INTERPOLATION FILTER DESIGN IN HEVC AND ITS CODING EFFICIENCY – COMPLEXITY ANALYSIS

Kemal Ugur[1], Alexander Alshin[2], Elena Alshina[2], Frank Bossen[3], Woo-Jin Han[4], Jeong-Hoon Park[2] and Jani Lainema[1]

Nokia Corporation[1], Samsung Electronics[2], DOCOMO Innovations[3], Gachon University[4]

ABSTRACT

Coding efficiency gains in the High Efficiency Video Coding (H.265/HEVC) standard are achieved by improving many aspects of the traditional hybrid coding framework. Motion compensated prediction, and in particular the interpolation filter, is one of the areas that was improved significantly over H.264/AVC. This paper presents the details of the motion compensation interpolation filter design of the H.265/HEVC standard and its improvements over the interpolation filter design of H.264/AVC. These improvements include discrete cosine transform based filter coefficient design, utilizing longer filter taps for luma and chroma interpolation and using higher precision operations in the intermediate computations. The computational complexity of HEVC interpolation filter is also analyzed both from theoretical and practical perspectives. Experimental results show that a 4.5% average bitrate reduction for the luma component and 13.0% average bitrate reduction for the chroma components are achieved compared to interpolation filter of H.264/AVC. The coding efficiency gains are significant for some video sequences and can reach up to 21.7%.

Index Terms— Video coding, standards, HEVC, interpolation filter

1. INTRODUCTION

Similar to H.264/AVC [1], the High Efficiency Video Coding (H.265/HEVC) standard supports motion vectors with quarter-pel accuracy [2]. If a motion vector of a block has a fractional value, then the reference block needs to be interpolated to obtain the prediction samples. The interpolation filter used in video coding standards are carefully designed taking into account many factors, such as coding efficiency, implementation complexity and visual quality [3]. The interpolation filter was one of the mostly worked aspects during the H.265/HEVC developments including many proposals for improvement [4][5][6][7]. Compared to H.264/AVC, the final motion compensation interpolation design in H.265/HEVC includes various improvements to the interpolation filter design. These improvements include filter design with very close to ideal frequency response, utilizing longer filter taps for luma and chroma components for preserving natural high-frequencies present in videos captured by modern devices and using higher precision operations in intermediate filter computations which prevents unnecessary accuracy loss. These modifications yield an average 4.0% bitrate reduction (and up-to measured 21.7% reduction) over the H.264/AVC interpolation filter for luma and 11.3% bitrate reduction for chroma components. The coding efficiency gains become very significant for some sequences and can reach a measured maximum of 21.7%. In this paper, the motion compensation interpolation filter process of H.265/HEVC is explained and the specific improvements over the interpolation filter design of H.264/AVC are described in detail. This paper is organized as follows. Section II describes the interpolation filtering process of the H.264/AVC video coding standard and explains its shortcomings. Section III discusses the details of the interpolation filter design of H.265/HEVC and describes the differences compared to H.264/AVC. Section IV presents a detailed complexity analysis using both theoretical study and practical study based on profiling software implementations. Section V presents experimental results and shows the coding efficiency gains of H.265/HEVC interpolation filter. Section VI concludes the paper.

2. OVERVIEW OF H.264/AVC FILTER DESIGN

H.264/AVC supports motion vectors with quarter-pel accuracy for the luma component and one-eighth pel accuracy for chroma components for coding video in the 4:2:0 color format [8]. H.264/AVC uses various combinations of separable one-dimensional filters according to the fractional sample position. For example, if one of the motion vector components is fractional but another is integer, then interpolation is applied along only one direction (vertical or horizontal). If both motion vector components are fractional, a horizontal (vertical) interpolation filtering is done followed by vertical (horizontal) filtering, while the intermediate results are stored in a buffer

The samples at half-pel positions, $b_{0,0}$ and $h_{0,0}$ (as indicated in Figure 1) are derived by applying a 6-tap filter. The samples at half-pel positions $j_{0,0}$ are computed similarly, but from the non-rounded intermediate half-pel samples rather than the integer-pel samples. The samples at quarter-pel positions are calculated by averaging the two nearest samples located at integer-pel and half-pel positions.

In H.264/AVC, chroma samples are obtained by bi-linearly averaging the nearest samples at integer-pel positions. The samples at the fractional positions are calculated directly from the surrounding integer position samples $A_{0,0}$, $A_{1,0}$, $A_{0,2}$ and $A_{1,1}$ in a weighted sum where weights are inversely proportional to the distance from fractional positions to the corresponding integer ones.

