Self-learning-based post-processing for image/video deblocking via sparse representation

Chia-Hung Yeh a, Li-Wei Kang b,c,*, Yi-Wen Chiou a, Chia-Wen Lin d,e, Shu-Jhen Fan Jiang a

a Department of Electrical Engineering, National Sun Yat-sen University, Kaohsiung 804, Taiwan
b Graduate School of Engineering Science and Technology-Doctoral Program, National Yunlin University of Science and Technology, Yunlin 640, Taiwan
c Department of Computer Science and Information Engineering, National Yunlin University of Science and Technology, Yunlin 640, Taiwan
d Department of Electrical Engineering, National Tsing Hua University, Hsinchu 300, Taiwan
e Graduate School of Engineering Science and Technology-Doctoral Program, National Yunlin University of Science and Technology, Yunlin 640, Taiwan
f Institute of Communications Engineering, National Tsing Hua University, Hsinchu 300, Taiwan

ARTICLE INFO

Article history:
Received 17 April 2013
Accepted 10 February 2014
Available online 26 February 2014

Keywords:
Blocking artifact
Deblocking
Sparse representation
Dictionary learning
Morphological component analysis
Image/video restoration
Image/video enhancement
Post-processing

ABSTRACT

Blocking artifact, characterized by visually noticeable changes in pixel values along block boundaries, is a common problem in block-based image/video compression, especially at low bitrate coding. Various post-processing techniques have been proposed to reduce blocking artifacts, but they usually introduce excessive blurring or ringing effects. This paper proposes a self-learning-based post-processing framework for image/video deblocking by properly formulating deblocking as an MCA (morphological component analysis)-based image decomposition problem via sparse representation. Without the need of any prior knowledge (e.g., the positions where blocking artifacts occur, the algorithm used for compression, or the characteristics of image to be processed) about the blocking artifacts to be removed, the proposed framework can automatically learn two dictionaries for decomposing an input decoded image into its “blocking component” and “non-blocking component.” More specifically, the proposed method first decomposes a frame into the low-frequency and high-frequency parts by applying BM3D (block-matching and 3D filtering) algorithm. The high-frequency part is then decomposed into a blocking component and a non-blocking component by performing dictionary learning and sparse coding based on MCA. As a result, the blocking component can be removed from the image/video frame successfully while preserving most original visual details. Experimental results demonstrate the efficacy of the proposed algorithm.

© 2014 Elsevier Inc. All rights reserved.

1. Introduction

Block-based transform coding has been among the most common compression techniques for image/video due to its regularity and simplicity for hardware implementation. It divides an image/video frame into several non-overlapping square blocks, followed by applying a transform operation to each individual block as part of the encoding process. Based on this methodology, several image/video coding standards (e.g., JPEG, MPEG-1/2/4, H.261–H.264, and HEVC) have been developed for various kinds of applications, including multimedia communication, videotelephony, videoconferencing, mobile video, video streaming, and high-definition television (HDTV). The major goal of these compression techniques is to efficiently compress a large amount of visual information to fit the bandwidth limits of communication channels while preserving acceptable quality of reconstructed data [1–3].

Among the transform operations employed in the current compression standards, the block DCT (discrete cosine transform) is the most popularly adopted one due to its good energy compaction and decorrelation properties. At low bitrates, the block DCT-based coding techniques exhibit a visually annoying phenomenon in reconstructed image/video, known as the blocking artifacts, which is characterized by visually noticeable changes in pixel values along block boundaries, as illustrated in Fig. 1. The major goal of this paper is to propose a framework to remove or reduce blocking artifacts to obtain visually acceptable quality for block DCT-based compressed image/video at low bitrates. This procedure is usually referred to as deblocking [4,5].

A large amount of research has been proposed to eliminate blocking artifacts [6–25]. Among them, post-processing techniques are the most practical ones, which can be performed after decoding and easily incorporated in any existing compression standards or paradigms. In addition, with the rapid increase of available computing power, more sophisticated post-processing methods can also be

* Corresponding author at: Graduate School of Engineering Science and Technology-Doctoral Program, and Department of Computer Science and Information Engineering, National Yunlin University of Science and Technology, Yunlin 640, Taiwan.
E-mail address: lwkang@yuntech.edu.tw (L.-W. Kang).
easily implemented at the decoder. Roughly speaking, current post-processing techniques for image/video deblocking can be derived from two different principles [4], namely, image enhancement [6–17] and image restoration [18–25], as briefly reviewed below.

