Composite Model-Based DC Dithering for Suppressing Contour Artifacts in Decompressed Video

Xin Jin, *Member, IEEE*, Satoshi Goto, *Life Fellow, IEEE*, and King Ngi Ngan, *Fellow, IEEE*

***Abstract*—Because of the outstanding contribution in improving compression efficiency, block-based quantization has been widely accepted in state-of-the-art image/video coding standards. However, false contour artifacts are introduced, which result in reducing the fidelity of the decoded image/video especially in terms of subjective quality. In this paper, a block-based decontouring method is proposed to reduce the false contour artifacts in the decoded image/video by automatically dithering its direct current (DC) value according to a composite model established between gradient smoothness and block-edge smoothness. Feature points on the model with the corresponding criteria in suppressing contour artifacts are compared to show a good consistency between the model and the actual processing effects. Discrete cosine transform (DCT)-based block level contour artifacts detection mechanism ensures the blocks within the texture region are not affected by the DC dithering. Both the implementation method and the algorithm complexity are analyzed to present the feasibility in integrating the proposed method into an existing video decoder on an embedded platform or system-on-chip (SoC). Experimental results demonstrate the effectiveness of the proposed method both in terms of subjective quality and processing complexity in comparison with the previous methods.**

***Index Terms*—Compressed video, contour artifact removal, DC dithering, decontouring.**

I. Introduction

QUANTIZATION has been widely adopted in the advanced image coding standards, like JPEG [1], and video coding standards, like MPEG-2 [2], H.264/AVC [3], AVS [4] etc. It maps the original signal, e.g., JPEG, or the transformed residual signal, e.g., H.264/AVC, with a wider range of values to a quantized signal with a reduced range of values to achieve a possible bit reduction in representing the signal. Quantization improves the compression efficiency obviously, but, meanwhile, artifacts are introduced into the compressed image because of quantization error, which especially degrades the subjective quality.

Blocking artifacts and false contour artifacts are two kinds of artifacts introduce by coarse quantization. Blocking artifacts are physically because of the brightness discontinuity across the block boundaries [5], which appear all over the compressed image; false contour artifacts are mainly caused by the loss of low amplitude detail during quantization [6], which usually appear in the smooth gradient region. In state-of-the-art video compression applications, in-loop filter (LF) [5] or post-filters [7], [8] have been adopted to remove the blocking artifacts for compression efficiency improvement both in terms of objective quality and subjective quality. However, those blocking-artifacts-effective methods present little effect on the visibility of false contour artifacts.

An example of the false contour artifacts is shown in Fig. 1 for a natural video with smooth gradient regions, e.g., the sky region [Fig. 1(a)]. As shown in the figure, after compressing the video using H.264/AVC, both blocking artifacts and false contour artifacts are introduced into the compressed video [Fig. 1(b)]. By enabling H.264/AVC in-loop filter [5], most of the blocking artifacts (obvious in the mountain region) can be effectively removed in the filtered image [Fig. 1(c)]. However, the false contour artifacts (obvious in the sky region) are still visible, which presents that the in-loop filter has little effect on the visibility of the contour artifacts. Consequently, in this paper, an approach to suppress false contour artifacts in the decompressed video is investigated to further improve the subjective quality for video compression applications.

Approaches to reduce the false contour artifacts can be classified into two categories according to the position it applied: one is contour prevention approaches, which affect the images prior to or during quantization to prevent the loss of low amplitude detail [9]–[13]; the other is decontouring approaches, which only affect the images after quantization to reduce the visibility of false contour artifacts [6], [14]–[18]. For video/image coding applications, especially at the user side, the compressed videos/images, which already have false contour artifacts, are usually provided. Reducing the contour artifacts for those decompressed contents is very important for the quality of service. Since the entropy loss of those low-amplitude details cannot be exactly restored at the decoder side, we focus on the decontouring approaches to reduce the visibility of the false contour artifacts.

Manuscript received May 21, 2010; revised December 02, 2010; accepted January 26, 2011. Date of publication February 14, 2011; date of current version July 15, 2011. This work was supported in part by the Chinese University of Hong Kong under the Focused Investment Scheme (Project 1903003) and in part by Core Research for Evolution Science and Technology (CREST), Japan Science and Technology Agency (JST), Japan. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Brendt Wohlberg.

X. Jin is with the Information Technology Research Organization, Waseda University, Fukuoka 808-0135, Japan (e-mail: xjin@aoni.waseda.jp).

S. Goto is with the Graduate School of Information, Production, and Systems, Waseda University, Kitakyushu 808-0135, Japan (e-mail: goto@waseda.jp).

K. N. Ngan is with the Department of Electronic Engineering, The Chinese University of Hong Kong, Hong Kong, China (e-mail: knngan@ee.cuhk.edu.hk).

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

Digital Object Identifier 10.1109/TIP.2011.2114356

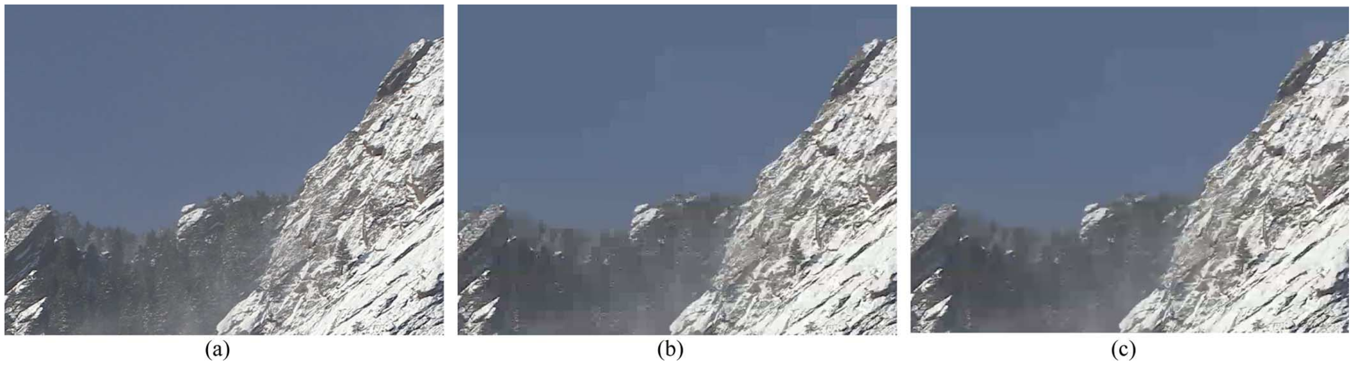


Fig. 1. Portion of a natural video frame. (a) The original uncompressed image. (b) H.264/AVC compressed image with LF off. (c) H.264/AVC compressed image with LF on.

A kind of general approach for decontouring is to apply a smoothing filter across the contour edge. Lee *et al.* [14] proposed to apply one-dimensional variable size directional smoothing filters orthogonal to the detected contour direction to reduce the false contours. However, when the contour artifacts are close to an actual edge, applying a smoothing filter may also blur the edge. Choi *et al.* [15] proposed to apply an edge-preserving filter using multi-dimensional weight functions to reduce the contour artifacts and preserve edges as well. However, the efficiency of applying smoothing filter is strongly depended on the design of the filter and also relied on the distribution of contours in the compressed image.

Besides applying smoothing filters, Ahn *et al.* [16] proposed a combined method based on random shuffling and low-pass filtering. Random shuffling sprinkles false contours over the neighboring regions; and low-pass filter tries to remove noise-like patterns introduced by random shuffling. Besides the noise which may be introduced during sprinkling, some tiny details in the low-gradient area may also be broken by random shuffling, and the low-pass filter may also blur the image.

Dithering can be applied in decontouring, like the work proposed by Boyce *et al.* [17] which applied temporally correlated noise signal to the decompressed image to improve the subjective quality. Also, Bhagavathy *et al.* [18] proposed a multi-scale probabilistic dithering method to first determine the presence and scale of contour artifacts around each pixel by multi-scale analysis, then to dither the pixel (if contour artifacts exist around it) based on the probability distribution of the color in its neighborhood. The method presents high computational complexity in determining the scale of contours.

Daly *et al.* [6] proposed a predictive decontouring method, which made a quantized low-pass filtered image a prediction of contour artifacts, and subtracted it from the input image to remove the contour artifacts. The processing quality of the algorithm depends on the low-pass filter it used. If noise exists in the video, which is very general to natural videos, the effectiveness of this algorithm will be constrained.

