A FAST TWO DIMENSIONAL DEBLOCKING FILTER FOR H.264/AVC VIDEO CODING

Mahdi Nazm Bojnordi, Mahmoud Reza Hashemi, Omid Fatemi
Multimedia Processing Laboratory, Nano-Electronics Center of Excellence
School of Electrical and Computer Engineering, University of Tehran, Tehran, Iran.
Email: m.bojnordi@ece.ut.ac.ir, hashemi@comnete.com, omid@fatemi.net

Abstract

H.264/AVC as the most recent video coding standard delivers significantly better performance compared to previous standards, supporting higher video quality over lower bit rate channels. The H.264 in-loop deblocking filter is one of the several complex techniques that have realized this superior coding quality. The deblocking filter is a computationally and data intensive tool resulting in increased execution time of both the encoding and decoding processes. In this paper and in order to reduce the deblocking complexity, we propose a new 2-D deblocking filtering algorithm based on the existing 1-D method of the H.264/AVC standard. Simulation results indicate that the proposed technique achieves a 40% speed improvement compared to the existing 1-D H.264/AVC deblocking filter, while affecting the SNR by 0.15% in average.

Keywords: *H.264/AVC, in-loop deblocking filter.*

1. Introduction

Many Video coding methods employ block-based prediction, transformation, and quantization for encoding video streams. The use of block-based processing, however, decreases inter-block correlation in video frames and adds visible blocking artifacts to the reconstructed frames [1]-[3]. In order to overcome this problem, two techniques are generally used by the current video coding algorithms: 1) avoidance strategies for reducing the occurrence of blocking artifacts, in H.263 [4] and MPEG-4 [5]; 2) artifact cleaning schemes after frame reconstruction in H.264/AVC [6].

Among various encoding tools of the H.264 standard, the in-loop deblocking filter has a significant impact on the perception quality of video [7][8]. As shown in [1], compared to post filtering algorithms, the in-loop deblocking filter reduces the bit-rate typically by 5%-10% preserving the same objective video quality. Because of the advantages of in-loop deblocking over post filtering, H.264 deblocking filter is used in the prediction loop of both the encoder and the decoder modules. However, this improvement is achieved at the cost of a large amount of computation and memory read/write operations resulting in a reduced speed of both encoding and decoding processes. The deblocking filter is described in more details in [9].

In order to present an efficient architecture for real-time applications, two problems need to be addressed:

i. filtering process latency;
ii. data memory access.

Recently, many efficient hardware implementations for deblocking filter in H.264 have been proposed [7][10]-[12]. To satisfy the real-time constraints, most existing architectures have used various fast memory accessing techniques. None of these solutions considers the algorithmic enhancements for overall speedup. On the other hand, there are many new deblocking filtering techniques in the literature that have resulted in better video quality. They are based on iterative algorithms. For instance, a non-linear loop filter algorithm has been proposed in [13] which performs an orthonormal linear transform over the decoded frame and establishes a de-noising rule for the individual transform coefficients. Experimental results show about 10% bit-rate reduction for this algorithm compared to the existing H.264 deblocking filter. Other examples of the iterative algorithms are "maximum a posteriori (MAP)" [14] and "projection onto convex sets (POCS)" [15]. Even though, the iterative methods result in better performance, they are not efficient for real-time applications or devices with low computational power. This paper follows the low-pass filtering strategy and proposes a fast 2-D deblocking algorithm for H.264/AVC. Many concepts from the existing 1-D deblocking filter are borrowed for defining the new 2-D filtering rules. A significant speedup of more than 40% is achieved at the cost of losing only 0.15% SNR in average.

The paper is organized as follows. The H.264/AVC deblocking filtering algorithm is reviewed in Section 2. Our proposed 2-D algorithm is described in Section 3. In Section 4, simulation results are presented and compared with the reference model. Finally, the paper ends with conclusion in Section 5.

2. The H.264/AVC Deblocking Algorithm

According to the last version of H.264 recommendation [6], the deblocking filtering algorithm is a conditional process that has to be applied to all $N \times N$ block boundaries of the picture. For luminance component (Y) N is selected equal to the size of the applied transform, four or eight. However, the algorithm filters the chrominance components (U and V) in terms of 4×4 blocks. Except for the picture boundary edges, the algorithm filters all the block edges inside the picture. Figure 1 shows the boundaries of three Y, U, and V macroblock data. Bold lines demonstrate the boundaries that should be filtered for both 4×4 and 8×8 transforms. The boundaries which are represented by

INTEL-1032
9,641,841

dotted lines are the boundaries that are filtered only when the transform size is 4×4.

