Application Map in the Noise Suppression Filter for Video Compression*

Roman Chernyak[1], Roman Meshcheryakov[2]

[1] Huawei Technologies Co., Ltd., Moscow, Russia
chernyak.roman@huawei.com

[2] V. A. Trapeznikov Institute of Control Sciences of Russian Academy of Sciences, Moscow, Russia
mrv@ieee.org

Abstract. This paper considers an application map method usage together with the Nose Suppression Filter (NSF) in-loop filter for video compression. The application map method is described in details and simulations results of basic NSF are provided as well as simulation results of NSF powered with the application map method. It is demonstrated that the application map method allows to significantly improve objective performance results of basic NSF and additionally decrease decoder average complexity. As a result, an average bd-rate saving of NSF with the application map reaches 2.6% for luma component in Low Delay P coding configuration in comparing to Versatile Video Coding reference implementation (VTM) version 1.0.

Keywords: Noise Suppression Filter, In-loop Filtering, Video Compression.

1 Introduction

At this moment digital video industry is keep extensively growing. Modern consumer devices already support Ultra High Definition (UHD) resolution and High Dynamic Range (HDR) video content. First 5G smartphones already available in the market and it is expected that 5G technology will shortly substitute the current one. New augmented and virtual reality (AR/VR) solution have been being demonstrated very frequently over all popular market shows. All such factors actively push video industry forward and it is expected that total amount of video traffic in the Network will exceed 80% already in 2020 [1]. In order to handle such big amount of video data and manage its storing and transmission, video coding experts such as VCEG and MPEG are finalizing several modern video compression standards, which are targeted to outperform currently available industry video compression standards Advanced Video Coding (AVC) [2] and High Efficiency Video Coding (HEVC) [3]. There are two recent video

* Supported by Huawei Technologies Co., Ltd.

compression standards are expected to be finalized in 2020: Versatile Video Coding (VVC) [4] and Essential Video Coding (EVC) [5]. Both of them utilize a hybrid coding scheme, which assumes having in a processing pipeline one or more in-loop filters targeted to improve subjective and objective quality of the coded video signal. Most of in-loop filters, which are currently used for video compression, somehow rely on the video signal information and sometimes on additional codec parameters in order to adjust filter strength and avoid over filtering or under filtering of the signal. However, such mechanisms can make wrong assessments, which, in turn, may lead coding performance drop. To avoid such situations in-loop filters may have some mechanisms of force disabling that specify filters applicability and prevent degradation of the signal quality. Based on these mechanisms all in-loop filters can be classified onto two big categories: 1) filters with explicit application specification; and 2) filters with implicit application specification. E.g. Sample Adaptive Offset (SAO) [6] filter and Adaptive Loop Filter (ALF) [7] from current VVC standard draft belong to the first category. In these both examples there is one bit of information transmitted per every coding tree unit (CTU), which indicates whether the filter is enabled in the current CTU or not. As another example, Bilateral Filter [8], which was studied during VVC standardization process, and Hadamard Transform Domain Filter (HTDF) [9] from current EVC standard draft do not have an explicit mechanism of the filters enabling. In these two cases, codec has some predefined rules of the filter applicability such as minimal block size etc., and always applies the filter for all blocks satisfying the rules. Thus, applicability of such filters is normally determined based on the rate-distortion optimization (RDO) performed at the encoder side.

Another in-loop filter from the first category is a Noise Suppression Filter (NSF) that was firstly proposed in VVC Call for Proposals (CfP) responses [10], [11] and was later studied in VVC Core Experiments (CE) [12]. The NSF filter also has the explicit filter control signaling, which is however implemented slightly differently than in SAO and ALF. Main NSF design principles are described in [13] and several specific aspects regarding the filtering process are described in [14]. This paper is mostly focused on the NSF application control mechanism, which is called the application map that allow significantly improve the filter performance comparing to a general NSF method without having any application control.

This paper is organized as follows. The NSF algorithm is described in section 2, where section 2.1 briefly summarizes a general NSF design and the application map approach detailed description is given section 2.2. Simulation results are provided in section 3. The conclusion of the paper is given in section 4.