Most of the above techniques performed pixel-by-pixel decontouring. Yoo *et al.* [19] proposed a 16×16 macroblock (MB) based in-loop decontouring method for H.264/AVC video en/decoding. The basic processing unit of the method is a 16×16 MB which is consistent with the basic unit of H.264/AVC video compression. MBs need to be processed are determined by the quantization parameter (QP), MB mode and

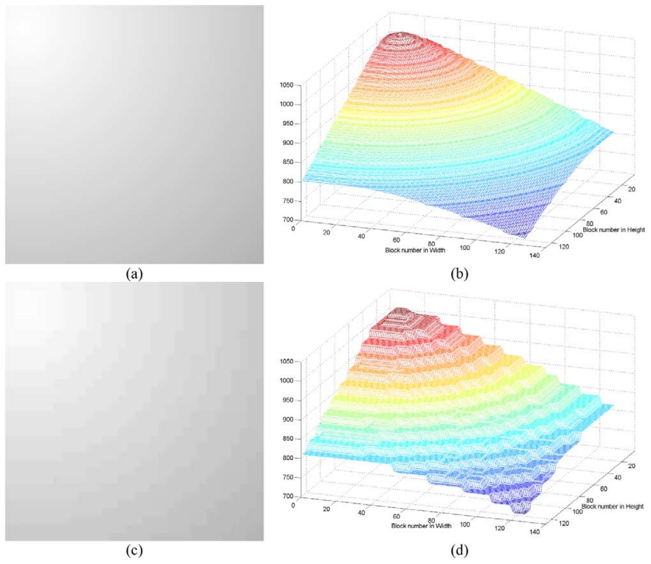


Fig. 2. (a) Original tungsten sensor shading image. (b) DC surface of (a). (c) H.264/AVC compressed (a) at 66 kbit/s with LF on. (d) DC surface of (c).

the gradients in four directions. Pseudorandom noise masks are added to the MB to dither the pixel. The algorithm is much simpler compared with the previous pixel-based approaches with a processing quality penalty.

As mentioned above, false contour artifacts are mainly caused by the loss of low amplitude detail during quantization [6], which breaks the regions with smooth luminance/color gradient into a number of “bands.” Considering the direct current (DC) coefficient after $N \times N$ DCT represents the average value (the average brightness or color) of the $N \times N$ block, the DC coefficient surface of the luminance signal of the original image and that of the H.264/AVC decoded image (with LF on) as N equals to 4 are compared in Fig. 2 for a tungsten sensor shading image [20]. Comparing the two DC surfaces, the DC surface of the decoded image [Fig. 2(d)] is not as smooth as that of the original image [Fig. 2(b)]. The stepped DC surface corresponds to the false contours in the pixel domain [Fig. 2(c)]. If the DC surface of the decoded image can be compensated to have the smooth gradient as the original, the contour artifacts in the pixel domain can be suppressed.

Consequently, a new decontouring approach is proposed in this paper for suppressing the contour artifacts in the H.264/AVC decoded video using DC dithering. It is a block-based algorithm to be consistent with block-based quantization defined in H.264/AVC encoding to reduce the processing complexity. Blocks belonging to contour artifacts are detected by the texture correlations among its neighboring blocks using their DCT coefficients. Thereafter, the DC value of the block is dithered based on a composite model between the gradient smoothness of the DC surface and the block-edge smoothness in the decoded image. Comparing with the existing approaches, the proposed method presents the following advantages:

- new approach to effectively suppress the contour artifacts based on block level processing;
- much lower complexity in comparison with the previous pixel-based approaches;
- better processing results with comparable complexity in comparison with the previous block-based approaches;
- SoC or embedded platform-friendly approach because of the block-based processing, the reusability of the calculated results during artifacts detection and DC dithering, and no requirements on intermediate decoded information;

The rest of this paper is organized as follows. In Section II, the composite model is derived for DC dithering. Accuracy of the proposed model is proven in Section III by comparing processing results of four feature points. In Section IV, DCT-based block level contour artifacts detection mechanism is designed. Algorithm implementation method is introduced in Section V. Final processing results are presented in Section VI to demonstrate the effectiveness and the advantages of the proposed algorithm. Section VII concludes the paper with a discussion on future work.

II. Model Derivation

The proposed decontouring method tries to adjust the pixel values of the block that belongs to the contour artifacts by automatically dithering its DC value. The adjustment on DC coefficient will introduce variations in the average brightness/color of the image and will also change the block-edge smoothness between the current block and its neighboring blocks. Consequently, the relationships among the quantity of DC value adjustment and its corresponding influence on both brightness/color variations and block-edge smoothness are first studied. In the following, the luminance signal of the decoded video is used as an instance for model derivation, while all of the models described in the following can be extended to other color components.

To evaluate the brightness variation in the decoded image, the concept of optical flow velocity at a pixel position (x, y) in an image [21] is extended for a block position (x_0, y_0) on the luminance DC surface of the decoded image from the temporal domain to the spatial domain as

$$f = \left(\frac{\partial D}{\partial x}\bigg|_{x_0}\right)^2 + \left(\frac{\partial D}{\partial y}\bigg|_{y_0}\right)^2 \tag{1}$$

of which D is the value of DC coefficient. According to the definition of $N \times N$ DCT, the DC coefficient of a Block C (as that shown in Fig. 3) is described as

$$D = \frac{1}{\sqrt{NN}} \sum_{i=0}^{N-1} \sum_{j=0}^{N-1} p_{i,j} \tag{2}$$

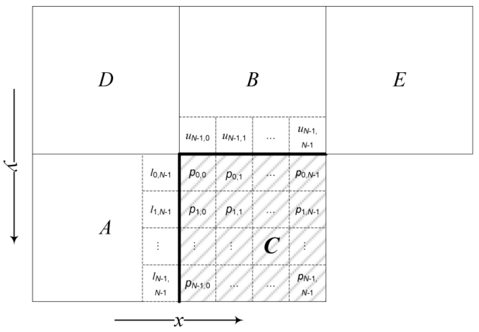


Fig. 3. Relative locations among the current block C and its neighboring blocks A, B, D, and E, and the pixels.

where $p_{i,j}$ is the (i, j)th pixel of Block C. When the optical flow velocity is calculated for an MB position, N equals to 16. As for a block position, N equals to 4.

As defined by (1), f measures the magnitude of 2-D spatial gradient of brightness variation at position (x_0, y_0) on the DC surface. Here, Robert Cross operator is used to calculate f because of its simplicity. Let Block C in Fig. 3 represent the block at position (x_0, y_0), f is further derived as

$$f = (D_d - D_c)^2 + (D_b - D_a)^2 \tag{3}$$

where D_a, D_b, D_c, and D_d represent the DC coefficient of Block A, B, C, and D, respectively. As a small adjustment δ is added to the DC value of Block C, f becomes

$$\begin{aligned} f(\delta) &= (D_d - D_c - \delta)^2 + (D_b - D_a)^2 \\ &= \delta^2 + 2\delta m + m^2 + n^2 \end{aligned} \tag{4}$$

where $m = D_c - D_d$, and $n = D_b - D_a$.

Adjustment on the DC value of the block will influence not only the smoothness of the brightness gradient at the block position but also changes the block edge smoothness. The block edge smoothness can be measured by the pixel value discontinuity across the block boundaries between the current block and its neighboring blocks, which can be formulated as

$$E = \|H\|_2^2 + \|V\|_2^2 \tag{5}$$

where H and V represent the horizontal edge difference vector and the vertical edge difference vector, respectively; $\| \bullet \|_2$ is the Euclidean norm of a vector. According to H.264/AVC video decoding order, as the current block is being decoded, the blocks right to it or below it have not been decoded yet. Consequently, if Block C is the block being processed, H evaluates the edge smoothness between Block C and Block B, while, V evaluates the edge smoothness between Block C and Block A. Then, we have

$$H = \begin{bmatrix} p_{0,0} & - & u_{N-1,0} \\ p_{0,1} & - & u_{N-1,1} \\ \vdots & & \\ p_{0,N-1} & - & u_{N-1,N-1} \end{bmatrix} \text{ and}$$

$$V = \begin{bmatrix} p_{0,0} & - & l_{0,N-1} \\ p_{1,0} & - & l_{1,N-1} \\ \vdots & & \\ p_{N-1,0} & - & l_{N-1,N-1} \end{bmatrix}. \tag{6}$$

TABLE I
COMPOSITE FUNCTIONAL RELATIONSHIP BETWEEN f AND g

	s	g	f	Case
$D_d \ge D_c$, $\delta \ge 0$, $m \le 0$	$[0,+\infty)$	$[g_0, g_1]$	$f=(G+m-s/2)^2+n^2$	1
	$(2m,0)$	$[g_{\min}, g_0]$	$f=(G\pm m\mp s/2)^2+n^2$	2
		$(g_0, g_1]$	$f=(G+m-s/2)^2+n^2$	
	$[4m,2m]$	$[g_{\min}, g_0]$	$f=(G\pm m\mp s/2)^2+n^2$	3
	$(-\infty,4m)$	$[g_1, g_0]$	$f=(G\pm m\mp s/2)^2+n^2$	4
$D_d \le D_c$, $\delta \le 0$, $m \ge 0$	$(4m,+\infty)$	$[g_1, g_0]$	$f=(G\pm m\mp s/2)^2+n^2$	5
	$[2m,4m]$	$[g_{\min}, g_0]$	$f=(G\pm m\mp s/2)^2+n^2$	6
	$(0,2m)$	$[g_{\min}, g_0]$	$f=(G\pm m\mp s/2)^2+n^2$	7
		$(g_0, g_1]$	$f=(G-m+s/2)^2+n^2$	
	$(-\infty,0]$	$[g_0, g_1]$	$f=(G-m+s/2)^2+n^2$	8

As the DC value of the block is adjusted by adding a δ, the block-edge smoothness g will be

$$g(\delta) = \|H + \delta \bullet M\|_2^2 + \|V + \delta \bullet M\|_2^2 \tag{7}$$

where $M^T = [1/N, 1/N, \ldots, 1/N]$. Putting M into (7), finally we have

$$g(\delta) = \left(\frac{2}{N}\right)\delta^2 + \left(\frac{2}{N}\right)\delta s + E \tag{8}$$

where $s = (\sum_{j=0}^{N-1} p_{0,j} - \sum_{j=0}^{N-1} u_{N-1,j} + \sum_{i=0}^{N-1} p_{i,0} - \sum_{i=0}^{N-1} l_{i,N-1})$, and $E = \|H\|_2^2 + \|V\|_2^2$.