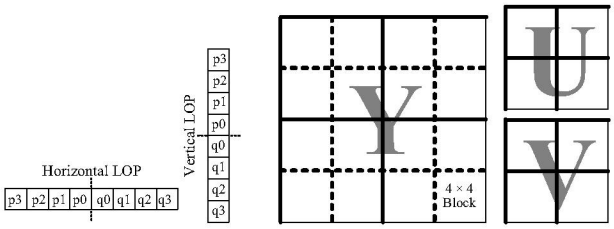

Figure 1. The macroblock's boundaries that should be filtered.

In order to filter vertical and horizontal block boundaries of a macroblock, the algorithm processes data in the form of horizontal and vertical line of pixels (LOP), respectively. Each LOP is composed of eight neighboring pixels across two 4×4 blocks, as illustrated in Figure 1. In each macroblock, the horizontal LOPs are filtered first. Then, the vertical LOPs are processed. We note that the H.264 deblocking filters are selected among a bank of adaptive low-pass filters. The proper filter is selected according to the boundary strength (bS) between the two 4×4 blocks, α and ß thresholds and the current LOP. The bS factor is determined using the procedure described in Table 1 for each two neighboring p and q blocks. a and ß depend on the quantization parameter (QP). Each LOP is filtered only if all the conditions in (1) are met.

$$bS \neq 0, |p0 - q0| < \alpha, |p1 - p0| < \beta, |q1 - q0| < \beta \qquad (1)$$

Table 1. Determining *bS* for a LOP.

Condition	*bS* value
p or *q* is intra coded and boundary is a macroblock boundary	4
p or *q* is intra coded and boundary is not a macroblock boundary	3
neither *p* or *q* is intra coded; *p* or *q* contain coded coefficients	2
neither *p* or *q* is intra coded; neither *p* or *q* contain coded coefficients; *p* and *q* have different reference frames or a different number of reference frames or different motion vector values	1
neither *p* or *q* is intra coded; neither *p* or *q* contain coded coefficients; *p* and *q* have same reference frame and identical motion vectors	0

According to the analysis in [16], 71% of the decoder's time is consumed by these three major units:

a. inverse transform and reconstruction (including inverse quantization and secondary DC transforms);
b. interpolation; and
c. deblocking filtering.

From Table 2, the deblocking unit has more impact on the consumed time compared to the other units. As presented in Figure 2, our experiments show that the existing H.264/AVC deblocking algorithm uses three major instruction types more frequently during the filtering process. According to these results, most-frequently-used operations for deblocking filtering are Add/Sub, Shift and Memory Access operations, respectively. More than 60% of the executed instructions are related to Add or Sub operations. Shift operations comprise about 20% of the instructions. Less than 17% of the executed instructions for deblocking filtering are needed for updating the pixels in the memory.

Table 2. Complexity of the various parts of H.264 decoder.

Decoder function	Complexity
Inverse Transforms and Reconstruction	13%
Interpolation	25%
Parsing and Entropy Decoding	13%
Deblocking Filtering	33%

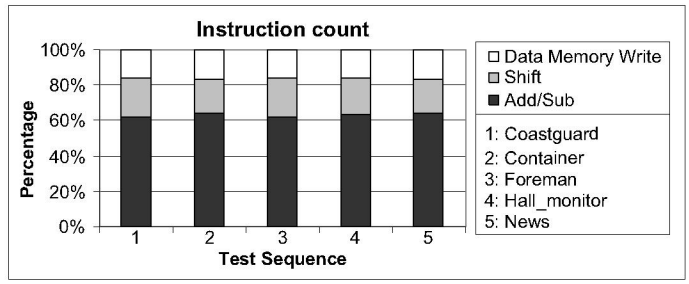

Figure 2. H.264/AVC deblocking algorithm complexity in terms of instruction count.

In order to reduce the overall execution time of the deblocking algorithm, all of these instruction types could be targeted for optimization. Of course, each instruction type has its own effects on the speed and filtering quality. In this paper we try to optimize all three cases.