2 Noise Suppression Filter

The Noise Suppression Filter (NSF) is a non-local collaborative filter, which can be used in video compression lossy systems as in-loop filter in order to improve reconstructed video frame quality. NSF is carried out after reconstruction process and based on a block matching procedure targeted to estimate and reduce quantization noise caused by quantization process of the hybrid video coding scheme. Basically, NSF comprises of a filter core part, which is shared by encoder and decoder, and the application map method, which checks whether NSF filtering is beneficial on a frame or block level at the encoder and transmit this information to the decoder. Such approach allows to amplify the filter benefits in terms of its complexity and performance by utilizing additional information available on the encoder side and remove complex filtering operations, which do not contribute to performance improvement at the decoder side. Fig. 1 gives a high-level NSF processing encoder side flowchart.

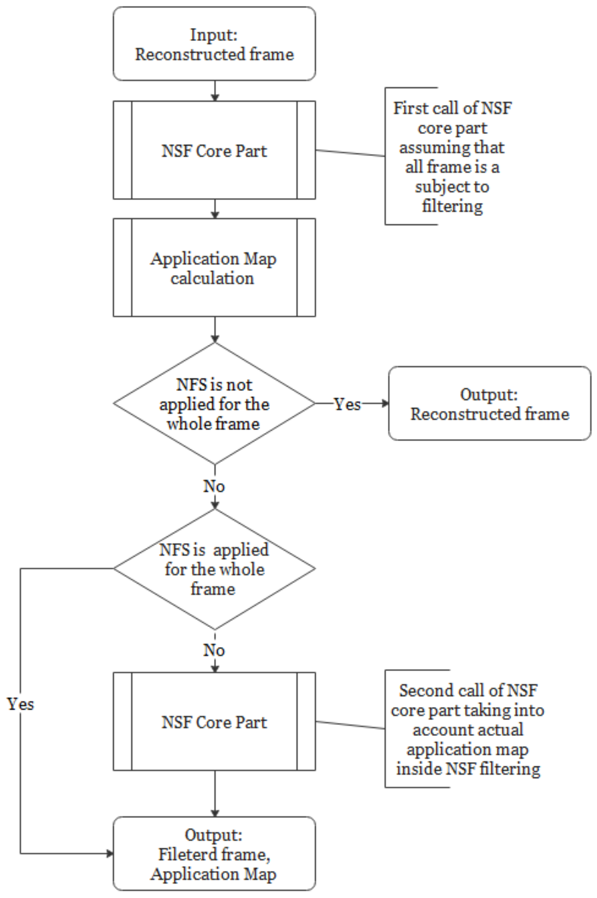
Input:
Reconstructed frame
NSF Core Part
First call of NSF core part assuming that all frame is a subject to filtering
Application Map calculation
NFS is not applied for the whole frame
Yes
Output:
Reconstructed frame
No
NFS is applied for the whole frame
No
Yes
NSF Core Part
Second call of NSF core part taking into account actual application map inside NSF filtering
Output:
Fileterd frame,
Application Map

Fig. 1. NSF in-loop filter encoder diagram

As can be seen from Fig. 1, NSF usage at the encoder side requires several additional operations besides NSF core part. It should be also noted, the NSF core part is applied twice at the encoder: first time without considerations of application map, assuming that whole frame is a subject to filtering; and second time considering actually calculated application map. Such design decision is determined by the fact that in video codecs encoder and decoder have to process data in the same way in order to guarantee that decoded results will be the same for any standard compliant decoders. Due to this reason and considering that in real systems decoder complexity is typically more critical than encoder one, it was decided to intentionally make encoder side more complex in order to simplify decoder.

Fig. 2 gives a high-level NSF processing flowchart for the decoder.