1.1. Deblocking via image enhancement

The goal of the image enhancement approach [6–17] is to subjectively improve the perceived image/video quality by considering the special structure of artifacts and human visual sensitivities. A typical solution is to perform post-filtering along block boundaries to reduce blockiness. In general, image enhancement approaches aiming at smoothing visible artifacts, instead of restoring original pixel value, is somewhat heuristic in the sense that no objective criterion is optimized. The main advantage of this kind of approach is usually its relatively low computational complexity.

More specifically, the method proposed in [6] extracts the high-frequency components of an image and then applied an adaptive filter on a scaled version of the obtained image. Tai et al. [9] proposed a deblocking algorithm based on three filtering modes in terms of the activity across block boundaries. Foi et al. [11] proposed an image deblocking approach based on the shape-adaptive DCT in conjunction with the anisotropic local polynomial approximation (LPA)-intersection of confidence intervals (ICI) technique, which defines the shape of the transform’s support in a point-wise adaptive manner. The image deblocking method proposed in [13] involves three parts: i.e., local AC coefficient regularization in the DCT domain, block-wise shape adaptive filtering in the spatial domain, and quantization constraint in the DCT domain. Yeh et al. [14] proposed to automatically detect blocking artifacts at block boundaries, where they provided four filter modes to eliminate the blocking artifacts in various frequency regions based on region activity analysis; to improve both subjective and objective video quality, they also considered the chrominance filtering. H.264/AVC in-loop filter [15] performs simple operations to detect and analyze artifacts on coded block boundaries and attenuates them by applying a selected filter. In [16], two image post-processing techniques, namely, DNLK (dual non-local Kuan’s) filter and OCDNLK (overcomplete dual non-local Kuan’s) filter, were presented based on a more accurate DCT domain Kuan’s filter with non-local parameter estimation to reduce blocking artifacts. In addition, a signal adaptive weighted sum (SAWS) technique for block boundary pixels was proposed in [17] to alleviate the blocking artifacts in a highly compressed image. The weights are adaptively adjusted according to the directional correlation and activities of local areas.

1.2. Deblocking via image restoration

For the image restoration approach [18–25], eliminating blocking artifacts is formulated as an image/video recovery problem. Image/video reconstruction is then performed based on some prior knowledge and observed data at the decoder. This kind of approach can usually be roughly classified into three categories [4], namely, criterion-based [18–20], constraint-based [21–24], and constrained optimization [25] methods. A criterion-based method usually finds the solution satisfying some predefined criterion, which may be an error measurement metric with high correlation to perceptual quality or a probability density of an image. For example, Wu and Gersho [18] proposed a nonlinear estimator utilizing nonlinear interpolation and vector quantization to improve the reconstructed image quality based on the JPEG baseline process. Ozcelik et al. [19] proposed an MAP (maximum a posteriori)-based method adopting a nonstationary Gauss-Markov model for image recovery.

The main idea of the constraint-based approach is to impose a number of constraints on a decoded image and restore it to its artifact-free version. Constraints may be obtained from some prior knowledge of the compression algorithm or properties of the input image. A typical example in this kind of approach is called the projection onto convex sets (POCS), where a constraint set is defined as a closed convex set whose members are consistent with the prior knowledge of an original image. Several POCS-based deblocking frameworks [21–24] have been proposed to restore decoded images with blocking artifacts. For example, in [24], to reduce blocking artifacts appearing in block-coded images, a novel quantization constraint set based on the theory of POCS was proposed, where the set can efficiently complement the drawbacks of the projection onto the other constraint sets, particularly the smoothness constraint set.

On the other hand, the constrained optimization approach optimizes an optimality criterion subject to some constraint(s) obtained from the prior knowledge of an image. For example, Jung et al. [25] proposed a deblocking method via sparse representation for JPEG image. First, they create a general dictionary for sparsely representing images using the K-SVD dictionary learning algorithm [26] based on a set of training images. To remove blocking artifacts of an image, the learned dictionary is used to sparsely represent the image via a sparse coding algorithm, namely OMP (orthogonal matching pursuit) [27]. The key is to impose an error threshold constraint for performing OMP to reconstruct the image without blocking artifacts, where the constraint is derived from the JPEG compression factor for compressing this image. In summary, the
applicability of [25] may be limited by: (i) only for a JPEG image with a known compression factor, which means it is not a pure post-processing approach; some prior knowledge about the used compression algorithm is required; and (ii) a set of pre-collected training images are required.

