Boundary-energy sensitive visual de-blocking for H.264/AVC coder

Sheng Ho Wang, Sung-Wen Wang, Yi-Chin Huang, Yi-Shin Tung, Ja-Ling Wu

The Communications and Multimedia Lab., Dept. CSIE, National Taiwan University

ABSTRACT

Finding out the better parameter set (*OffsetA* and *OffsetB*) for conducting the de-blocking process of H.264/AVC, is capable of improving visual quality, said eliminating the resultant blocking artifact. Identifying which edges belong to blocking regions relies on the perceptual judging process of human beings. In fact, this subjective assessment may not exactly match existing objective assessing measurements, and the meaning of high PSNR does not always stand for less blocking artifacts. In this paper, we first introduce a new criterion for measuring the block boundary distortion by comparing the source video and the reconstructed video prior to the deblocking process. By jointly optimizing the objective picture quality and the blocky energy, the deblocking parameters decision process can find out a good balance between signal matching and blocky elimination, and therefore, maximize the effect of the built-in deblocking process. In our experiments, the proposed method can efficiently pick a better deblocking parameter set from all 169 possibilities for each coded frame and result in a better visual quality.

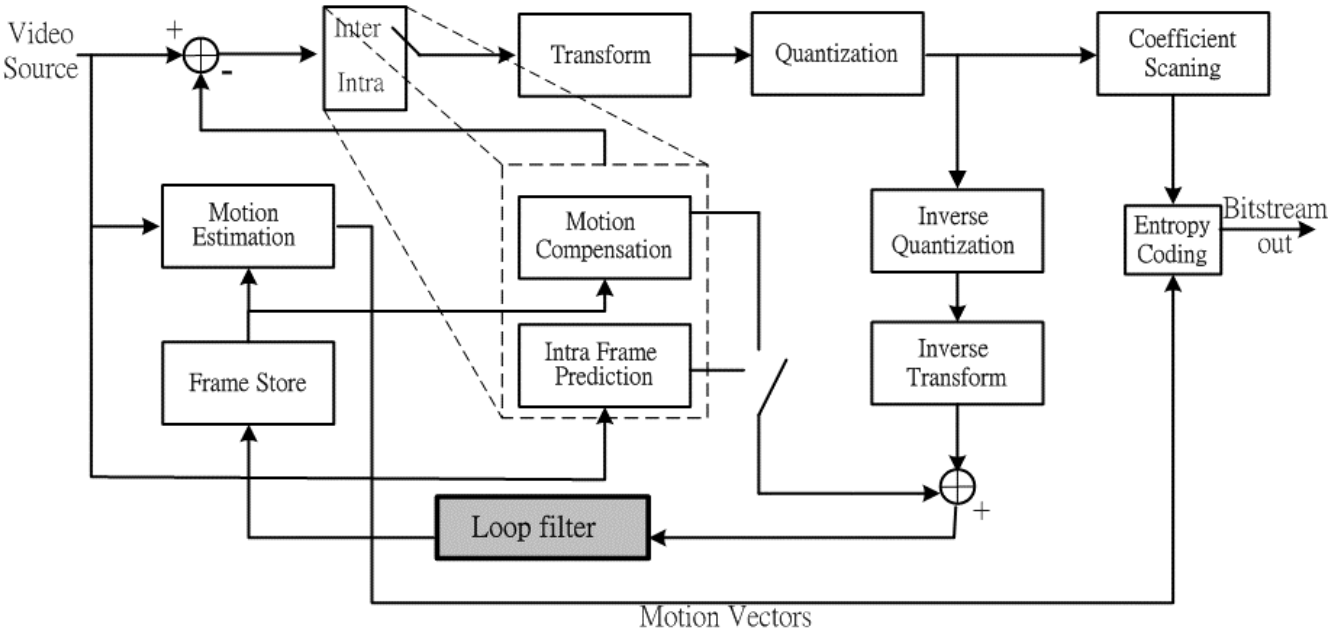


Figure 1: The block diagram of hybrid video coder, H.264/AVC. The loop filter is an additional new tool as compared with other previous coding standards.

1. INTRODUCTION

ISO/IEC MPEG and ITU-T VCEG co-organized a working group, called Joint Video Team (JVT), which aims to define a new video coding standard. This task is just completed in March 2003 and finally comes out the new standard, H.264/AVC [1]. During the development process, a lot of new advances are considered, and some elaborate combinations of these tools are conducted empirically. Finally, only those novel but computation-efficient coding tools are adopted. Generally speaking, although the decoding framework of H.264/AVC is analogous to all previous hybrid video coders, the whole process changes dramatically after integrating the new advances into different functional blocks. These new advances include 4×4 block transform, spatial prediction, variable block size motion compensation, long term motion compensation, context-adaptive VLC, context-adaptive arithmetic coding and so on. As a result,

H.264/AVC improves the coding performance a lot. It is a general belief that H.264/AVC outperforms the previous standards such as MPEG-2, MPEG-4 Visual [2] and H.263 and can save about 50% bitrate at the same visual quality [3]. The block diagram of the H.264/AVC decoding process is shown in Figure 1, and the only major change is adding an adaptive de-blocking filter inside the coding loop.

As abovementioned, H.264/AVC, as well as traditional standards, are all block-based coding. In this coding scheme, independent block-based coding can probably lead to noticeable discontinuities between the block boundaries of the reconstructed image or video (a.k.a. "blocking" artifact). Actually, it is the most annoying artifact incurred by block-based coding. Therefore, various post-processing and pre-processing algorithms [4] are proposed to eliminate the blocking artifact such as overlapped motion compensation (OBMC) [5], lapped orthogonal transform (LOT) [6], projection onto convex sets (POCS) [7], maximum a posterior (MAP) [8], block boundary filtering [9], filtering in transform domain [10]. While some video coding standards regard the deblocking filter as an optional post-processing, H.264/AVC makes it mandatory. The reason H.264/AVC forces de-blocking filter within the coding loop is to guarantee visual quality at a certain level. Furthermore, the de-blocking filter adopted by H.264/AVC provides two control variables, allowing it to flexibly alter the filter strength or even disable the filter. This paper aims to devise a method for determining the two variables, with an emphasis on subjective issues. The organization of this paper is as follows. Section 2 briefly explains why blocking artifact comes up. Next, before starting our work, we introduce the adaptive de-blocking filter used in H.264/AVC in Section 3. The approach we proposed to optimize video quality under the deblocking filtering is described in Section 4. The experimental results and some related discussions are provided in Section 5. Finally, Section 6 concludes this write-up and points out the directions of our future research.

