

Classified Quadtree-based Adaptive Loop Filter

Qian Chen[†], Yunfei Zheng[‡], Peng Yin[‡], Xiaoan Lu[‡], Joel Solé[‡], Qian Xu[‡], Edouard Francois[‡], and Dapeng Wu[†]

[†] School of Electrical and Computer Engineering, University of Florida, Gainesville, FL USA

[‡] Research and Innovations, Technicolor, Princeton, NJ USA

Abstract—In this paper, we propose a classified quadtree-based adaptive loop filter (CQALF) in video coding. Pixels in a picture are classified into two categories by considering the impact of the deblocking filter, the pixels that are modified and the pixels that are not modified by the deblocking filter. A wiener filter is carefully designed for each category and the filter coefficients are transmitted to decoder. For the pixels that are modified by the deblocking filter, the filter is estimated at encoder by minimizing the mean square error between the original input frame and a combined frame which is a weighted average of the reconstructed frames before and after the deblocking filter. For pixels that the deblocking filter does not modify, the filter is estimated by minimizing the mean square error between the original frame and the reconstructed frame. The proposed algorithm is implemented on top of KTA software and compatible with the quadtree-based adaptive loop filter. Compared with kta2.6r1 anchor, the proposed CQALF achieves 10.05%, 7.55%, and 6.19% BD bitrate reduction in average for intra only, IPPP, and HB coding structures respectively.

I. INTRODUCTION

Many techniques have been developed to improve the coding performance of H.264/AVC for high definition video contents. Among these techniques, the adaptive in-loop filter (ALF) is proposed and achieves an attractive gain over H.264/AVC. The basic idea of ALF is to estimate one or multiple filters at encoder by restoring the reconstructed frame towards its original version. This technique can effectively remove more general coding artifacts when working together with deblocking filter and prepare better reference frames for coding future frames, therefore improving the coding performance.

In [1], the wiener filter is used as the post filter to improve the subjective and objective qualities. [2] and [3] are two earliest proposals of ALF schemes to place the wiener filter inside the coding loop. Although the filter is temporally adaptive by updating the filter coefficients frame by frame, it is usually kept spatially invariant. To achieve better adaptation, several techniques are proposed in recent academic and standardization activities.

In [4], a block-based adaptive in-loop filter (BALF) was proposed. The reconstructed frame is partitioned to non-overlapped and equal-size blocks. For each block, a flag is used to signal whether the block will be filtered or not. Quadtree-based adaptive in-loop filter (QALF) further improves the filter performance by allowing blocks to be of variable size, and provides a signaling scheme in a quadtree data structure [5]. Based on QALF, multiple filters scheme is proposed by estimating a set of filters [6] instead of only one filter in QALF. Whenever the QALF partition is indicated to be filtered, for

each pixel in the block, a specific filter from the set is chosen based on a measure of local characteristic of the pixel. The filters take local characteristic of a frame into account and allow more spatial adaptation. In [7], the concept of multiple inputs filter was proposed by taking the reconstructed frame, prediction frame, and prediction residual frame as the inputs of the filter in order to incorporate more hypothesis and consider different characteristics of input signals for restoration. It automatically adapts the filter coefficients to different inputs during the filter estimation.

Although the wiener filter can efficiently remove some artifacts that are caused during compression, it is less efficient to remove blocky artifacts compared with the traditional deblocking filter. So in all previous works, the wiener filter is designed after the deblocking filter. These two filters work sequentially but are designed independently, which may cause both to be sub-optimal. In [12], it shows that the joint design of deblocking filter and ALF is important in both performance and complexity.

Motivated by the joint design of the filtering framework, in this paper, we propose a classified quadtree-based adaptive in-loop filter (CQALF) that takes advantage of both multiple filters and multiple inputs filter schemes. The proposed scheme is based on the hybrid video coding framework, for example H.264/AVC. Instead of unifying the deblocking filter and adaptive in-loop filter into one filter as in [12], we keep the traditional structure which makes the problem simpler. However, the impact of the deblocking filter of H.264/AVC is considered during the wiener filter restoration. The output pixels of deblocking filter are classified into two categories: those modified by the deblocking filter and those untouched. A filter will be designed for each category. For the pixels modified by the deblocking filter, the adaptive loop filter takes two inputs, frames before and after the deblocking filter. A weighted average of the two inputs are used to estimate the filter coefficients, which avoids increasing the number of filter coefficients and hence saves the coding overhead. For the pixels which are not modified by deblocking filter, only the frame after the deblocking filter is exploited to estimate the filter coefficients. CQALF classifies pixels by taking the impact of the deblocking filter into account in order to avoid the possible over-filtering problem. No overhead is needed to indicate the classification of each pixel because the classification is based on the reconstructed frame which is available at both encoder and decoder.