Generally speaking, the image restoration approach achieves better quality with higher computational complexity. For example, POCS-based methods may exhibit extremely high computational complexity due to its iterative process, where a pair of DCT and inverse DCT operations is performed in each iteration. However, the complexity problem may be solved algorithmically or with the rapid increase of available computing power.

1.3. Contribution of proposed deblocking method

In this paper, we propose a novel deblocking framework for image/video by formulating deblocking as an MCA (morphological component analysis)-based image decomposition problem via sparse representation. In our method, an image/video frame is first decomposed into the low-frequency and high-frequency parts using BM3D (block-matching and 3D filtering) algorithm [28]. The high-frequency part is then decomposed into ’’blocking component’’ and ’’non-blocking component’’ by performing dictionary learning and sparse coding based on MCA. The proposed method is kind of constrained optimization approach. Compared to state-of-the-art techniques (e.g., [25]) in this class, the major contribution of this paper is three-fold: (i) we propose the first self-learning-based image decomposition framework based on MCA for blocking artifact removal; (ii) the learning of the dictionary for removing blocking artifacts from an image is fully automatic and self-contained, where no extra training samples are required; and (iii) the proposed method can be adapted to any block transform-based compressed image/video without the need of any prior knowledge from data source, coding bitrates, and compression algorithms.

It should be emphasized that the proposed deblocking method is a ‘’pure’’ post-processing approach, where the input can be any decoded image/video, instead of a compressed bitstream. Moreover, it does not assume any prior knowledge about the locations (i.e., the block size) of blocking artifacts in an image/video, which is beneficial because, in recent video compression standards, the block size used for DCT can vary from 4 × 4 to 32 × 32. For example, in HEVC [3], the block size can be 4 × 4, 8 × 8, 16 × 16, and 32 × 32. Hence, blocking artifacts will no longer occur only in fixed block boundaries in an image/video. In addition, we will not perform blocking artifact detection first to prevent poor deblocking results induced by possibly inaccurate detection. In fact, the only prior knowledge used in the proposed method is that blocking artifacts act as horizontal or vertical lines in an image/video which is true for block-based coding. Nevertheless, we will also discuss how to improve the performance of the proposed deblocking method if the blocking artifacts in an image/video can be accurately detected in advance.

The rest of this paper is organized as follows. In Section 2, we briefly review the concepts of MCA-based image decomposition, sparse representation, and dictionary learning techniques. Section 3 presents the proposed image/video deblocking framework. In Section 4, experimental results are demonstrated. Finally, Section 5 concludes this paper.

2. MCA-based image decomposition, sparse representation, and dictionary learning

The major idea of MCA is to utilize the morphological diversity of different features contained in the data to decompose and to associate each morphological component to a dictionary of atoms.

In this section, we briefly introduce the conventional MCA-based image decomposition approach [29–31], sparse representation [27,32,33], and dictionary learning [26,34] techniques. More examples of MCA-based image decomposition applications can be found in [29,35–37]. The symbols used in this paper are listed in Table 1.

2.1. MCA-based image decomposition

Suppose an image I of N pixels is a superposition of S layers (called morphological components), denoted by I = \sum_{s=1}^{S} I_s, where I_s denotes the sth component, such as the geometric or textural component of I. To decompose the image I into \{I_s\}_{s=1}^{S}, the MCA algorithms [29–31] iteratively minimize the following energy function:

\[ E\left(\{I_s\}_{s=1}^{S}, \{\theta_s\}_{s=1}^{S}\right) = \frac{1}{2} \left\| \mathbf{I} - \sum_{s=1}^{S} \mathbf{I}_s \right\|_2^2 + \tau \sum_{s=1}^{S} E(s, \theta_s), \]  

where \(\theta_s\) denotes the sparse coefficients corresponding to \(I_s\) with respect to the dictionary \(D_s\), \(\tau\) is a regularization parameter, and \(E(s, \theta_s)\) is the energy function defined according to the type of \(D_s\) (global or local dictionary).

The MCA algorithms solve (1) by iteratively performing for each component \(I_s\), the following two steps: (i) update of the sparse coefficients: this step performs sparse coding to solve \(\theta_s\) or \(\{\phi_k\}_{k=1}^{K}\), where \(\phi_k\) represents the sparse coefficients of the kth patch \(b^k_s\) extracted from \(I_s\), and \(K\) is the total number of extracted patches, to minimize \(E(I_s, \theta_s)\) while fixing \(I_s\); and (ii) update of the components: this step updates \(I_s\) or \(\{b^k_s\}_{k=1}^{K}\) while fixing \(\theta_s\) or \(\{\phi_k\}_{k=1}^{K}\). More details about MCA can be found in [29–31]. In [35–38], we showed that MCA can be used to successfully remove structural noise (e.g., rain streaks) in a single image.