2. BLOCKING EFFECTS

The phenomenon of artificial discontinuity across the block boundaries on the reconstructed image or video is so-called blocking effects. There are two possible causes for the blocking effects. First, during the block-based coding process, the frame is divided into several non-overlapped blocks, and each is transformed into frequency domain and then quantized into several quantization bins. Usually, neighboring pixels within a nature scene are highly correlated. If they are separated into two different blocks and then quantized independently, a small difference of the original pixel pair possibly makes them falling into different bins. In other words, the quantization magnifies the error terms in some cases. After accumulating all the effects from different frequencies, a noticeable discontinuity arises. Second, more seriously, in the inter coding referenced blocks are compensated from different portions of previous frames. Inevitably, the discontinuity between adjacent motion blocks arises. The current motion estimation algorithm applying equal weight to all pixels enlarges the boundary mismatch. As a result, the annoying blocking effect becomes more obvious in this case.

To retain the inter-block correlations, i.e. to eliminate blocking effect, there are two principal classes of methods. One is adopting the overlapped blocks as the basic coding unit in the encoding process, for motion compensation [5] or transform [6] or both. Another is applying the post-processing, which finds out the occurrences of blocking edges and then eliminates them by using signal processing techniques. Generally speaking, the technique of using overlapped blocks usually has higher complexity, and thus, adds a lot of implementation costs to both encoder and decoder. Further, this approach, such as OBMC and LOT, usually changes the decoding process dramatically, which is not desired if we want to make the decoding structure unchanged. On the other hand, post-processing can be treated as an independent task outside the coding process, which means it is usually optional and not specified in the standard decoding process. MPEG-1, MPEG-2 and MPEG-4 Visual are examples. Another advantage for the choice is that it is free for implementers to apply different approaches to reflect the state-of-art deblocking technologies and adapt to the available computation power. In the literature, various methods are applicable for removing the blocking effect. They are projection onto convex sets (POCS) [7], maximum a posterior (MAP) [8], block boundary filtering [9] and filtering in transform domain.

Although the deblocking filter applied outside the coding loop can be free from too many computation requirements and has largest extent of freedom to adopt different algorithms, it cannot guarantee the reconstructed video free from annoying blocking effect. To retain high coding performance, it becomes necessary for codec design to consider the inclusion of deblocking. As a result, the latest codec H.264/AVC finally decides to perform a content-adaptive deblocking filter, which is thought of the lowest computation among all applicable approaches. The new standard

enforces the deblocking inside the coding loop, immediately after frame reconstruction. Through this way, it also ensures all motion compensated references are used after removing blocking effect

3. ADAPTIVE DEBLOCKING OF H.264/AVC

In order to eliminate the blocking effect and enhance the compression efficiency, H.264/AVC puts the deblocking filter inside the main coding loop. With adopting the in-loop deblocking filter, the encoder takes the deblocked frame as reference in the motion estimation process. Comparing to the frame before applying the deblocking filter, the frame after performing the filtering process is not only closer to the original frame but also contains less blocking artifacts. Thus, the improvement in quality used in inter-frame prediction can make the estimated motion vector more accurate, hereafter. Basically, the 4×4 block is the minimal processing unit of block prediction and block transform. The deblocking process should scan all edges of 4×4 blocks within each macroblock so as to smooth out all occurrences of artificial blocking boundaries. Additionally, according to the design, the adopted deblocking filter is adaptive to the coding condition and content properties, either in filtering type or its strength. To understand the whole deblocking process, we need to view it from three hierarchical levels: the slice level, the block-edge level and finally the sample level, as described below.

3.1 The slice level

The parameters in the slice level delineate the global characteristics of the current video slice. In H.264/AVC, a frame may be split into one or several slices, and each has its own characteristics. The adaptability of H.264/AVC deblocking filter is achieved by adjusting encoder-selectable offsets, referred to as *OffsetA* and *OffsetB*. These offsets are both even numbers within the range from -12 to 12, inclusive, and are used to adjust the filter strength. Since the fixed filtering operation may under- or over-smooth the reconstructed frame, finding a good parameter set, i.e. a pair of (*OffsetA*, *OffsetB*), can help to regulate the filtering thresholds manually, and therefore optimizes the subjective quality. As a result, how to classify similar macroblocks into the same slice and find out the best parameter set is the major work for bringing the deblocking to its maximal effect. Moreover, by using the tool, flexible macroblock ordering (FMO), it is possible to segment different objects and then code them in each independent slice, which provides the chance for further improvements.

3.2 The block-edge level

At this level, for each edge between two adjacent 4×4 blocks, a Boundary-Strength (*Bs*) parameter is assigned an integer value from 0 to 4 according to some pre-defined criteria at the block or macroblock level. For example, is the neighboring block intra- or inter-coded, is the edge a macroblock edge, does the block have coded residuals, and are their motion vectors and reference frames the same? The detail description for deciding the Boundary-Strength parameter is described in Table 1. In the filtering implementation, *Bs* determines the filter strength performed on the edge: *Bs* with 0 implies no filtering applied, while *Bs* equal to 4 allows the strongest filtering. In addition, the deblocking filter is also dependent on the average quantization parameter (QP) of two blocks adjacent to the edge. With a larger QP, the filtering strength is likely to be stronger to smooth out the large block discontinuity incurred in the low-quality reconstruction; while with a smaller QP the visual quality of the decoded frame is guaranteed above a certain extent, thus the deblocking filter is applied less probably and with less strength.

Table 1. The coding mode based decision of the Boundary-Strength (Bs).

Block modes and conditions	*Bs*
One of the blocks is Intra and the edge is a macroblock edge	4
One of the blocks is Intra	3
One of the blocks has coded residuals	2
Difference of block motion $\geqq$ 1 luma sample distance	1
Motion compensation from different reference frames	1
Otherwise	0

Table 2. The relationships between enabling flags and pixels to be filtered, when (a) Bs = 1 to 3 and (b) Bs = 4, where "**T**", "**F**" and "-" stands for that the specified flag must be "true", "false" and "don't care", respectively. The filter coefficients are shown in the last column.

