Selective Deblocking Method Using a Transform Table of Different Dimension DCT

Taehwan Lim Jiman Ryu Jechang Jeong
Department of electronics and computer engineering, Hanyang University
17 Haengdang-dong, Seongdong-gu
133-791, Seoul, Korea
taehwan.lim@ece.hanyang.ac.kr

ABSTRACT
In this paper, we propose a selective deblocking algorithm that reduces block discontinuities in DCT domain. Our algorithm applies a deblocking procedure to each line of adjacent 3 blocks, so the block is divided into several line vectors. There are three Low Pass filters that are applied differently to 1×24 DCT values according to each condition of adjacent 3 vectors for conserving image details, and we use a transform table between different dimension DCTs (1×8 and 1×24 DCT) for reducing a computational cost. The experimental results show that the proposed algorithm makes good results on an improvement of subjective image quality and a computational efficiency.

Keywords
Blocking artifacts, post-processing, DCT domain, transform table.

1. INTRODUCTION
Because the discrete cosine transform (DCT) provides a good compromise between an information packing ability and computational complexity, the discrete cosine transform is a kernel in many industry standards of image and video compression, such as JPEG, MPEG, and H.26x [Gon02]. Images are divided into blocks of a size N×N in most standards. The blocks are transformed from the spatial domain to the frequency domain by DCT, and the DCT coefficients are quantized by a quantization table.
Because the original values of the DCT coefficients are necessarily changed by inverse quantization, the quantization is a lossy step. Each block is separately encoded and transmitted. Since blocks are treated as single objects, compression coding ignores a correlation of neighboring blocks.

It brings about a degradation of the block based transform coding, and the blocking artifact appears at block boundaries. It is visible when the decoded image is reconstructed. For example, a smooth change of a luminance across a border can result in a step in the decoded image if neighboring samples fall into different quantization intervals [Tri02].
To overcome a block based transform coding's weakness, "Blocking artifact", numerous methods have been researched. These can be classified into two groups: The in-loop processing uses different encoding schemes, such as the interleaved block transform, the lapped transform, and the combined transform. The other method is a post-processing which uses a reconstructed image. Because the post-processing does not require a change of the existing standards, it attains more practical solutions [Gao02]. The post-processing should preserve object edges and keep the sharpness of image and be simple for real-time applications, which are important features of the post-processing. In [Luo03], a classification procedure is used for distinguishing between a smooth region and a non-smooth region. The non-smooth region is not regulated because humans are more sensitive to a low frequency than to high frequency information. In deblocking processing, they change only 5 coefficients, so they prevent over-blurring and save computation efforts. However, there are still some blocking artifacts which are unprocessed. Signal decomposition is used in

[Wan06] to cope with a over-blurring. However, it loses some signal information because of discarding of high frequency values. Furthermore, it ignores new blocking artifacts that are produced in a next vector processing.

In this paper, we propose a DCT based algorithm to reduce both computational cost and over-blurring using transform table of different dimension DCT. We decompose images to two vectors, a low-frequency (LF) vector and a high-frequency (HF) vector, and adjust only a LF vector to preserve image details because the HF coefficients have lots of image detail. Our proposed method also uses selective filters in a selected region instead of discarding of high frequency coefficients. With such a process, the edges are preserved while the blocking artifacts are removed. We consider three 1×8 DCT vector as a one 1×24 DCT vector, so we use a transform table of different dimension DCT for high computational efficiency. This helps us remove meaningless calculations that exist in a transformation between 1×8 DCT vector and 1×24 DCT vector. Finally, we add some refined values determined by a difference of pixel to remove new blocking artifacts that produced in next vector process.

2. PROPOSED ALGORITHM

2. 1. Overview of the proposed algorithm

We explicate a proposed algorithm only in the horizontal direction because the blocking artifacts have similar properties in the horizontal and vertical direction. Below, the uppercase X[n] and the lowercase x[n] represent a DCT coefficient and its pixel value, respectively.

The deblocking process has a trade-off between edge-preserving and a over-blurring. To overcome it, we decompose images into two vectors. Most high-frequency DCT coefficients contain image details; therefore, we decompose an image into two domains, LF and HF, using either an 8×8 DCT domain image data or a spatial domain image data as Fig. 1, and our proposed algorithm changes only the LF data for preserving image details.

The LF and HF data are vector sets that consist of 1×8 DCT vectors because our procedure deals with the image as a line instead of a block. The line level procedure is more complex than the block level procedure, but it helps conserve image details. The LF data contains only the adjusted first 2 coefficients in each 1×8 DCT vector; otherwise, the HF data consists of adjusted 8 coefficients in each vector. This will be presented later in Section 2.2.