The rest of the paper is organized as follows. Section II briefly revisits the deblocking filter in H.264/AVC and ALF

technique in KTA. In Section III, the proposed CQALF and its implementation details are elaborated. Section IV shows simulation results of the CQALF. Finally, conclusions are drawn in section V.

II. DEBLOCKING FILTER AND ADAPTIVE IN-LOOP FILTER

A. Deblocking Filter

In H.264/AVC, a deblocking filter is exploited to remove the blocky artifacts around the block boundaries in order to improve the quality of a coding frame. To preserve image details and remove blockiness, a boundary strength (Bs) parameter is assigned to every boundary between two blocks. If filtering conditions are met, a pre-defined filter will be applied. Filtering does not take place for edges with $Bs = 0$. For nonzero Bs values, filtering on the edge only takes place when the following 3 conditions are met:

$$|p_0 - q_0| < \alpha(Index_A) \quad (1)$$

$$|p_1 - p_0| < \beta(Index_B) \quad (2)$$

$$|q_1 - q_0| < \beta(Index_B) \quad (3)$$

where p_0, q_0 are block boundary pixels located in Fig.1, α and β are dependent on the quantization parameter (QP) employed over the edge as well as the encoder offset value to control the properties of the deblocking filter on slice level [8]. The deblocking filter can successfully remove blocky artifacts. It is not efficient to remove more general artifacts, like ringing artifacts, blurring, and so on, which may be produced during the compression process.

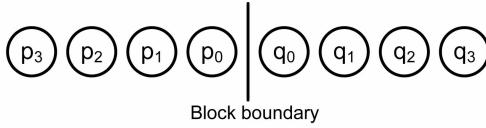


Fig. 1. Notation of pixels across a block boundary.

B. Adaptive in-loop Filter

In recent video coding standardization activities, a wiener filter based adaptive in-loop filter (ALF) was proposed in order to further improve the quality of the reconstructed frame. It restores the reconstructed frame output from the deblocking filter towards its corresponding original frame. At encoder, a filter is adaptively estimated for each frame using wiener filtering theory. The filter coefficients are transmitted to decoder as the overhead.

Suppose s is a pixel in the original frame; x_0 is its corresponding degraded pixel in the reconstructed frame; $X_0 = [x_0, x_1, \dots, x_{n-1}]^T$ is a pixel set that consists of the pixels surrounding x_0 with a defined filter support N . The filtered pixel of x_0 can be formulated as:

$$\hat{x}_0 = \sum_{i=0}^{n-1} w_i x_i = H^T X_0 \quad (4)$$

where $H = [w_0, w_1, \dots, w_{n-1}]^T$ is the filter coefficients. Assume the filter is spatial invariant, the wiener filter is the one which minimize the mean square error of the whole frame:

$$MSE = \sum_{i=0}^{M-1} \|s_i - \hat{x}_i\|_2 = \sum_{i=0}^{M-1} \|s_i - H^T X_i\|_2 \quad (5)$$

where M is the number of pixels in the frame. The wiener filter can be estimated by:

$$H^* = \underset{H}{\arg \min} MSE = (C^T C)^{-1} C^T S \quad (6)$$

where $C = [X_0, X_1, \dots, X_{M-1}]^T$, $S = [s_0, s_1, \dots, s_{M-1}]^T$.

As mentioned in I, the filter coefficients will be updated frame by frame to achieve the temporal adaptation. A quadtree based filtering signaling scheme is proposed to turn on/off the filtering operation to enable spatial adaptation.

III. CLASSIFIED QUADTREE-BASED ADAPTIVE IN-LOOP FILTER

A. Problem Formulation

In KTA or recent released TMuC software, the deblocking filter and adaptive in-loop filter is concatenated to filter the reconstructed frame sequentially but the two filters are designed independently. Under this strategy, some pixels may be filtered twice, which may cause the over-filtering. The over-filtering can degrade the performance of both filters, thus these filters should design and work jointly in order to achieve the best possible performance.

