel value $\hat{p}$ in the original counterpart A (Figure 2). Since the artifact pattern is reordered and stored in a vector form, we usually called it artifact vector. From now on, we shall use the two terms, artifact pattern and artifact vector, interchangeably.

Selecting an appropriate size of artifact vector (the size of neighborhood) is a trade-off between the computational cost and the accuracy. Larger neighborhood in general provides more local information and hence returns better result, but the computation is more expensive. Since 8×8 blocks are used in BDCT, the size of neighborhood should not exceed 8×8. From our experiment, a 5×5 neighborhood provides adequate local information and tractable computation. Detail discussion on the neighborhood size can be found in Section 5.5.

Since the ringing artifacts mainly appear at region with strong edges, we only need to extract distinct artifact patterns from edge region. This can drastically reduce the number of distinct artifact patterns. Hence, the first step is to identify the edge block by applying standard edge detector as shown in Figure 4. In our current implementation, we use Sobel edge detector. More sophisticated edge detectors such as [Perona and Malik 1990] can also be used. Moreover, we can use multiple edge detectors to reduce the chance of

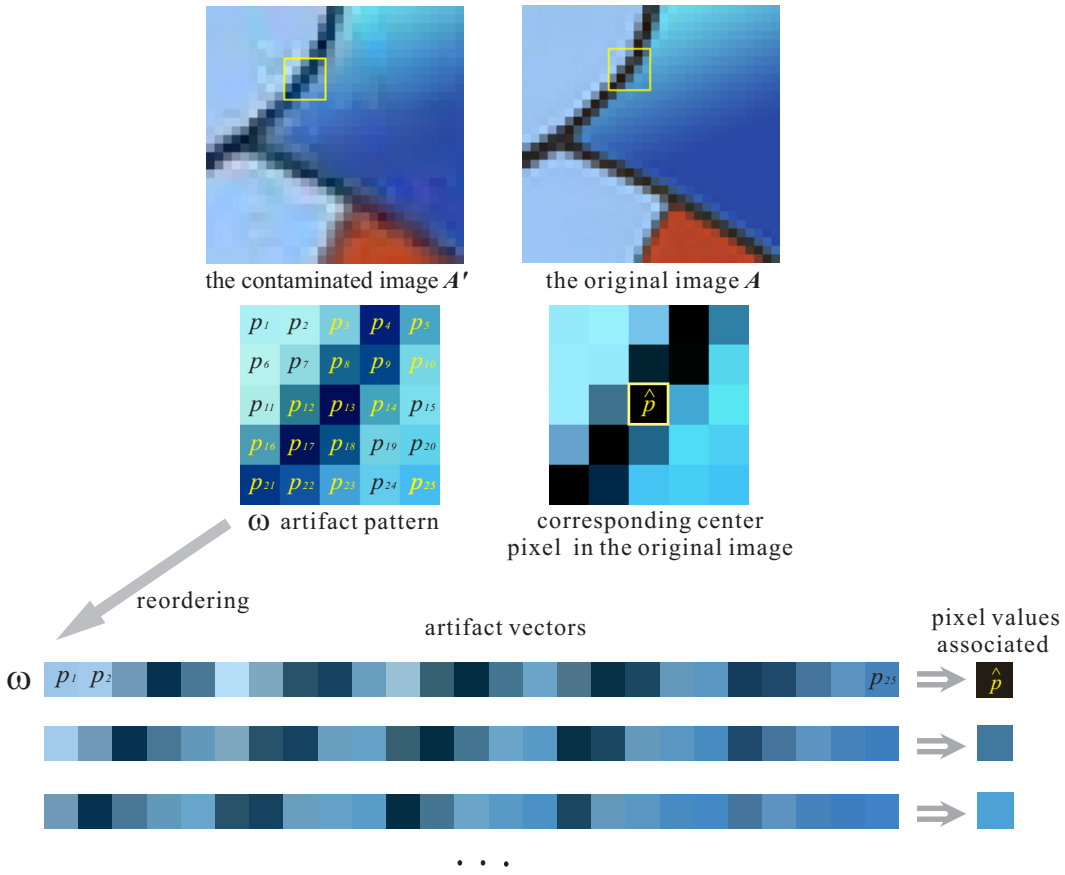


Fig. 5. Formation of artifact vectors. The original image A (top right) and its contaminated counterpart A' (top left) are shown on the top row. The current pixel $\hat{p}$ and its corresponding neighborhood ω in A' are blowed up in the second row. The 25 pixels in ω are reordered and stored together with $\hat{p}$.

missing edge detection. Note that, we should detect edges in the *contaminated* images (Figure 4(a)) in the training set and edge pixels are marked (Figure 4(b)). The reason we do not detect edges in the original images is because the original images are not available during artifact removal. Next, we check each non-overlapping 8×8 block in the BDCT-encoded image to see if it contains any edge pixel. If so, the corresponding 8×8 block in the *original* image is tagged as *edge block* (Figure 4(c)).

For each pixel $\hat{p}$ in the edge block of the *original* image, we look up its corresponding 5×5 neighborhood ω in the *contaminated* counterpart (Figure 5). This neighborhood is then reordered to form the artifact vector. If the artifact vector is distinct, it is stored together with $\hat{p}$. We say ω implies $\hat{p}$, and denoted as,

$$\omega \Rightarrow \hat{p}.$$

Note that only ω is searched and matched during synthesis. The implied pixel value $\hat{p}$ is the data to retrieve. During the extraction of artifact vectors, the pixels near the image boundary may not be able to form artifact vectors due to the clipping by the image boundary.

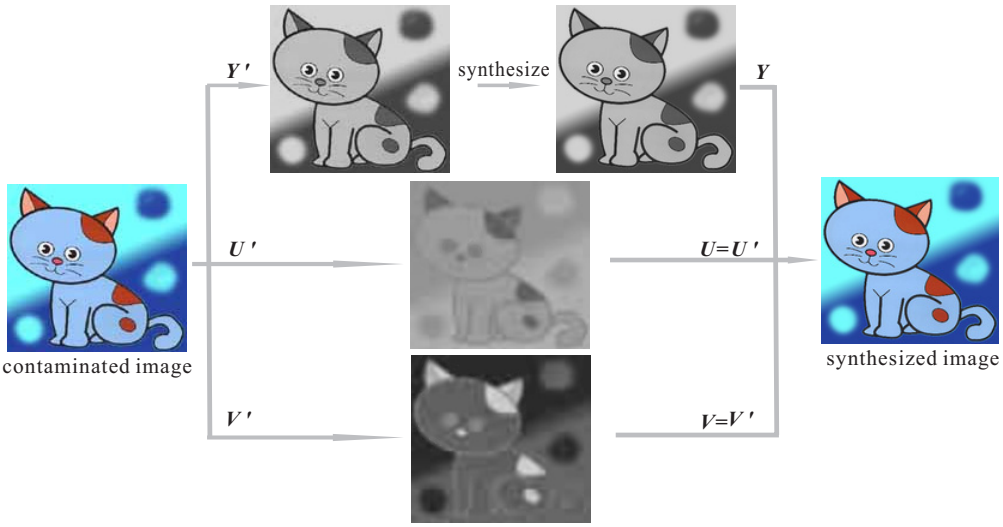


Fig. 6. Heterogeneous strategy for different color components. The visual-sensitive luminance component Y is applied with the proposed method while the visual-insensitive chrominance components U and V are not processed. For visualization purpose, U and V components are shifted to $[0, 1]$.