Analyses on $f(0)$ and $g(0)$ for H.264/AVC decoded images (with LF on) indicate that: LF tries to reduce the value of $g(0)$ which corresponds to improve the block edge smoothness, but it does not take the smoothness of gradient into consideration, which shows inefficiency in suppressing contour artifacts. Consequently, the proposed decontouring method aims to find a suitable δ for the decoded block to improve the smoothness of the gradient with the tradeoff in block-edge smoothness.

Though both of function f and g are quadratic functions of DC adjustment δ, the greatest lower bound and the least upper bound of the function f are different from those of the function g as δ varies. Taking $[0, 2(D_d - D_c)]$ as the adjustment range of δ, an instance of $f(\delta)$ (solid blue curve) and $g(\delta)$(dashed red curve) of a block is depicted in Fig. 4. As shown in the figure, f and g cannot reach the minimum simultaneously, which reveals that the brightness variation of the decoded image and the pixel value variation across the block boundary cannot achieve the best smoothness concurrently during the adjustment of δ.

In order to find out the relative correlation between the gradient smoothness and the block-edge smoothness when altering δ, the functional relationship between f and g is further derived as

$$f(G) = \begin{cases} \left(G + m - \frac{s}{2}\right)^2 + n^2, & \delta = G - \frac{s}{2} \\ \left(G - m + \frac{s}{2}\right)^2 + n^2, & \delta = -G - \frac{s}{2} \end{cases} \tag{9}$$

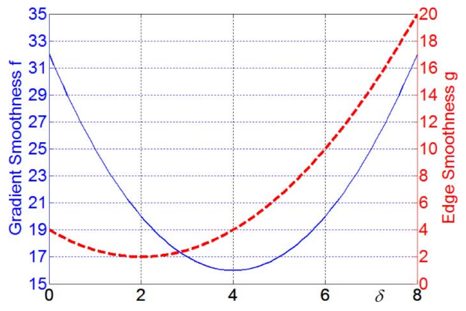


Fig. 4. Instance of f and g as δ varies in $[0, 2(D_d - D_c)]$. Blue curve is $f(\delta)$ with f-axis labeling on the left. Red curve is $g(\delta)$ with g-axis labeling on the right.

where $G = \sqrt{N(g-E)/2 + s^2/4}$. Taking $2(D_d - D_c)$ as the upper/lower bound of $\delta (\delta \in [0, 2(D_d - D_c)]$ or $\delta \in [2(D_d - D_c), 0])$, the composite functional relationship between the gradient smoothness f and the block-edge smoothness g can be derived as that listed in Table I. Using $2(D_d - D_c)$ as the range limit is to guarantee improvement in the smoothness of gradient during the adjustment. As listed in the table, the function style differs with the relative variation among D_d, D_c, s, and m, which are determined by the current block and its neighboring blocks. For example, when $m < 0$, as s changes from $[2m, 0)$ to $[4m, 2m]$, the value range of g and the function style of f changes accordingly. $g_{\min}$, g_0 and g_1 are $g_{\min} = g(-s/2) = E - (1/2N) \times s^2$, $g_0 = g(0) = E$, and $g_1 = g(-2m) = (8/N) \times m^2 - (4/N) \times m \times s + E$.

Within these cases, the ones with the highest probability of occurrence for natural video content are Case 1 and Case 2, which occupies an average of 63.78% and 17.00%, respectively. The composite functions of these two representative cases are depicted in Fig. 5. As shown in the figure, when the edge smoothness is improved, shown as g moves from right to left, the gradient smoothness f will also become smaller. While after g passes the point at which the curve is tangent to the horizontal axis, f starts to increase with the reduction in g. Consequently, finding a point on the model that shows better tradeoff between f and g is critical for artifacts reduction. The

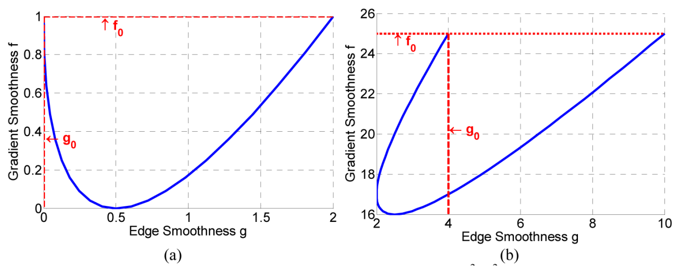


Fig. 5. Composite relationship between f and g for: (a) Case 1; (b) Case 2. $f_0 = f(0) = m^2 + n^2$.

TABLE II
FEATURE POINTS DEFINITION FOR PERFORMANCE COMPARISON

Point	g	f	δ
a	g_{min}	$f(g_{min})$	$\arg\min(g(\delta))$
b	$f^{-1}(f_{min})$	f_{min}	$\arg\min(f(\delta))$
c	-	-	$\arg\min\left(\frac{f(\delta)}{\max(f(\delta))}+\frac{g(\delta)}{\max(g(\delta))}\right)$
d	-	-	$\arg\min\left(\left(\frac{f(\delta)}{\max(f(\delta))}\right)^2+\left(\frac{g(\delta)}{\max(g(\delta))}\right)^2\right)$

consistency between the model definition and the actual effects on false contour suppression is evaluated in the following.

III. Feature Points Evaluation

Four feature points on the proposed composite model, denoted as Point a to d, are selected for consistency evaluation. Definition of each point is listed in Table II. As listed in the table, Point a only guarantees the best smoothness of the block-edge; Point b only guarantees the best smoothness of gradient; Point c and Point d optimize the smoothness of gradient and the smoothness of block-edge jointly to provide a processed result without deteriorating any of them. Compared with Point c, d is more sensitive to variations in either of the two factors.

We first use the tungsten sensor shading image [Fig. 6(a)] to evaluate the performance of the above four feature points to prove the accuracy of the proposed model. The H.264/AVC encoded image with LF on [5] shows obvious contour artifacts [Fig. 6(b)]. The proposed decontouring is applied to all the 4×4 blocks $(N = 4)$ following the decoding order of H.264/AVC. Processed results displayed in (c)–(f) prove that: only optimizing the smoothness of the block-edge as Point a is insufficient for decontouring [Fig. 6(c)]; only optimizing the smoothness of gradient as Point b will result in over-blurring problem [Fig. 6(d)], since it flats the brightness too much; Point c [Fig. 6(e)] and Point d [Fig. 6(f)] provide better tradeoffs between the gradient smoothness and edge smoothness, which remove the false contours while preserve the gradual brightness variation to achieve a more natural image. The result of Point c is slightly better than that of Point d by comparing the similarity between the processed image and the original one. All of the processing results are consistent with the theoretical meanings of the feature points that derived from the model. Fig. 7 shows more results demonstrating the good consistency between the processing effects and the model definition by applying the four feature points on natural video content. Improvements in

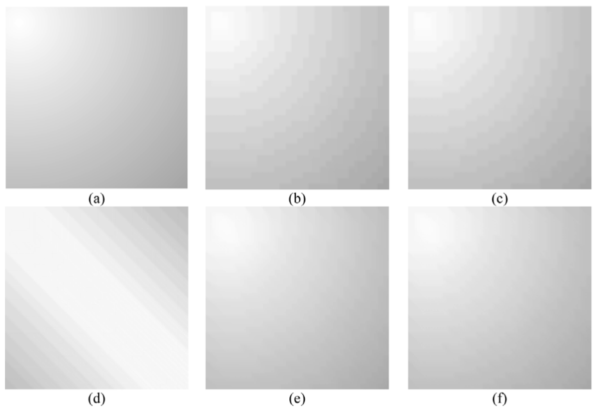


Fig. 6. (a) Original tungsten sensor shading image; (b) H.264/AVC decoded image at 66 Kbps with LF on; results of DC dithering according to: (c) point a; (d) point b; (e) point c; (f) point d.

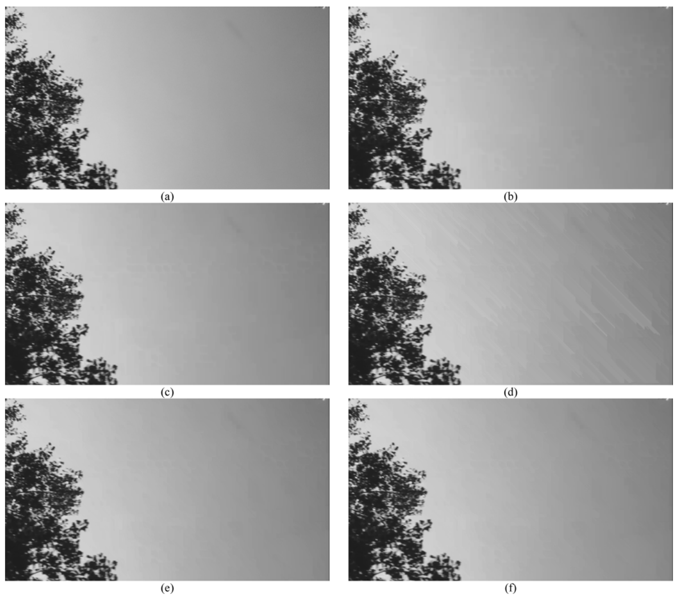


Fig. 7. (a) Original image; (b) H.264 decoded image by at 3 Mbps with LF on; results of DC dithering according to: (c) point a; (d) point b; (e) point c; (f) point d.