We propose to consider the deblocking filter and ALF jointly by classifying the pixels into two categories: pixels that are modified by deblocking filter and pixels that are untouched by deblocking filter. To avoid the over-filtering, the idea is to design a different filtering operation for the pixels that have already been filtered by the deblocking filter. Let us denote the original input frame as s , the reconstructed frame before deblocking filter as s' , and the reconstruction after deblocking filter as s'' in the rest of the paper. According to the filtering rule of deblocking filter, only some pixels (especially around the block boundaries) are filtered, while the rest are unchanged. The pixel classification is done by comparing the frame s' and s'' pixel by pixel. We denote Category I as the pixels modified by the deblocking filter and Category II as the pixels not modified by the deblocking filter.

The coding diagram of CQALF is illustrated in Fig.2. For Category I, each pixel has two hypothesis from s' and s'' . We take both of them as the input of the filter corresponding to Category I. To avoid too much additional overhead, we apply the same set of filter coefficient to both inputs instead of using two sets of filter coefficients as in [7]. To handle the two inputs differently based on their characteristics, we propose to combine them by weighting average, which is controlled by a weighting factor ω . Based on off-line training, we found that there is a relationship between weighting factor and quantization parameter (QP) for each frame type, which will be elaborated in Section III-B. For the pixels in Category II, only one input s'' is used. Though either s' or s'' can

be used for input as they are equal for every pixel to be filtered, we choose to use s'' . This is because the surrounding pixels within its filter support \mathcal{N} are generally of better quality in s'' than in s' , which is based on the observation that the reconstructed frame after deblocking filter shows overall improvement in both objective and subjective quality than that before deblocking filter.

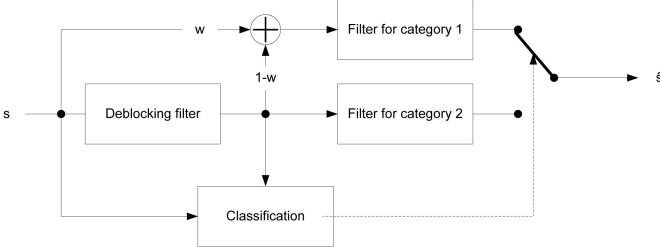


Fig. 2. Block Diagram of Encoder with CQALF.

Suppose filter length is n for both filters, the filter of Category I is $H_1 = [w_{10}, w_{11}, \dots, w_{1,n-1}]^T$, the filter of Category II is $H_2 = [w_{20}, w_{21}, \dots, w_{2,n-1}]^T$, input from s' as $X_i = [x_0, x_1, \dots, x_{n-1}]_i^T$, input from s'' as $Y_i = [y_0, y_1, \dots, y_{n-1}]_i^T$, and \hat{s}_i as the output of the i th pixel filtered. For Category I, a pixel will be filtered as

$$\hat{s}_i = H_1^T (\omega X_i + (1 - \omega) Y_i) \quad (7)$$

the filter can be estimated by

$$H_1 = \arg \min_{H_1} \sum_{i=0}^{M_1-1} \|s_i - H_1^T (\omega X_i + (1 - \omega) Y_i)\|_2 \quad (8)$$

where M_1 denotes the number of pixels in Category I. The pixels in Category II will be filtered as

$$\hat{s}_i = H_2^T Y_i \quad (9)$$

the filter can be estimated by

$$H_2 = \arg \min_{H_2} \sum_{i=0}^{M_2-1} \|s_i - H_2^T Y_i\|_2 \quad (10)$$

where M_2 denotes the number of pixels in Category II.

B. Parameter Settings

It can be seen in 1 that the deblocking filtering strength is controlled by boundary strength and two parameters, α and β . The filtering process only takes place when the difference between neighboring pixels is bigger than corresponding threshold α or β . Their values increase with QP and are obtained from a look-up table which is specified by H.264/AVC. In particular, at the low end of table where $Index_A < 16$ and $Index_B < 16$, one or both α and β are clipped to 0 and filtering is turned off [8]. That implies few pixels will be modified in deblocking filter due to small α and β when QP is very small. It results in few different pixels in s' and s'' . Under such circumstances, Category I is almost empty which leads to CQALF being the same algorithm as ALF.