4.2 Color Processing and Luminance Remapping

As JPEG adopts YUV color model and the luminance component Y is more human vision sensitive, we only dering the Y component and leave U and V untouched (Figure 6).

To focus on the artifact patterns rather than the absolute gray values, we normalize them using the *luminance remapping* in [Hertzmann et al. 2001], a linear mapping that matches the means and variances of the luminance distributions. If p is the luminance of a pixel in artifact pattern, and p' is the remapped luminance, luminance remapping is described by:

$$\frac{p - \mu_p}{\sigma_p} = \frac{p' - \mu_n}{\sigma_n}\,, \tag{2}$$

where μ_p and σ_p are the mean luminance and the standard deviation of the original pattern respectively; μ_n and σ_n are the mean luminance and standard deviation of the normalized pattern and predefined as 0.5 and 0.3 respectively. These normalized patterns are then stored. Note that this luminance remapping is also needed during the artifact removal.

4.3 Dominant Implication

Theoretically, one artifact pattern ω may not imply (be associated with) a unique pixel value $\hat{p}$. An artifact pattern may in fact imply a set of k possible pixel values, each with different probability.

$$\omega \;\Rightarrow\; \{\hat{p}_1, \hat{p}_2, \cdots, \hat{p}_k\}$$

and

$$\sum_{i=1}^{k} \mathrm{Prob}(\hat{p}_i) = 1$$

where $\mathrm{Prob}(\hat{p}_i)$ is the probability of $\hat{p}_i$.

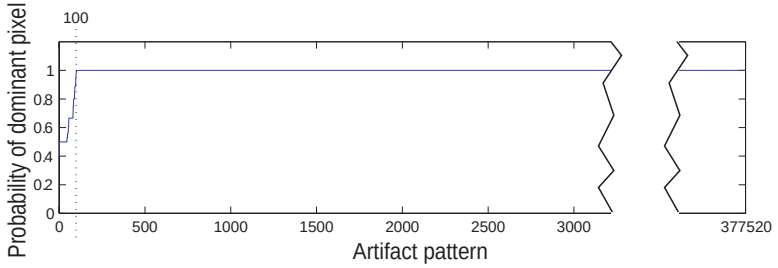


Fig. 7. Probabilities of dominant pixels. For illustration purpose, artifact patterns are sorted increasingly according to the probabilities of dominant pixels.

The probability of each $\hat{p}_i$ can be obtained from the training set. This probability distribution can be stored together with the artifact pattern. During synthesis, the pixel value can be synthesized according to this probability distribution as in the original image analogies method.

However, as our goal is to recover the error-free images instead of the aesthetic purpose, it would be more reasonable to synthesize the most probable pixel value instead of a variety of possible pixel values. Interestingly, we also observed a phenomenon that there is always a dominant pixel value $\hat{p}^*$ associated with each artifact pattern. One reason is related to the deterministic properties of DCT and quantization. Another possible reason is because we focus on cartoon images which usually have less variation in pixel values. Cartoon images usually have large smooth color regions and strong edges/outlines.

To verify our finding, we plot the probabilities of the dominant pixel values of all artifact patterns from our training set in Figure 7. In this training set of 112 cartoon image pairs, there are 377,520 distinct artifact patterns. Among them, 377,420 patterns (99.97%) are associated with a unique pixel value. Only 100 patterns associate with two or more pixel values. In Figure 7, we sort the artifact patterns according to the probabilities of their dominant pixel values, so that the smallest is on the left.

This observation of dominance suggests that we can simply keep the dominant pixel value $\hat{p}^*$ and discard all other pixel values $\{\hat{p}_1, \hat{p}_2, \cdots, \hat{p}_k\} - \{\hat{p}^*\}$. Hence, there is no need to store the probability distribution. This simplification reduces the space consumed by probability distributions, so that more distinct artifact patterns can be kept.

4.4 Tree-Structured Vector Quantization

Even though we only extract artifact patterns from the Y component of color images, there are still enormous number of patterns from the training set. Management of such huge number of artifact patterns is crucial for fast query, hence fast synthesis. Brute-force searching among the sea of artifact patterns is inefficient. Therefore, an indexing scheme is needed. Furthermore, even though we have a lot of extracted artifact patterns, we may not find an exact match in some cases. This is especially true when the training set is small or it contains images with less diversity. In case no exact matching exists, the indexing scheme should allow us to locate the closest artifact pattern.

To solve this problem, we employ an indexing technique called tree-structured vector quantization (TSVQ) [Gersho and Gray 1992]. Note that other indexing schemes such as kD-tree [Bentley 1975] with approximate nearest neighbor (ANN) can also be adopted. TSVQ is usually used in data compression. Because of its hierarchical tree structure, an artifact pattern can be rapidly looked up. It also has a nice feature that allows efficient searching of the close pattern when the exact match does not exist. Note that the close pattern returned by TSVQ without ANN may not be the true nearest neighbor because the true one may locate in another node. The TSVQ with ANN approach [Arya et al. 1998] may obtain closer neighbor than the TSVQ

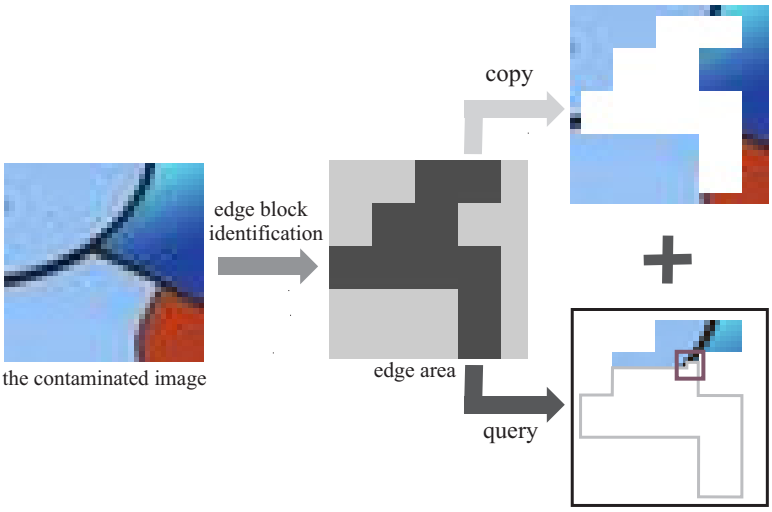


Fig. 8. Pixels in the contaminated image are classified into two types. Pixels in the non-edge blocks are copied directly as the ringing artifacts are not apparent. Pixels in the edge blocks are synthesized by querying the TSVQ.

without ANN. Nevertheless, TSVQ without ANN is empirically sufficient for our deringing application and it is simple to implement.

