Adaptive Guided Image Filter for Improved In-Loop Filtering in Video Coding

Chen Chen #1, Zexiang Miao #2, Bing Zeng #3,4

# Department of Electronic and Computer Engineering, The Hong Kong University of Science and Technology
Clear Water Bay, Kowloon, Hong Kong, China
1,3 {cchenaf,eezeng}@ust.hk

* Institute of Image Processing, University of Electronic Science and Technology of China
No.2006, Xiyuan Avenue, West Hi-tech Zone, Chengdu, Sichuan, China
2 zmiao@uestc.edu.cn
3 eezeng@uestc.edu.cn

Abstract—This paper proposes a new adaptive sharpening filter based on guided image filter and improves HEVC’s in-loop filter architecture by embedding sharpening filter between deblocking filter and SAO. The proposed algorithm classifies pixels of a frame into several groups according to uniform quantization of each pixel’s Sum-Modified-Laplacian value and assigns identical optimal filtering parameters to the pixels belonging to the same group based on rate-distortion optimization. Simulation results show that our proposed algorithm achieves 0.7% on average and up to 8% BD-rate reduction with respect to the original HEVC in-loop filtering method. Encoding time increases slightly by about 15% and decoding time increases by 70% on average without special optimization of C++ program integrated in HM-16.5.

I. INTRODUCTION

With the rapid growth of demand for high-definition videos with 1080p or even 4K resolutions and the limited storage space or network bandwidth, more and more people are inspired to dedicated in research on video compression. High Efficiency Video Coding (HEVC) is a newly established video coding standard in Jan. 2013 as the successor of H.264/AVC by joint collaborative team on video coding (JCT-VC), whose primary goal is to achieve 50% bitrate reduction as compared with H.264/AVC under the same perceptual quality [1].

While inheriting most of H.264/AVC’s features, HEVC is still based on a hybrid coding scheme using block-based predictions and transforms. Unlike the fixed 16×16 macroblock coding basic unit [2], HEVC introduces a flexible quadtree structure called Coding Tree Unit (CTU), predefined by configuration with size 16×16, 32×32 or 64×64, which can be further split into different sizes of Coding Unit (CU) varying from 8×8 to 64×64, Prediction Unit (PU) and Transform Unit (TU) varying from 4×4 to 32×32 adaptively according to complexity of frame textures [1].

Such a block-based coding scheme adopted by H.264/AVC and HEVC will be no doubt that leads to blocking artifacts because of the discontinues occurred on block boundaries caused by quantization errors and differences among motion vectors of adjacent blocks [3]. Besides, another artifact—ringing artifact will also be introduced after compression, which is analogous to the Gibbs phenomenon and results from the high frequency detail loss due to quantization [4]. In the most recent version of HEVC codec, two kinds of in-loop filters are adopted in the standard—deblocking filter [5] whose role is to reduce the blocking artifact and sample adaptive offset (SAO) [6] aimed at removing the ringing artifact. Both the two in-loop filters are designed to improve the subjective visual quality and objective coding efficiency while reducing the implement complexity, especially considering the parallel operations of encoder and decoder. Another in-loop filter adopted by early HEVC version is adaptive loop filter (ALF), which is a Wiener-based adaptive filter in order to minimize the mean square error between original samples and decoded samples [7].

The essence of current deblocking filter is still low-pass filtering by eliminating high frequency noise occurred on block boundaries. Even the HEVC standard has employed some smart strategy to identify the discontinues of block boundaries and real image edges, it still blurs some exact image textures. So the most important part of in-loop filter is to preserve the original image edges or even recover the real image textures as far as possible while removing the blocking and ringing artifacts at the same time. So, in recent years, many people spent a lot of efforts on pursing some sharpening filters, playing an opposite role of deblocking filter, which can sharpen image edges and restore image textures.

In [8], a famous bilateral filter is proposed to smooth image while preserving edges by means of a nonlinear combination of nearby image values. Based on such edge-preserving smooth image filter, paper [9] proposes a new type of trained bilateral filter by least mean square optimization to reduce coding artifacts. Paper [10] integrates bilateral filer into early HEVC coding standard, doing rate-distortion optimization (RDO) combined with ALF to enhance coding efficiency.