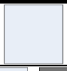
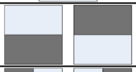
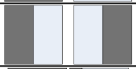
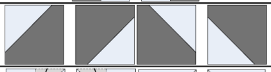
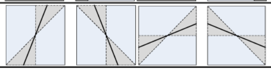
subjective quality are shown in the processed results both of Point c and Point d, while more noticeable to Point c.

IV. DCT-Based Block Level Contour Artifacts Detection

With respect to reducing contour artifacts, it is in general desirable to affect only regions where false contours occur, while preserve other regions, especially those with textures or details. So, before performing decontouring, the regions where false contours are likely to be present need to be detected. In this section, a block-level contour artifacts detecting algorithm is presented by exploiting DCT coefficients.

As mentioned above, contour artifacts always appear in the smooth gradient region because of loss of low amplitude detail. Smooth gradient implies two features of the region: first, the region should not contain directional texture or details, which is called a smooth content region in the following; second, the brightness/color variation in the region is smooth. Therefore, to detect the blocks involved in contour artifacts, an algorithm

TABLE III
PROPOSED EDGE FEATURE DEFINITION

Ph	*Pv*	Texture Direction	*T*(*x*, *y*)	Possible Edge Patterns
0	0	No directional	0	
≠ 0	0	Horizontal	1	
0	≠ 0	Vertical	2	
Ph = *Pv* or *Ph* = -*Pv*		Diagonal	3	
Else		Other directional	4	

consisting of two analyses, so-called DCT-based texture distribution analysis (DTDA) and brightness/color variation distribution analysis (BVDA), is proposed to first distinguish the smooth content region which barely contains directional texture, then to find out the block presenting brightness/color variation feature like that around the "band" boundary.

A. DTDA

DTDA tries to find out the region has few textures through analyzing the texture distribution in the region. Considering the edge pattern of a block can be decomposed into vertical and horizontal directional components by DCT transform, 2D-DCT transform is introduced to design DTDA.

The spatial edge features in a block can be represented by two edge feature sets of 2-D DCT coefficients: $\{f_{u,0} : u = 1, \ldots, N-1\}$ represents horizontal features, where u is the row index; $\{f_{0,v} : v = 1, \ldots, N-1\}$ represents vertical features, where v is the column index, respectively [22]. It has been proven that: the horizontal edges of the block only correspond to $f_{u,0}$; the vertical edges only correspond to $f_{0,v}$; other directional edges correspond to both of them with balanced value. For the proposed method, only coarse texture feature is needed to decide whether the current MB/block belongs to a smooth content region or not. Consequently, only the first two most significant AC coefficients, $f_{1,0}$ and $f_{0,1}$, are used in block-level edge feature retrieval. In order to omit the floating point operation, the edge feature decision using the exact value of $f_{1,0}$ and $f_{0,1}$ is further approximated by using Ph and Pv, which are given by

$$\begin{aligned} Ph &= Ph_0 - Ph_{N-1} = \sum_{j=0}^{N-1} p_{0,j} - \sum_{j=0}^{N-1} p_{N-1,j} \\ Pv &= Pv_0 - Pv_{N-1} = \sum_{i=0}^{N-1} p_{i,0} - \sum_{i=0}^{N-1} p_{i,N-1}. \end{aligned} \tag{10}$$

Finally, the edge features of a block (x, y), denoted by $T(x, y)$, is defined in Table III based on the value of Ph and Pv.

Experimental results proved that, as N equals to 4, 100% of $T(x, y)$ using the approximation method described by (10) are the same with that using $f_{1,0}$ and $f_{0,1}$. As the block size becomes larger, like $N = 16$, the accuracy is an average of 97.85%, which is still acceptable. However, the complexity of (10). is much lower than using $f_{1,0}$ and $f_{0,1}$ by omitting floating point operation and reducing the number of additions.

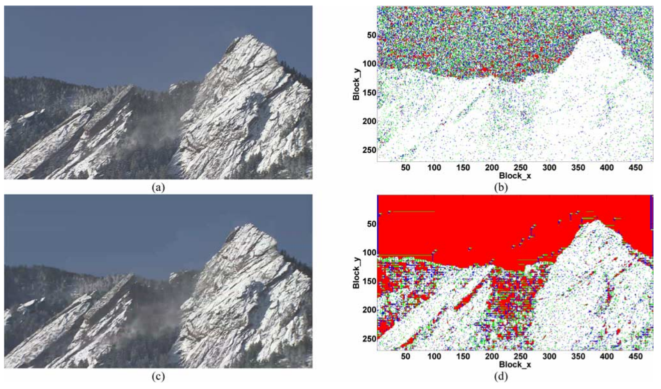


Fig. 8. Block level edge feature decision results. (a) Original image; (b) $T(x, y)$ of image (a); (c) H.264/AVC decompressed image; (d) $T(x, y)$ of image (c). Red, green, blue, gray, and white dots correspond to $T(x, y)$ at this block position equals to 0, 1, 2, 3, and 4, respectively.

Applying the proposed simplified edge feature decision method on each block, Fig. 8 presents instances for $T(x, y)$ on both an original image and an H.264/AVC decompressed image using $N = 4$. Comparing $T(x, y)$ retrieved both for the smooth gradient region (the sky) and texture region (the mountain) in Fig. 8(b) and (d): after compression, the smooth gradient region becomes to the region almost fulfilled by red dots [Fig. 8(d)] because of loss of tiny details; while, the texture region still mainly consists of directional texture features, reflected as mountain region is mainly filled by white dots both in Fig. 8(b) and (d). Therefore, blocks belonging to the region which shows high probability of having directional texture blocks can be regarded as the blocks within texture regions. The others will be the ones belonging to smooth content regions, which will be the output of DTDA.

Consequently, let block (x_0, y_0) in the decompressed image be the current block being analyzed, a directional texture distribution probability is computed for DTDA as

$$p_T(x_0, y_0) = \frac{\sum\limits_{\{(x,y)\in R_L(x_0,y_0)\}} \omega(T(x,y), 3)}{\sum\limits_{\{(x,y)\in R_L(x_0,y_0)\}} 1} \tag{11}$$

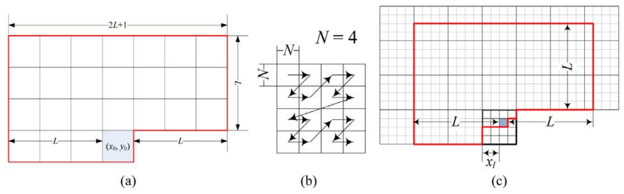


Fig. 9. (a) $R_L(x_0, y_0)$ definition for $N = 16$; (b) 4 × 4($N = 4$) block decoding order defined in H.264/AVC. (c) An instance of $R_L(x_0, y_0)$ for $N = 4$. Gray block represents the current block (x_0, y_0). Region circled by the red line is $R_L(x_0, y_0)$.

where $R_L(x_0, y_0)$ corresponds to a L-block-sized neighborhood around block (x_0, y_0) in the decompressed image and $\omega(\cdot)$ is an indicator function defined as

$$\omega(a, b) = \begin{cases} 1, & \text{if } a \geq b \\ 0, & \text{else.} \end{cases} \tag{12}$$

Region $R_L(x_0, y_0)$ only consists of decoded blocks around the current block to endue the proposed algorithm with feasibility to process each decoded block immediately to reduce the memory consumption. Definitions of $R_L(x_0, y_0)$ corresponding to different Ns are shown in Fig. 9. As $N = 16$, $R_L(x_0, y_0)$ is a $(2L + 1) \times L + L$ region as depicted in Fig. 9(a). While as $N = 4$, since H.264/AVC decoding order for 4×4 blocks within an MB [Fig. 9(b)] results in the available blocks (already decoded blocks) varies with the position of the current block inside the MB, $R_L(x_0, y_0)$ will be a region consisting of all of the decoded blocks within the same macroblock with block (x_0, y_0) and $(2L + 1) \times L + (L - x_l) \times N$ blocks around it [Fig. 9(c)].

$p_T(x_0, y_0)$ indicates the distribution probability of directional edges, other than horizontal and vertical edges, in the region $R_L(x_0, y_0)$ around block (x_0, y_0). Then, a threshold T_t is set between 0 and 1 as a constraint for $p_T(x_0, y_0)$. If $p_T(x_0, y_0)$ is higher than T_t, block (x_0, y_0) belongs to a texture region, otherwise, it belongs to a smooth content region.