The construction of TSVQ tree is as follows. Firstly, the centroid of all artifact vectors is computed, and this centroid represents all artifact vectors. A node holding this centroid is formed and is assigned as the root of TSVQ tree. Then we use k-mean clustering to divide the vectors into τ_c groups, where τ_c is user-defined constant. For each group, we compute the centroid and let it be the representative of that group. To do so, we create a node for each centroid and connect it as a child node of the root node. Each child node is actually the root of the subtree representing the corresponding group. Each group (subtree) is then recursively subdivided (branched) until one of the stopping criteria is reached.

To subdivide the set of artifact vectors into groups, we need a metric to measure the similarity among vectors. We employ the weighted Euclidean distance function.

$$D_e(\omega_a, \omega_b) = \sqrt{\sum_i \sum_j G(i,j)(\omega_a(i,j) - \omega_b(i,j))^2}\ , \tag{3}$$

where ω_a and ω_b are artifact patterns being compared, and G is a 2D Gaussian kernel with higher weight at the center.

There are also two user-controllable stopping criteria: the maximum number of subdivided groups (number of children), τ_c, and the maximum tree depth τ_d. If any one of them is satisfied, the subdivision should be stopped.

To query a pattern in the tree, we start the comparison with the child nodes of the tree root. The closest child node is then located and its branch is traversed. The process continues until a leaf node is encountered. The closest vector in the leaf node is matched and returned as the query result. The time complexity of querying a vector is $O(\log_{\tau_c} n)$, where n is the total number of artifact patterns.

4.5 Artifact Removal

Once the TSVQ tree is constructed, we can perform the artifact removal. The constructed TSVQ tree is actually the *priori* knowledge of BDCT-encoding artifacts. Given a contaminated color image, we apply the

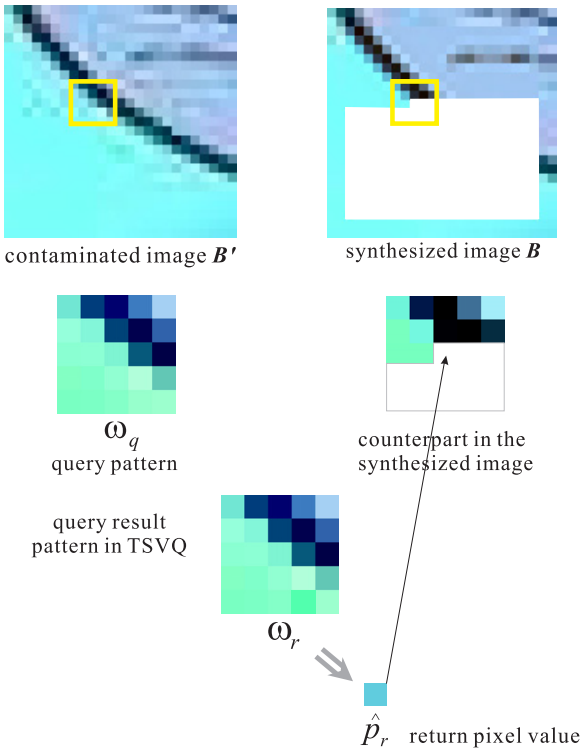


Fig. 9. Artifact removal. Given the edge region in contaminated image B', we synthesize the artifact-reduced image B in a pixel-by-pixel manner. For illustration purpose, artifact vectors are drawn as a block of pixels.

synthesis on the Y component only as explained in Section 4.2.

As the BDCT encoding can effectively handle smooth content (non-edge block), the ringing artifact mainly locates in edge blocks. So we first separate the image into the edge and non-edge blocks. Such separation not just speeds up the process, but also avoids the method from incorrectly introducing error into non-edge blocks which have a less chance of being contaminated by ringing artifacts. The same identification method as in Section 4.1 is used to identify edge blocks (Figure 8). All pixels in the non-edge blocks of the contaminated image are simply copied to the output as illustrated on the upper path of Figure 8. All pixels in the edge blocks have to be synthesized by querying the TSVQ tree.

The query process is illustrated in Figure 9. Given a contaminated image B', we try to synthesize an artifact-reduced image B in a pixel-by-pixel manner for each pixel in edge blocks. For each pixel being synthesized, we form a query pattern ω_q by reordering its corresponding 5×5 neighborhood in image B'. Image B is initialized with a frame of width of 2 pixels copied from the contaminated image B'. The reason is that pixels on this frame cannot form a 5×5 neighborhood for query. This is one restriction of our current approach.

We then query the artifact pattern ω_q, by traversing the TSVQ tree. Firstly, we compare the query vector ω_q with every child node of the root. The one with the smallest D_e (Equation 3) is selected. The search continues to the next level of the branch until a leaf node is encountered. We then sequentially compare all patterns in the leaf node and return the closest one. This sequential matching process does not take much time as the number of patterns inside a leaf node is bounded by τ_c. If ω_r is the closest pattern, its implied

pixel value $\hat{p}_r$ is returned and copied to image B. The synthesis continues until the whole image B is filled.

From experiments, we found that if we accept every returned value, the proposed method may introduce error (over-aggressive change) to the synthesized image. This is especially true when the size of artifact patterns is small. One can suppress such error by increasing the size of artifact patterns, but with the increase of computation and storage requirement. In the original image analogies approach, a coherence search is performed to ensure coherence. Instead, we suppress such error by rejecting over-aggressive change of pixel values according to a threshold based on statistics. To obtain a proper threshold, we study the pixel value change of the original and the contaminated images due to BDCT encoding of different bit rates. The average changes at different bit rates can be determined and served as the guideline in choosing the threshold. Without the coherence search, our method has an extra advantage of parallelism.

5. RESULTS

5.1 Statistical and Visual Comparisons

To verify the proposed method, we first create a training set of 112 images, as shown in Figure 3. The original artifact-free images are compressed by the JPEG codec from Independent JPEG Group to obtain 112 contaminated images. Then the TSVQ tree is constructed with $\tau_c = 9$ and $\tau_d = 7$, as we found this configuration maintains a balance between the query efficiency and the complexity of tree. To measure the quality of synthesized images, we employ peak signal-noise ratio (PSNR) and Structural SIMilarity (SSIM) [Wang et al. 2004]. PSNR is one of the most widely used image quality assessments. However, it does not fully capture the perceptual quality. Hence, we also make use of SSIM index which measures the similarity of structure rather than pure pixel differences. Larger SSIM value implies better image quality. The maximum SSIM value is 1.

Firstly, we test our method with 39 cartoon images compressed with JPEG. None of them is inside the training set. The control images are either originally stored in lossless image formats or rasterized from vector graphics. With our method, all input images are improved in terms of PSNR and SSIM. We obtain an average *net* PSNR improvement of 0.379 dB and average *net* SSIM improvement of 0.017. Figure 10 shows 4 tested images, "dog", "sunflower", "elephant" and "tripod". Columns 1, 2, & 3 show the original, JPEG compressed, and our synthesized images, respectively. Substantial visual improvement is achieved without over-blurring or exaggerating the edges. The corresponding PSNR and SSIM values are shown underneath each blow-up image. They confirm with the visual comparison.