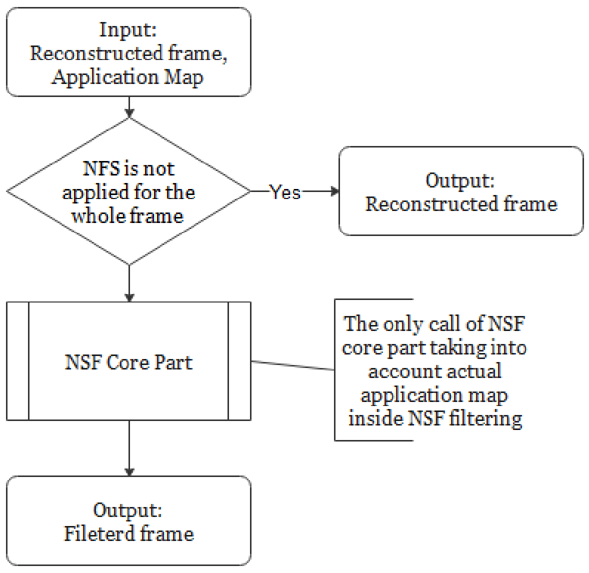

Fig. 2. NSF in-loop filter decoder diagram

As can be seen from Fig. 2, the NSF usage at the decoder side is much simpler than the encoder one. The only complex operation of NSF core part is performed under the application map condition and even if the condition is satisfied, NSF core part considers actual values of application map inside the process.

2.1 NSF Core Part

The NSF core part is a main filtering procedure inside NSF. It takes whole reconstructed frame and the application map as inputs, performs filtering and outputs the filtered frame. The filtering procedure comprises of the following main steps.

1. Dividing the frame into a set of blocks;
2. Performing blocks classification;
3. Finding a set of similar blocks (stack) for each block by using a block matching procedure;
4. Performing a Frequency Domain Filtering (FDF) procedure within each stack;
5. Performing a pixel domain averaging of every blocks corresponding to a certain location.

It should be noted that the application map is widely used during NSF core part methods as an early skip mechanism. At the first encoder call of NSF core part, the application map is considered to specify whole frame to be filtered and at the second call of NSF core part instant values of the calculated application map are being used.

A general flowchart of NSF core part is demonstrated in Fig. 3.

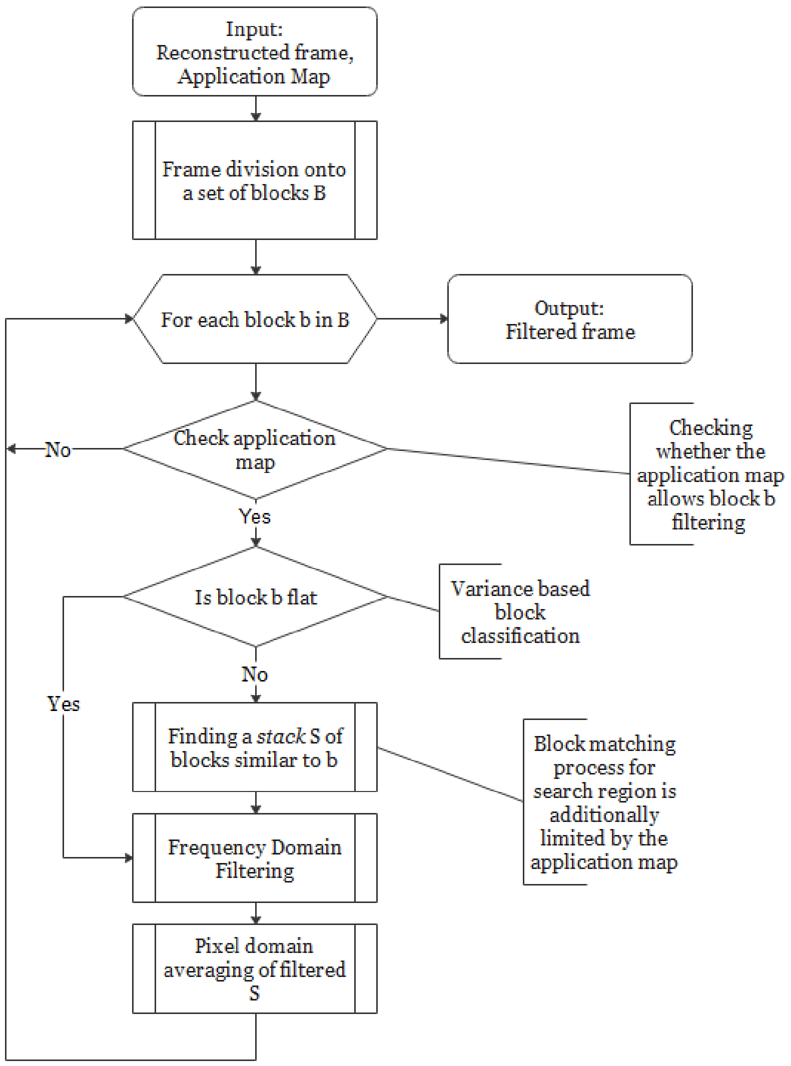
Input: Reconstructed frame, Application Map
Frame division onto a set of blocks B
For each block b in B
Output: Filtered frame
Check application map
No
Checking whether the application map allows block b filtering
Yes
Is block b flat
Variance based block classification
No
Yes
Finding a *stack* S of blocks similar to b
Block matching process for search region is additionally limited by the application map
Frequency Domain Filtering
Pixel domain averaging of filtered S