2.2. Sparse representation and dictionary learning

Sparse coding [27,32,33] is a technique of finding a sparse representation for a signal with a small number of nonzero or significant coefficients corresponding to the atoms in a dictionary. To construct a dictionary \(D\), to sparsely represent each patch \(b^k_s\) extracted from the component \(I_s\) of the image \(I\), we may use a set of available training exemplars \(y^k, k = 1, 2, ..., P\), to learn a dictionary \(D_s\) sparsifying \(y^k\) by solving the following optimization problem [26,34]:

\[ \min_{\phi_k} \sum_{k=1}^{P} \left( \frac{1}{2} \left\| y^k - D_s \phi_k \right\|_2^2 + \beta \right), \]

where \(\phi_k\) denotes the sparse coefficient vector of \(y^k\) with respect to \(D_s\) and \(\beta\) is a regularization parameter. Eq. (2) can be efficiently solved by performing a dictionary learning algorithm, such as the online dictionary learning [34] or K-SVD [26] algorithms, where the sparse coding step is usually achieved via OMP [27]. Finally, the image decomposition is achieved by iteratively performing the MCA algorithm to solve \(I_s\) (while fixing \(D_s\)) described in Section 2.1 and the dictionary learning algorithm to learn \(D_s\) (while fixing \(I_s\)) until convergence.

3. Proposed self-learning-based deblocking framework via sparse representation

Fig. 2 shows the proposed self-learning-based deblocking framework via sparse representation, where deblocking is formulated as an MCA-based image decomposition problem via sparse representation.
representation. In our method, an input image with blocking artifacts is first roughly decomposed into the low-frequency (LF) part and the high-frequency (HF) part using the BM3D algorithm [28], where the most basic information will be retained in the LF part while the blocking artifacts and the other edge/texture details will be included in the HF part of the image as illustrated in Fig. 2(b) and (c), respectively. Then, we perform the proposed MCA-based image decomposition to further decompose the HF part into the non-blocking component [see Fig. 2(d)] and the blocking component [see Fig. 2(e)]. In the image decomposition step, a dictionary learned from the training exemplars extracted from the HF part of the image itself can be divided into two sub-dictionaries [see Fig. 2(f) and (g)] by performing our modified HOG (histograms of oriented gradients [39]) feature-based dictionary atom clustering. Then, we perform sparse coding [27] based on the two sub-dictionaries to achieve MCA-based image decomposition, where the non-blocking component in the HF part can be obtained, followed by integrating with the LF part of the image to obtain the de-blocked version of this image as illustrated in Fig. 2(h). The detailed method will be elaborated below.

3.1. Preprocessing and problem formulation

For an input image \( I \) with blocking artifacts, in the preprocessing step, we apply the BM3D image denoising algorithm [28] to decompose \( I \) into the LF part \( (I_{\text{LF}}) \) and HF part \( (I_{\text{HF}}) \), i.e., \( I = I_{\text{LF}} + I_{\text{HF}} \). BM3D is an efficient image denoising strategy based on an enhanced sparse representation in the transform-domain, where sparsity is enhanced by grouping similar 2D image fragments into 3D data arrays called “groups.” Then, collaborative filtering is developed to deal with these 3D groups. In our deblinking framework, we first treat the blocking artifacts in \( I \) as noises and apply BM3D to remove all of them to obtain its LF part \( I_{\text{LF}} \).

Then, our deblinking method learns a dictionary \( D_{\text{HF}} \) based on the training exemplar patches extracted from \( I_{\text{HF}} \) to further decompose \( I_{\text{HF}} \), where \( D_{\text{HF}} \) can be further divided into two sub-dictionaries, \( D_{\text{HF},C} \) and \( D_{\text{HF},B} \), for sparsely representing the non-blocking and blocking components of \( I_{\text{HF}} \), respectively. As a result, we formulate the problem of deblinking for image \( I \) as an MCA-based image decomposition problem via sparse representation:

\[
\min_{\theta} \left\| b^L_{\text{HF}} - D_{\text{HF}} \theta^L_{\text{HF}} \right\|_2^2 \quad \text{s.t.} \quad \left\| \theta^L_{\text{HF}} \right\|_0 \leq L,
\]

where \( b^L_{\text{HF}} \) represents the \( k \)th patch extracted from \( I_{\text{HF}} \), \( \theta^L_{\text{HF}} \) is the sparse coefficient vector of \( b^L_{\text{HF}} \) with respect to \( D_{\text{HF}} \), and \( L \) denotes the sparsity or maximum number of nonzero coefficients of \( \theta^L_{\text{HF}} \). Before solving (3), we should first learn \( D_{\text{HF}} \) and further partition it into \( D_{\text{HF},C} \) and \( D_{\text{HF},B} \), as described below.