Every deblock procedure regards adjacent 3 vectors (1×8) as one long vector (1×24) like Fig. 2, and the proposed method deblocks these connecting vectors whose centers are located at the even vectors. This means that the C vector will become the first vector when the deblock procedure starting at the vector A is ended.

After deblocking, we overwrite the LF vectors that are used in processing. The new A and B vectors are overwritten by 8 new coefficients whereas the new C vector has only the first 3 coefficients computed by deblock process. This C vector will become an A vector when we deblock next H; therefore, we get 7 coefficients from 3 vectors used in next deblock process.

Fig. 3 is a flowchart of the proposed algorithm. If the differences among the vectors are larger than a threshold *T1* after obtaining 7 coefficients, it means that they may contain two real edges at their boundaries. In this condition, we skip these vectors. Otherwise, we examine whether these vectors have one or no real edge. If they have a real edge, we deblock using *Filter3*. The other cases operate *Filter1* or *Filter2* according to the AC coefficients. Following the filtering, we refine the first vector and overwrite the LF vectors. After all deblock process of the LF data, we summate the LF and the HF data and deblock vertical direction the same way.

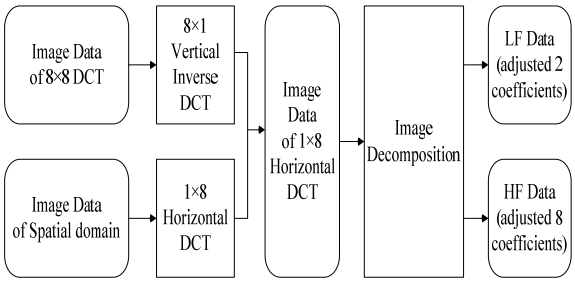


Figure 1. Image Decomposition Block Diagram.

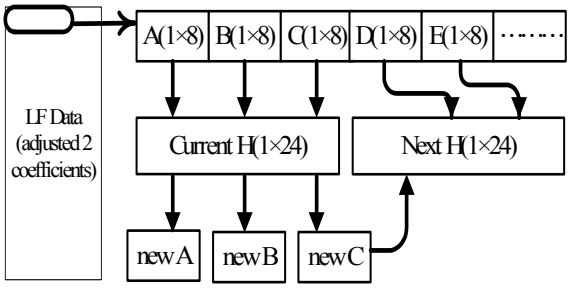


Figure 2. Horizontal vectors and their combination H

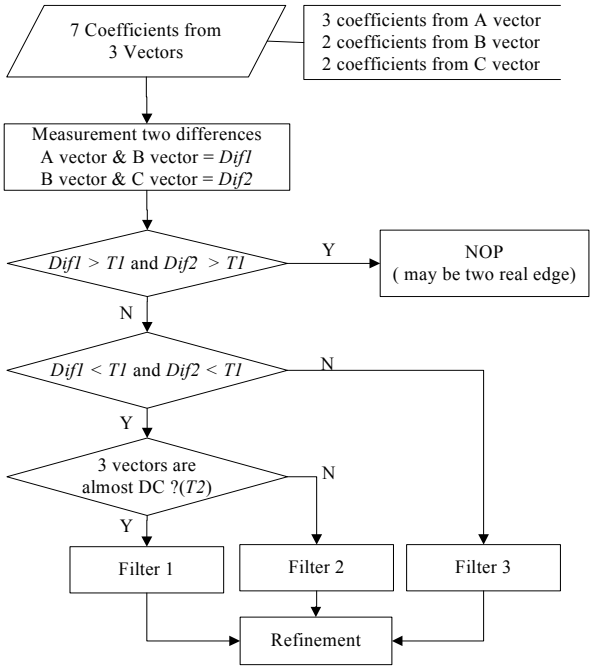


FIGURE 3. FLOWCHART OF THE PROPOSED DEBLOCKING ALGORITHM

2. 2. Image decomposition

Our method uses only the LF vectors and reduces a discontinuity of LF vector; hence, there must be no discontinuities in the HF vectors. In the method presented in [Wan06], they divide original 1×8 DCT vectors into first 2 coefficients (LF) and the others (HF). Then they set the first and last pixel values of the HF vector to zero by using the resultant average and difference of the HF coefficients. If there is an original 1×8 DCT vector, X[n], then we decompose as

$$\begin{aligned}
\eta &= \sum_{n=2,4,6} \frac{X[n]\cdot\cos(n\pi/16)}{2} \\
\lambda &= \sum_{n=3,5,7} \frac{X[n]\cdot\cos(n\pi/16)}{2} \\
LF_X[0] &= X[0] + \eta\cdot\sqrt{8} \\
LF_X[1] &= X[1] + 2\lambda/\cos(\pi/16) \\
LF_X[n] &= 0, \qquad 2 \le n \le 7 \\
HF_X[0] &= -\eta\cdot\sqrt{8} \\
HF_X[1] &= -2\lambda/\cos(\pi/16) \\
HF_X[n] &= X[n], \qquad 2 \le n \le 7
\end{aligned} \tag{1}$$