5.2 Comparison with Existing Methods

In order to evaluate the performance of the proposed method, we also compare it with existing methods that may handle ringing artifacts, including bilateral filtering [Tomasi and Manduchi 1998], signal adaptive filtering (SAF) [Lee et al. 1998], the POCS-based Zakhor method [Zakhor 1992], and the baseline image analogies [Hertzmann et al. 2001]. Figure 10 visually compares our method with these methods. Bilateral filtering averages gray levels based on both geometric closeness and photometric similarity. In our comparison, the geometric spread σ_d is set to 5 pixels, which is the same size of neighborhood in the proposed method. The photometric spread σ_r is set to 30. When σ_r is small, bilateral filtering may not effectively reduce ringing artifacts. On the other hand, when σ_r is large, edges may be destroyed. We have tried different combinations of parameter values in bilateral filtering and, finally, we came up with the above setting. SAF filters the image adaptively by the signal of edge. Zakhor method applies a low-pass filter to the contaminated image and constrains the change of each pixel by the given quantization table. We also compared our method with the baseline image analogies [Hertzmann et al. 2001].

Among all tested methods, our method produces the best result, not only visually but also statistically (in terms of PSNR and SSIM). Our method can nicely preserve the small dots in Figure 10A and the weak edges

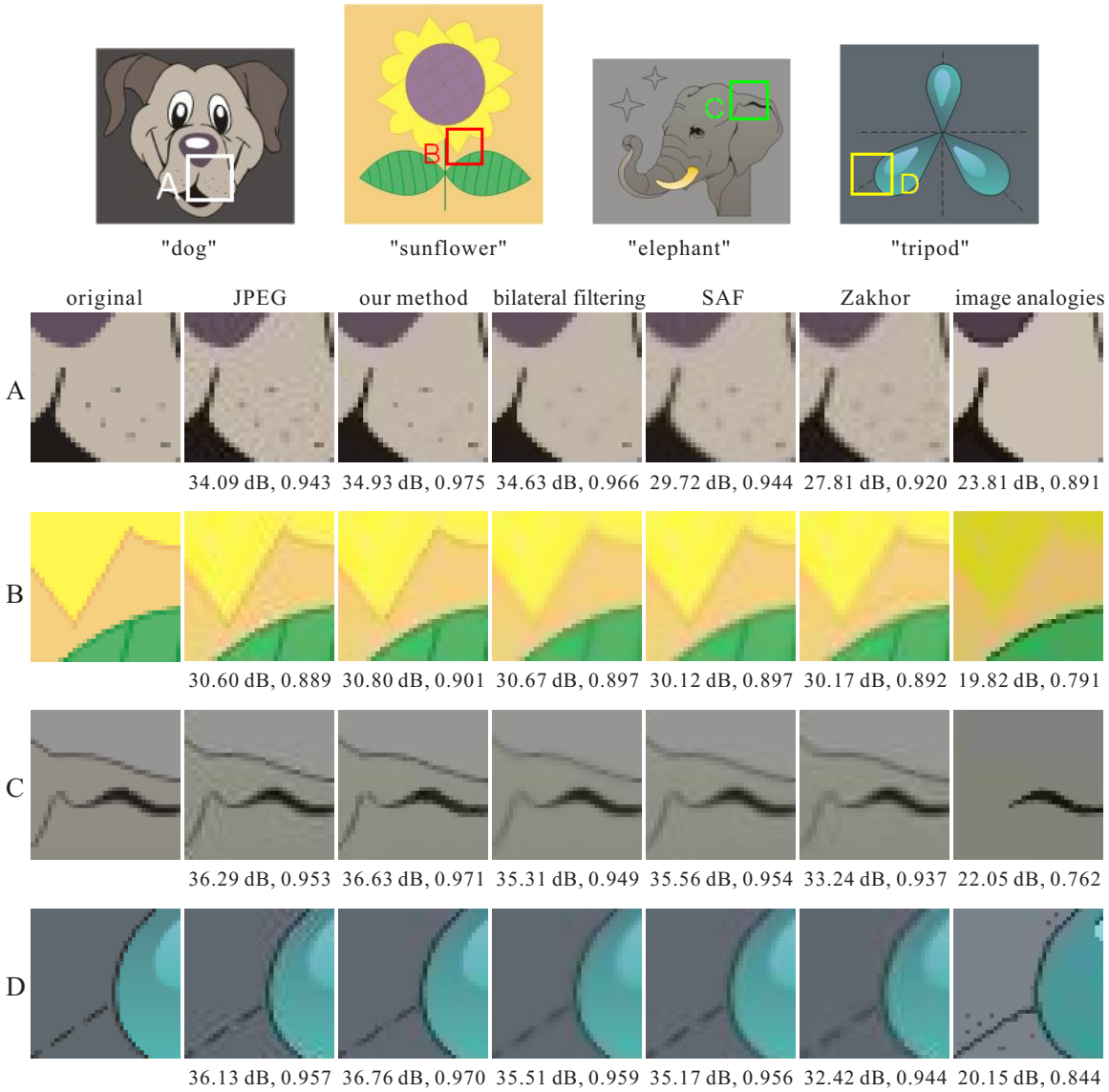
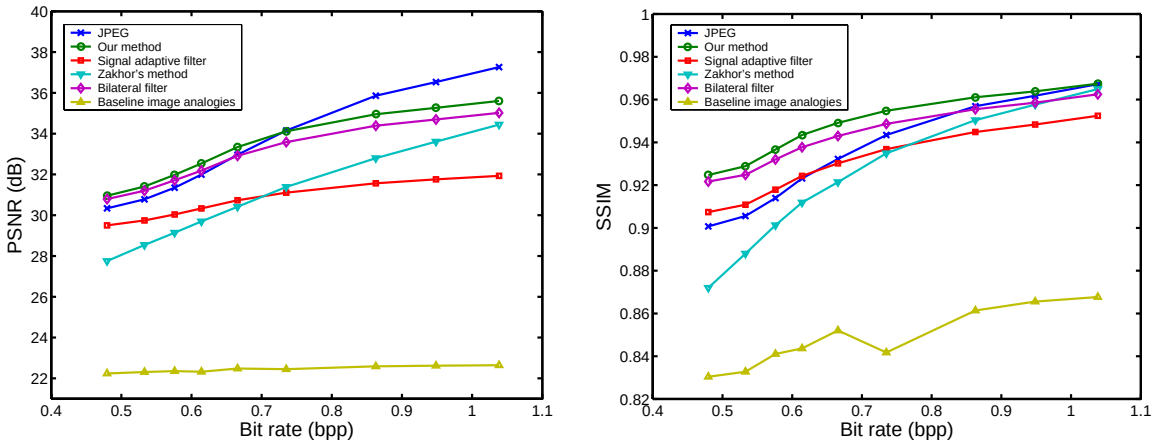