3.2. Dictionary learning and partition

In this step, we extract a set of overlapping patches from \( I_{\text{HF}} \) as the training exemplars \( y^L \) for learning dictionary \( D_{\text{HF}} \) by the dictionary learning technique described in Section 2.2.1 using the online dictionary learning algorithm [34] to obtain \( D_{\text{HF}} \). More specifically, we extract all square patches of size \( n \) overlappedly from \( I_{\text{HF}} \) of size \( N = N_1 \times N_2 \), and hence, we totally extract \( P = (N_1 - \sqrt{n} + 1) \times (N_2 - \sqrt{n} + 1) \) patches as training samples. In this training patch extraction process, we do not consider where the blocking artifacts may occur in \( I \) based on the fact that the block sizes used in the compression process may be variable and the proposed method is a pure post-processing approach. We analyze the atoms constituting the blocking and non-blocking components, respectively, of \( I \) via the dictionary learning and partition process, detailed in the following paragraphs. However, if the positions of the blocking artifacts can be accurately known in advance, it may be beneficial to the process of removing blocking artifacts, as discussed in Section 3.3.

To analyze the atoms constituting the learned dictionary \( D_{\text{HF}} \), we find that these atoms can be roughly divided into two clusters (sub-dictionaries) for representing the non-blocking and blocking components of \( I_{\text{HF}} \), respectively. Intuitively, the most significant feature for a “blocking atom” can be extracted via “image gradient.” In our method, we utilize and modify the HOG descriptor [39] to describe each atom in \( D_{\text{HF}} \). The HOG descriptor [39] is
briefly introduced as follows. The basic idea of HOG is that local object appearance and shape can be usually well characterized by the distribution of local intensity gradients or edge directions, without precisely knowing the corresponding gradient or edge positions [39]. To extract the HOG feature from an image, the image can be divided into several small spatial regions or cells. For each cell, a local 1-D histogram of gradient directions or edge orientations over the pixels of the cell can be accumulated. The combined histogram entries of all cells from the HOG representation of the image.

Based on the fact that blocking artifacts are characterized by visually noticeable changes in pixel values along block boundaries, it is only required to consider the horizontal and vertical gradients in each dictionary atom. Hence, we modify the original HOG [39] to calculate only the histogram over the intervals of angles around 0°, 180°, 90°, and 270°. In our implementation, we calculate the horizontal HOG descriptor over the two intervals, [355°, 5°] and [175°, 185°], and the vertical HOG descriptor over the two intervals, [85°, 95°] and [265°, 275°], respectively, for each atom in \( D_{HF} \).

After extracting the HOG feature for each atom in \( D_{HF} \), we first apply the K-means algorithm to classify all of the atoms in \( D_{HF} \) into two clusters, \( D_1 \) and \( D_2 \), based on their horizontal HOG feature descriptors. Then, we calculate the variance of gradient directions for each atom \( d_{ij} \) in cluster \( D_i \), as \( VG_{ij} \), \( i = 1, 2 \). Then, we calculate the mean of \( VG_{ij} \) for each cluster \( D_i \) as \( MVG_i \). Given the fact that the edge directions of the samples of an atom with horizontal (or vertical) blocking artifacts should be consistent, i.e., the variance of gradient directions for a “horizontal (or vertical) blocking” atom should be smaller than those of the other atoms with no remarkably dominating edge direction, we identify the cluster with the smaller \( MVG_i \) as the “horizontal (or vertical) blocking” atom and obtain the “horizontal blocking” sub-dictionary \( D_{HF,B1} \) (or \( D_{HF,BV} \) to from the blocking sub-dictionary \( D_{HF,B} = [D_{HF,B1}|D_{HF,BV}] \) [see Fig. 2(g)]. That is, we first classify all of the atoms in \( D_{HF} \) into two clusters via the K-means algorithm and the horizontal HOG feature descriptor for each atom to obtain the “horizontal blocking” sub-dictionary \( D_{HF,B1} \) and the other sub-dictionary. Then, the sub-dictionary can be further similarly classified again to obtain the “vertical blocking” sub-dictionary \( D_{HF,BV} \) and non-blocking sub-dictionary \( D_{HF,C} \) [see Fig. 2(f)]. Fig. 3 exemplifies the highlighted versions of the learned non-blocking and blocking dictionaries in Fig. 2.