Fig. 3. NSF core part flowchart

At the first step of the algorithm, whole frame is divided onto the set of regular square 8×8 blocks. In order to increase efficiency an overlapping of blocks can be used at this stage.

Then for each block two conditions are checked. First, whether the block is marked "to be filtered" by the application map. In negative case the block is simply bypassed, and algorithm moves to the next block. If the block is marked "to be filtered", the second condition checks whether it is flat or not. Insofar as NSF demonstrates the biggest efficiency being applied among objects edges, blocks classification allows to avoid all further NSF processing for the blocks where filtering would not be efficient.

The third step of the algorithm performs block matching procedure to collect a stack S of k blocks similar to the current within a search range also considering the application map limitations by removing picture areas from the search region, where NSF is not applicable. Specific block matching method, which is used at this step is described in detail in [14].

At the fourth step of the algorithm the collected stack S is being filtered by a Frequency Domain Filtering (FDF) procedure. In general, FDF procedure performs a collaborative filtering of each sample within stack S, where filter strength is determined by the only additional parameter σ that, in turn, is a function of codec quantization parameter (QP). Thus, FDF process can be summarized as follows (1):

$$FDF: S \times \sigma \rightarrow \hat{S}, \text{ were } \hat{S} \text{ is a filtered stack.} \quad (1)$$

More specifically for a given stack S of k blocks, where each block includes n^2 samples, FDF process first makes n^2 lines L_i of k elements, where each element j of the line L_i is an $i-th$ samples of $j-th$ block inside S. E.g. $L_0 = \{p_0^0, p_0^1, \dots, p_0^{k-1}\}$, where p_0^0 is a $0-th$ sample of $0-th$ block, p_0^1 is a $0-th\ sample$ of $1-st$ block, etc. Thus, stack whole S can be represented as n^2 lines L_i of k elements. Next FDF performs Hadamard transform of each line L_i and then for each frequency element f_i^j staring from first AC it calculates a gain value g by the following formula (2):

$$g = \frac{\left(f_i^j\right)^2}{\left(f_i^j\right)^2 + \sigma^2}. \quad (2)$$

Finally, FDF filters each component f_i^j in a following way (3):

$$\hat{f}_i^j = f_i^j \times g. \quad (3)$$

In the end of the process, FDF performs all required backward operations and computes filtered stack $\hat{S}$.

It should be noted that since $\sigma = f(QP)$, and QP value is already presented in the bitstream, the whole FDF process does not require any additional information to be signaled in the bitstream. In other words, most of FDF parameters are implicitly derived from the video signal itself.

At the fifth step all filtered blocks within different stacks are being collected back to the frame into their original locations.