Fig. 10. Four tested images are shown on the first row. The boxed regions in these images are labeled and blown up in the matrix of sub-images. From left to right, the original images, corresponding JPEG images, the results of our method, bilateral filtering, SAF, Zakhor's method, and baseline image analogies are shown accordingly. Below each blow-up, we also list the corresponding PSNR and SSIM of the whole image (not only the blow-up area).

in Figure 10B, while other methods fail. The first runner-up is bilateral filtering which effectively derings and preserves details when the edge signal is strong (column 4 in Figure 10). However, when obvious edge is absent (such as the dotted pattern in Figure 10A), the bilateral filtering may smooth out or even remove the dots. From the results, SAF and Zakhor method usually over-blur the edges, or cannot effectively remove the ringing artifacts. The baseline image analogies may introduce certain artifacts including changes in overall



(a) (b)

Fig. 11. Comparison with other methods in terms of (a) PSNR and (b) SSIM. A set of tested JPEG images are encoded with different bit rates. We evaluate how these methods perform as the bit rate of the input contaminated images changes. Our method out-performs all other methods, in terms of both PSNR and SSIM. In (a), our curve (the green one with circle markers) is above the curve of JPEG (the blue one with cross markers) before 0.735 bpp. The decrease in PSNR after 0.735 bpp not necessarily means the visual quality decreases. (b) shows that our method still improves the visual quality after 0.735 bpp.

color tone and removal and/or introduction of details (as shown in the rightmost column in Figure 10). Although we have tried different combinations of parameter values in the image analogies software, such artifacts still exist. One possible source of artifacts is the coherence search which does not exist in our proposed method. However, we have to emphasize that the baseline method is not tailormade and optimized for deringing.

For a comprehensive comparison, we also tested all five methods on JPEG images encoded with different bit rates. This allows us to evaluate how the methods perform as the bit rate (hence the compression ratio) of the input contaminated images changes. We compare the methods in terms of PSNR and SSIM ((a) and (b) of Figure 11 respectively). Our method out-performs all other methods. Note that artifact is apparent mainly at low bit rate. Our curve in Figure 11(a) is above the curve of JPEG before 0.735 bpp. The decrease in PSNR after 0.735 bpp not necessarily means the visual quality decreases. Figure 11(b) shows that our method still improves the visual quality after 0.735 bpp. Interestingly, although SAF method does not improve the PSNR (its curve is below the JPEG curve in Figure 11(a)), it still improves the visual quality in terms of SSIM (its curve is above the JPEG curve in Figure 11(b), in lower bit rates). Figure 11(b) also shows that our method achieves larger net improvement (larger gap between our curve and that of JPEG) when the bit rate is low. This makes sense as there is not much room for visual quality improvement when the images are encoded at high bit rate. The poor statistics of the baseline image analogies are mainly caused by the color tone change.

5.3 Size of Training Set

It is natural to expect the performance is directly affected by the size of training set. To observe the effect of the size of training set, we conduct another experiment. During the experiment, we enlarge the size of training set gradually, from 50,000 to 200,000 (number of distinct artifact patterns). We then measure the average PSNR and SSIM improvements between the contaminated input images and the synthesized images. The result is shown in Figure 12. As the number of artifact patterns increases, the PSNR and SSIM improvements increase. However the PSNR and SSIM improvements tend to be saturated after certain point.

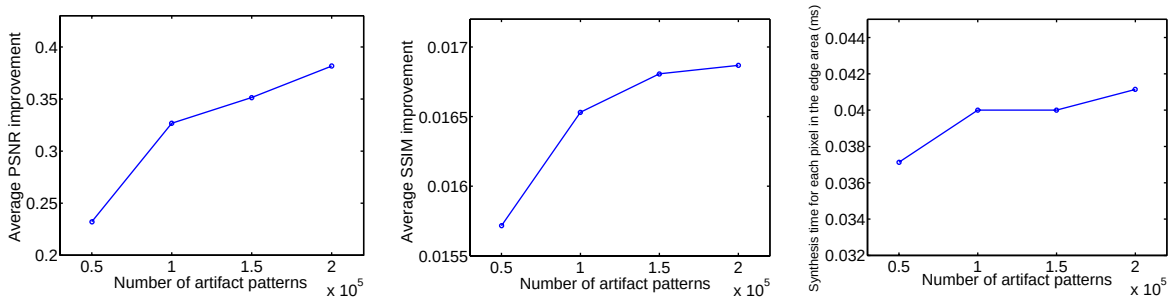


Fig. 12. Statistics as the number of artifact patterns increases from 50,000 to 200,000. As the number of artifact patterns increases, the PSNR (a) and SSIM (b) improvements increase. However such improvements tend to be saturated after certain point. The processing time (c) slightly increases as the number of artifact patterns increases.

5.4 Timing Statistics

The time of synthesis depends on the size of TSVQ tree, the resolution of input image, and the image content. Copying of non-edge block takes very little amount of time. The major computation is spent on the synthesis of pixels in the edge blocks.

To measure the time of synthesis, we compute the average query time for a pixel. It equals to the total time of synthesizing edge pixels divided by the total number of edge pixels. We plot a graph of this average query time against the number of distinct artifact vectors in TSVQ tree in Figure 12(c). The timing statistics are recorded on Pentium III 870 MHz with 256 MB memory. Obviously, the time slightly increases as the number of artifact patterns increases. It confirms with the $O(\log_{\tau_c} n)$ running time property of TSVQ. Since only a small portion of the image requires query, the total time for artifact removal is usually much smaller than the multiplication of the total number of pixel and the average query time per pixel. As an example, the time for synthesizing a 256×256 artifact-reduced image takes only 0.5 second.

5.5 Size of Neighborhood

As mentioned in Section 4.1, selecting the size of neighborhood is a trade-off. Larger neighborhood in general captures more local information, but the computational cost is higher. Since the block size adopted in practical compression standards, like JPEG and MPEG, is 8×8 and each block is transformed and quantized independently, the size of neighborhood should not exceed 8×8. We tested 3 sizes in our experiment: 3×3, 5×5, and 7×7. Statistics are shown in Figure 13. Both PSNR (Figure 13(a)) and SSIM (Figure 13(b)) values reach the maximum when the size is 5×5. A 3×3 neighborhood cannot capture sufficient local information. When the size is raised to 7×7 (49 pixels), the curse of dimension appears. As the ringing artifact normally occurs at edge region which only occupies a small portion of the 7×7 region, the Euclidean distance between two artifact vectors no longer effectively measures their difference as it has been "diluted" by the dominant non-edge pixels. This will reduce the accuracy of artifact pattern query and degrade the deringing results. Besides, the computational cost increases as the neighborhood size increases (Figure 13(c)). From the figures, the choice of 5×5 is optimal.