(a)

Pixel	Flag1	Flag2	Flag3	Flag4	Flag5	CFlag	Filter Coefficient $[p_2,p_1,p_0,q_0,q_1,q_2]$
p_0	T	T	T	-	-	-	$[0,\frac{1}{8},\frac{1}{2},\frac{1}{2},-\frac{1}{8},0]$
q_0	T	T	T	-	-	-	$[0,-\frac{1}{8},\frac{1}{2},\frac{1}{2},\frac{1}{8},0]$
p_1	T	T	T	T	-	T	$[\frac{1}{2},0,\frac{1}{4},\frac{1}{4},0,0]$
q_1	T	T	T	-	T	T	$[0,0,\frac{1}{4},\frac{1}{4},0\frac{1}{2}]$

(b)

Pixel	Flag4	Flag5	Flag6	CFlag	Filter Coefficient $[p_3,p_2,p_1,p_0,q_0,q_1,q_2,q_3]$
p_0	T	-	T	T	$[0,\frac{1}{8},\frac{1}{4},\frac{1}{4},\frac{1}{4},\frac{1}{8},0,0]$
p_1	T	-	T	T	$[0,\frac{1}{4},\frac{1}{4},\frac{1}{4},\frac{1}{4},0,0,0]$
p_2	T	-	T	T	$[\frac{1}{4},\frac{3}{8},\frac{1}{8},\frac{1}{8},\frac{1}{8},0,0,0]$
q_0	-	T	T	T	$[0,0,\frac{1}{8},\frac{1}{4},\frac{1}{4},\frac{1}{4},\frac{1}{8},0]$
q_1	-	T	T	T	$[0,0,0,\frac{1}{4},\frac{1}{4},\frac{1}{4},\frac{1}{4},0]$
q_2	-	T	T	T	$[0,0,0,\frac{1}{8},\frac{1}{8},\frac{1}{8},\frac{3}{8},\frac{1}{4}]$
p_0	-	-	F	-	$[0,0,\frac{1}{2},\frac{1}{4},0,\frac{1}{4},0,0]$
	F	-	-	-	
	-	-	-	F	
q_0	-	-	F	-	$[0,0,\frac{1}{4},0,\frac{1}{4},\frac{1}{2},0,0]$
	-	F	-	-	
	-	-	-	F	

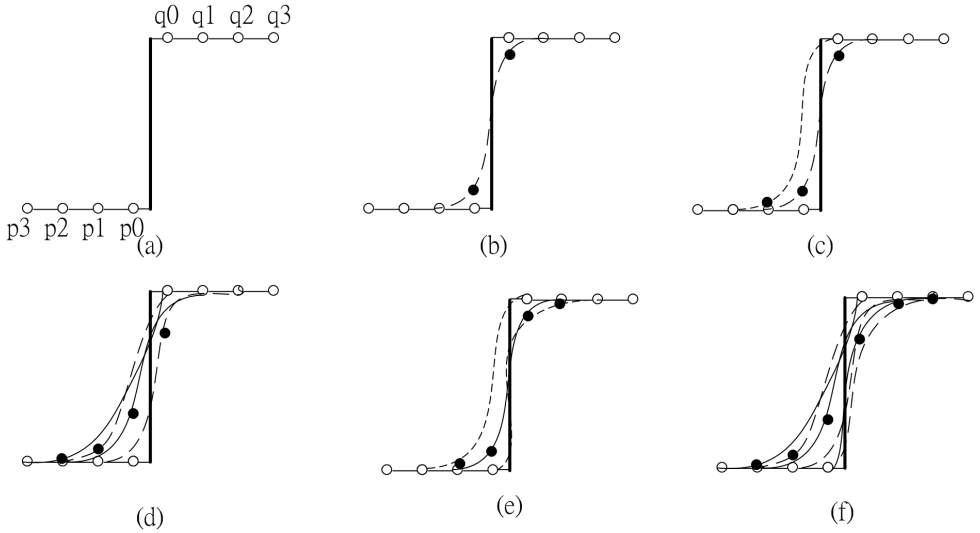


Figure. 2. **(a)** When Boundary-Strength (*Bs*) is 0, no pixel at both sides will change. **(b)** The nearest pixel next to the edge at both sides may change, while *Bs* ranges from 1 to 4. **(c~f)** The second (and third) nearest pixel(s) may change at one side or both sides while *Bs* is 1 to 3 (4). There are at most 4 and 6 pixels changed for *Bs* is equal to 1~3 and 4, respectively.

3.3 The sample level

After deciding the strength of each edge, the next step is to dynamically enable the chosen filters, which are applied to samples across every 4×4 block boundary. For each set of boundary pixels, it first analyzes the discontinuity across the block boundaries, and then judges that the discontinuity is the blocking artifact or the real object/texture edge. This help

to reduce the visibility of the artificial edges and, at the same time, preserve the sharpness of the object edges. The judgment is accomplished by through checking absolute difference between several pairs of samples across the block boundary, as formulated in Eqns. 1 to 7 where p_0, p_1, p_2, q_0, q_1 and q_2 are sample values inside the two neighboring 4×4 block and threshold values will be explained in the following paragraph. If the differences of all pixel pairs are smaller than the corresponding threshold, as shown in Table 2, this boundary is marked as an artificial edge and filtered based on the Boundary-Strength (*Bs*) determined at the block-edge level. Otherwise, it is marked as an object edge and no smoothing operation will be applied. 9 possible deblocking situations with 0 to 6 pixel changes are shown in Figure 2.

$$Flag1 = Abs(p_0 - q_0) < \alpha(Index_A) \quad (1)$$

$$Flag2 = Abs(q_1 - q_0) < \beta(Index_B) \quad (2)$$

$$Flag3 = Abs(p_1 - p_0) < \beta(Index_B) \quad (3)$$

$$Flag4 = (Abs(p_2 - p_0) < \beta(Index_B)) \quad (4)$$

$$Flag5 = (Abs(q_2 - q_0) < \beta(Index_B)) \quad (5)$$

$$Flag6 = (Abs(p_0 - q_0) < ((\alpha >> 2) + 2) \quad (6)$$

$$cFlag = Is\ chromaEdgeFlag\ Zero? \quad (7)$$

The determination of the thresholds is crucial since it directly affects the enabling flag. The values of the thresholds specified in H.264/AVC are functions of *IndexA* and *IndexB*, and the values of α and β are defined approximately according to Eqns. 8 and 9. In fact, *IndexA* and *IndexB* are decided by the value of the quantization parameters (*QP*) and the slice level parameters, *OffsetA* and *OffsetB*, as shown by Eqns. 10 and 11.