To evaluate the effectiveness of the proposed dictionary partition process, we calculate the average precision rate, average recall rate, and average F-measure for the classification

---

**Fig. 2.** An example demonstrating the flowchart of the proposed self-learning-based image deblocking framework via sparse representation: (a) original image with blocking artifacts, PSNR = 35.93 dB; (b) low-frequency part of (a), PSNR = 35.39 dB; (c) high-frequency part of (a); (d) non-blocking component of (c); (e) blocking component of (c); (f) non-blocking dictionary; (g) blocking dictionary; and (h) blocking artifact-removed version of (a), PSNR = 36.10 dB.

**Fig. 3.** Highlighted versions of the two learned dictionaries for the example shown in Fig. 2: (a) the non-blocking dictionary, and (b) the blocking dictionary.
of dictionary atoms conducted on several test images, as shown in Table 2. Because no ground truth is available for dictionary partition, we manually label each dictionary atom to be “blocking” or “non-blocking” for the test images. It can be observed from Table 2 and the experimental results presented in Section 4 that the performance of dictionary atom classification based on HOG feature extraction and K-means algorithm is efficient for deblocking purpose.

### Table 2

<table>
<thead>
<tr>
<th></th>
<th>Precision rate (%)</th>
<th>Recall rate (%)</th>
<th>F-measure</th>
</tr>
</thead>
<tbody>
<tr>
<td>Blocking atom</td>
<td>97.87</td>
<td>93.31</td>
<td>0.9554</td>
</tr>
<tr>
<td>Non-blocking atom</td>
<td>98.37</td>
<td>99.30</td>
<td>0.9883</td>
</tr>
</tbody>
</table>

3.3. Removal of blocking artifacts

Based on the two self-learned dictionaries $D_{HF,G}$ and $D_{HF,B}$, we perform sparse coding by applying the OMP algorithm [27] for each patch $b_{HF}$ extracted from $I_{HF}$ via minimization of (3), where $D_{HF} = [D_{HF,G} | D_{HF,B}]$, to find its sparse coefficient vector $\theta_{HF}$. Then, each reconstructed patch $b_{HF}$ can be used to recover either non-blocking component $I_{G}$ or blocking component $I_{B}$ of $I_{HF}$ based on the sparse coefficient vector $\theta_{HF}$ as follows. We set the coefficients corresponding to the atoms in $D_{HF,G}$ in $\theta_{HF}$ to zeros to obtain $\theta_{HF,G}$, while the coefficients corresponding to the atoms in $D_{HF,B}$ in $\theta_{HF}$ to zeros to obtain $\theta_{HF,B}$. Therefore, each patch $b_{HF}$ can be re-expressed as either $b_{HF,G} = D_{HF,G} \times \theta_{HF,G}$ or $b_{HF,B} = D_{HF,B} \times \theta_{HF,B}$, which can be used to recover $I_{G}$ or $I_{B}$, respectively. Each pixel in $I_{G}$ or $I_{B}$ may be recovered by several overlappedly reconstructed patches and we calculate its final value by averaging all of

![Fig. 4](image1.png)

**Fig. 4.** Highlighted versions of the two reconstructed components for the example shown in Fig. 2: (a) the non-blocking component, and (b) the blocking component.

![Fig. 5](image2.png)

**Fig. 5.** Selected regions of the original and reconstructed images for the example shown in Fig. 2: (a) the original image with blocking artifacts; (b) the low-frequency part obtained by BM3D; and (c) the deblocked version obtained by the proposed method.
the pixel values in the same overlapping position [see Fig. 2(d) and (e)]. Fig. 4 illustrates the highlighted versions of the decomposed $I_{HF}^e$ in Fig. 2. Finally, the deblocked version of the image $I$ can be obtained via $I_{HF}^{\text{blocking removal}} = I_{HF} + I_{HF}^e$. [see Fig. 2(h)]. As shown in Fig. 5, although BM3D can remove blocking artifacts, it usually makes the image oversmoothed, whereas the proposed scheme provides much more details in the deblocked image as also illustrated in Fig. 4(a). The proposed self-learning-based deblocking method is summarized in the flowchart shown in Fig. 6 and Table 3.