5.6 Video Deringing

Unlike the wide variety of still image coding methods (which may not be BDCT-based), popular video coding standards, like MPEG, usually adopt BDCT encoding. The proposed method would be even more useful in deringing cartoon videos. To verify, we applied the proposed method to dering a MPEG-encoded cartoon

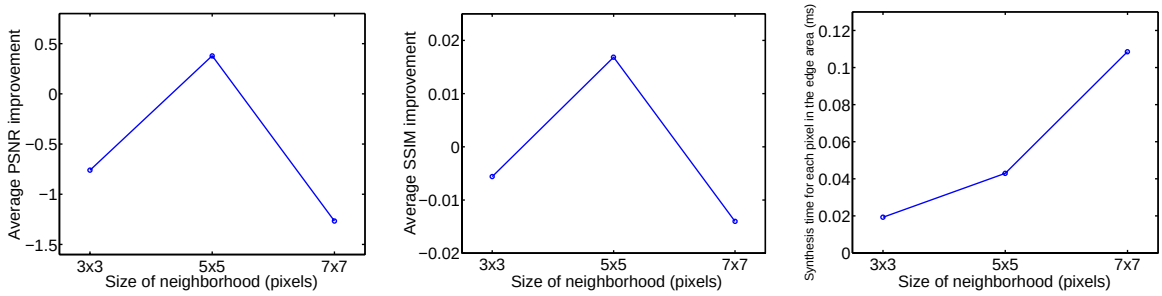


Fig. 13. Statistics as the size of neighborhood increases from 3×3 to 7×7. Both PSNR (a) and SSIM (b) values reach the maximum when the neighborhood size is 5×5. As the size increases, the computational cost increases (c).

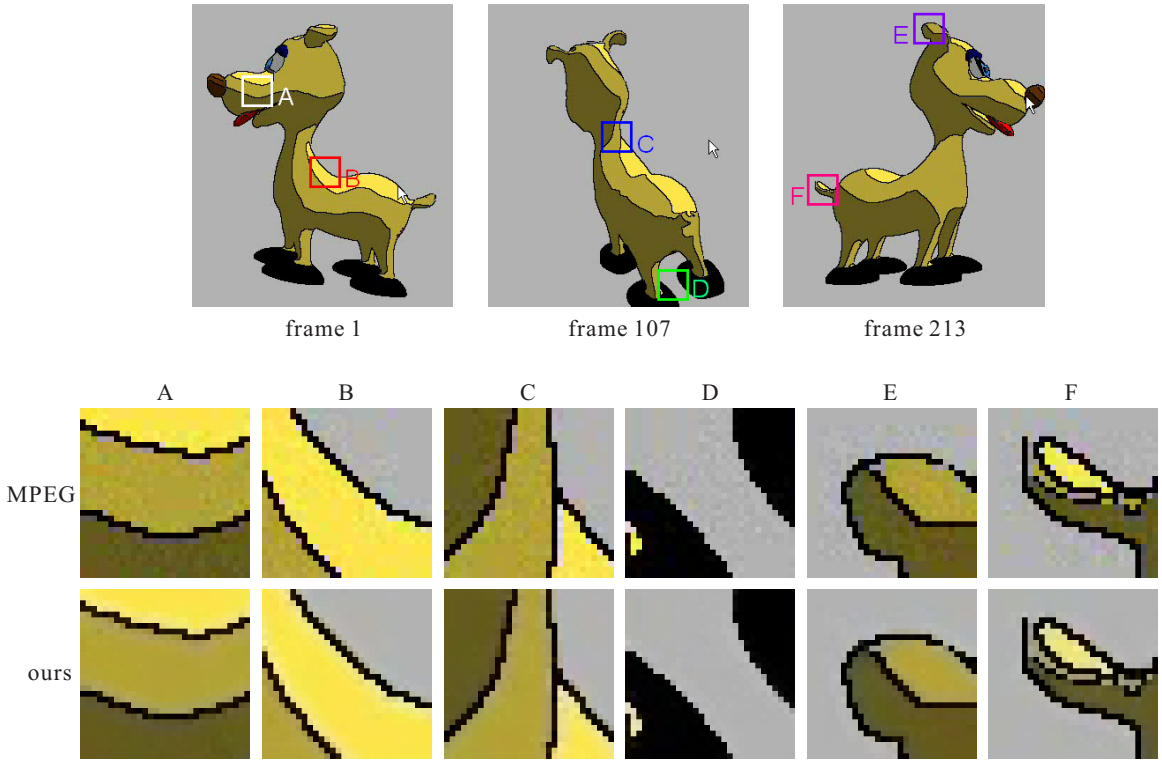
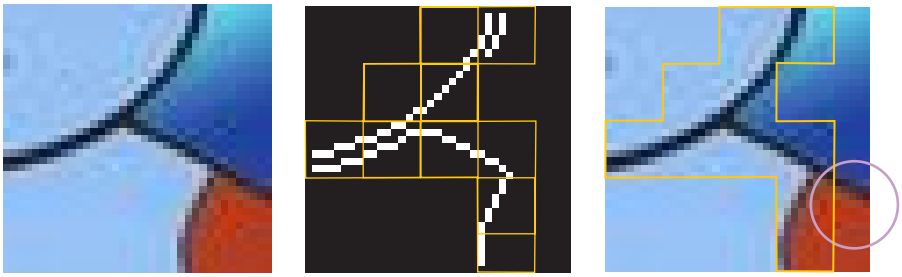


Fig. 14. Application to MPEG video. Three example frames are shown on the upper row. The boxed regions in these frames are blown up in the following rows. The middle row shows the original MPEG-contaminated frames while the bottom row shows our results.

video by deringing each frame separately. There are totally 271 frames. In Figure 14, three example frames are shown on the upper row. The boxed regions in these frames are blown up in the following rows. The middle row shows the input MPEG-contaminated frames. The bottom row shows our results. The ringing artifacts are significantly reduced while details are preserved.



(a) (b) (c)

Fig. 15. Some edges (in the circle of (c)) may not be identified and hence cannot be synthesized.

6. CONCLUSIONS

By modeling the contaminated image as MRF, we propose an example-based method for reducing the ringing artifacts in contaminated cartoon images. Instead of post-filtering the contaminated image, we synthesize an artifact-reduced image. Using our method, the image quality of all tested cases is substantially improved, not only subjectively but also objectively. Our method effectively handles the ringing artifacts which cannot be effectively solved by previous methods. Like other example-based methods, the size of training data affects the performance. In general, more training samples will give better PSNR improvement.

One limitation of our current method is illustrated in Figure 15. In this example, the edge detection fails to detect the edge circled in Figure 15(c). Therefore that particular block is tagged as non-edge block and no synthesis is performed. A partial solution is to make use of multiple edge detectors. However, no matter what kind of edge detection filter is used, it is still possible that some edges are missed. Even worse, edges in the input contaminated images may be already blurred by the BDCT encoding. In that case, a different strategy of contaminated block detection is needed. For instance, the gradient sum of a block may be useful. Further investigation in finding a robust contamination detection method is needed. Currently, our method synthesizes images in a pixel-by-pixel manner. We believe the speed can be substantially increased if a patch-based approach is used. However whether the quality of synthesized images is still preserved requires further investigation.