$$\alpha(Index_A) = 0.8(2^{Index_A/6} - 1) \quad (8)$$

$$\beta(Index_B) = 0.5 \times Index_B - 7 \quad (9)$$

$$Index_A = Min(Max(0, QP + Offset_A), 51) \quad (10)$$

$$Index_B = Min(Max(0, QP + Offset_B), 51) \quad (11)$$

4. BOUNDARY-EN ERGY SENSITIVE DEBLOCKING

4.1 Maximize effects of content-adaptive deblocking filter

Based on the design of H.264/AVC deblocking filter, the encoder can filter individual frames or individual parts of a frame (coded in a slice) with different settings. The encoder can adjust the values of *OffsetA* and *OffsetB* in the slice level to control the filtering enabling and to vary filter strengths. Since each frame has different texture/color characteristics and will be coded under different bitrate budgets, applying the same thresholds throughout the whole video sequence as the reference software [11] did will not bring the in-loop deblocking process into its full play. Consequently, an algorithm to explore the most adequate threshold offsets is desired, which can refine the decoded frame to the one having the best visual quality. The proposed algorithm provides an efficient way for finding out the most suitable *OffsetA* and *OffsetB* for each individual frame (or slice) that optimizes the resultant objective quality or minimizes the perceived blocking energy or both

4.2 The proposed algorithm

Because the blocking effect results from the discontinuity across the block boundaries, investigating the energy of block boundary mismatch will help to determine the parameters (*OffsetA*, *OffsetB*). In H.264/AVC, it allows these two offsets to be any even numbers ranging from -12 to 12 inclusive, and as a result, there are totally 169 (13×13) combinations. It is a time consuming work if the encoder tries all possibilities for obtaining the best pair of (*OffsetA*, *OffsetB*). To avoid it, the proposed algorithm translates the original problem into a search problem in a 2-dimensional space, with *OffsetA* as one axis and *OffsetB* as the other, and finds the best pair of (*OffsetA*, *OffsetB*) based on a specific cost function. The exploration of the cost function, which reflects the perceived visual quality, and the detail of the search algorithm are described in Sections 4.2.1 and 4.2.2, respectively.

4.2.1 The discontinuity energy of block boundary mismatch

In order to determine the best deblocking parameters, a criterion function for representing the frame's fineness is required. Although the overall distortion energy, at a certain extent, represents a frame's quality, it is not effective for the case that the certain type of error patterns is known in advance. For example, two reconstructions of the same image may have similar PSNR values but one may suffer from obvious blocking artifact while the other does not. Thus, in our application, we need to define a measurement, *blocking_degree* (*BD*), for representing how serious the blocking effect is perceived. By analyzing the distortion frame (*D*), the block discontinuity energy is calculated by summing differences of those pixel pairs, which belong to boundary pixels and are filtered by H.264/AVC, as formulated below.

$$BD = \frac{\sum_{i=0}^{height-1}\sum_{j=0,j\%4==0}^{j=width-4}\{[D(i,j)-D(i,j-1)]^2\times(1-\delta(Bs))\}+\sum_{j=0}^{j=width-1}\sum_{i=0,i\%4==0}^{i=height-4}\{[D(i,j)-D(i-1,j)]^2\times(1-\delta(Bs))\}}{\sum_{i=0}^{height-1}\sum_{j=0,j\%4==0}^{j=width-4}(1-\delta(Bs))+\sum_{j=0}^{j=width-1}\sum_{i=0,i\%4==0}^{i=height-4}(1-\delta(Bs))} \quad (13)$$

where *D* is obtained by subtracting the original frame (*I*) from its reconstruction (*U*) for every column and row index *i* and *j*, and δ is the delta function as shown in Eqns. 14 and 15.

$$D(i,j) = U(i,j) - I(i,j) \quad (14)$$

$$\delta(Bs) = \begin{cases} 1, & Bs = 0 \\ 0, & others \end{cases} \quad (15)$$

Note that *D*(*i*, *j*)-*D*(*i*, *j*-1) denotes the neighboring distortion gradient, and a large value of this term implies apparent blocking effect. Hence, *blocking_degree* is obtained by computing the gradient difference energy across block boundaries, and the larger it is, the more perceptible the blocking effect will be.

As prescribed in Section 2, the visual deblocking by H.264/AVC at most smoothes out 6 pixels across each boundary location, and correctly detecting the occurrence of blocking boundaries can reduce mismatch energy. Thus, mean square error representing the average error signal energy is also considered in our criterion function. The final criterion used throughout this work is the *weighted_blocking_degree* (*WBD*), which is a linearly weighted result of *BD* and *MSE* as shown in Eqn. 17. The value of γ is set to 0.7 empirically.

$$MSE = \sum_{i=0,j=0}^{i=height-1,\,j=width-1} D^2(i,j) \Big/ height \times width \quad (16)$$

$$WBD = \gamma \times BD + (1.0 - \gamma) \times MSE \quad (17)$$

4.2.2 Predicted diamond search (PDS)

PDS is a predictor-centric coarse-to-fine search method. The searching procedure is composed of four steps as shown in Figure 3. They are start-point initialization, large diamond search, small diamond search and linear search on OffsetA-axis in order.

In the first step, start-point initialization, the initial value of (*OffsetA*, *OffsetB*) is set to that of the slice at the same location of the previous frame. For the case of the first frame or the scene-change frame, (0, 0) is adopted instead. Then, the initial search point is treated as the center point of the large diamond search pattern. The concept of the large diamond search and the small diamond search are similar. First, the search is conducted on all points of the search pattern. The searching procedure iterates one to several times, which moves its center to the point having the optimal cost until the center of the searching pattern is the optimal among all neighbors. And in the follow-up, small diamond is applied for finer adjustment. The search patterns of the large and small diamond search are illustrated in Figure 4.

Finally, after the diamond search, linear search on *OffsetA*-axis is conducted. Since H.264/AVC deblocking filter is more sensitive to the variation of *OffsetA* than that of *OffsetB*, the proposed algorithm linearly searches all possible

values for *OffsetA* along the fixed *OffsetB* value, which prevents the search from falling into local optimal. After the linear search on *OffsetA*-axis, the most appropriate (*OffsetA*, *OffsetB*) for the current frame is found. It is also taken as the initial start point for the next frame.