B. BVDA

For the blocks determined to be within a smooth content region by DTDA, BVDA is performed by computing a brightness/color variation probability $p_B(x_0, y_0)$ to determine whether the block is involved in contour artifacts or not. $p_B(x_0, y_0)$ is given by

$$p_B(x_0, y_0) = \frac{\sum\limits_{\{(x,y) \in R_L(x_0,y_0)\}} \omega(B(x, y), 0)}{\sum\limits_{\{(x,y) \in R_L(x_0,y_0)\}} 1} \tag{13}$$

where $B(x, y)$ represents the brightness/color variation among the block (x, y) and its neighboring block. It is given by

$$B(x, y) = \begin{cases} -1, & \left(\sum\limits_{D_x \in \{D_a, D_b, D_d, D_e\}} \omega(|D_c - D_x|, D_T) \right) \geq 1 \\ 0, & |m| + |n| = 0, \\ 1, & \text{else} \end{cases} \tag{14}$$

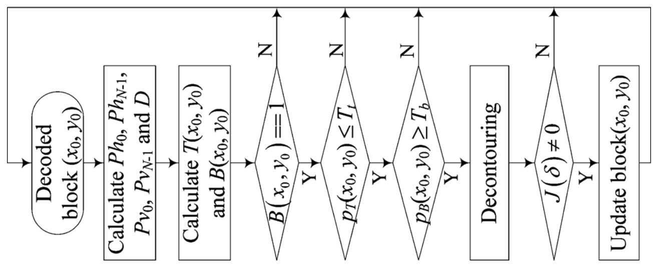


Fig. 10. Overall procedure of the proposed decontouring algorithm for a block (x_0, y_0).

where D_c represents the DC value of block (x, y), D_a, D_b, D_d, and D_e represent the DC value of Block A, B, D, and E around block (x, y) according to Fig. 3, respectively. D_T is a threshold defined to distinguish the brightness/color variation caused by an actual difference in content. After analyzing natural video content, D_T is set to 14 which corresponds to each pixel of a block differs by a value of 3.5 compared with the pixels in the other block.

$p_B(x_0, y_0)$ indicates the distribution probability of small brightness/color variation, which is smooth in contrast with that caused by an actual content change, in the neighborhood around block (x_0, y_0). Also, a threshold T_b is set between 0 and 1 as a constraint for $p_B(x_0, y_0)$. If $p_B(x_0, y_0)$ is higher than T_b, block (x_0, y_0) may be involved in contour artifacts and will be processed by the proposed decontouring algorithm.

With processing, an additional gradient constraint, $B(x_0, y_0)$ equals to 1, is introduced prior to DTDA and BVDA to omit processing for those already smoothed blocks or those blocks with content difference to accelerate the processing.

V. Decontouring Implementation

The final decontouring procedure is shown in Fig. 10. As depicted in the figure, the output value of decontouring, δ, is always a floating point result, while, pixels in the block can only take integral values. So, a dithering is performed in $J(\delta)$ to generate the final value to be added to each pixel. $J(\delta)$ is given by

$$J(\delta) = \begin{cases} \operatorname{sgn}(\delta) \left\lceil \frac{|\delta|}{N} \right\rceil, & \frac{|\delta|}{N} - \left\lfloor \frac{|\delta|}{N} \right\rfloor \geq 0.5 \\ \operatorname{sgn}(\delta) \left\lfloor \frac{|\delta|}{N} \right\rfloor, & \text{else.} \end{cases} \tag{15}$$

where $\lfloor \cdot \rfloor$ and $\lceil \cdot \rceil$ represents floor and ceiling functions, respectively.

The proposed algorithm processes the decoded video block-by-block using the same processing order with video decoding and the processing only exploits the decoded pixel information. These two features imply that the algorithm can be easily integrated into a video decoder without any additional requirements on buffering the decoded whole image, or buffering other intermediate information. Consequently, a set of buffer managements and update methods like that used in [23] can be designed to make the proposed method be friendly to embedded system or VLSI chip, on which video decoder is usually implemented. It is very important for the algorithm to work together with a video decoder.

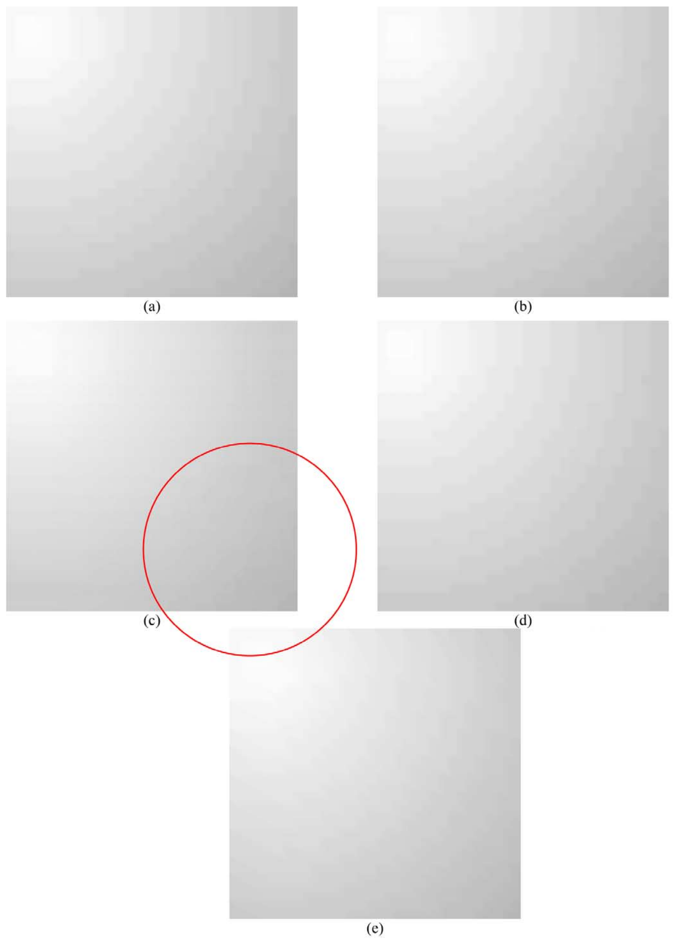


Fig. 11. (a) Input H.264/AVC decompressed tungsten sensor shading image $(\mathrm{CR} = 1420 : 1)$ with contour artifacts; (b) result of Bhagavathy *et al.* [18] using the original setting $k = 1$; (c) result of Bhagavathy *et al.* [18] using a changed setting $k = 5$; (d) result of Yoo *et al.* [19]; (e) result of the proposed decontouring method.

VI. Experimental Results

In this section, we demonstrate the effectiveness of the proposed decontouring algorithm. The effectiveness in suppressing the contour artifacts together with low complexity are illustrated in comparison with two representative decontouring algorithms: one is "Multiscale decontouring" method proposed in Bhagavathy *et al.* [18], which is a pixel-based approach; the other is "In-loop decontouring" algorithm proposed by Yoo *et al.* [19], which is also a block-based approach for decompressed H.264/AVC videos.

First, the processing results are compared for H.264/AVC decompressed tungsten sensor shading image (512×512) at compression ratio (CR) 1420:1 in Fig. 11(a). Using the same parameter settings as that mentioned in [18], the decontouring result using "Multiscale decontouring" is shown Fig. 11(b). However, the effectiveness in processing this image is limited.

It is found that the quality of the results provided by Bhagavathy *et al.* [18] depends on the pixel value difference defined around the contour. The original parameters provided in [18] set the pixel difference, k, as 1, which represents that pixels on either side of a contour differ by a value of 1. This setting assumed that the false contours are caused by simple color quantization (bit truncation), which results in limited loss in the amplitude [18]. However, for the video compressed by H.264/AVC, larger pixel value discontinuity between two "bands" may be resulted in especially for being encoded at low bit rate. We experimented with different ks. Fig. 11(c) shows an instance that $k = 5$ which seems to be more effective than the original definition [Fig. 11(b)]. It can be observed that although using our selected k can reduce the existed false contours, it also introduces new ones at the bottom-right region.

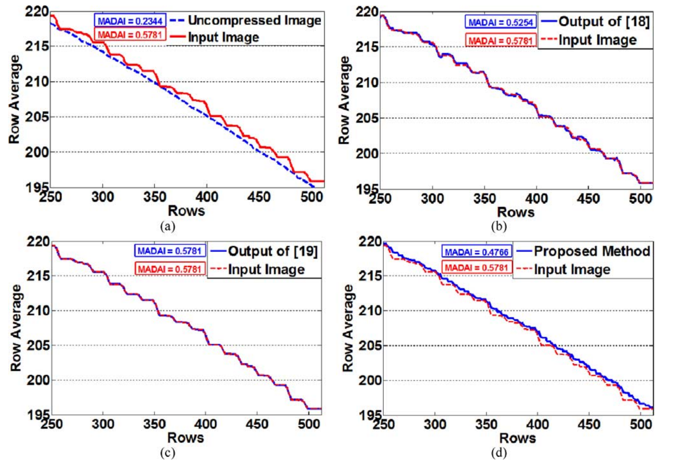


Fig. 12. Row averages of intensities for the image: (a) Fig. 11(a) the input image; (b) Fig. 11(b), output image of Bhagavathy *et al.* [18]; (c) Fig. 11(d), output image of Yoo *et al.* [19]; (d) Fig. 11(e), output image of the proposed decontouring method.

Fig. 11(d) shows the result of applying the block-based in-loop decontouring method proposed in [19]. The subjective performance improvement is limited.