2. 3. Selecting filter step

There are 3 filters in our algorithm, *Filter1*, *Filter2*, and *Filter3*. To prevent over-blurring, *Filter1* is designed for a smooth region that has almost only DC coefficients and *Filter2* is used for a non-smooth region that has high AC values. If there is a real edge at boundaries, we use *Filter 3*. Therefore, each filter is selected by the differences between vectors (*Dif1* and *Dif2*) and AC coefficients of the LF vectors. There are 3 horizontal neighboring vectors in Fig. 2. We take 7 values from 3 LF DCT vectors. The A vector gives us three values, *A[0]*, *A[1]*, and *A[2]*, and the other vectors transfer two coefficients each, *B[0]*, *B[1]*, *C[0]*, and *C[1]*. From these values, we can find boundary pixel values of each vector and differences (*Dif1*, *Dif2*) using (2).

$$\begin{aligned}
a[7] &= \frac{A[0]}{\sqrt{8}} + \frac{A[1]\cdot\cos(15\pi/16)}{2} + \frac{A[2]\cdot\cos(30\pi/16)}{2} \\
b[0] &= \frac{B[0]}{\sqrt{8}} + \frac{B[1]\cdot\cos(\pi/16)}{2} \\
b[7] &= \frac{B[0]}{\sqrt{8}} + \frac{B[1]\cdot\cos(15\pi/16)}{2} \\
c[0] &= \frac{C[0]}{\sqrt{8}} + \frac{C[1]\cdot\cos(\pi/16)}{2} \\
Dif1 &= |a[7] - b[0]| \\
Dif2 &= |b[7] - c[0]|
\end{aligned} \tag{2}$$

Two thresholds, *T1* and *T2*, are needed for a filter selection, and if they are adapted to image, we get enhanced results. We measure optimal thresholds according to an image quality to find threshold curves, and we have thresholds as (3) by fitting the data with a line formula.

$$T1 = \alpha\cdot(qt(0,0)/8) \tag{3}$$

The *qt(0,0)* is the quantization table's first value which is used in DC quantization. If the DC coefficient of vector A has a one interval difference from other vector (they have no AC coefficients) then the pixel of A has a difference as *qt(0,0)/8*. In our simulation, α is 3.5 and the other threshold (*T2*) is 25, which was founded to give best results. We select between *Filter1* and *Filter2* according to the *T2*, and the filter selection is shown as follows.

Step 1) Calculate *Dif1* and *Dif2*
Step 2) Comparing *T1* with *Dif1* and *Dif2*
if *Dif1* > *T1* and *Dif2* > *T1* then *NOP*
else if *Dif1* < *T1* and *Dif2* < *T1* then step 3)
else *Filter3*
Step 3) Comparing AC values with *T2*
if abs(*A[1]*) < *T2* and abs(*B[1]*) < *T2*
and abs(*C[1]*) < *T2* then *Filter1*
else *Filter2*

 ISBN 978-80-86943-93-0

2. 4. Transform of different dimension DCT

Our filters use a 1×24 DCT vector; therefore, we should infer a 1×24 DCT vector from three 1×8 DCT vectors as Fig. 2. There is a matrix that transforms DCT dimensions in [Jia02]; thus, we use it to make our transform matrix, and we optimize the matrix calculations for saving computational efforts. First, we make two matrices according to (4).

$$T_{24}[i,j]=\begin{cases} 1/\sqrt{24}, & i=0,\ 0\le j\le 23, \\ \frac{1}{\sqrt{12}}\cdot\cos\left(\frac{(2j+1)\cdot i\cdot\pi}{48}\right), & 1\le i\le 23,\ 0\le j\le 23 \end{cases} \quad (4)$$

$$T_8[i,j]=\begin{cases} 1/\sqrt{8}, & 0\le i\le 7,\ j=0, \\ \frac{1}{2}\cdot\cos\left(\frac{(2i+1)\cdot j\cdot\pi}{16}\right), & 0\le i\le 7,\ 1\le j\le 7 \end{cases}$$