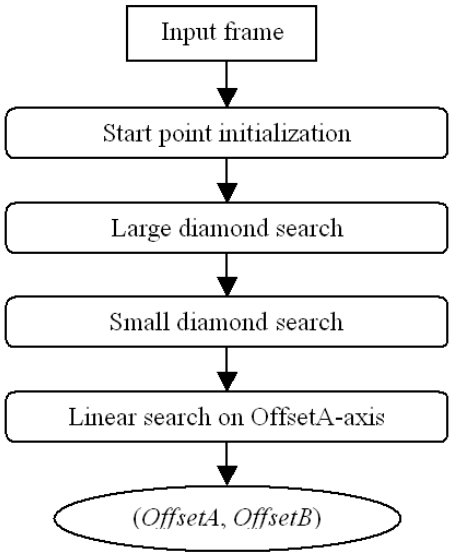


Figure 3. The steps of the proposed predicted diamond search (PDS).

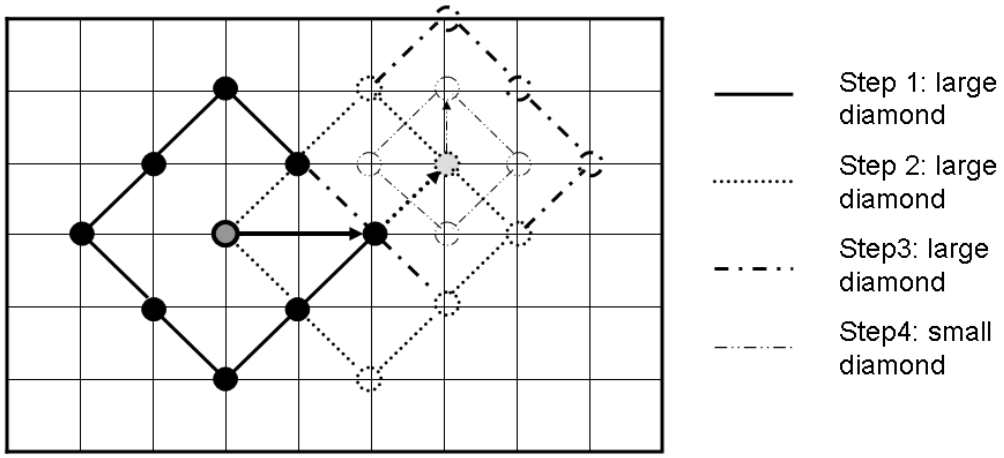


Figure 4. An example to show search patterns during the large and small diamond search employed by the PDS. The dark grey and light grey points are the center point of the large and small diamond search.

4.2.3 Predicted local squared search (PLSS)

PLSS is a predictor-constrained local search method. The search is simpler and faster than PDS, and its procedure is shown in Figure 5. In this method, if the input frame is the leading frame of a sequence or a first frame of a new scene, we apply PDS to get a precise initiation of (*OffsetA*, *OffsetB*). Otherwise, we initiate the value of (*OffsetA*, *OffsetB*) to the one copied from the previous frame. After the initiating step, we take that as a center point to apply first squared search process. The behavior of our approach can be described as follows. The surroundings of center point are viewed as candidates to search. Once finding the best value at first process we let that point as an initiate of the process of applying second squared search to find the value of (*OffsetA*, *OffsetB*). One example can be found in Figure 6. The white point is the initial value at first initial step. Black points surrounding center point are candidates to be decided at the step of first squared search process. The remaining operations of second squared search process can be classified three cases. If the best point is horizontal or vertical to central point three other points are searched at the second

squared search. Five points are considered when the best point is at the diagonal position of central point. Otherwise, it is no need to process second squared search.

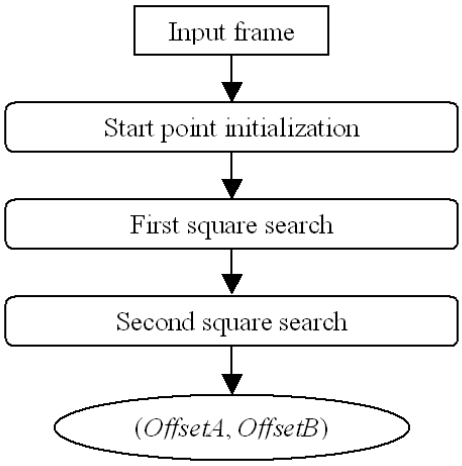


Figure 5. The steps of the proposed predicted local squared search (PLSS).

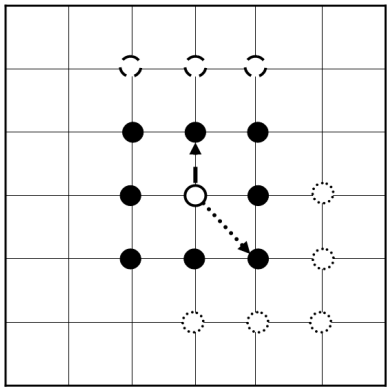
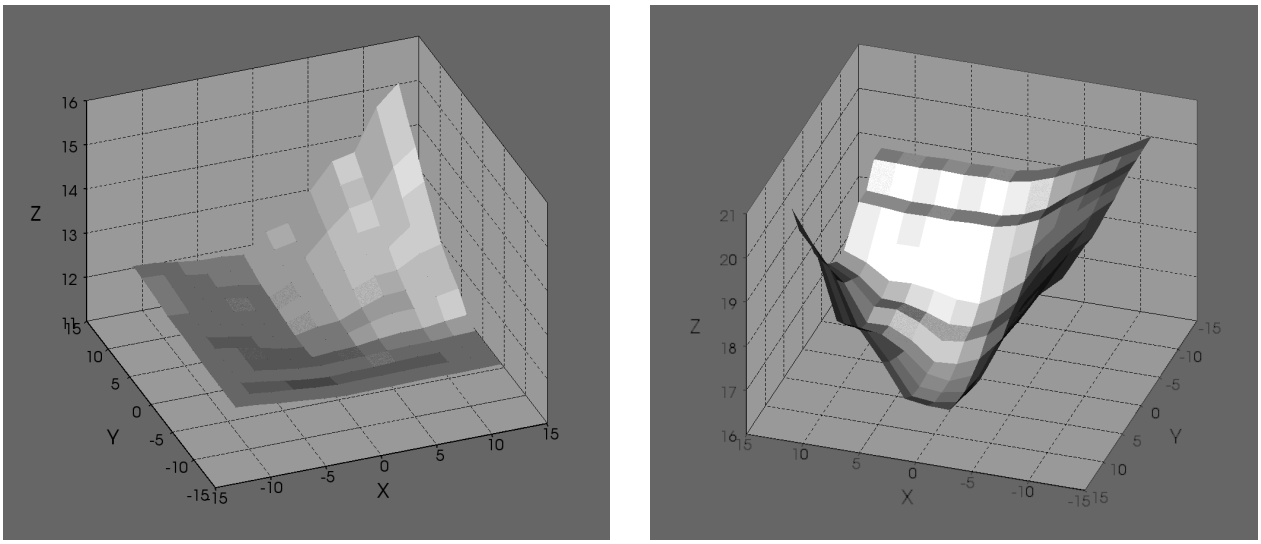


Figure 6. An example to show search patterns by applying 1~2 iterations of the squared search employed by the PLSS.