TABLE I
 ω VALUE FOR DIFFERENT FRAME TYPE AND QP

QP	I	P	B
$QP \leq 25$	0.9	0.5	0.5
$QP > 25$	0.5	0.5	0.5

To improve the overall performance, we observed that to increase the deblocking parameter α and β slightly will give better result in relatively low QP range. Similar strategy has also been reported in [9]. In our algorithm, the better case of the two: with or without increasing deblocking parameter, is chosen at encoder based on the rate-distortion (RD) cost. A flag at slice header is used to signal the selected deblocking parameter setting if CQALF is enabled. Simulation results show that to choose α and β on an RD basis will have BD bitrate saving of 0.8% for IPPP mode and 0.58% for hierarchical-B (HB) mode.

As proposed in CQALF, the filter corresponding to Category I has two inputs. A weighting factor ω is used to combine them as in Eq.(7). When ω varies from 0 to 1, input s' will gradually get involved in the filter. Considering s' is usually of lower quality than s'' , how much it is involved should influence the performance of CQALF. The off-line training shows that for P and B frames, the coding gain reaches its peak at $\omega = 0.5$ regardless of QP.

On the other hand, the quality of s' differs significantly in intra frame with QP. Hence we should analyze the effect of QP when determining the best ω for intra frame. We test the intra only mode for all test sequences described in Section IV of 50 frames at QP from 22 to 37 with a step of 5. For each QP point, we vary ω from 0 – 1, select the ω that give the highest gain, and plot QP vs. ω . Fig.3 shows the average result over all test sequences. It can be seen that the optimal ω drops with QP's increase. When QP is small, the quality of s' does not differ much from s'' , i.e. the quality improvement of deblocking filter is not significant. Meanwhile, small QP is more likely to have slight increase of α, β , so that more pixels will be modified by deblocking filter and makes over-filtering the major concern in s'' . Hence, a more involvement of good quality s' helps to overcome over-filtering without loss to the filter output. When QP becomes large, the loss caused by low quality s' becomes noticeable and should be involved less in the filter input, as is suggested by the decrease of ω .

We summarize the analysis above in Table I which suggests the choice of ω that achieves the best coding performance.

C. Compatibility with QALF

As the estimated filter is optimal in the sense of minimizing the MSE of the whole frame, it may be far from optimal for each pixel. Actually, degradation can be observed for some pixels that are filtered by the estimated filter. To solve this issue, QALF introduces a region based signaling scheme to indicate whether a region is filtered or not. It adopts a bottom-up method for rate-distortion optimization (RDO) of quadtree data structure to signal the block partition and filtering decision [5].

TABLE II
ADAPTIVE LOOP FILTER DATA SYNTAX

adaptive_loopfilter_data{	Descriptor
pred_coef_mode	u(1)
cqalf_on_flag	u(1)
cqalf2_offset_2	u(1)
alf_tap_size_luma	ue(v)
for(i=0;i<num_of_coeff_luma;i++) {	
filter_coeff_luma[i]	se(v)
}	
if (cqalf_on_flag ==1)	
for(i=0;i<num_of_coeff_luma;i++) {	
filter_coeff_luma2[i]	se(v)
}	
...	
}	

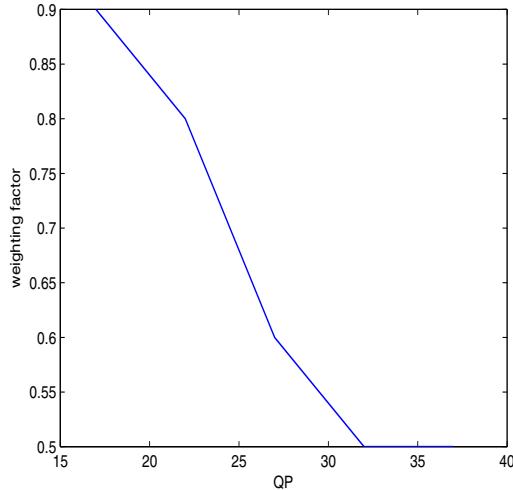


Fig. 3. Average QP vs. weighting factor ω over all test sequences for intra only mode.