Fig. 11(e) shows the result of applying our proposed block-based decontouring method by setting $N = 4$ and the size of $R_L(x_0, y_0)$, L, equals to 3. After analyzing H.264 compressed video sequences (the sequences are not used in the following test) with contour artifacts, T_t for $p_T(x_0, y_0)$ and T_b for $p_B(x_0, y_0)$ are set to 0.625 and 0.4, respectively. The optimization criterion corresponding to Point c on the composite model is used. Contour artifacts detection is first performed only in Y component. The detection result, whether the block has contour artifacts or not, is applied equally to all the three components (Y, U, and V) for reducing the complexity of the algorithm. Then, decontouring is performed in each component separately. It can be seen that the proposed method is effective in breaking down false contours and making the artifacts less visible.

Since the decontouring process should result in the reintroduction of smooth gradient, we use the smoothness of the output intensity gradients [14], [18] as an objective evaluation method to compare the proposed method with that of Bhagavathy *et al.* and Yoo *et al.* further. Fig. 12(a) plots the intensity average of each row of the input image shown in Fig. 11(a). Fig. 12(b)–(d) shows the row-wise intensity averages of the output images after applying the decontouring algorithms. It is noted that the staircase profile of the input image is smoothed out effectively by the proposed method. The maximum absolute difference in the average intensity between the neighboring two rows, denoted by MADAI, is calculated as a quantitative value to evaluate the smoothness of the intensity gradient objectively. The lower

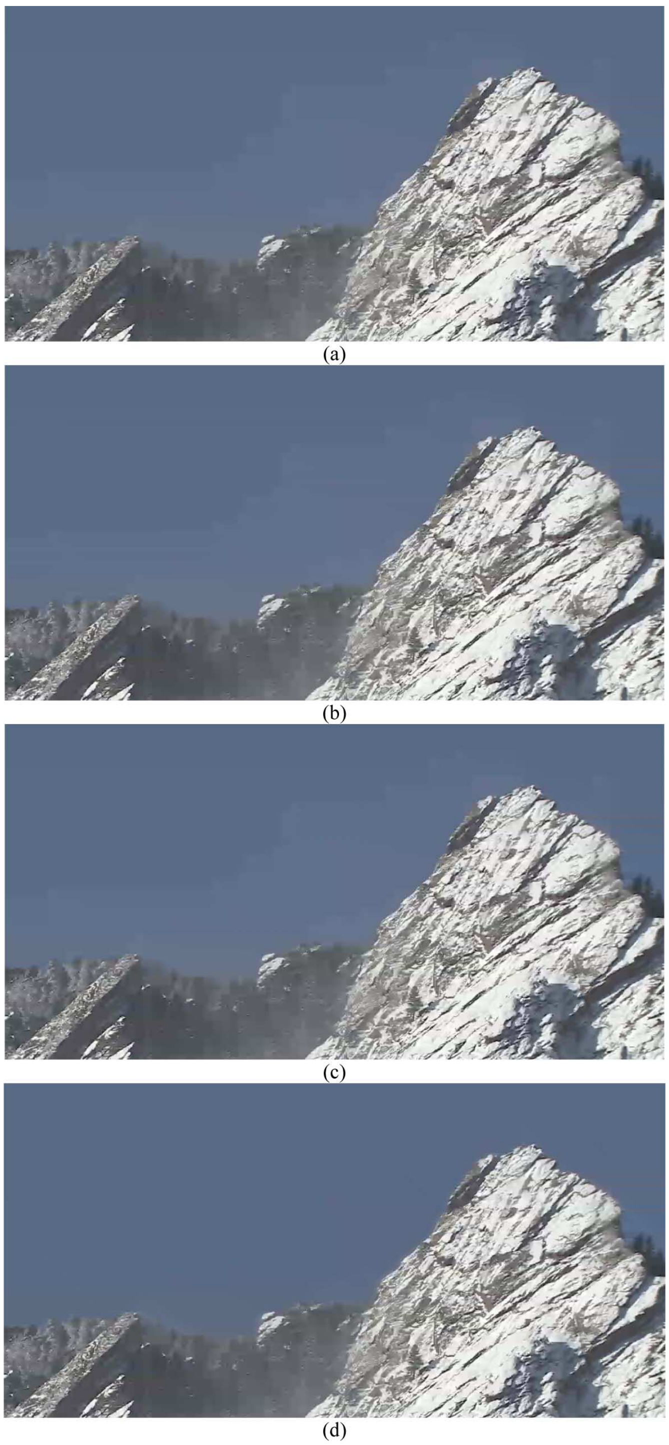


Fig. 13. Frame 3 of (a) Input H.264/AVC decompressed video ($\mathrm{CR} = 3347 : 1$) with contour artifacts, $\mathrm{MADAI} = 0.7821$; (b) result of Bhagavathy *et al.* [18] using original settings, $\mathrm{MAMDAI} = 0.9242$; (c) result of Yoo *et al.* [19], $\mathrm{MADAI} = 0.7087$; (d) result of the proposed decontouring method, $\mathrm{MADAI} = 0.3376$. MADAI of the uncompressed frame equals to 0.1556.

the value is, the higher the smoothness of the intensity gradient. It proved that the proposed algorithm always provides the best MADAI value approaching the value of the uncompressed video, which represents the best smoothness in the output intensity gradients.

Figs. 13–15 show more results of processing the natural video content (1920×1080) to demonstrate the effectiveness of the

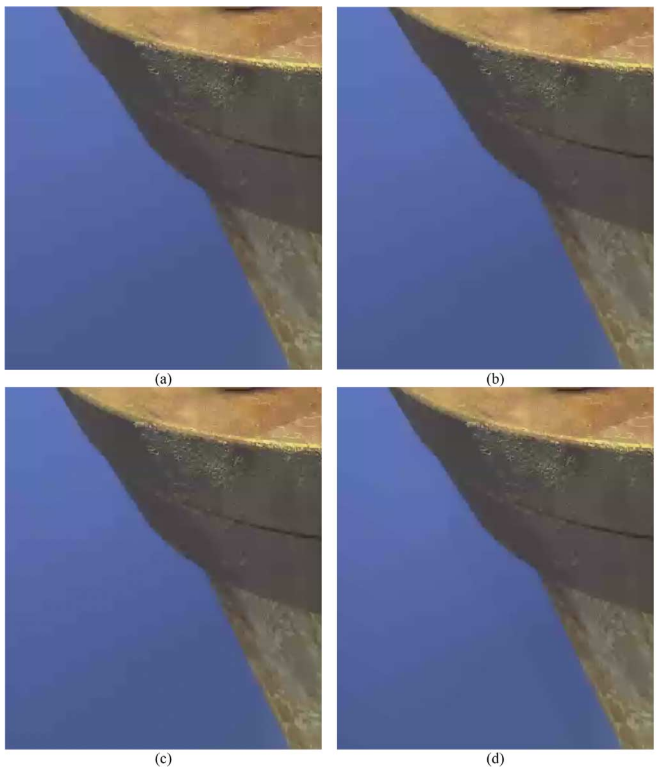


Fig. 14. Frame 1 of (a) Input H.264/AVC decompressed video ($\mathrm{CR} = 753 : 1$) with contour artifacts, $\mathrm{MADAI} = 0.4889$; (b) result of Bhagavathy *et al.* [18] using original settings, $\mathrm{MADAI} = 0.5222$; (c) result of Yoo *et al.* [19], $\mathrm{MADAI} = 0.9704$; (d) result of the proposed decontouring method, $\mathrm{MADAI} = 0.4741$. MADAI of the uncompressed frame equals to 0.0852.

proposed algorithm. Due to limitation on the paper length, only portions of frames together with MADAI values are shown. MADAI value is calculated for the smooth region in each video. For Multiscale decontouring method [18], since the efficiency of video compression is content dependant, the amplitude loss is also content depended which results in k should vary with video content. The processing quality varies a little bit with different settings in k, but such variation is not as visible as that in processing the tungsten sensor shading image. Consequently, only the results using its original settings in [18] are presented. As shown in these figures, the proposed decontouring method provides effective results in reducing the false contours (with the lowest MADAI value compared with the other algorithms), and simultaneously preserves the texture region correctly.