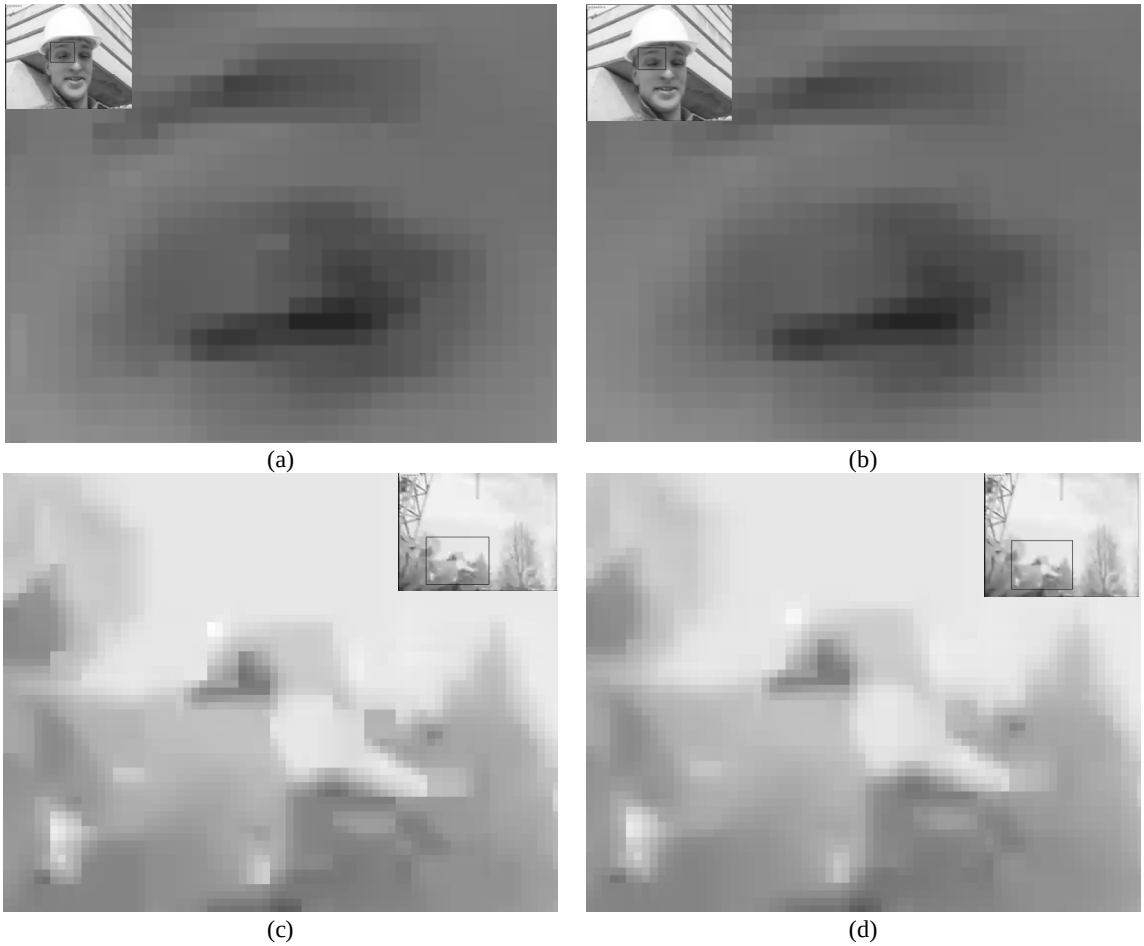


(a) (b)

Figure 7. This figure shows the values of *MSE* and *WBD* for different deblocking parameter sets (*OffsetA*, *OffsetB*). (a) *MSE*. (b) *WBD*. X axis and Y axis indicate *OffsetA* and *OffsetB*, respectively, and Z axis is the output value of the measurement functions.

5. EXPERIMENTAL RESULTS

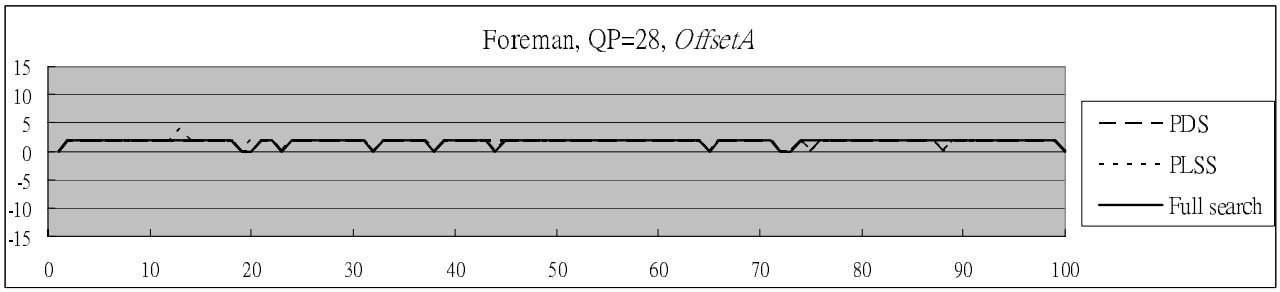
Computer simulations were conducted using the reference software of H.264/AVC, JM 8.0 [11]. The following experiments reveal that 1) using *weighted_blocking_degree* as the measurement criterion is better than just using *MSE* in subjective quality and 2) our proposed fast search schemes in the *OffsetA*-*OffserB* space performs almost the same as the exhaustive search while the spent time decreases significantly. Figure 7(a) and (b) exhibit examples of the *MSE* and *WBD* variations with different deblocking parameters, respectively. For the case of WBD, γ is set to 0.7. The result is consistent with our basic assumption that adjusting *OffsetA* and *OffsetB* can improve the reconstruction quality, and dynamically changing these parameters for different frames can make it perform better. The figure also shows that in most cases it will not trap into local optimum, so the found parameter set is surely the best. Figure 8 shows the reconstructed frames of applying the best pair of (*OffsetA*, *OffsetB*) found by using two different criteria. It is evident that the blocking effects are less noticeable when using *WBD* as the criterion. Since the complexity decides the applicability, the fast search algorithms should be employed. The performance comparison among exhaustive search, predicted diamond search, and predicted local squared search are depicted in Table 3. The test sequences are Foreman and Football sequences whose formats are QCIF and CIF, respectively. They are coded by different *QPs* for evaluating how the *QP* affects the deblocking parameters and the search results. It can be seen in Figure 9 that the three search algorithms have similar resultant *OffsetA* and *OffsetB* regardless of the value of *QP*.