We integrate our scheme on top of QALF framework to take advantage of the region based filtering scheme. Note that in CQALF, though the classification of the pixels makes more adaptive filter design and usually lower distortion than no classification, it sacrifices the bit rate by transmitting two sets of filter coefficients per frame. Consequently, for each frame, we compare the RD cost with and without classification, and select the one with a lower RD cost. If classification is chosen, it enables the proposed CQALF, otherwise, the algorithm goes back to QALF. The quadtree signaling scheme is adopted to indicate the partition and filtering decision, which is similar as that of QALF. Since the pixels are classified into two categories, for each block to be filtered, pixels within each category will be filtered by its corresponding filter.

D. Syntax and Semantics of CQALF

Table II gives the syntax of adaptive in-loop filter data supporting CQALF, which is sent in slice header syntax structure. The change is marked by the bold font. Refer to semantics below for specification of each new element. Note that if both of the two sets of coefficients exist, they are of the same tap size num_of_coeff_luma.

- 1) **calf_on_flag**: specifies the use of CQALF for the slice. **calf_on_flag** equals 1 indicates CQALF is applied for the slice. **calf_on_flag** equals 0 otherwise.
- 2) **calf2_offset_2**: specifies the use of deblocking parameter α and β . **calf2_offset_2** equals 1 indicates α and β offset by 2. **calf2_offset_2** equals 0 indicates no α and β offset.
- 3) **filter_coeff_luma[i]**: specifies filter coefficients for luminance in Category I.
- 4) **filter_coeff_luma2[i]**: specifies filter coefficients for luminance in Category II.

IV. EXPERIMENTAL RESULTS

In this section, intensive experiments are conducted to verify the performance of the proposed CQALF. Simulations are conducted using class B, C, D and E test sequences that are recommended by MPEG Call for Proposal. The test condition is based on [10]. KTA software version kta2.6r1 is used as the reference with all KTA tools enabled except for QALF when conducting the proposed CQALF tests. BD-PSNR is measured and compared according to [11].

We compare the BD bitrate saving of QALF and CQALF respectively over kta2.6r1 anchor without adaptive in-loop filter. We tested three coding structures, intra only, IPPP, and HB for the first 50 frames of each sequence. QP ranges from 22 to 37 with a step of 5. Weighting factor ω is set according to Table I. As shown in Table III, CQALF always outperforms QALF in all three cases. We also demonstrate the rate distortion (RD) performance of CQALF and the anchor for two selected sequences: Kimono1_1920x1080_24 of IPPP coding structure in Fig.5, and vidyo1_1280x720_60 of HB coding structure in Fig.6.

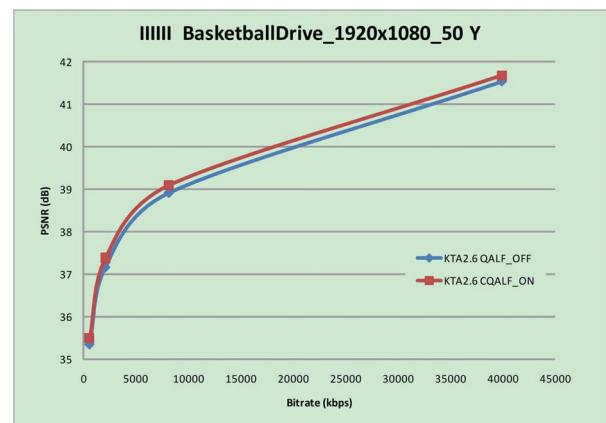


Fig. 4. RD performance of CQALF vs. kta2.6r1 anchor of BasketballDrive_1920x1080_50 in intra only coding structure. BD bitrate saving 11.59%.