Furthermore, the effectiveness of the proposed algorithm is analyzed for the same video content compressed at different ratios. Fig. 16(a) and (c) shows portions of H.264/AVC compressed video at $\mathrm{CR} = 2382 : 1$ (much lower than that in Fig. 13) and 4459:1 (much higher than that in Fig. 13), respectively. Fig. 16(b) shows the results of the proposed algorithm for the video in (a). The false contours are effectively reduced without loss in the texture region. However, for the processing results of (c) shown in (d), the proposed algorithm shows some insufficiency in some regions close to the mountain boundary. Such insufficiency is mainly caused by the big losses in signal amplitude introduced by coarse quantization (very high compression ratio). The video shown in Fig. 16(c) is compressed by H.264/AVC using quantization parameter equal to 44, which

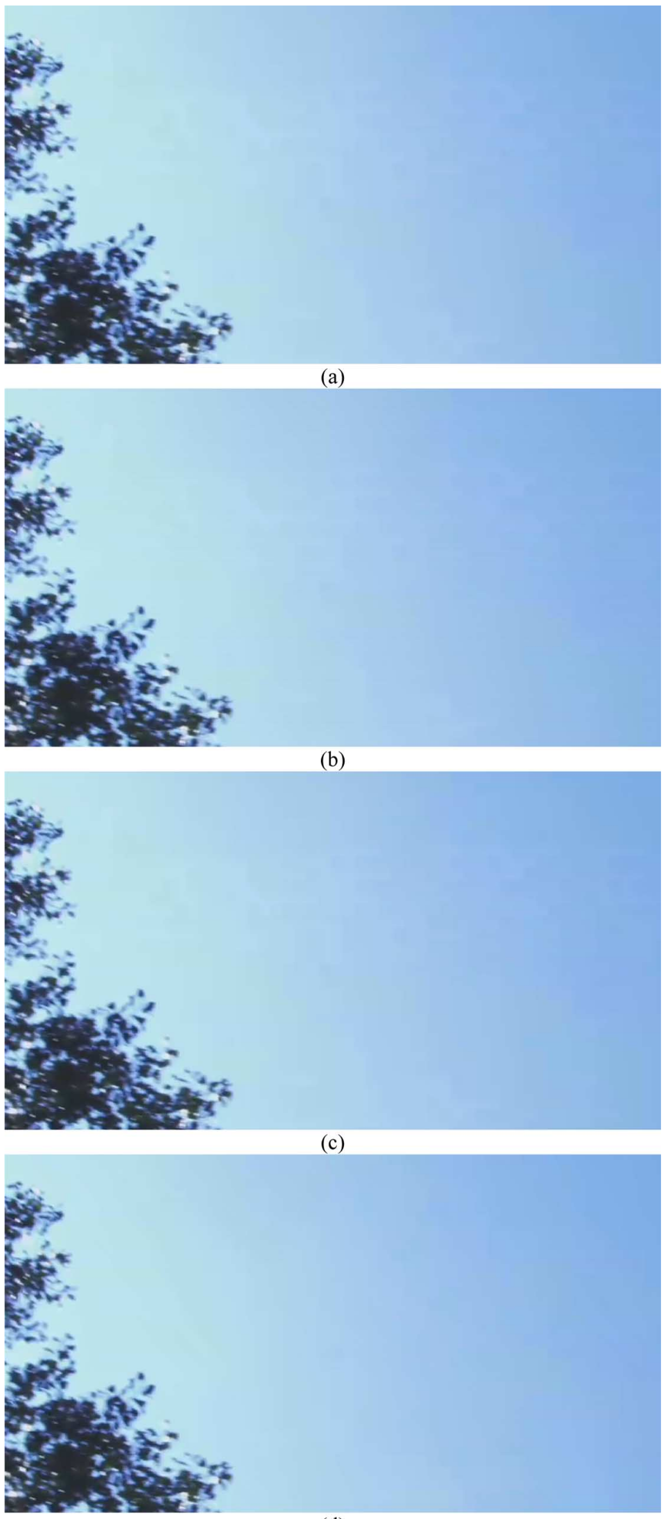


Fig. 15. Frame 15 of (a) Input H.264/AVC decompressed video $(\mathrm{CR} = 738:1)$ with contour artifacts, $\mathrm{MADAI} = 0.4741$; (b) result of Bhagavathy *et al.* [18] using original settings, $\mathrm{MADAI} = 0.3840$; (c) result of Yoo *et al.* [19], $\mathrm{MADAI} = 0.4696$; (d) result of the proposed decontouring method, $\mathrm{MADAI} = 0.3570$. MADAI of the uncompressed frame equals to 0.2782.

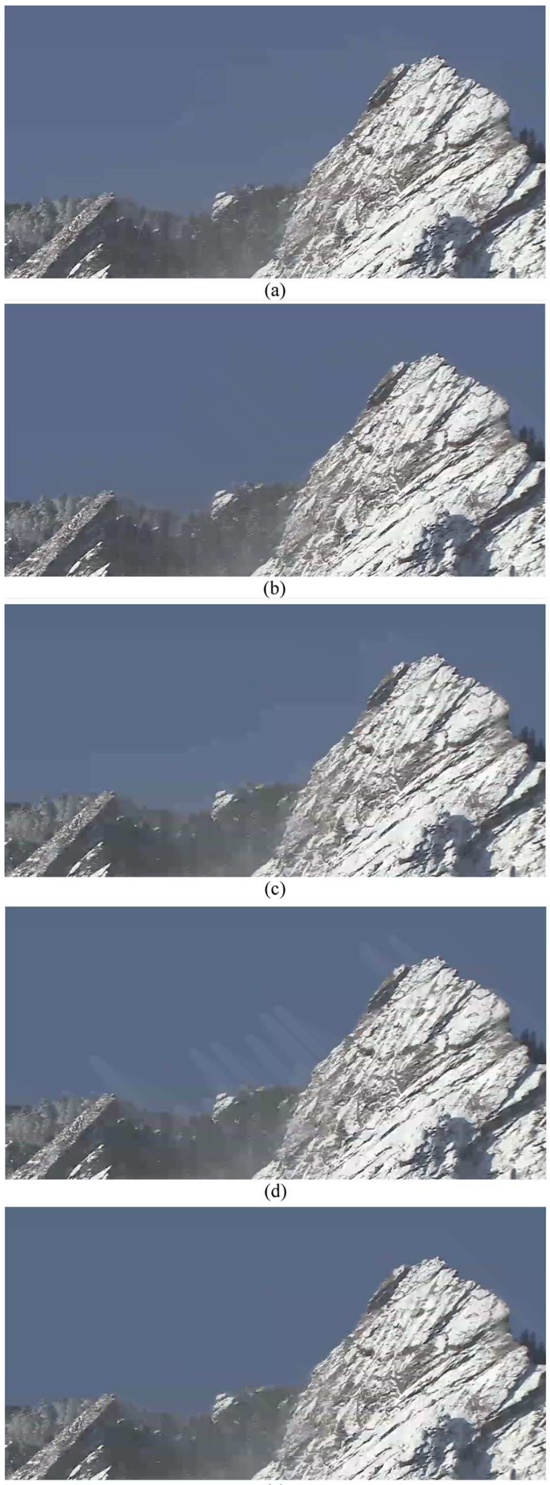


Fig. 16. Frame 3 of (a) Input H.264/AVC decompressed video at $\mathrm{CR} = 2382:1$, $\mathrm{MADAI} = 0.6177$; (b) decontoured output of the proposed method for (a), $\mathrm{MADAI} = 0.4270$; (c) Input H.264/AVC decompressed video at $\mathrm{CR} = 4459:1$, $\mathrm{MADAI} = 1.0000$; (d) decontoured output of the proposed method for (c), $\mathrm{MADAI} = 0.4410$; (e) decontoured output of the proposed method for (c) using $D_T = 20$, $\mathrm{MADAI} = 0.3379$.

already approaches the upper boundary, 51, of the standard. At such a high compression ratio, not only the low amplitude detail but also some medium amplitude information will be lost. Such a big loss in signal amplitude makes the brightness/color discontinuity around the "band" exceeds the threshold D_T, which is used to distinguish whether the brightness/color variation is

TABLE IV
COMPARISON IN PROCESSING TIME CONSUMPTION AND AVERAGED MADAI

Sequence Info.		Average Time Consumption (ms)			Average MADAI				
No. of Frames	Content	[18]	[19]	Proposed	Original Video	Compressed Video	[18]	[19]	Proposed
1	Shown in Fig. 11	40969.8	4.3	161.0	0.2344	0.5781	0.5254	0.5781	**0.4766**
60	Shown in Fig. 13	357371.0	67.9	369.0	0.1889	0.7981	0.7863	0.8711	**0.4003**
60	Shown in Fig. 14	387745.2	35.2	190.7	0.0959	0.5259	0.6613	0.8113	**0.5109**
60	Shown in Fig. 15	386937.4	66.9	116.2	0.2934	0.4637	0.4089	0.4607	**0.3375**

caused by content variation or not during contour artifacts detection. Since the original setting of D_T is retrieved from natural video contents, it needs to be increased when such a big quantization error occurs. Fig. 16(e) shows the processing results of the proposed algorithm after increasing D_T from 14 to 20. It shows effective results in reducing the false contours. Note that such a high compression ratio will only be applied to an extreme bandwidth constrained scenario, which is not that common.

This section is concluded by measuring the frame average processing time and frame average MADAI for the above decompressed videos in Table IV. The frame averaged MADAI (MADAI value averaged over all the frames of the video sequence) is used to evaluate the effectiveness in reducing the contour artifacts along the temporal direction. Each algorithm is implemented as a single thread C program and executed on a PC with 2.93-GHz Intel Core™ i7 CPU and 8-GB RAM. As listed in the table, the proposed algorithm always provides improved and the best MADAI (compared with that of the H.264 compressed videos) for all the video sequences, which indicates the improved smoothness in intensity gradient in the decontoured output video. The processing complexity of the proposed algorithm is much lower than Bhagavathy *et al.* [18]. Computations of Bhagavathy *et al.* [18] are mainly consumed in detecting the scale of the contour artifacts, which is performed on every pixel of the image. Most of the computations of our proposed method are spent on finding δ. However, such searching is performed only on the blocks determined as being involved in contour artifacts. So, the complexity of our proposed algorithm varies with video contents. Even though it is a little bit higher than Yoo *et al.* [19], the subjective decontouring quality derived by our proposed algorithm is much better than that of [19].