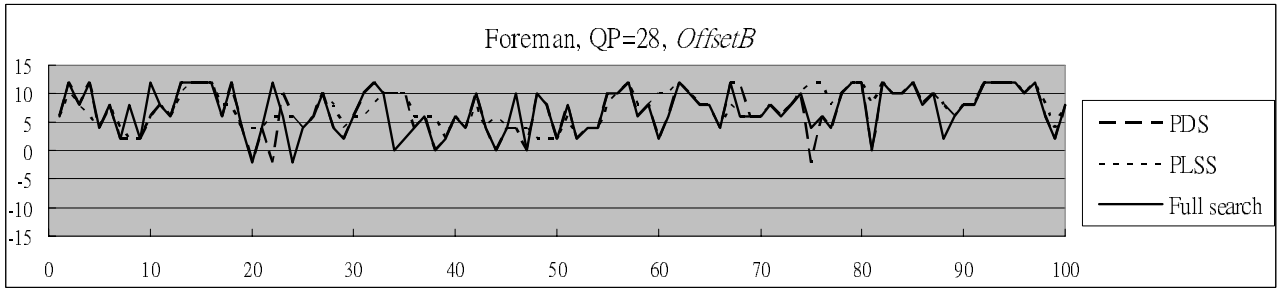
(a) (b)

(c) (d)

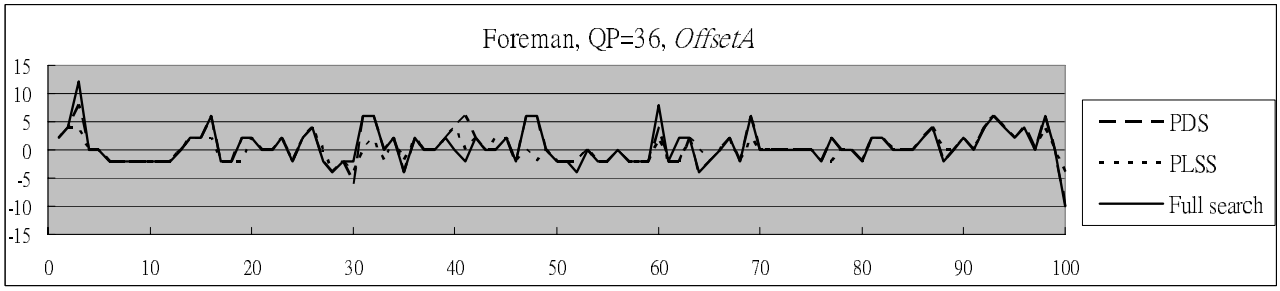
Figure 8. Regions enlarged of some reconstructed frames (QP=28) of the foreman sequence by different criterion functions. (a) Frame #0 of the foreman sequence using MSE as the criterion (*OffsetA*=-2, *OffsetB*=-2). (b) Frame #0 using *WBD* as the criterion (*OffsetA*=0, *OffsetB*=6). (c) Frame #200 using MSE as criterion (*OffsetA*=-2, *OffsetB*=-4). (d) Frame #200 using *WBD* as the criterion (*OffsetA*=2, *OffsetB*=0).



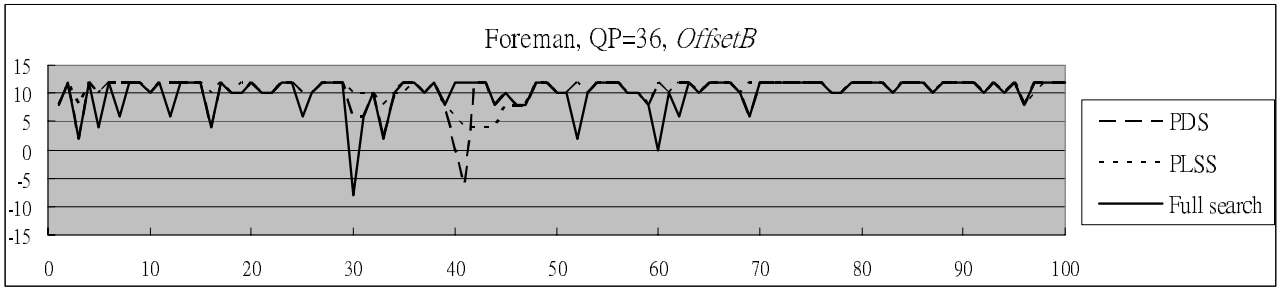
(a)



(b)



(c)



(d)

Figure 9. The figures draw the trace of the output (*OffsetA*, *OffsetB*) using three searching algorithms. (a, b) *OffsetA* and *OffsetB* for Foreman sequence coded at QP=28. (c, d) *OffsetA* and *OffsetB* for Foreman sequence coded at QP=36.

As Table 3 illustrates, the reduction in the number of search points is up to 85% compared to all 169 trials by the exhaustive search. Despite the fewer search points, the pair of (*OffsetA*, *OffsetB*) obtained from our proposed algorithms is almost identical to those from exhaustive search, except that the Forman sequence based on the PLSS algorithm has less than 60% of accuracy. Here, the accuracy stands for the percentage that the obtained pair of (*OffsetA*, *OffsetB*) is the same as the exhaustive search. Among these different best pairs between the exhaustive and fast search algorithm, we further measure the average relative distance by the formula:

$$\sqrt{(OffsetA_{full}-OffsetA_{fast})^2+(OffsetB_{full}-OffsetB_{fast})^2}\Big/diff\_num \quad (18)$$

where *diff_num* is the number of frames which are of different parameter sets in comparison with those acquired from the exhaustive search. According to Table 3, it shows the value is considerably small, which means that even some pairs of (*OffsetA*, *OffsetB*) are not identical to those obtained from exhaustive search, they are only of little difference. Moreover, the increase in *WBD* is small enough to neglect it. Again, these observations prove that our proposed algorithm can find *OffsetA* and *OffsetB* very close to those obtained from the exhaustive search, while the spent time is much less than it. As a result, our proposed algorithm achieves comparable performance with the exhaustive search. Generally speaking, PLSS has worse performance compared to PDS, but the time saving becomes further more. Thus, we may apply two search algorithms to different application scenarios, or integrate them for getting a good balance, which is one of our future works. In addition, we also compare our results with the one adopting the fixed parameter set in Table 4. There is little reduction in the objective quality assessment, PSNR. Based on the mean of *WBD*, it is undoubted that the approaches, which dynamically change the deblocking parameters can eliminate the blocking effects better than the fixed-parameter approach. As for the resultant bitrate, there is no obvious variation among all approaches. With regard to the encoding time measured in second per each frame, it increases less than 14% and 10% in PDS and PLSS, while it almost doubles in the exhaustive search, compared to the fixed-parameter approach.