2.2 Application Map

One of the main features of NSF core part is an absence of additional filter parameters. Indeed, the only external parameters for the core part of NSF is σ value, which is function of QP, so it does not bring any additional overhead for parameters transmission. However, on the other hand that also means that NSF core part does not have any mechanisms to utilize information from the original picture, which is available at the encoder so the core part always relies only on reconstructed frame that would inevitably lead over filtered and under filtered regions in the frame.

In order to increase NSF filter flexibility and minimize a number of filtering errors the application map method was introduced. This method allows to estimate how beneficial is NSF core part filtering process on a frame or block level at the encoder side based on the original picture and transmit this information to the decoder side. More specifically, after NSF core part the encoder checks several following options of NSF applicability:

1. Apply NSF for the whole frame;
2. Do not apply NSF for the whole frame;
3. Apply NSF on a block-based level:
   a. With the block size equal to CTU;
   b. With the block size equal to half of CTU;
   c. With the block size equal to quarter of CTU;
   d. With the block size equal to one eighth of CTU.

In options 1 and 2 encoder simply choose whether or not to use NSF filter for the whole frame. This encoder decision should be signaled in the bitstream, however signaling overhead is only 2 bits per picture, which normally does not affect whole codec performance. In a) – d) options under category 3 encoder introduces the application map with different block size, where each item of the application map determines whether the NSF filter is used for the current block. Fig. 4 gives an example of an application map for the picture, where green blocks demonstrate picture areas to be filtered by NSF and red block demonstrate picture areas to be kept without filtering.

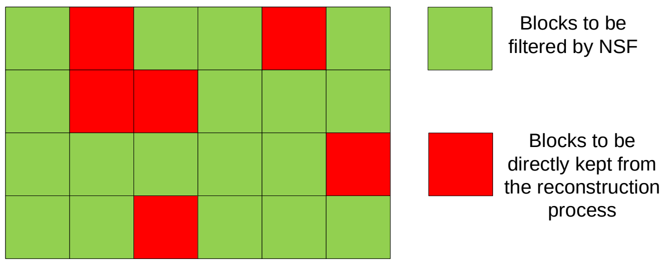
Blocks to be filtered by NSF
Blocks to be directly kept from the reconstruction process

Fig. 4. An application map example

Obviously, the application map usage and especially option 3 d) will always lead better performance comparing to other options. However, the application map information requires additional 1 bit for each block, where block size depending on CTU size. In modern video codecs CTU is normally square blocks of 128 samples, so option 3 d) will require additional signaling for each 16×16 block, which can visibly affect total bitrate. Thus, in order to find an optimal solution of NSF application map option, encoder should consider all possibilities taking into account both parameters: filtered picture distortion and additional bit budget required to transmit the decision.

Since the application map is signaled in the bitstream, decoder can similarly read and apply it within the decoding process. In addition, the application map approach also allows to reduce NSF decoder complexity. Indeed, since decoder already has the information about blocks that are not filtered, these blocks can be removed from NSF core part and thus reduce the number of filtering operations at the decoder.

3 Modeling

The Noise Suppression filter was implemented in VVC Test Model (VTM) version 1.0 and simulation results were carried out within JVET Common Test Condition (CTC) [15].

According to CTC, the simulations should be performed in 4 different test configurations corresponding to 4 following application scenarios: 1) All intra (AI) configuration that allows only Intra frames usage within the coded sequence; 2) Random Access (RA) configuration that allows both Intra and Inter frames and also allows frames reordering; 3) Low Delay B (LDB) configuration that allows both Intra and Inter frames, but disallows frames reordering; and 4) Low Delay P (LDP) configuration, which is similar to LDB, but allows only unidirectional prediction (P) for Inter frames.

CTC simulations are performed on a CTC video test sequences data set, which consists of 7 following test subsets of YCbCr video sequences: class A1, class A2, class B, class C, class D, class E and class F. Classes from A1 to D include natural camera video content with different resolution, where classes A1 and A2 include UHD content (3840×2160), class B – FHD content (1920×1080), class C – 832×480 resolution content, and class D – 416×240 resolution content. Class E includes HD resolution (1280×720) teleconferences content and class F includes screen content sequences of a different resolution. CTC defines some of the classes as optional for some configurations. Namely, for RA configuration, class E is not mandatory as well as classes A1 and A2 for both low delay configurations.