Note, the method described above can be used for “blind” deblocking, that is, we have no knowledge about the locations where blocking artifacts occur. If the locations of blocking artifacts are known in advance, the reconstruction process for each block with blocking artifact can be modified as follows. We consider the $k$th block $B_k$ with blocking artifact in $I$ and its corresponding deblocked version $\bar{B}_k$ in $I_{HF}^{\text{blocking removal}}$, where the block size may range from $4 \times 4$ to $64 \times 64$ based on the current compression paradigms (not necessarily the same as the patch size $n$ used in our method). The block $\bar{B}_k$ can be further reconstructed by the linear combination of $B_k$ and $\bar{B}_k$ as:

$$\bar{B}_k(i,j) = w(i,j)B_k(i,j) + (1 - w(i,j))\bar{B}_k(i,j),$$

(4)

where $\bar{B}_k$ denotes the final reconstructed block for $B_k$, $w(i,j)$ denotes the position for the $(i,j)$th pixel, and $w$ and $(1 - w)$ denote the weights for $B_k$ and $\bar{B}_k$, respectively, where $0 \leq w(i,j) \leq 1$. That is, if the $(i,j)$th pixel is at the block boundary, we set $w(i,j)$ to 0 to only use $B_k(i,j)$ obtained by our MCA-based method to recover this pixel without including any percentage of its original pixel value (with blocking artifact). For the pixels located from the block boundary to inner region, we increasingly set $w(i,j)$ from 0 to 1, while $[1 - w(i,j)]$ is decreasingly from 1 to 0. On the other hand, even if the positions of blocking artifacts can be well derived in advance, it is not always better to only replace the affected pixels based on the following reason. Blocking artifacts usually occur in low bitrate transmission quality in the case that the locations of blocking artifacts are available.

3.4. Extension to blocking artifact removal for video

The proposed image deblocking method can then be easily extended to video deblocking by performing image decomposition for each frame. We first perform BM3D’s video denoising mode [28] to a video sequence by which the temporal consistency of the filtered video can be automatically maintained. The learned dictionaries for a frame can be re-used by latter frames during a certain period (e.g., a group of pictures or GOP) in the video to reduce the computational complexity induced by dictionary learning.

More specifically, we apply the proposed image deblocking method to each intra-frame (I frame) in a video and then the learned dictionaries for this frame can be re-used by the subsequent frames for deblocking in the same GOP. Using the same pair of dictionaries for a GOP can either save the computational complexity of dictionary learning or maintain the temporal consistency by using the same set of dictionary atoms for patch reconstructions in the consecutive frames of the GOP. In addition, similar to our image deblocking approach, our video deblocking method is also a pure post-processing technique without relying on any prior knowledge about video coding. The proposed
The deblurring results for the H.264/AVC-decoded Foreman, Football, and Stefan video sequences obtained by the H.264/AVC without (denoted by “No filter”) and with (denoted by “Loop filter”) performing the in-loop filter, DNLK, SAWS, and proposed methods are shown in Figs. 8–10, respectively. It can be found from Figs. 8–10 that the proposed method outperforms the “No filter,” “Loop filter [15],” DNLK [16], and SAWS [17] methods. In fact, it is not easy to outperform H.264/AVC in-loop deblocking filter owing to the fact that the filter is conducted on both the encoding and decoding paths, and so the in-loop effects of the filter are taken into account in reference frames used for prediction, resulting in good deblurring results. However, as a pure post-processing approach, the proposed deblurring method can still outperform the H.264/AVC in-loop filter, mainly benefiting from the proposed self-learning-based image decomposition strategy. On the other hand, the DNLK method [16] employs the used quantization tables for compression to decide some filtering parameters, while the proposed method as a pure post-processing strategy does not rely on any prior knowledge.