Inspired by the classical bilateral filter, an explicit image filter named guided filter is proposed in [11]. Derived
from a local model, the guided filter generates output image considering the content of a guidance image (can be the input image itself), which can perform as an edge-preserving smooth operator or even sharpening operator and has better behavior as compared with the bilateral filter. Paper [12] has employed this guided image filter on screen content coding. Consequently, the filter is with large probability useful for video coding, reducing coding artifacts while preserving real image edges and finally resulting in significant enhancement of coding efficiency. Therefore, in this article, we will propose an adaptive sharpening filter to improve in-loop filter of video coding based on guided image filtering.

The rest of the paper is organized as follows. In Section 2, we briefly introduce the related work of the guided image filter. In Section 3, we detailed present our proposed sharpening filter integrated with HEVC standard. Some experimental results are shown in Section 4. Finally, Section 5 concludes this paper.

II. REVIEW OF GUIDED IMAGE FILTER

Guided image filter (GIF) is proposed in [11]. In this algorithm, a guidance image $G$, which could also be identical to the input image $I$ itself, is introduced. GIF assumes that the filter is a local linear model between the guidance image and the filter output image $Y$ as

$$Y(i) = a_k G(i) + b_k, \forall k \in \omega_k$$  \hspace{1cm} (1)

where $\{a_k, b_k\}$ are some linear coefficients considered to be constant in a square window $\omega_k$ (can be $3 \times 3$, $5 \times 5$, $7 \times 7$, \cdots) which contains image pixel $i$.

In order to solve out the coefficients $\{a_k, b_k\}$, the GIF algorithm proposes a cost function that minimizes the square error between input pixel value $I(i)$ and filter output pixel value $Y(i)$ by

$$E(a_k, b_k) = \sum_{i \in \omega_k} \left((a_k G(i) + b_k - I(i))^2 + \epsilon a_k^2\right)$$  \hspace{1cm} (2)

where $\epsilon a_k^2$ is a regularization term preventing $a_k$ from being too large. Simply computing $\frac{\partial E(a_k, b_k)}{\partial a_k} = 0$ and $\frac{\partial E(a_k, b_k)}{\partial b_k} = 0$, the solution of Eq. (2) are given by:

$$a_k = \frac{1}{|\omega_k|} \sum_{i \in \omega_k} G(i) I(i) - \mu_k \bar{I}_k$$ \hspace{1cm} (3)

$$b_k = \bar{I}_k - a_k \mu_k$$ \hspace{1cm} (4)

where $|\omega_k|$ means the pixel number in window $\omega_k$, $\mu_k$ and $\sigma_k^2$ are the mean and variance of guidance image $G$ in window $\omega_k$, respectively and $\bar{I}_k$ represents the mean value of input image $I$ in window $\omega_k$.

Finally, after averaging all the possible output values $Y(i)$ for all windows $\omega_k$ that contains pixel $i$, the filter output is given by:

$$Y(i) = \frac{1}{|\omega_k|} \sum_{k : i \in \omega_k} (a_k G(i) + b_k) = \bar{\pi}(i) G(i) + \bar{b}(i)$$ \hspace{1cm} (5)

III. PROPOSED ADAPTIVE SHARPENING FILTER

A. Proposed In-loop Filter Architecture

As shown in Fig. 1(a), the in-loop filter part of traditional HEVC encoder architecture consists of two filters—deblocking filter and SAO. Inheriting the two in-loop filters, our new proposed encoder architecture embeds a sharpening filter based on guided image filter between the deblocking filter and SAO. The three filters aim at reducing blocking artifact, sharpening or restoring image edges and eliminating ringing artifact, respectively.

Like SAO, the sharpening filter also needs be controlled by some filtering parameters that are estimated by previous already encoded frames and chosen in current frame according to rate-distortion optimization. And of course, such parameters should be packed into bitstream by entropy coding and transmitted to decoder. Detailed algorithm of this sharpening filter will be described in the following subsections.