T_{24} is a 24×24 matrix and T_8 is a 8×8 matrix; however, the *Filter1* has 6 valid coefficients, and the *Filter2* and the *Filter3* have 12 valid coefficients. The filters are presented in Section 2.5. We discard T_{24}'s lower 12 rows because we are only concerned with the first 12 coefficients; therefore, the T_{24} becomes a 12×24 matrix. We split the T_{24} into three 12×8 matrices (T_{24a}, T_{24b}, and T_{24c}) and multiply by the T_8.

$$\begin{aligned} &T_{24a}[i,j]=T_{24}[i,j], \\ &T_{24b}[i,j]=T_{24}[i,j+8], \\ &T_{24c}[i,j]=T_{24}[i,j+16], \\ &\qquad i=0,1,\cdots,11 \quad j=0,1,\cdots,7 \\ &T_A=T_{24a}\cdot T_8,\ T_B=T_{24b}\cdot T_8,\ T_C=T_{24c}\cdot T_8 \end{aligned} \quad (5)$$

These matrices (T_A ,T_B, and T_C) represent relation factors between a 1×24 DCT vector and a 1×8 DCT vector. Each i_{th} row stands for the relation factors that show how the 1×24 DCT's i_{th} coefficient has an effect on the each coefficient of 1×8 DCT vector. Likewise, each i_{th} column signifies how the i_{th} coefficient of 1×8 DCT contributes to coefficients of a 1×24 DCT vector.

$$\begin{aligned} &H[j]=\sum_{i=0}^{7}(T_A[j,i]\cdot A[i]+T_B[j,i]\cdot B[i]+T_C[j,i]\cdot C[i]), \\ &\qquad\qquad j=0,1,\cdots,11 \\ &(A/B/C)[i]=\sum_{j=0}^{11}T_{(A/B/C)}[j,i]\cdot H[j], \qquad i=0,1,\cdots,7 \end{aligned} \quad (6)$$

Finally, we have a transform method among the different dimension DCTs as above (6). At the same time, we know that there are many zero coefficients in T_A ,T_B, and T_C; furthermore, we only have 7 values from three 1×8 DCT vectors, A, B, and C. Due to these reasons and the similarity between T_A and T_C, we diminish a lot of computations. For example, we get *H[6],* and B*[0]* as (7).

$$\begin{aligned} &H[6]=T_A[6,2]\cdot A[2] \\ &B[0]=T_B[0,0]\cdot H[0]+T_B[2,0]\cdot H[2]+T_B[4,0]\cdot H[4] \\ &\qquad +T_B[8,0]\cdot H[8]+T_B[10,0]\cdot H[10] \end{aligned} \quad (7)$$

2. 5. Filter stage

Removing the high frequencies is very powerful to eliminate blocking artifacts. However, it has a loss of image information and an unwanted overshoot like a Gibbs' phenomenon. Because of these reasons, we apply a kind of average filter instead of discarding high frequency coefficients. The average or smoothing function (*f'(t)*) is mentioned below.

$$\begin{aligned} f'(t) &= \frac{1}{2T}\int_{t-T}^{t+T} f(\tau)\,d\tau \\ F'(\omega) &= F(\omega)\frac{\sin\omega T}{\omega T} \end{aligned} \quad (8)$$

We assume the blocking artifacts vectors have a discontinuity of less than *T1*, and this means that there are no huge high frequency coefficients. In this respect, we can adaptively discard some values of high frequency components. If the Fourier transform of *f'(t)* is truncated above $|\omega| = \Omega$, the truncated function $F'_{\Omega}(\omega)$ is $F'(\omega)P_{\Omega}(\omega)$, $P_{\Omega}(\omega)$ is the rectangular pulse function, and its inverse transform $f'_{\Omega}(t)$ is the weighted average of *f'(t)* [Pap62]. Hence, we can obtain a good result through adaptive decisions of a truncated frequency ω and an averaging period *T*.

$$\begin{aligned} &Filter1 \\ &F_1[i]=\begin{cases} 1 & i=0 \\ \frac{\sin(10\pi\cdot i/48)}{(10\pi\cdot i/48)} & i=1,2,\cdots,5 \end{cases} \\ &Filter2 \\ &F_2[i]=\begin{cases} 1 & i=0 \\ \frac{\sin(3\pi\cdot i/48)}{(3\pi\cdot i/48)} & i=1,2,\cdots,11 \end{cases} \\ &Filter3 \\ &F_3[i]=\begin{cases} 1 & i=0 \\ \frac{\sin(\pi\cdot i/48)}{(\pi\cdot i/48)} & i=1,2,\cdots,11 \end{cases} \end{aligned} \quad (9)$$

If there is a strong edge inside of vector, which is determined by *T2*, we should conserve this information, and the human eyes are more sensitive to find errors in smooth region. Because of these reasons, *Filter 2* has no need to be much rougher than *Filter1*. Since we consider these conditions, our filters are expressed by using (9).