VII. CONCLUSION

In this paper, a decontouring method is proposed by dithering the DC value of the decoded block according to a model established between the smoothness of gradient of the DC surface and the smoothness of the block-edge in the decoded image. Experimental results demonstrated the effectiveness of the proposed method in decontouring and lower complexity in comparison with the previous algorithms. Without using any other information from the bitstream, and with the same processing order like video decoding, the proposed algorithm provides platform friendliness to be integrated into the video decoder.

Besides the four feature points defined in the paper for δ using the composite model, other feature points will be further investigated by introducing additional constraints, e.g., visual significance of edge impairment, to provide a decontouring effect which may adapt to human perception. Furthermore, introducing the temporal properties into the decontouring model is also under investigation. Temporal correlation in video content can be introduced into artifacts detection to accelerate the decision. Introducing the temporal properties of human visual system into the decontouring model is also an interesting direction of future research to enhance temporal consistency in the processed video, while the tradeoff between introducing new features and the increase in algorithm complexity is also needed to be considered. The DC dithering idea also shows the potential in being integrated into the quantizer of the image/video en/decoder to improve the subjective quality concurrently, which will also be investigated further.

REFERENCES

[1] *JPEG Standard*, (JPEG ISO/IEC 10918-1 ITU-T Rec. T.81).

[2] *Generic Coding of Moving Pictures and Associated Audio Inform.-Part 2: Video*, ITU-T and ISO/IEC JTC1 ITU-T Rec. H.262 and ISO/IEC 13818-2 (MPEG-2), Nov. 1994.

[3] *Draft ITU-T Recommendation and Final Draft International Standard of Joint Video Specification*, (ITU-T Rec. H.264/ISO/IEC 14 496-10 AVC), Mar. 2003.

[4] *Information Technology-Advanced Audio Video Coding Standard Part 2: Video*, AVS-P2 Standard Draft, Mar. 2005.

[5] P. List, A. Joch, J. Lainema, G. Bjontegaard, and M. Karczewicz, "Adaptive deblocking filter," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 13, no. 7, pp. 614–619, Jul. 2003.

[6] S. Daly and X. Feng, "Decontouring: Prevention and removal of false contour artifacts," in *Proc. SPIE-IS&T Electron. Image, SPIE*, 2004, vol. 5292, pp. 130–149.

[7] T. Chen, H. R. Wu, and B. Qiu, "Adaptive post-filtering of transform coefficients for the reduction of blocking artifacts," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 11, no. 5, pp. 594–602, May 2001.

[8] S. Liu and A. C. Bovik, "Efficient DCT-domain blind measurement and reduction of blocking artifacts," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 12, no. 12, pp. 1139–1149, Dec. 2002.

[9] L. G. Roberts, "Picture coding using pseudo-random noise," *IRE Trans. Inf. Theory*, vol. IT-8, no. 2, pp. 145–154, Feb. 1962.

[10] S. Daly and X. Feng, "Bit-depth extension using spatiotemporal microdither based on models of the equivalent input noise of the visual system," in *Proc. SPIE*, 2003, vol. 5008, pp. 455–466.

[11] R. W. Floyd and L. Steinberg, "An adaptive algorithm for spatial grayscale," *Proc. Soc. Inf. Display*, vol. 17, no. 2, pp. 75–77, 1976.

[12] V. Ostromoukhov, "A simple and efficient error-diffusion algorithm," in *Proc. SIGGRAPH ACM Comput. Graph., Annu. Conf. Ser.*, 2001, pp. 567–572.

[13] G. Joy and Z. Xiang, "Reducing false contours in quantized color images," *Comput. Graph.*, vol. 20, no. 2, pp. 231–242, 1996.

[14] J. W. Lee, B. R. Lim, R.-H. Park, J.-S. Kim, and W. Ahn, "Two-stage false contour detection using directional contrast and its application to adaptive false contour reduction," *IEEE Trans. Consum. Electron.*, vol. 52, no. 1, pp. 179–188, Feb. 2006.

[15] H.-R. Choi, J. W. Lee, R.-H. Park, and J.-S. Kim, "False contour reduction using directional dilation and edge-preserving filtering," *IEEE Trans. Consum. Electron.*, vol. 52, no. 3, pp. 1099–1106, Aug. 2006.

[16] W. Ahn and J.-S. Kim, "Flat-region detection and false contour removal in the digital TV display," in *Proc. IEEE Int. Conf. Multimedia Expo*, 2005, pp. 1338–1341.

[17] J. M. Boyce and A. M. Tourapis, "Comfort noise for compressed video," in *Proc. ICCE*, Jan. 2005, pp. 323–324.

[18] S. Bhagavathy, J. Llach, and J. Zhai, "Multiscale probabilistic dithering for suppressing contour artifacts in digital images," *IEEE Trans. Image Process.*, vol. 18, no. 9, pp. 1936–1945, Sep. 2009.

[19] K. Yoo, H. Song, and K. Sohn, "In-loop selective processing for contour artifact reduction in video coding," *Electron. Lett.*, vol. 45, pp. 1020–1022, Sep. 2009.

[20] Digital Image Processing 3rd Edition-Book Images Downloads. [Online]. Available: http://www.imageprocessingplace.com/DIP-3E/dip3e_book_images_downloads.htm

[21] B. Horn and B. Schunck, "Determining optical flow," *Artif. Intell.*, vol. 17, pp. 185–204, Feb. 1981.

[22] M. Lee, S. Nepal, and U. Srinivasan, "Role of edge detection in video semantics," in *Proc. Pan Sydney Workshop Vis. Inf. Process. (VIP2002)*, Sydney, Australia, 2003, pp. 59–68, Conf. Res. Practice Inf. Technol..

[23] X. Jin, S. Li, and K. N. Ngan, "Platform-independent MB-based AVS video standard implementation," *Signal Process.: Image Commun.*, vol. 24, pp. 312–323, Apr. 2009.

Xin Jin (S'03–M'09) received the M.S. degree in communication and information system and the Ph.D. degree in information and communication engineering, both from Huazhong University of Science and Technology, Wuhan, China, in 2002 and 2005, respectively.

She is a Visiting Lecturer with the Information Technology Research Organization, Waseda University, Fukouoka, Japan. From 2006 to 2008, she was a Postdoctoral Fellow with The Chinese University of Hong Kong. From 2004 to 2005, she was an Intern with the Internet Multimedia Group, Microsoft Research Asia, Beijing, China. Her current research interests include power-rate-distortion theory, power-constrained video processing and compression, and multimedia applications. She has published over 50 conference and journal papers.

Dr. Jin was a member of the Audio Video Standard Workgroup of China (AVS) for the development of the video coding standard and received AVS Outstanding Contributor Award of year 2004. She is a member of SPIE and ACM. She received the ISOCC Best Paper Award in 2010.

Satoshi Goto (M'77–SM'84–F'87–LF'11) received the M.S. degree and Doctoral degree in electronics and communication engineering from Waseda University, Fukuoka, Japan, in 1970 and 1981, respectively.

He is currently a Professor with the Graduate School of Information, Production, and Systems, Waseda University, Kitakyushu, Japan. He joined NEC Laboratories in 1970, where he worked for LSI design, Multimedia System and Software as GM and Vice President. He has published seven monographs, 80 journal papers and over 250 international conference papers in the areas of LSI design methodology and multimedia processing. His current research interest is system LSI design methodology and new design tools for multimedia processing, networking functions with security or cryptography for the Next Generation Internet or Super Internet.

Prof. Goto received a number of awards and honors, including Distinguish Achievement Awards from IEICE, the Best Paper Award from ICCC and Jubilee Medal from the IEEE. He was a member of the Board of Director of the IEEE CIRCUITS AND SYSTEMS. He has served on numerous conference committees, e.g., as General Chair and Program Chair of ICCAD, General Chair of ASPDAC, ASICON, VLSI-DAT, and executive committee member of DAC and ISCAS. He is an IEICE Fellow.

King Ngi Ngan (M'79–SM'91–F'00) received the Ph.D. degree in electrical engineering from the Loughborough University, Leicestershire , U.K.

He is currently a Chair Professor at the Department of Electronic Engineering, The Chinese University of Hong Kong. He was previously a Full Professor at Nanyang Technological University, Singapore, and the University of Western Australia, Crawley. He holds honorary and visiting professorships of numerous universities in China, Australia, and South East Asia.

Prof. Ngan is an associate editor of the *Journal on Visual Communications and Image Representation*, as well as an area editor of the *EURASIP Journal of Signal Processing: Image Communication*, and served as an associate editor of the IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS FOR VIDEO TECHNOLOGY and *Journal of Applied Signal Processing*. He chaired a number of prestigious international conferences on video signal processing and communications, and served on the advisory and technical committees of numerous professional organizations. He was a general co-chair of the IEEE International Conference on Image Processing (ICIP) held in Hong Kong in September 2010. He has published extensively including three authored books, five edited volumes, over 300 refereed technical papers, and edited nine special issues in journals. In addition, he holds ten patents in the areas of image/video coding and communications. He is a Fellow of IET (U.K.) and IEAust (Australia), and was an IEEE Distinguished Lecturer in 2006–2007.