CTC evaluation methodology assumes encoding all required sequences by reference (anchor) and test encoders with 4 predefines quantization parameters (QP) equal to 22, 27, 32 and 37. Then for both anchor and test materials rate-distortion (RD) curve is being built for each of video signal components with the piece-wise cubic interpolation and BD-rate metrics [16] are being calculated. Finally results for all sequences inside one class are averaged and the average results for classes A1, class A2, class B, class C and class E are averaged once again to overall configuration result.

Table 1 - 4 demonstrate CTC simulation results for a general NSF without the application map method. More specifically, they correspond to encoder flowchart (Fig. 1)

modification, where both application map conditions are always forced to decide whole frame to be filtered by NSF.

Positive numbers in the result tables represent bitrate increase with the same video quality between anchor and test and mean codec performance degradation. Similarly, negative numbers represent bitrate reduction with the same video quality between anchor and test and mean codec performance improvement.

Table 1. AI Simulation results of NSF without the application map

	All Intra				
	Y	U	V	EncT	DecT
Class A1	-0.32%	-2.70%	-2.96%	100%	162%
Class A2	-0.21%	-1.95%	-1.37%	100%	149%
Class B	-0.03%	-2.14%	-2.72%	100%	148%
Class C	-0.34%	-1.75%	-2.15%	101%	134%
Class E	-0.97%	-2.63%	-3.10%	101%	145%
Overall	-0.33%	-2.20%	-2.47%	100%	147%
Class D	-0.09%	-1.51%	-1.80%	101%	131%
Class F	2.94%	-1.36%	-1.44%	101%	130%

Table 2. RA Simulation results of NSF without the application map

	Random Access				
	Y	U	V	EncT	DecT
Class A1	1.49%	-3.64%	-3.43%	101%	211%
Class A2	4.44%	0.74%	3.26%	100%	208%
Class B	4.04%	-0.41%	-0.52%	100%	211%
Class C	2.72%	0.16%	-0.52%	100%	187%
Class E					
Overall	3.26%	-0.67%	-0.34%	100%	204%
Class D	3.10%	1.35%	0.38%	100%	179%
Class F	6.11%	-0.20%	-0.56%	100%	194%

Table 3. LDB Simulation results of NSF without the application map

	Low delay B				
	Y	U	V	EncT	DecT
Class A1					
Class A2					
Class B	6.70%	5.05%	4.89%	102%	200%
Class C	3.26%	2.11%	1.48%	102%	179%
Class E	26.28%	37.03%	39.15%	106%	242%
Overall	10.45%	12.07%	12.32%	103%	202%
Class D	2.57%	2.48%	1.81%	101%	173%
Class F	8.54%	4.58%	5.03%	103%	208%

Table 4. LDP Simulation results of NSF without the application map

			Low delay P		
	Y	U	V	EncT	DecT
Class A1					
Class A2					
Class B	2.28%	0.98%	0.47%	101%	195%
Class C	1.59%	0.53%	-0.74%	101%	174%
Class E	19.53%	29.86%	32.61%	104%	232%
Overall	6.36%	8.05%	8.10%	102%	196%
Class D	1.40%	0.95%	0.53%	101%	169%
Class F	7.82%	2.79%	3.48%	103%	203%

As can be seen from Table 1 - 4, a straightforward usage of NSF without the application map is certainly only beneficial for All Intra configuration, however in all other configurations there are many places where NS filtering actually contributes to codec performance degradation.

Table 5 - 8 demonstrate CTC simulation results for NSF method with the application map.