The filters apply DCT coefficients of H vector as (10), and not-defined values of filters are zero. This helps to reduce the amount of calculation.

$$H'[i] = H[i] \cdot F_{1/2/3}[i], \quad i = 0,1,\cdots,5 \ or \ 0,1,\cdots,11 \tag{10}$$

Each filtering stage makes a new H vector (*H'*), and it is separated into new three vectors (*A'*, *B'*, and *C'*). This procedure uses the transform of different dimension DCTs in Section 2.4. After filtering stage, vector *A'* and vector *B'* have 8 coefficients; otherwise, vector *C'* has only first 3 coefficients (remained 5 coefficients are discarded).

2. 6. Refinement stage

Our deblocking function is applied every two vector distances, so we deal with only three vectors in each execution. This might create a new blocking artifact between the vector B and C as shown in Fig. 2. It comes from changes of the vector C when the next deblocking executes with starting at the vector C, and the pixels that are close to the vector B are especially important causes. For reducing this new artifact, we define a pixel vector, R_{efine}, as (11), and its DCT transform coefficients are $R_p[i]$.

$$R_{efine}[i] = 2^{-i}, \quad i = 0,1,\cdots,7 \tag{11}$$

We suppose that our next deblock process starts at vector A after former deblock process ended. Before the filtering, we calculate the vector A's first pixel value.

$$a[0] = \frac{A[0]}{\sqrt{8}} + \frac{A[1]\cdot\cos(\pi/16)}{2} + \frac{A[2]\cdot\cos(2\pi/16)}{2} \tag{12}$$

After the filtering, we calculate the new vector A's first pixel value *a'[0]* using (13), and we refine the vector A following (14).

$$a'[0] = \frac{A'[0]}{\sqrt{8}} + \sum_{i=1}^{7} \frac{A'[i]\cdot\cos(i\cdot\pi/16)}{2} \tag{13}$$

$$A_{refine}[i] = A'[i] + (a[0] - a'[0]) \cdot R_p[i], \quad i = 0,1,\cdots,7 \tag{14}$$

Clearly, the refined vector A has no change at the first pixel value, and other pixels are adjusted according to a difference between the *a[0]* and the *a'[0]*.

As a result of refinement process, we get refined vector A and filtered vector B and C, which are not refined, and these three deblocked vectors overwrite on the LF DCT data. After horizontal deblock process, we summate the HF DCT data and the LF DCT data.

3. EXPERIMENT RESULT

We applied our proposed algorithm on 512×512 monochrome images that are compressed by JPEG. First, we demonstrate the computational performance of the proposed algorithm. And then the effectiveness of the blocking artifacts reducing algorithm is explained and compared with other algorithms.

3. 1. Computational cost performance

First of all, we will make an inquiry into our proposed algorithm's computational cost. The image decomposition needs 6 additions (A) and 8 multiplications (M) at every 1×8 vector. It also takes 7A and 9M to obtain the *Dif1* and the *Dif2* at every period of 2 vector distances. In the case of comparing the *Dif1* and the *Dif2* with the *T1*, if we regard a comparing function as an addition function, it takes 2A when the algorithm chose between NOP and Filter choice stage, and the next choice stage also spends 2A.

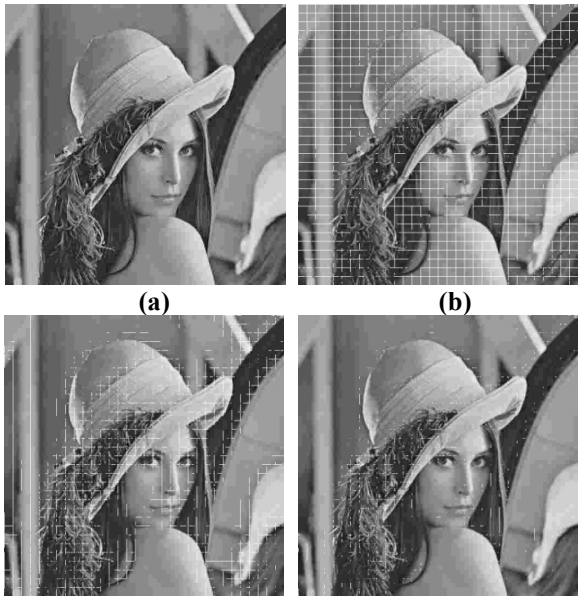
Before the filtering, it needs 2A and 3M to compute the *a[0]* for a refinement. The last choice stage between *Filter1* and *Filter2* needs 2A more. After filtering, 15A and 16M are needed to set the *a'[0]* and to refine a vector, and 8A are needed for summation of the LF_vector and the HF_vector.