Although our method is initially designed for cartoon images, it can also be applied to natural images in theory. In our preliminary experiment on natural images "calendar" and "palace" (Figure 16), the effectiveness of the proposed method is not conclusive. In Figure 16, the first row shows the JPEG contaminated images, labeled with boxes. We blow up the boxed regions on the following rows for visual comparison. Our results are on the bottom row. It can be shown that at high-contrast regions (Figures 16 A, B, C and D), the ringing artifacts are reduced. However in Figure 16E, the details are incorrectly blurred. Our program mistakenly regards the details in Figure 16E as artifacts and performs synthesis. The major reason is that our current training set is tailored for cartoon images.

To solve this problem, we can first increase the number of distinct artifact patterns in TSVQ tree. It would be interesting to study, as the pattern number increases, whether the dominant pixel phenomenon stills hold. Furthermore, a more robust mean to detect edge regions is needed to avoid the detection failure of blurry edges (such as in Figure 15(c)) frequently found in natural images.

A demonstrative artifact removal program is available for evaluation at

`http://www.cse.cuhk.edu.hk/~ttwong/demo/arti/arti.html`

Companion video and other results are available at

`http://www.cse.cuhk.edu.hk/~ttwong/papers/artifact/artifact.html`

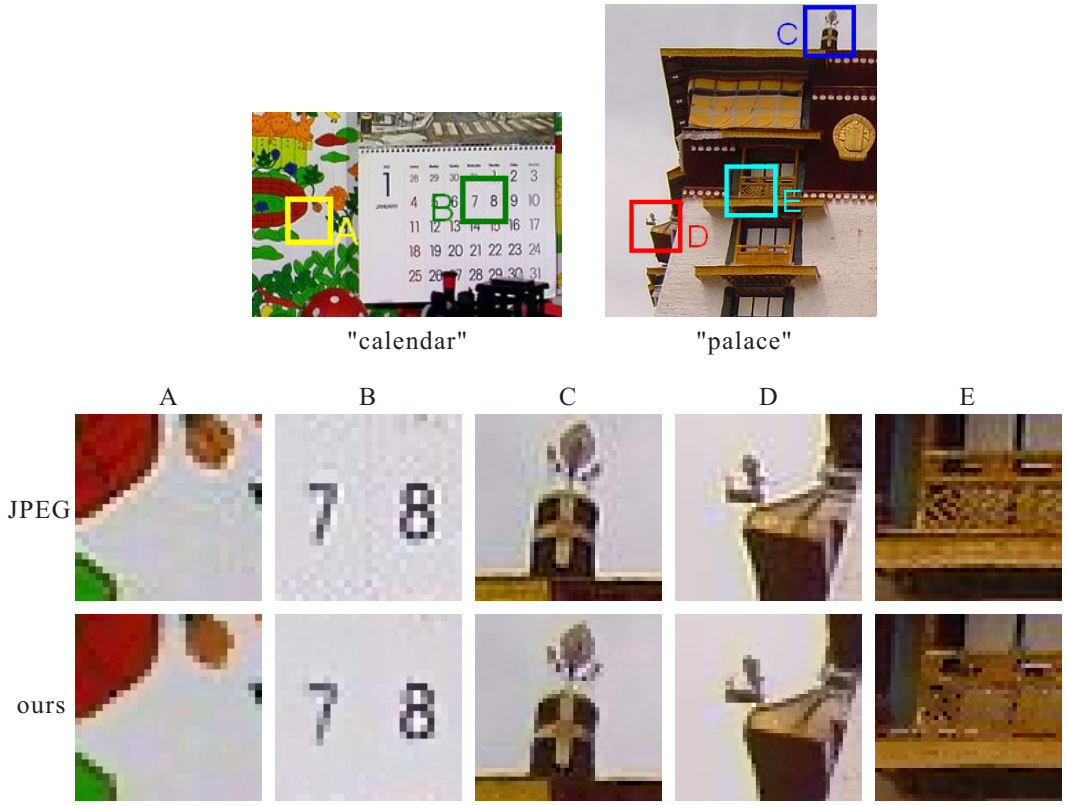


Fig. 16. Application to natural images. The first row shows the JPEG contaminated images, labeled with boxes. We blow up the boxed regions in the second row for visual comparison. The results processed by our method are on the bottom row.

REFERENCES

Arya, S., Mount, D. M., Netanyahu, N. S., Silverman, R., and Wu, A. Y. 1998. An optimal algorithm for approximate nearest neighbor searching fixed dimensions. *Journal of the ACM 45*, 891–923.

Bar-Joseph, Z., El-Yaniv, R., Lischinski, D., and Werman, M. 2001. Texture mixing and texture movie synthesis using statistical learning. *IEEE Trans. Visual. Comput. Graphics 7,* 2, 120–135.

Bentley, J. L. 1975. Multidimensional binary search trees used for associative searching. *Communication of the ACM 18,* 9 (Sep.), 509–517.

Chan, Y. H., Hong, S. W., and Siu, W. C. 1998. A practical postprocessing technique for real-time block-based coding system. *IEEE Trans. Circuits Syst. Video Technol. 8,* 1 (Feb.), 4–8.

Chen, T., Wu, H. R., and Qiu, B. 2001. Adaptive postfiltering of transform coefficients for the reduction of blocking artifacts. *IEEE Trans. Circuits Syst. Video Technol. 11,* 5 (May), 594–602.

Cohen, M. F., Shade, J., Hiller, S., and Deussen, O. 2003. Wang tiles for image and texture generation. In *SIGGRAPH 2003 Conference Proceedings*. 287–294.

Combettes, P. L. 1993. The foundation of set theoretic estimation. *Proc. IEEE 81,* 2 (Feb.), 182–208.

Efros, A. A. and Freeman, W. T. 2001. Image quilting for texture synthesis and transfer. In *SIGGRAPH 2001 Conference Proceedings*. 341–346.

Efros, A. A. and Leung, T. K. 1999. Texture synthesis by non-parametric sampling. In *Proc. IEEE Int'l Conf. Computer Vision*. Corfu, Greece, 1033–1038.

Freeman, W. T., Jones, T. R., and Pasztor, E. C. 2002. Example-based super-resolution. *IEEE Comput. Graph. Appl. 22,* 2 (Mar./Apr.), 56–65.

Freeman, W. T., Pasztor, E. C., and Carmichael, O. T. 2000. Learning low-level vision. *International Journal of Computer Vision 40,* 1, 25–47.

Gall, D. L. 1991. MPEG: A video compression standard for multimedia applications. *Communications of the ACM 34,* 4 (Apr.), 46–58.

Gersho, A. and Gray, R. M. 1992. *Vector Quantization and Signal Compression.* Kluwer Academic Publishers, Boston.

Gunturk, B. K., Altunbasak, Y., and Mersereau, R. M. 2002. Multiframe blocking-artifact reduction for transform-coded video. *IEEE Trans. Circuits Syst. Video Technol. 12,* 4 (Apr.), 276–282.

Hertzmann, A., Jacobs, C., Oliver, N., Curless, B., and Salesin, D. 2001. Image analogies. In *SIGGRAPH 2001 Conference Proceedings.* 327–340.

Hertzmann, A., Oliver, N., Curless, B., and Seitz, S. M. 2002. Curve analogies. In *Proc. 13th Eurographics Workshop on Rendering.* Pisa, Italy, 233–245.