Table 5. AI Simulation results of NSF with the application map

			All Intra		
	Y	U	V	EncT	DecT
Class A1	-0.71%	-1.85%	-2.04%	100%	144%
Class A2	-0.52%	-1.24%	-0.91%	100%	132%
Class B	-0.48%	-1.42%	-1.80%	100%	131%
Class C	-0.45%	-1.23%	-1.49%	100%	126%
Class E	-1.02%	-2.38%	-2.82%	101%	140%
Overall	-0.61%	-1.58%	-1.79%	100%	134%
Class D	-0.24%	-0.94%	-1.16%	100%	122%
Class F	-0.48%	-0.94%	-1.03%	100%	123%

Table 6. RA Simulation results of NSF with the application map

			Random Access		
	Y	U	V	EncT	DecT
Class A1	-1.49%	-3.52%	-3.58%	100%	156%
Class A2	-1.13%	-1.71%	-0.44%	100%	130%
Class B	-1.16%	-1.88%	-1.86%	100%	132%
Class C	-0.46%	-1.37%	-1.75%	100%	123%
Class E					
Overall	-1.03%	-2.04%	-1.89%	100%	134%
Class D	-0.12%	-0.83%	-1.13%	100%	117%
Class F	-0.45%	-1.02%	-1.14%	100%	117%

Table 7. LDB Simulation results of NSF with the application map

	Low delay B				
	Y	U	V	EncT	DecT
Class A1					
Class A2					
Class B	-0.89%	-1.58%	-1.50%	100%	134%
Class C	-0.48%	-1.74%	-2.17%	100%	129%
Class E	-0.56%	-1.01%	-0.98%	101%	118%
Overall	-0.67%	-1.49%	-1.59%	100%	128%
Class D	-0.28%	-1.31%	-2.27%	100%	125%
Class F	-0.52%	-1.00%	-0.31%	100%	121%

Table 8. LDP Simulation results of NSF with the application map

	Low delay P				
	Y	U	V	EncT	DecT
Class A1					
Class A2					
Class B	-3.06%	-3.87%	-4.23%	100%	145%
Class C	-1.18%	-2.52%	-3.60%	99%	132%
Class E	-3.76%	-3.78%	-2.03%	99%	123%
Overall	-2.61%	-3.40%	-3.47%	99%	135%
Class D	-0.75%	-2.23%	-3.20%	99%	128%
Class F	-0.85%	-1.78%	-1.51%	100%	123%

As can be seen the application map usage allows to significantly improve NSF performance. The NS filter powered with the application map always contribute visible performance improvement to overall luma (Y) component bd-rate saving for 4 typical encoder configurations. Chroma (U and V components) modeling results sometimes vary that is explained by differences in RDO processes between single NSF and NSF with the application map tests.

4 Conclusion

The Noise Suppression filter is a powerful in-loop that can be used in lossy video systems in order to improve reconstructed video quality without any additional filter parameters signaling. However, in all coding scenarios utilizing Inter frames redundancy removal, Nose Suppression Filter cannot be directly used due to a negative impact caused by over filtering that in many cases is more significant than the filter benefits. In order to eliminate such drawback and amplify NSF benefits for AI configuration the application map approach was introduced as an additional step for general NSF design. Together with the application map method, NSF demonstrates clear beneficial effect on top of VVC video codec for all CTC configurations. Namely, comparing simulation results for All Intra, where NSF improves quality even without the application map, it can be concluded that application map makes NSF performance roughly 2 times more efficient. For the other 3 Inter configuration the application map method allows to completely eliminate over filtering for all test classes. As a summarized result of NSF on top of VTM it demonstrated from 0.61% to 2.61% of luma bd-rate saving on average from 4 CTC test scenarios. In addition, the application map usage allows to significantly reduce decoder complexity especially such complexity reduction is observed on RA and both LD configurations.

Thus, the application map approach is an essential part of Noise Suppression Filter, which makes this filter practically applicable for typical video compression scenarios. In this paper NSF was considered as a normative part of VVC codec because filtered frame was used for further temporal prediction. However, NSF can be also used as a pure post filter, where filtered results will be only applied for output and unfiltered frame will be used for further temporal prediction. It is expected that compression efficiency of such approach will be less than one presented in this paper, on the other hand it will not require a standardization and such design can be used in wider video systems as a non-normative part. This exploration can become one of the further working directions towards NSF investigations. It should be also noted that introducing application map to the NSF method allows situation where some blocks of the current picture are filtered by NSF but some other blocks within the same picture are directly kept from the reconstruction process, which may theoretically lead visual blocking effects. Although several examples of NSF subjective results given in [13] demonstrate absence of such drawback, a detailed subjective assessment of NS filter with application map has not been performed and can become another direction for further related work.