On the other hand, it has been shown that the performance improvement gaps over JPEG of the existing sparse representation-based deblurring method for JPEG images reported in [25] slightly outperform the proposed method for JPEG images. However, in [25], only the results obtained from the range of smaller compression factors (lower PSNR values) were reported, while our method can adapt to larger range of JPEG compression factors. In addition, it is required to collect extra training data and perform some pre-statistics about JPEG compression in [25], while our method can adapt to larger range of JPEG compression factors (lower PSNR values) were reported, while our method can adapt to larger range of JPEG compression factors. In addition, it is required to collect extra training data and perform some pre-statistics about JPEG compression in [25], while our method can adapt to larger range of JPEG compression factors.
in an image. The other steps (e.g., dictionary partition) are relatively simple and their complexities can be ignored. Moreover, the dictionary learning process essentially includes a sequence of sparse coding operations. Hence, the core operation of the proposed method is the sparse coding with complexity $O(nrL + rL^2)$ based on the analysis in [34], where $n$ is the atom size (patch size), $r$ is the dictionary size (number of atoms), and $L$ is the number of nonzero coefficients. Recall from the parameter settings in our experiments, $n$, $r$, and $L$ are set to $16 \times 16$, 1024, and 10, respectively.

Similar to several existing sparse coding-based algorithms designed for different applications (e.g., image denoising [26,40], image decomposition [29–31,35–38], or super-resolution [41–42]), the computational complexity of the proposed
deblocking algorithm is currently indeed expensive. Hence, the proposed algorithm is mainly designed for post-processing applications with possibly higher computational power supplied. However, similar to [41], the complexity of the patch reconstruction step based on sparse coding can be further reduced via the following three strategies. (1) Selective patch processing: only patches in the salient regions in an image are recovered using the proposed sparse recovery approach. For the patches in the less notable regions, only simple deblocking method will be applied. (2) Fast sparse inference: the computation of sparse coding can be fast and approximately inferred by a pre-trained feedforward model [43], which can be also used to speed up the sparse coding step in the dictionary learning process. (3) Number of overlapping pixels for patch recovery: our recovery strategy intends to reconstruct each overlapping image patch independently, and then fuse the multiple pixel predictions in overlapping regions by simply averaging. However, the reconstruction performance and the number of overlapping pixels (proportional to the computational complexity of image recovery) should be well compromised. In addition, the dictionary learning step in our method may be also speeded up via fast dictionary learning (e.g., [44]), while efficient hardware implementation for both sparse coding and dictionary learning would be also a choice (e.g., [45]).

Moreover, the other goal of designing such algorithm is for our future work to develop a unified self-learning-based sparse

---

Fig. 8. Debloking results for the Foreman sequence: (a) the rate-PSNR performances; (b) the H.264/AVC-decoded frame with No filter; (c) the H.264/AVC-decoded frame with Loop filter; the debloked frames obtained by the: (d) DNLK; (e) SAWS; and (f) proposed methods.
representation framework for joint super-resolution and deblocking of a highly compressed image. In the prototype [46] of this novel framework, we embedded the deblocking functionality into a sparse representation-based image super-resolution framework and showed that no additional computational complexity will be induced, compared with the original pure sparse representation-based super-resolution framework.

5. Conclusions

In this paper, we have proposed a self-learning-based post-processing framework for image/video deblocking by formulating deblocking as an MCA-based image decomposition problem solved by performing sparse coding and dictionary learning algorithms. The dictionary learning of the proposed method is fully automatic and self-contained; no extra training samples are required in the dictionary learning stage. Our experimental results show that the proposed method achieves better performance than the existing deblocking algorithms, including the H.264/AVC in-loop deblocking filter. As a pure post-processing approach without any prior knowledge (e.g., data source, coding bitrates, and compression algorithms) needed, the proposed deblocking method can be applied to any block-based image/video compression paradigms. Moreover, the proposed image decomposition framework can be easily extended to remove other undesired component with regular or near-regular patterns by incorporating suitable feature extraction technique.

Fig. 9. Deblocking results for the Football sequence: (a) the rate-PSNR performances; (b) the H.264/AVC-decoded frame with No filter; (c) the H.264/AVC-decoded frame with Loop filter; the deblocked frames obtained by the: (d) DNLK; (e) SAWS; and (f) proposed methods.
Acknowledgments

This work was supported in part by the National Science Council, Taiwan, under Grants NSC102-2221-E-110-032-MY3, NSC101-2221-E-110-093-MY2, and NSC100-2218-E-224-017-MY3. Our thanks to Wen-Hung Xu for executing the program on the test data in this work; his timely assistance is greatly appreciated.

References


Fig. 10. Deblocking results for the Stefan sequence: (a) the rate-PSNR performances; (b) the H.264/AVC-decoded frame with No filter; (c) the H.264/AVC-decoded frame with Loop filter; the deblocked frames obtained by the: (d) DNLK; (e) SAWS; and (f) proposed methods.