3. The Proposed Two Dimensional Filter

Based on the standard deblocking algorithm, we propose a new 2-D deblocking filtering algorithm for H.264. The existing 1-D algorithm suffers from large amount of memory accesses during the filtering process. In our proposed algorithm, an 8×8 filtering window is employed resulting in decreased number of memory accesses. The filtering window is formed of an 8×8 box of pixels (BOP). In each filtering pass, a new BOP is selected by the algorithm for deblocking. BOP and a filtering window are shown in Figure 3.

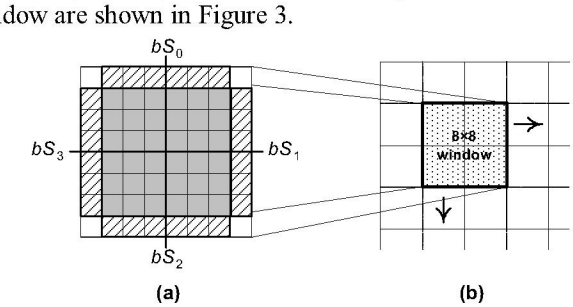

Figure 3. (a) Box of Pixels; (b) Position of the filtering window on the reconstructed frame.

In order to accelerate the filtering process, two filtering regions are considered for each BOP. As depicted in part (a) of Figure 3, each BOP is composed of two types of pixels shown as hatched and shaded pixels. The hatched pixels are updated using the LOP filtering rules with the same rules as used in the

existing H.264/AVC deblocking algorithm. For the shaded pixels, however, the LOP filtering rules are combined and new 2-D filtering rules are obtained. The existing 1-D algorithm only uses the pixels of the LOP to update each pixel. The new rules, however, use the pixels of horizontal and vertical LOPs containing the desired pixel, simultaneously. Therefore, each pixel is updated according to two bS values of horizontal and vertical LOPs. Using two bS values results in four different filtering conditions. In case of each condition, the vertical and horizontal filtering rules are combined and simplified. Applying the 2-D filtering rules results in reducing the number of redundant operations; consequently the number of needed Add/Sub and Shift instructions for deblocking filtering decreases. Each BOP is equivalent to eight vertical or horizontal LOPs. In our proposed algorithm, a maximum of 60 pixels are needed to be updated. This number is significantly less than 96 updates when the existing deblocking algorithm is applied to the same window. For the proposed algorithm reduction in all three types of instructions is considered. The proposed algorithm is shown in Figure 4.

```
procedure two_dim_filter;
  if transform size of the first macroblock is 4×4 then
    initialize the filtering window at (0, 0);
  else
    initialize the filtering window at (4, 4);
  end if;
  while the entire frame is not filtered do
    a. evaluate filtering parameters;
    b. update the hatched pixels;
    c. update the shaded pixels;
    if transform size for the current block is 4×4 then
      move the filter by 4 pixels;
    else
      move the filter by 8 pixels;
    end if;
  end while;
end two_dim_filter;
```

Figure 4. Proposed algorithm for deblocking filter.

4. Experimental Results

For our simulations, we have used the H.264/AVC reference software, JM 9.5. The proposed 2-D algorithm is implemented in software and is compared in terms of processing time with the reference model. In addition for each algorithm, the number of executed instructions for all instruction types is calculated. As presented in Figure 5, the proposed technique has reduced the number for all instruction types. The instruction count for Add/Sub, Shift and Memory Access types is reduced by more than 55%, 25% and 22%, respectively. Consequently, the total filtering execution time is reduced by the proposed algorithm. In order to compare total filtering time of our proposed algorithm with the existing standard deblocking algorithm, the algorithms are applied to several test sequences encoded at various bit-rates. As depicted in Figure 6 the proposed 2-D deblocking algorithm offers more than 40% speed improvement.

The inter-block co-relation increases at higher bit-rates since they have smaller QP. The deblocking algorithm is adapted to skip the filtering in the cases in which the inter-block co-relation is more than a predefined threshold. Experimental results show that our proposed algorithm is even more efficient than the existing algorithm for reducing the computations at higher bit-rates. Figure 7 shows typical execution time of the two algorithms applied to the test sequences encoded at various SNR qualities. The least speed improvement is achieved for the sequences encoded at lower bit-rates.