Now, we conclude our proposed algorithm's computation amount with estimates of filters. Every filter must have a procedure of the different dimension DCT transform, and the coefficients that are used in each filter are different. In addition, we can reduce the addition function by the preceding addition calculations of *A[0]* and *C[0]* because of the similarity that is exited in T_A and T_C.

In this condition, our filter procedures have a low computation cost such as – *Filter 1* : 65A and 90M, *Filter 2* and *Filter3* : 142A and 178M. Finally, the computational cost for one row deblocking process,

CC_{row}, is (15). The α, β, and γ are the number of times each filter is executed. In Fig. 4, there are the Lena images of each filter classification. As you know, our method has different computation amounts according to image conditions because each filter has a different computational cost. There are totally 32564 times for application of filters in 0.247bpp Lena image. The *Filter1* is 20064; *Filter2* is 10132, and *Filter3* is 2368.

$$\begin{aligned} CC_{row} &= decompose \\ &+ select1 + (\alpha + \beta + \gamma) select2 \\ &+ (\alpha + \beta) select3 \\ &+ \alpha \cdot Filter1 + \beta \cdot Filter2 + \gamma \cdot Filter3 \\ &+ summate \\ &+ (\alpha + \beta + \gamma) refinement \end{aligned} \quad (15)$$



(a) **(b)** **(c)** **(d)**

Figure 4. Filter Classification of Lena (0.247bpp), (a) JPEG original image, (b) Filter1 region, (c) Filter2 region, (d) Filter3 region.

From this result, we can calculate a cost for a one row deblocking: 4918.2A and 5340.5M. Table I is a comparison of ours with others whose data comes from [Wan06]. It is easy to find that the proposed algorithm has much lower addition computations than [Wan06], but the computation of multiplication increase slightly.

	[Wan06]	[Pae98]	[Pae00]	proposed
One row	11136A 4224M	13608A 12096M	27720A 28224M	4918.2A 5340.5 M

Table 1. Comparison of computational cost (8×8 DCT based 512×512 image)

Most processors take more time for executing multiplications than for addition computations. However, the proposed algorithm reduces additions by 55 percent (6210.6 times) while the multiplications are increased by just 26 percent (1124.96 times). Therefore, we have an advantage of a low computing cost where the multiplication does not take about sextuple times more than the addition.

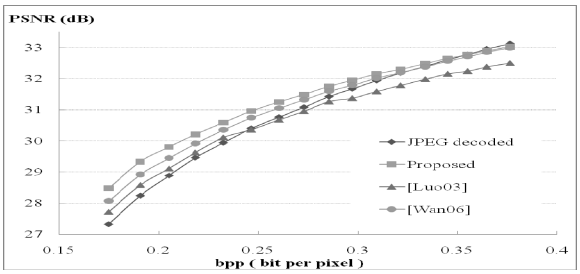
3. 2. Performance of deblocking

In this part, we examine the deblocking effectiveness of our proposed algorithm and compare it with the JPEG decoded image and others. The traditional way to evaluate the quality of the reconstructed image is the peak signal-to-noise ratio (PSNR), which uses the mean squared error as the distortion measure. However, it is obvious that the PSNR is not always a good measurement of the image quality. Because of this, we also use another measurement, MSDS. The MSDS, which was introduced in [Min95], calculates the intensity gradient of the boundary of the adjacent two blocks. It is used to evaluate the outcome of reducing blocking artifacts.

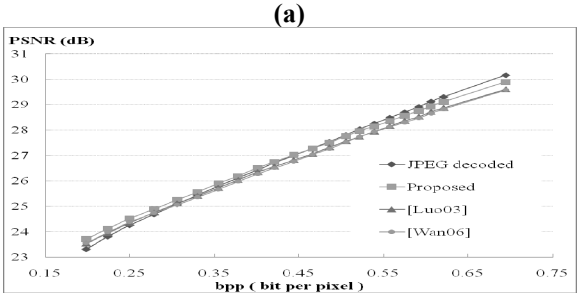
The experiment images are three JPEG coded images, Lena, Barbara, and Pepper which are 512×512 monochrome images. The simulation results of PSNR and MSDS are shown in Fig. 5 and Fig. 6, respectively. As shown in Fig. 5, The PSNR of our proposed algorithm has higher performances than other methods in [Luo03] and [Wan06] under all sample images which are coded by a different bit per pixel (bpp), and the intersection which is between the JPEG decoded image line and the other lines has a higher bpp value than the other's one.