TABLE III
BD BITRATE SAVING (%) OF QALF & CQALF OVER ANCHOR

Sequence	IIII		IPPPP		HB	
	QALF	CQALF	QALF	CQALF	QALF	CQALF
Kimono1_1920x1080_24	21.39	24.91	5.42	5.70	6.17	6.57
ParkScene_1920x1080_24	6.39	11.13	1.87	3.67	2.27	4.03
cactus_1920x1080_50	5.36	8.42	4.48	5.38	4.31	5.03
BasketballDrive_1920x1080_50	3.74	11.59	7.29	9.78	5.45	7.80
BQTerrace_1920x1080_60	3.58	7.04	5.22	6.43	7.61	8.49
BasketballDrill_832x480_50	10.30	11.96	10.06	11.96	5.69	7.90
BQMall_832x480_60	3.05	5.33	3.71	4.71	2.58	3.26
PartyScene_832x480_50	1.53	3.57	3.53	4.07	5.68	5.96
RaceHorses_832x480_30	5.68	7.99	2.18	2.39	1.55	1.83
BasketballPass_416x240_50	5.36	9.91	4.44	6.72	3.35	4.90
BQSquare_416x240_60	1.61	4.18	5.56	7.14	13.15	13.75
BlowingBubbles_416x240_50	1.94	3.32	2.48	3.32	2.89	3.31
RaceHorses_416x240_30	7.30	10.06	1.96	2.19	1.18	1.60
vidyo1_1280x720_60	8.54	11.99	8.92	20.53	5.06	6.76
vidyo3_1280x720_60	12.10	15.03	11.46	12.84	8.96	9.75
vidyo4_1280x720_60	7.25	11.33	10.37	14.02	6.78	8.02
Average	6.57	10.05	5.56	7.55	5.17	6.19

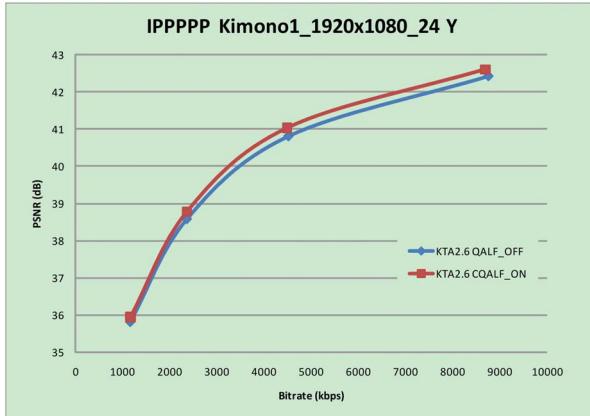


Fig. 5. RD performance of CQALF vs. kta2.6r1 anchor of Kimono1_1920x1080_24 in IPPP coding structure. BD bitrate saving 5.7%.

We observe that the coding gain of intra only case achieved by CQALF is the largest among all other cases. In intra frame, there are more pixels that may be modified by deblocking filter than in P and B frames. Hence the over filtering is more severe in I frame. As CQALF proposes to solve this problem, it can be foreseen that intra only case is the most effective to demonstrate the improvement of CQALF.

Fig.7 are selected partial frames extracted from two sequences to demonstrate the visual performance of CQALF. It can be seen the proposed CQALF achieves competitive or even better subjective quality than QALF.

V. CONCLUSION

We propose a classified quadtree-based adaptive loop filter to improve the coding efficiency by jointly considering the design of deblocking filter and adaptive in-loop filter. Pixels in a coded frame are classified into two categories, pixels modified and not modified by the deblocking filter. A wiener

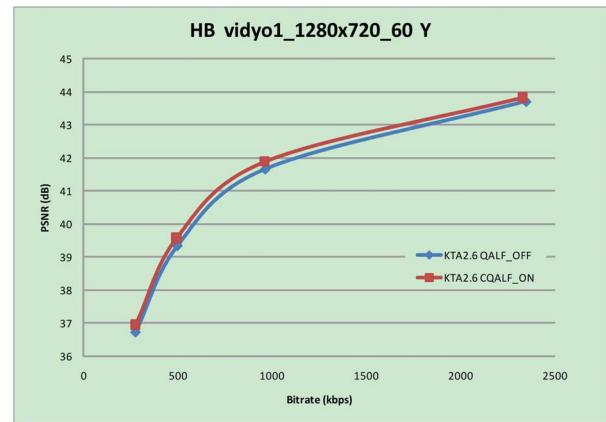


Fig. 6. RD performance of CQALF vs. kta2.6r1 anchor of vidyo1_1280x720_60 in HB coding structure. BD bitrate saving 6.76%.