B. Derivation of Self-guided Image Filter

In term of video coding, the only known information for the decoder is the already decoded frames, so the most straightforward way to handle the GIF algorithm is regarding decoded frame itself as guidance image directly. Therefore, the GIF algorithm can be simplified as self-guided image filter. Replacing guidance image $G(i)$ as input image itself $I(i)$ in Eq. (3) and Eq. (4), the solutions of parameters $\{a_k, b_k\}$ are
given by:

\[ a_k = \frac{1}{|\omega_k|} \sum_{i \in \omega_k} I^2(i) - \bar{T}_k^2 = \frac{\sigma_k^2}{\sigma_k^2 + \epsilon} \tag{6} \]

\[ b_k = \bar{I}_k - a_k \bar{I}_k = \frac{\epsilon}{\sigma_k^2 + \epsilon} \bar{I}_k \tag{7} \]

where \( \sigma_k^2 \) and \( \bar{T}_k \) represents variance and mean of input image in window \( \omega_k \). Then, it's easy to derive the filter output as:

\[ Y(i) = \frac{1}{|\omega_k|} \sum_{k \in \omega_k} \left( \frac{\sigma_k^2 I(i) + \epsilon \bar{T}_k}{\sigma_k^2 + \epsilon} \right) \tag{8} \]

Observing the solution of \( \{a_k, b_k\} \) in Eq. (6) and Eq. (7), suppose the parameters \( \omega_k \) and \( \epsilon \) are determined, it’s obvious that in some “flat areas” the value of \( a_k \) will close to 0 while \( b_k \) will close to \( \bar{T}_k \), which means that the pixel located in the middle of a “flat areas” will be smoothed by the pixels nearby. On the other hand, in some “fluctuating area” the value of \( a_k \) will close to 1 while \( b_k \) will close to 0, which means that the pixel located in the middle of a “fluctuating area” will remain unchange. Such features ensure that original image edges will not be smoothed, achieving the goal—edge-preserving. Furthermore, changing of parameters \( \{\omega_k, \epsilon\} \) will adjust the degree of edge sharpness.

In term of the algorithm complexity, Eq. (8) clearly shows that only some local variances and means should be pre-calculated and then a linear combination of pixel values in a local window is regarded as the filter output of each pixel. So, such algorithm has a \( O(N) \) computation complexity (\( N \) is the total pixel number), which is a great advantage for video coding that requires fast encoding and decoding speed.

### C. Performance of Pixel-adaptive Self-guided Image Filter

Original GIF algorithm applies the filter on the whole image using the same parameters [11], but it’s a very coarse strategy to enhance quality of a reconstructed frame after compression since there is an original frame as a reference and a strict objective criteria—PSNR. So the most accurate solution to achieve the best filter output is determining parameters for each pixel respectively, namely pixel-adaptive self-guided image filter. Fortunately, only two parameters—local window size \( \omega \) and coefficient \( \epsilon \) need to be handled (refer to Eq. (8)).

Experiments in [11] detailed demonstrate that larger window
size results in smoother filter output. While our goal is not to smooth the flat area but to sharpen the image edges, we should spend no efforts on searching the optimal window size for each pixel but fix the window size $\omega$ as the smallest $3 \times 3$. Therefore, the only parameter should be chosen adaptively is $\epsilon$. We conduct a much more accurate experiment that assigns an adaptive parameter $\epsilon$ to each pixel. $\epsilon$ is optimized by calculation square errors after filtering over five possible options—$\epsilon = \{5, 10, 15, 20\}$ and don’t filtering. Notice that the guided filter is only applied on Luma component.

Three sequences (BasketballDrill, BasketballPass and Johnny) are coded by default encoder_lowdelay_P_main configuration with quantization parameter (QP) equaling 27. Experiment results are shown in Fig. 2–4. Sub-images (a) (b) (c) (d) represent reconstructed frames after deblocking filter, reconstructed frames after deblocking filter-pixel-adaptive self-guided image filter, difference map between (a) and (b) and optimal $\epsilon$ map, respectively.