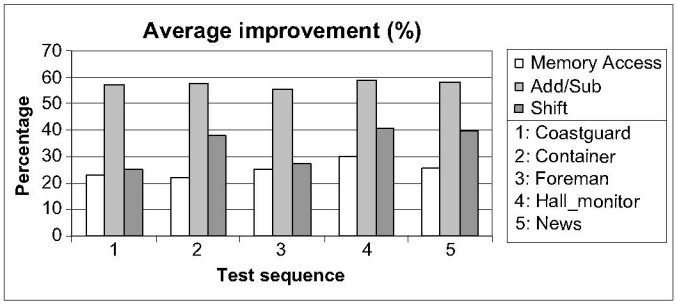

Figure 5. Improvement achieved by the proposed algorithm in contrast with the existing H.264/AVC deblocking algorithm.

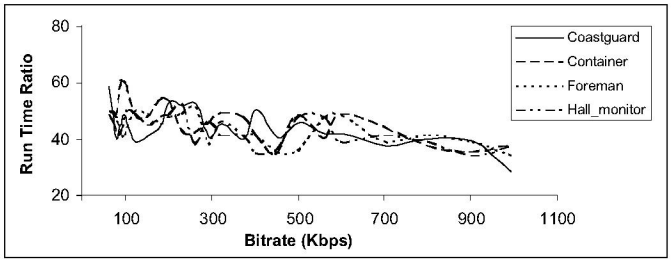

Figure 6. Run time ratio for the proposed algorithm to the H.264/AVC deblocking algorithm.

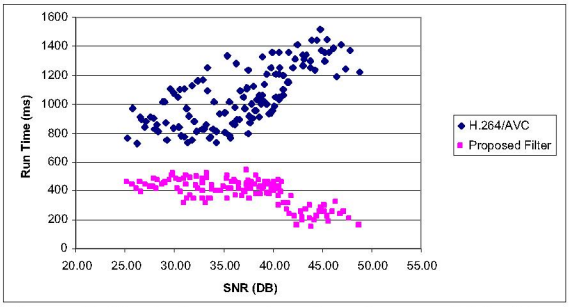

Figure 7. Typical execution time for two deblocking algorithms applied to sequences with various SNR qualities.

Detailed quality comparison between the two algorithms for five CIF test sequences is shown in Table 3. The test sequences are shown by numbers 1, 2, 3, 4 and 5 representing *Coastguard*, *Container*, *Foreman*, *Hall_monitor*, and *News*, respectively. This table shows the SNR degradation of the proposed algorithm compared to the reference model. Also,

minimum, maximum and average SNR degradation is measured for each test sequence. According to the results, the proposed algorithm offers more quality improvements, compared to the standard algorithm, for the sequences with fixed and uniform content, e.g. container. The most degradation is occurred for the input sequences containing sharp-edge objects, such as the *hall_monitor* which is encoded at 122*Kbps*. But this degradation is just 0.68% of the sequence SNR measure. In average, 0.15% SNR degradation is expected by the proposed algorithm in comparison with the H.264/AVC deblocking algorithm. In Figure 8, the result of applying the proposed 2-D algorithm to the CIF format of the foreman sequence encoded at 60*Kbps* is shown.

Table 3. Degradation in SNR of the proposed algorithm compared to the reference software model.

SNR Degradation (dB)						
Bit-rate (Kbps)	**1**	**2**	**3**	**4**	**5**	
37	0.07	0.06	-0.03	0.12	0.02	
61	0.04	-0.03	0.16	0.16	0.15	
122	-0.03	-0.11	0.09	0.24	0.11	
152	-0.01	0.01	0.1	0.1	-0.01	
202	0.01	-0.01	0.09	0.18	0.08	
262	0.05	0.04	0.04	0.11	-0.11	
292	0.04	0.01	0.07	0.1	0.06	
372	-0.01	-0.05	0.06	0.11	0.07	
402	0.04	-0.04	0.02	0.1	-0.04	
451	0.02	-0.03	0.02	0.1	-0.04	
502	-0.03	-0.05	0.04	0.11	0.05	**Total**
Min	-0.04	-0.05	-0.03	0.1	-0.11	-0.11
Max	0.07	0.06	0.16	0.18	0.15	0.18
Average	0.02	0.00	0.06	0.12	0.03	0.05

(a) (b)

Figure 8. Applying the 2-D deblocking filter to CIF foreman test sequence. (a) not-filtered sequence; (b) filtered sequence.