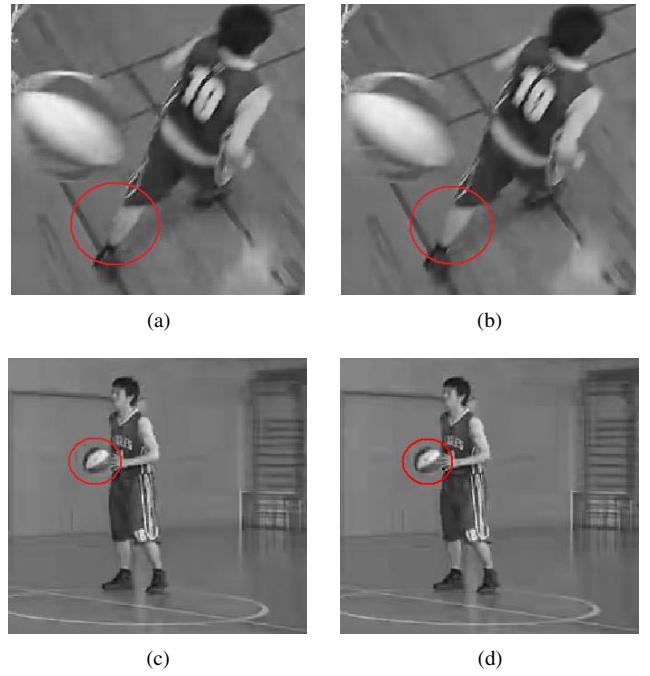


Fig. 7. Reconstructed BasketballDrill (832x480) sequence at the 10th frame with QP=37. (a) QALF: 32.12dB; (b) CQALF: 32.72dB; Reconstructed Basketballpass (416x240) sequence at the 9th frame with QP=37. (c)QALF: 31.96dB; (d)CQALF: 32.00dB

filter is carefully designed for each category. For the pixels modified by the deblocking filter, the reconstructed pixels before and after deblocking filter are combined as the input of the filter. For the pixels not modified by the deblocking filter, only reconstructed pixel after deblocking filter is used. Extensive experiments are conducted to verify the performance of the proposed CQALF. Compared with kta2.6r1 anchor, CQALF provides an average of 10.05% bitrate deduction for intra only, 7.55% for IPPP and 6.19% for HB coding structure.

REFERENCES

- [1] S. Wittmann and T. Wedi, "Transmission of post-filter hints for video coding schemes", Proceedings of 2007 IEEE International Conference

- on Image Processing.
- [2] T. Chujo, A. Tanizawa and T. Yamakage, "Adaptive Loop Filter for Improving Coding Efficiency", ITU-T SG16 Contribution, C402, Geneva, April 2008.
 - [3] Yi-Jen Chiu and L. Xu, "Adaptive (Wiener) Filter for Video Compression", ITU-T SG16 Contribution, C437, Geneva, April 2008.
 - [4] T. Chujo, A. Tanizawa and T. Yamakage, "Block-based Adaptive loop Filter", ITU-T SG16 Q.6 Document, VCEG-AJ13, San Diego, October, 2008.
 - [5] T. Chujo, N. Wada, T. Watanabe, G. Yasuda, T. Yamakage, "Specification and Experimental Results of Quadtree-based Adaptive Loop Filter", ITU-T SG16, Q.6 Document, VCEG-AK22, Japan, April, 2009.
 - [6] M. Karczewicz, P. Chen, Rajan, *Video coding technology proposal by Qualcomm Inc.*, ITU-T SG16, JCTVC-A121, Germany, April, 2010.
 - [7] J. Jung, T.K. Tan, S. Wittmann, E. Francois, and etc., *Description of video coding technology proposal by France Telecom, NTT, NTT DOCOMO, Panasonic and Technicolor*, ITU-T SG16, JCTVC-A114, German, April, 2010.
 - [8] P. List, A. Joch, J. Lainema, G. Bjøntegaard, and M. Karczewicz, "Adaptive Deblocking Filter", IEEE Trans. on Circuits and System for Video Technology, vol.13, No.7, July 2003.
 - [9] Wen Yang, "Image/video Processing Techniques: Header Compression and Post-processing", MPhil Thesis, Hong Kong University of Science and Technology, May, 2010.
 - [10] T.K. Tan, G.J. Sullivan, and T. Wedi, "Recommended Simulation Common Conditions for Coding Efficiency Experiments Revision 1", ITU-T SG16, Q.6 Document, VCEG-AJ10r1, San Diego, October 2008.
 - [11] S. Pateux and J. Jung, "An Excel Add-in for Computing Bjøntegaard Metric and its Evolution", ITU-T SG16 Q.6 Document, VECG-AE07, Marrakech, January 2007.
 - [12] Y. Liu, *Unified Loop Filter for Video Compression*, IEEE Transaction on Circuits and System for Video Technology, vol.20, No.10, pp1378-1382, Oct. 2010