In the MSDS results, the proposed algorithm's MSDS is more than [Wan06] but is less than JPEG decoded image and [Luo03]. However, this result is acceptable and is not a defect of our algorithm because we propose our algorithm to conserve the details of image. In high bpp image, most deblocking algorithms are used to lose image's detail information, but the proposed algorithm preserves the image's detail. As shown in Fig. 6, every MSDS result increases according to image's bpp increases, and they even follow the MSDS line of the original decoded image in the results of Barbara.

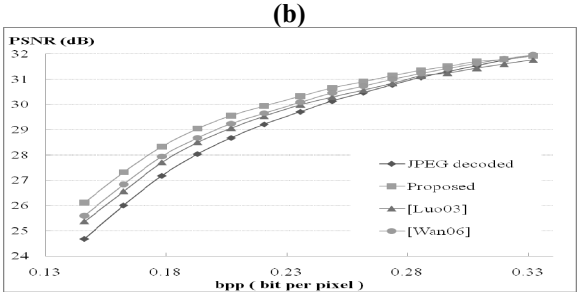
Finally, the proposed algorithm has both higher PSNR results than other algorithms and enough MSDS results to reduce the block discontinuities at the boundaries. This means that our method has smoother boundary conditions than other comparable methods and conserves more image details. We can also confirm visual effectiveness from Fig. 7-9.



(a)

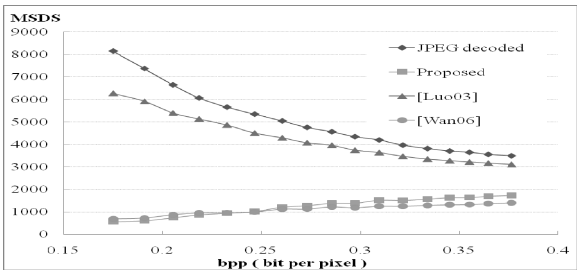


(b)

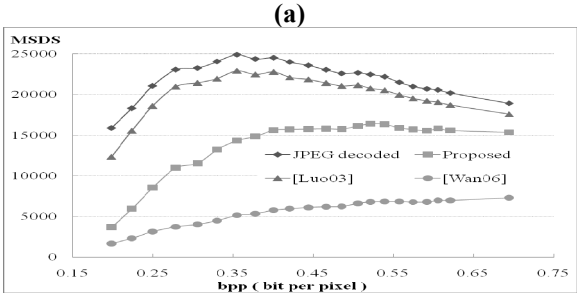


(c)

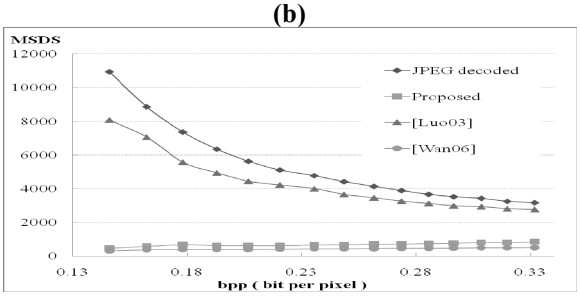
Figure 5. PSNR versus bpp : (a) Lena, (b) Barbara, (c) Pepper.



(a)



(b)

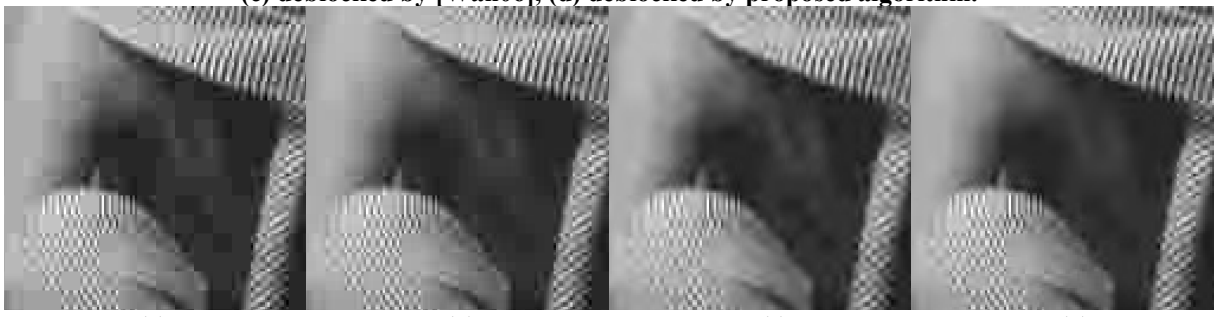


(c)

Figure 6. MSDS versus bpp : (a) Lena, (b) Barbara, (c) Pepper.