Table 3. The results of the proposed search algorithms compared with the exhaustive search. The source consists of 100 frames at 30 fps. All frames are coded as P frame except for the leading I frame.

Sequence	Search algorithm	Reduction in number of search points	Increase in *WBD*	Ratio of accuracy	Average relative distance
Foreman, QCIF, QP=28	PDS	85.32%	1.2×10^{-2} %	86%	7.93
	PLSS	93.48%	7.8×10^{-2} %	57%	5.05
Foreman, QCIF, QP=36	PDS	87.04%	6.1×10^{-3} %	89%	9.36
	PLSS	94.11%	3.9×10^{-2} %	72%	6.93
Football, CIF, QP=28	PDS	85.94%	1.4×10^{-4} %	99%	6.00
	PLSS	93.33%	7.6×10^{-3} %	86%	3.00
Football, CIF, QP=36	PDS	89.71%	0%	100%	0.00
	PLSS	95.56%	3.8×10^{-4} %	95%	4.80

Table 4. The time spent of different search algorithms. Among these approaches, "Fixed" means set *OffestA* and *OffestB* equal to zero. The first 100 frames of each sequence are encoded.

	Criterion	Search algorithm	Average PSNR	Average *WBD*	Files size (Bytes)	Encoding time/frame (sec)
Foreman, QCIF, QP=28	*N.A.*	*Fixed*	36.487	17.725	52160	1.15567
	WBD	*Full Search*	36.482	16.852	52254	2.10534
		PDS	36.445	16.854	52117	1.29954
		PLSS	36.448	16.866	52220	1.22985
Foreman, QCIF, QP=36	*N.A.*	*Fixed*	31.007	47.634	19760	1.11988
	WBD	*Full Search*	30.981	47.168	19750	1.96956
		PDS	30.985	47.171	19680	1.24814
		PLSS	30.986	47.186	19817	1.18782
Football, CIF, QP=28	*N.A.*	*Fixed*	34.232	37.228	728517	4.76772
	WBD	*Full Search*	34.240	36.914	729505	9.32912
		PDS	34.240	36.914	729128	5.65504
		PLSS	34.234	36.917	728867	5.26369
Football, CIF, QP=36	*N.A.*	*Fixed*	28.456	119.364	244537	4.55422
	WBD	*Full Search*	28.428	117.584	243932	8.65531
		PDS	28.428	117.584	243932	5.15972
		PLSS	28.441	117.588	244492	4.91099

6. CONCLUSIONS

In this paper, we addressed an approach to decide the parameter set, *OffsetA* and *OffsetB*, to adaptively control the filtering strength of H.264/AVC de-blocking. We consider blocking effect sensitive Human Visual System (HVS), as well as PSNR, as the criterion to improve visual quality in both objective and subjective way. Moreover, characterized by its efficiency, our approach selects the optimal parameter without exhaustive search. The experimental results show that our approach defeats the fix-offset approach in both PSNR and subjective quality, with only introducing a little computational overhead.

ACKNOWLEDGMENTS

This work was partially supported by the National Science Council and the Ministry of Education of ROC under the contract No. NSC92-2622-E-002-002, NSC92-2213-E-002-023 and 89E-FA06-2-4-8. Additionally, we specially appreciate Ya-Ting Yang and Chia-Ying Li for their help to polish this work.

REFERENCES

1. "Text of ISO/IEC FDIS 14496-10/Draft ITU-T H.264: Information Technology – Coding of Audio-Visual Objects: Advanced Video Coding," International Organization for Standardization, 2003.
2. "Information Technology – Coding of Audio-Visual Objects, Part 2: Visual," ISO/IEC 14496-2:2001, Second Edition, International Organization for Standardization, 2001.
3. "MPEG-4 – The Media Standard, the landscape of advanced multimedia coding," White Paper by MPEG-4 Industry Forum, available via http://www.m4if.org
4. W.F.Cheung and Y.H.Chan, "Improving MPEG-4 coding performance by jointly optimizing compression and blocking effect elimination", IEE Proceedings-Vision, Image and Signal Processing, vol.148, , pp.194-201, Jun 2001.
5. Tao, B.; Orchard, M.T. "A parametric solution for optimal overlapped block motion compensation," IEEE Trans. on Image Processing, vol.10, pp.341 -350,March 2001.
6. Fong, W.C.; Chan, S.C.; Nallanathan, A.; Ho, K.L.,"Integer lapped transforms and their applications to image coding," IEEE Trans. on Image Processing, vol. 11, pp.1152-1159, Oct. 2002.
7. A. ZAKHOR, "Iterative procedures for reduction of blocking effects in transform image coding", IEEE Trans. on Circuits Syst. Video Technology, vol.2, pp.91-95, Mar 1992.
8. T.P. O'ROURKE, and R.L. STEVENSON, "Improved image decompression for reduced transform coding artifacts", IEEE Trans. on Circuits Syst. Video Technology, vol.5, pp.490-499, Dec 1995,
9. Y.H. CHAN, S.W.HONG, and W.C SIU., "A practical post-processing technique for real-time block-based coding system", IEEE Trans. on Circuits Syst. Video Technology, vol.8, , pp.4-8, Feb 1998.
10. Z. XIONG, M.T. ORCHARD, and Y.Q. ZHANG, "A deblocking algorithm for jpeg compressed images using overcomplete wavelet representations", IEEE Trans. on Circuits Syst. Video Technology, vol. 7, pp.433-437, Apr 1997.
11. "Information technology: Coding of audio-visual objects, Part 5: Reference software Amendment 6: Advanced video coding and high efficiency advanced audio coding reference software", ISO/IEC JTC 1/SC 29/WG 11 Docs. No. 6248.