Observing these figures, especially the subparts in small red windows, the optimal pixel-adaptive self-guided image filter sharpens the image edges more naturally and continuously, further reducing some blocking artifacts and even ringing artifacts. In addition, objective qualities are also significantly improved by fantastic more than 1dB PSNR gain. Difference maps clearly illustrate that main enhancements occurred on image edges and high variance areas. Finally, the most important $\epsilon$ maps also indicate that the “flat areas” are assigned with small $\epsilon$ values or even no filtering while the “fluctuating areas” or image edges are filtered with large $\epsilon$ values.

Although such pixel-adaptive self-guided image filter will result in significant PSNR gain with optimal parameter assignment, it’s a huge bit-rate expense when transmitting the $\epsilon$ maps shown in Fig. 2–4(d). Therefore, a feasible alternative strategy is to estimate the optimal $\epsilon$ map using already known information—previous decoded frames, which can keep a balance between increasing of extra bit-rate and improvement of video quality.

### D. Estimation of Parameters

The above section detailed discusses that the only parameter need to be estimated in entire guided image filtering algorithm is the $\epsilon$ and experiments demonstrate that the chosen of $\epsilon$ are highly correlated with the texture complexity. Consequently, a logical idea is to consider each pixel’s gradient intensity.

The Laplace operator is a differential operator given by the divergence of the gradient of a continuous function on Euclidean space. A discrete approximation to the modified Laplacian is obtained as:

$$ML(x, y) = [2I(x, y) - I(x - 1, y) - I(x + 1, y)]$$
$$+ [2I(x, y) - I(x, y - 1) - I(x, y + 1)]$$

Then the Sum-Modified-Laplacian (SML) \cite{13} operator is computed in a small window $(2N + 1) \times (2N + 1)$ as below:

$$SML(i, j) = \sum_{x=i-N}^{i+N} \sum_{y=j-N}^{j+N} ML(x, y)$$

(10)

All pixels in an entire frame are classified into $M$ groups according to uniform quantization of their SML values. Pixels assigned in each group share one parameter—$\epsilon$ selected from a candidate list $E$ and determined by rate-distortion optimization (RDO).

According to a large number of experiments, finally in our proposed adaptive guided filter algorithm, the small window for SML calculation is defines as $7 \times 7$ ($N = 3$), window of guided filter local averaging is fixed as $3 \times 3$ ($|\omega| = 3 \times 3$), each frame is divided into 16 groups ($M = 16$) and the $\epsilon$ candidate list is $E = \{0, 1, 2, 3, \cdots, 31\}$, where $\epsilon = 0$ means applying on guided filter on these pixels.

Summarily, the process of adaptive guided image filtering optimization is as follow:

- Divide all pixels of a frame into 16 groups based on calculation of SML.
- For all the pixels associated to group $i$, do
  - Define $MinRDCCost(i) = \lambda \cdot rate(0)$ (means no guided filtering)
  - For each candidate $\epsilon$ in set $E$, do
    * compute $RDCCost(i) = \Delta SSE(i, \epsilon) + \lambda \cdot rate(\epsilon)$ on the image pixels belonging to group $i$ before and after guided filter.
    * if $RDCCost(i) < MinRDCCost(i)$, store the best parameter $\epsilon$ and update $MinRDCCost(i) = RDCCost(i)$.
  - Go to next candidate $\epsilon$ in set $E$
- Go to next group $i$
- Perform guided image filter on all the pixels using the optimal parameters for each group.

Notice that the $rate(\epsilon)$ represents the bit cost when coding $\epsilon$, which will be detailed explained in the next subsection.

### E. Syntax Design and Entropy Coding of Parameters

The coded information filter coefficient $\epsilon$ need to be transmitted. As shown in Fig. 5, the parameters are located in the picture parameter set (PPS) since the coefficient set of $\epsilon$ is used for an entire frame and will change among different pictures.