(a) (b) (c) (d)

Figure 7. Enlarged Lena : (a) JPEG decoded (0.233bpp), (b) deblocked by [Luo03], (c) deblocked by [Wan06], (d) deblocked by proposed algorithm.



(a) (b) (c) (d)

Figure 8. Enlarged Barbar : (a) JPEG decoded (0.306bpp), (b) deblocked by [Luo03], (c) deblocked by [Wan06], (d) deblocked by proposed algorithm.

(a) (b) (c) (d)

Figure 9. Enlarged Pepper : (a) JPEG decoded (0.236bpp), (b) deblocked by [Luo03], (c) deblocked by [Wan06], (d) deblocked by proposed algorithm.

4. CONCLUSION

When images are coded by the block based DCT transform, the reconstructed images mainly include blocking artifacts caused by coarse quantization. There have been many suggestions to overcome this problem, but it is a very hard work to reduce both the over-blurring and computational cost. This paper proposes a novel coding artifact reduction method based on both a specific region classification and image decomposition. The decomposition helps to preserve the details that mostly exist in HF. The classification has functions to avoid over-blurring and to reduce unnecessary computational cost. The refinement stage makes a deblocking process to eliminate the new discontinuity which occurs in next deblocking process. The experimental results show that the proposed algorithm decreases the computational cost while still achieving both a reducing blocking artifacts and preserving image details. Finally, all processing are carried out in DCT domain, so they are particularly suitable for image processing in the compressed domain.

5. ACKNOWLEDGMENTS

This research was supported by Seoul Future Contents Convergence (SFCC) Cluster established by Seoul R&BD Program.

6. REFERENCES

[Gon02] R. C. Gonzalez and R. E. Woods*, Digital image Processing*. NJ: Prentice Hall, 2002.

[Tri02] G. A. Triantafyllidis, D. Tzovaras, and M. G. Strintzis, "Blocking Artifact Detection and Reduction in Compressed Data," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 12, no. 12, pp.877-890, Oct. 2002.

[Gao02] W. F. Gao and Y. M. Kim, "A de-blocking algorithm and a blockiness metric for highly compressed images," *IEEE Trans. Circuits Syst. Video Technol.,* vol. 12, no. 12, pp. 1150-1159, Dec. 2002.

[Luo03] Y. Luo, and R. K. Ward, "Removing the Blocking Artifacts of Block-Based DCT Compressed Images," *IEEE Trans. Image Process.*, vol. 12, no. 7, pp. 838-842, Jul. 2003.

[Wan06] C. Wang, P. Xue, W. Lin, W. Zhang, and S. Yu, "Fast Edge-Preserved Postprocessing for Compressed Image," *IEEE Trans. Circuits Syst. Video Technol.,* vol. 16, no. 9, pp. 1142-1147, Sep. 2006.

[Jia02] J. M. Jiang and G. C. Feng, "The spatial relationship of DCT coefficients between a block and its sub-blocks," *IEEE Trans. Signal process.,* vol. 50, no. 5, pp. 1160-1169, May. 2002.

[Pap62] A. Papoulis. *The Fourier integral and its applications*. New York: McGraw-Hill, 1962.

[Pae98] H. Paek, R. C. Kim, and S. U. Lee, "On the POCS-based postprocessing technique to reduce the blocking artifacts in transform coded images," *IEEE Trans. Circuits Syst. Video Technol.,* vol. 8, no. 3, pp. 358-367, Jun. 1998.

[Pae00] H. Paek, R. C. Kim, and S. U. Lee, "A DCT-based spatially adaptive post-processing technique to reduce the blocking artifacts in transform coded images, " *IEEE Trans. Circuits Syst. Video Technol.,* vol. 10, no. 1, pp. 36-41, Feb. 2000.

[Min95] S. Minami and A. Zakhor, "An optimization approach for removing blocking effects in transform coding," *IEEE Trans. Circuits Syst. Video Technol*., vol. 5, pp. 74-82, Apr. 1995.