References

1. T3-Global - 2020 Forecast Highlights, https://www.cisco.com/c/dam/m/en_us/solutions/service-provider/vni-forecast-highlights/pdf/Global_2020_Forecast_Highlights.pdf. Last assessed 15 Jun 2020.
2. ITU-T Rec. H.264 and ISO/IEC 14496-10, ITU-T and ISO/IEC JTC 1: Advanced Video Coding for Generic Audio-Visual Services, May 2003.
3. ITU-T Rec. H.265 and ISO/IEC 23008-2, ITU-T and ISO/IEC JTC 1: High efficiency video coding, 2014.
4. Bross, B., Chen, J., Liu, S., Wang, Y.-K.: Versatile Video Coding (Draft 9). Document of Joint Video Experts Team, JVET-R2001, 18th Meeting of the JVET, by teleconference, Apr. 2020.
5. ISO/IEC FDIS 23094-1: Information technology — General video coding — Part 1: Essential video coding.
6. Fu, C-M., Alshina, E., Alshin A., et al.: Sample Adaptive Offset in the HEVC Standard. IEEE Transactions on Circuits and Systems for Video Technology, vol. 22, no. 12, Dec. 2012.
7. Tsai, C-Y., Chen, C-Y., Yamakage, T., et al.: Adaptive Loop Filtering for Video Coding. IEEE Transactions on Circuits and Systems for Video Technology, vol. 7, no. 6, Dec. 2013.
8. Wennersten, P., Ström, J., Wang, Y.: Bilateral Filter for Video Coding. Proc Visual Communications and Image Processing (VCIP) 2017, St. Petersburg, USA, Dec. 2017.

9. Ikonin, S., Stepin, V., Chernyak, R., Chen, J.: Hadamard transform domain filter for video coding. Proc. SPIE 11137, Applications of Digital Image Processing XLII, 1113711, San Diego, Sep. 2019.
10. Alshin, A., Alshina, E., Choi, K., et at: Description of SDR, HDR and 360° video coding technology proposal by Samsung, Huawei, GoPro, and HiSilicon – mobile application scenario. Document of Joint Video Experts Team, JVET-J0024, 10th Meeting of the JVET, San Diego, Apr. 2018.
11. Chen, H., Chen, J., Chernyak, R., et al.: Description of SDR, HDR and 360° video coding technology proposal by Huawei, GoPro, HiSilicon, and Samsung – general application scenario, Document of Joint Video Experts Team, JVET-J0024, 10th Meeting of the JVET, San Diego, Apr. 2018.
12. Chernyak, R., Stepin, V., Ikonin, S., Chen, J.: CE2: Noise Suppression Filter (Test 2.5.3). Document of Joint Video Experts Team, JVET-K0053, 11th Meeting of the JVET, Ljubljana, Jul. 2018.
13. Chernyak, R., Stepin, V., Ikonin, S., Chen, J.: Noise suppression filter for video coding. Proc. SPIE 11137, Applications of Digital Image Processing XLII, 1113712, San Diego, Sep. 2019.
14. Chernyak, R., Mullakhmetov, R., Stepin, V., Ikonin, S.: Block Matching in Noise Suppression Filter for Video Coding. 2019 International Multi-Conference on Engineering, Computer and Information Sciences (SIBIRCON), Tomsk, Oct. 2019.
15. Bossen, F., Boyce, J., Li, X., Seregin, V., Sühring, K.: JVET common test conditions and software reference configurations for SDR video. Document of Joint Video Experts Team, JVET-N1010, 14th Meeting of the JVET, Geneva, Jul. 2018.
16. Bjøntegaard, G.: Calculation of average PSNR differences between RD-curves. Document of Video Coding Experts Group, VCEG-M33, 13th Meeting of the VCEG, Austin, Texas, Apr. 2001.