Since adjacent frames usually share similar parameters, for B or P frame, a most probable $\epsilon$ candidate list for each group $i$ is defined as $E^{best}_i = \{\epsilon_1, \epsilon_2, \epsilon_3, \epsilon_4\}$ which are estimated by corresponding group of previous already coded reference frames (up to 4). For the remaining $\epsilon$ in candidate list $E$, calculate the smallest difference between $\epsilon$ and the value in $E^{best}_i$, then transmit the index in $E^{best}_i$ and the difference value.
In this paper, we propose a new sharpening filter based on guided image filter and embeds this filter between deblocking filter and sample adaptive offset (SAO) in the HEVC encoding structure. Our algorithm classifies pixels of a frame into several groups according to uniform quantization of each pixel’s Sum-Modified-Laplacian (SML) value and assigns optimal filtering parameters to the pixels belonging to each group based on rate-distortion optimization (RDO). Finally, we apply self-guided filtering on Luma component of images and each group shares the same parameters. Experimental results employing C++ program integrated in HM-16.5 show that our proposed algorithm achieves 0.7% on average and up to 8% best under encoder_lowdelay_P_main configuration which can achieve 1.1% BD-rate reduction on average while the proposed filter save only 0.5% BD-rate under encoder_randomaccess_main configuration. Besides, this sharpening filter is more effective for low bit-rate video coding since the low quality test performs better than high quality test under all configurations.

In terms of the algorithm complexity, the encoding time will increase by 10% on average because of introduction of the sharpening filter and the decoding time will increase by average 70%.

## V. Conclusion

Simulations are conducted on a range of HEVC standard test video sequences including natural and synthetic scenes. Performance of the proposed algorithm is compared against the HEVC algorithm implemented in the new established HM-16.5 reference software [14] using the default encoder_intra_main, encoder_randomaccess_main, encoder_lowdelay_main and encoder_lowdelay_P_main configurations. Furthermore, we take the average over 60 frames to ensure the correctness of experimental results.

The whole algorithm is implemented on Luma component of 24 test sequences which are arranged into six classes: Class A (4K), Class B (1080p), Class C (WVGA), Class D (WQVGA), Class E (720p) and Class F (synthetic). The quantization parameters are set to $Q_P = \{22, 27, 32, 37\}$ for the high quality test and $Q_P = \{27, 32, 37, 42\}$ for the low quality test. The coding efficiency results are presented in Table I–IV as the percentage of bit-rate savings (BD-rate [15]) with respect to the HM-16.5 main profile anchor. Statistics reveal that our algorithm obtains about 0.7% performance gain on average and can be up to 8% for the sequence “Kimono” under encoder_lowdelay_P_main configuration, as compared with the original HEVC’s coding scheme.

Generally speaking, the proposed algorithm performs the best under encoder_lowdelay_P_main configuration which can achieve 1.1% BD-rate reduction on average while the proposed filter saves only 0.5% BD-rate under encoder_randomaccess_main configuration. Besides, this sharpening filter is more effective for low bit-rate video coding since the low quality test performs better than high quality test under all configurations.

In terms of the algorithm complexity, the encoding time will increase by 10% on average because of introduction of the sharpening filter and the decoding time will increase by average 70%.

## V. Conclusion

In this paper, we propose a new sharpening filter based on guided image filter and embeds this filter between deblocking filter and sample adaptive offset (SAO) in the HEVC encoding structure. Our algorithm classifies pixels of a frame into several groups according to uniform quantization of each pixel’s Sum-Modified-Laplacian (SML) value and assigns optimal filtering parameters to the pixels belonging to each group based on rate-distortion optimization (RDO). Finally, we apply self-guided filtering on Luma component of images and each group shares the same parameters. Experimental results employing C++ program integrated in HM-16.5 show that our proposed algorithm achieves 0.7% on average and up to 8% best under encoder_lowdelay_P_main configuration which can achieve 1.1% BD-rate reduction on average while the proposed filter saves only 0.5% BD-rate under encoder_randomaccess_main configuration. Besides, this sharpening filter is more effective for low bit-rate video coding since the low quality test performs better than high quality test under all configurations.

In terms of the algorithm complexity, the encoding time will increase by 10% on average because of introduction of the sharpening filter and the decoding time will increase by average 70%.
BD-rate reduction with respect to the original HEVC coding method. Encoding time increases slightly by about 15% and decoding time increases by 70% on average.

Our future work will focus on more accurate estimation of filtering parameters which can achieve more significant enhancement of compression efficiency and further optimization of program’s running efficiency for the purpose of decreasing coding time.

**REFERENCES**