5. Conclusion

Introducing the concept of BOP, instead of LOP; we have presented a new fast 2-D deblocking filtering algorithm for H.264/AVC. The proposed method has improved the deblocking speed by minimizing the usage of the three most frequently used instructions. Experimental results indicate that the proposed technique has improved the speed by more than 40%. The speed improvement is achieved at the cost of losing only 0.15% of the objective video quality in average.

Acknowledgments

The authors wish to express their gratitude to Iran Telecommunication Research Center (ITRC) for their financial support during the course of this research.

References

[1] T. Wiegand, G. J. Sullivan, G. Bjontegaard, A. Luthra, "Overview of the H.264/AVC Video Coding Standard," *IEEE Trans. on Circuits and Systems for Video Technology*, vol. 13, no.7, pp. 560-576, Jul. 2003.

[2] S. D. Kim, J. Yi, H. M. Kim, J. B. Ra, "A Deblocking Filter with Two Separate Modes in Block-Based Video Coding," *IEEE Trans. on Circuits and Systems for Video Technology*, vol. 9, no. 1, pp. 156- 160, Feb. 1999.

[3] R. Schäfer, T. Wiegand, H. Schwarz, "The Emerging H.264 /AVC Standard," *EBU Technical Review*, Jan. 2003.

[4] Video Coding for Low Bit Rate Communication. ITU-T Recommend H.263, Feb. 1998.

[5] Information Technology - Coding of Audio-Visual Objects – Part. 2; Visual, ISONEC 144496-2, 1999.

[6] "Advanced video coding for generic audiovisual services," ITU-T Rec. H.264/ISO/IEC 14496-10, Mar. 2005.

[7] S.-C. Chang, W.-H. Peng, S.-H. Wang, T. Chiang, "A Platform Based Bus-interleaved Architecture for De-blocking Filter in H.264/MPEG-4 AVC," *IEEE Trans. on Consumer Electronics*, vol. 51, no. 1, pp. 249- 255, Feb. 2005.

[8] J. Ostermann, J. Bormans, P. List, D. Marpe, M. Narroschke, F. Pereira, T. Stockhammer, T. Wedi, "Video coding with H.264/AVC: tools, performance, and complexity," *IEEE Circuits and Systems Magazine*, vol. 4, no. 1, 2004.

[9] P. List, A. Joch, J. Lainema, G. Bjontegaard, M. Karczewicz, "Adaptive deblocking filter," *IEEE Trans. on Circuits and Systems for Video Technology*, vol. 13, pp. 614- 619, July 2003.

[10] B. Sheng, W. Gao, D. Wu, "An implemented architecture of deblocking filter for H.264/AVC," *IEEE International Conference on Image Processing*, vol. 1, pp. 665-668, oct. 2004.

[11] M. Sima, Y. Zhou, W. Zhang, "An efficient architecture for adaptive deblocking filter of H.264/AVC video coding," *IEEE Trans. on Consumer Electronics*, vol. 50, no. 1, pp. 292- 296, Feb. 2004.

[12] T. Liu, W. Lee, T. Lin, C. Lee, "A Memory-Efficient Deblocking Filter for H.264/Avc Video Coding," *IEEE International Symposium on Circuits and Systems*, pp. 2140 - 2143, 2005.

[13] Onur Guleryuz "A Nonlinear Loop Filter for Quantization Noise Removal in Hybrid Video Compression," *IEEE International Conference on Image Processing*, Sep. 2005.

[14] R. Stevenson, "Reduction of Coding Artifacts in Transform Image Coding," *ICASSP*, pp. 2363-2366, Mar. 1995.

[15] R. Rosenholtz and A. Zakhor, "Iterative Procedures for Reduction of Blocking Effects in Transform Image Coding," *IEEE Trans. on Circuits and Systems for Video Technology*, pp. 91-95, Mar. 1992.

[16] M. Horowitz, A. Joch, F. Kossentini, A. Hallapuro, "H.264/AVC Baseline Profile Decoder Complexity Analysis," *IEEE Trans. on Circuits and Systems for Video Technology*, vol. 13, no. 7, pp. 704-716, Jul. 2003.