Hsu, Y. F. and Chen, Y. C. 1993. A new adaptive separable median filter for removing blocking effects. *IEEE Trans. Consumer Electron. 39,* 3 (Aug.), 510–513.

Jarske, T., Haavisto, P., and Defee, I. 1994. Post-filtering methods for reducing blocking effects from coded images. *IEEE Trans. Consumer Electron. 40,* 3 (Jun.), 521–526.

Jeong, Y., Kim, I., and Kang, H. 2000. A practical projection-based postprocessing of block-coded images with fast convergence rate. *IEEE Trans. Circuits Syst. Video Technol. 10,* 4 (Jun.), 617–623.

Kong, H.-S., Vetro, A., and Sun, H. 2004. Edge map guided adaptive post-filter for blocking and ringing artifacts removal. In *IEEE International Symposium on Circuits and Systems (ISCAS).* Vol. 3. 929–932.

Kuo, C. J. and Hsieh, R. J. 1995. Adaptive postprocessor for block encoded images. *IEEE Trans. Circuits Syst. Video Technol. 5,* 4 (Aug.), 298–304.

Kwatra, V., Schödl, A., Essa, I., Turk, G., and Bobick, A. 2003. Graphcut textures: Image and video synthesis using graph cuts. In *SIGGRAPH 2003 Conference Proceedings.* 277–286.

Lee, Y. L., Kim, H. C., and Park, H. W. 1998. Blocking effect reduction of JPEG images by signal adaptive filtering. *IEEE Trans. Image Processing 7,* 2 (Feb.), 229–234.

Lee, Y.-L., Park, H.-W., Park, D.-S., and Kim, Y.-S. 1998. Loop filtering for reducing blocking effects and ringing effects. Fifth Meeting, Video Coding Experts Group, ITU - Telecommunications Standardization Sector, ITU Document No. Q15-E-22.

Li, S. Z. 1995. *Markov Random Field Modeling in Computer Vision.* Springer-Verlag, New York.

Liu, C., Wang, C., and Lin, J. 1998. A new postprocessing method for the block-based DCT coding based on the convex-projection theory. *IEEE Trans. Consumer Electron. 44,* 3 (Aug.), 1054 –1061.

Liu, X., Hu, Y., Zhang, J., Tong, X., Guo, B., and Shum, H.-Y. 2004. Synthesis and rendering of bidirectional texture functions on arbitrary surfaces. *IEEE Trans. Visual. Comput. Graphics 10,* 3, 278–289.

Luo, J., Chen, C. W., Parker, K. J., and Huang, T. S. 1996. Artifact reduction in low bit rate DCT-based image compression. *IEEE Trans. Image Processing 5,* 9 (Sep.), 1363–1368.

Mese, M. and Vaidyanathan, P. P. 2001. Look-up table (LUT) method for inverse halftoning. *IEEE Trans. Image Processing 10,* 10 (Oct.), 1566–1578.

Minami, S. and Zakhor, A. 1995. An optimization approach for removing blocking effects in transform coding. *IEEE Trans. Circuits Syst. Video Technol. 5,* 2 (Apr.), 74–82.

Paek, H., Kim, R. C., and Lee, S. U. 1998. On the POCS-based postprocessing technique to reduce the blocking artifacts in transform coded images. *IEEE Trans. Circuits Syst. Video Technol. 8,* 3 (Jun.), 358–367.

Paek, H., Kim, R. C., and Lee, S. U. 2000. A DCT-based spatially adaptive post-processing technique to reduce the blocking artifacts in transform coded images. *IEEE Trans. Circuits Syst. Video Technol. 10,* 1 (Feb.), 36–41.

Perona, P. and Malik, J. 1990. Scale-space and edge detection using anisotropic diffusion. *IEEE Trans. Pattern Anal. Machine Intell. 12,* 7 (Jul.), 629–639.

Praun, E., Finkelstein, A., and Hoppe, H. 2000. Lapped textures. In *SIGGRAPH 2000 Conference Proceedings.* 465–470.

Qiu, G. 2000. MLP for adaptive postprocessing block-coded images. *IEEE Trans. Circuits Syst. Video Technol. 10,* 8 (Dec.), 1450–1454.

Ramamurthi, B. and Gersho, A. 1986. Nonlinear space-variant postprocessing of block coded images. *IEEE Trans. Acoust., Speech, Signal Processing 34,* 5 (Oct.), 1258–1268.

Reeve, H. C. and Lim, J. S. 1984. Reduction of blocking effect in image coding. *Opt. Eng. 23,* 1 (Jan./Feb.), 34–37.

Tomasi, C. and Manduchi, R. 1998. Bilateral filtering for gray and color images. In *Proc. IEEE Int'l Conf. Computer Vision.* Bombay, India, 839–846.

Triantafyllidis, G. A., Tzovaras, D., and Strintzis, M. G. 2002. Blocking artifact detection and reduction in compressed data. *IEEE Trans. Circuits Syst. Video Technol. 12,* 10 (Oct.), 877–890.

Wallace, G. K. 1991. The JPEG still picture compression standard. *Communications of the ACM 34,* 4 (Apr.), 30–44.

Wang, H. 1965. Games, logic and computers. *Scientific American* (Nov.), 98–106.

Wang, Z., Bovik, A. C., Sheikh, H. R., and Simoncelli, E. P. 2004. Image quality assessment: From error visibility to structural similarity. *IEEE Trans. Image Processing 13,* 1 (Jan.), 600–612.

Weerasinghe, C., Liew, A. W., and Yan, H. 2002. Artifact reduction in compressed images based on region homogeneity constraints using the projection onto convex sets algorithm. *IEEE Trans. Circuits Syst. Video Technol. 12,* 10 (Oct.), 891–897.

Wei, L. Y. and Levoy, M. 2000. Fast texture synthesis using tree-structured vector quantization. In *SIGGRAPH 2000 Conference Proceedings*. 479–488.

Welsh, T., Ashikhmin, M., and Mueller, K. 2002. Transferring color to greyscale images. In *SIGGRAPH 2002 Conference Proceedings*. 277–280.

Yang, S., Hu, Y. H., Nguyen, T. Q., and Tull, D. L. 2001. Maximum-likelihood parameter estimation for image ringing-artifact removal. *IEEE Trans. Circuits Syst. Video Technol. 11,* 8 (Aug.), 963–973.

Yang, Y. and Galatsanos, N. P. 1997. Removal of compression artifacts using projections onto convex sets and line process modeling. *IEEE Trans. Image Processing 6,* 10 (Oct.), 1345–1357.

Yang, Y., Galatsanos, N. P., and Katsaggelos, A. K. 1995. Projection-based spatially adaptive reconstruction of block-transform compressed images. *IEEE Trans. Image Processing 4,* 7 (Jul.), 896–908.

Zakhor, A. 1992. Iterative procedures for reduction of blocking effects in transform image coding. *IEEE Trans. Circuits Syst. Video Technol. 2,* 1 (Mar.), 91–95.