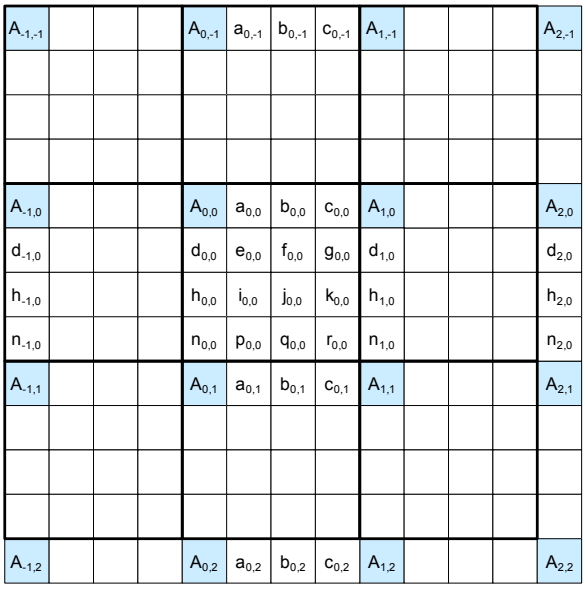

Figure 1 Fractional positions in Luma motion compensation with 1/4 pel accuracy.

Several issues were later identified with the filter design of H.264/AVC which motivated further improvements.

- **Number of filter coefficients:** a six-tap filter is used for half-pel positions of luma samples and a bi-linear filter for eighth-pel positions of chroma samples and this may not be sufficient for video sequences acquired with modern recording devices, which typically contain more high frequency information than older video sequences.
- **Cascaded process for quarter-pel positions:** samples at quarter-pel positions are generated by averaging two neighboring samples at either half-pel or integer-pel positions. This cascaded process introduces an intermediate step with rounding of the half-pel samples, which leads to undesirable latency and accuracy losses. In addition, samples at quarter-pel positions are derived differently according to their fractional positions. This complicates the overall motion compensation design as many sub-functions are implemented separately according to the quarter-pel position.
- **Loss of accuracy from cascaded rounding operations:** the interpolation filter defined in H.264/AVC has a large number of intermediate rounding operations. The number of rounding operations can go up to 7 when specific quarter-pel positions are used with bi-directional prediction. Every rounding operation introduces an undesirable rounding error that accumulates over frames.

3. MOTION COMPENSATION INTERPOLATION FILTER DESIGN IN H.265/HEVC

To overcome the above issues, HEVC introduces several new features including newly designed interpolation filters for luma and chroma as well as a high-accuracy motion compensation process for uni- and bi-directional prediction. The key differences between H.264/AVC and HEVC interpolation can be summarized as

- **Re-designed luma and chroma interpolation filter:** to improve the filter response in both the middle and the high frequency range, luma and chroma interpolation filters are re-designed. The luma interpolation process uses a symmetric 8-tap filter for half-pel positions and an asymmetric 7-tap filter for quarter-pel positions. For chroma samples, a 4-tap filter is introduced.
- **Non-cascaded process for quarter-pel positions:** rather than averaging two neighboring samples, HEVC directly derives quarter-pel samples by applying an asymmetric interpolation filter.
- **High-accuracy motion compensation operation:** in HEVC, intermediate values used in interpolation are kept at a higher accuracy. In addition, the rounding of two prediction blocks used in bi-directional prediction is delayed and merged with the rounding in the bi-directional averaging process.

3.1. Improved filter design

Finite impulse response (FIR) filters are used for luma and chroma interpolation in HEVC. The coefficients of the FIR filters are designed using a Fourier decomposition of the discrete cosine transform. The resulting interpolation filter is thus named DCT-based interpolation filter (DCTIF). The filter coefficients are shown in the following tables for luma and chroma components, where α indicates the sub-pixel position.

Table 1 Interpolation filter coefficients for luma

α	$filter_l(\alpha)$
1/4	{-1, 4, -10, 58, 17, -5, 1}
1/2	{-1, 4, -11, 40, 40, -11, 4, -1}

Table 2 Interpolation filter coefficients for chroma

α	$filter_C(\alpha)$
1/8	{-2, 58, 10,-2}
1/4	{-4, 54, 16, -2}
3/8	{-6, 46, 28, -4}
1/2	{-4, 36, 36, -4}

Compared to H.264/AVC, H.265/HEVC utilizes longer filter taps for interpolating the luma and chroma components. This means that the high frequency contents of the picture can be better preserved. This is illustrated in Figure 2. The main target during interpolation filter design was to minimize the difference between ideal and proposed filters frequency responses. So H.265/HEVC interpolation filter has better than H.264/AVC filter accuracy in the middle part of the spectrum. In particular H.265/HEVC interpolation filter has less noise amplification.

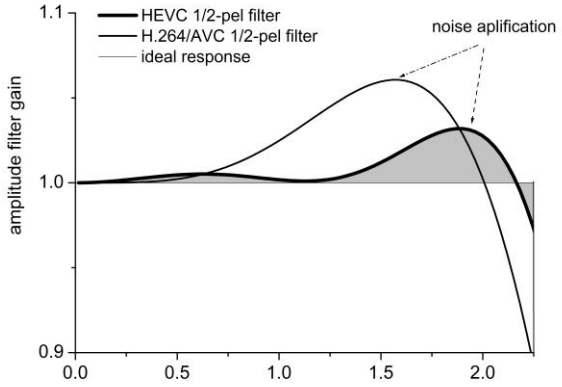

Figure 2 Amplitude frequency response of 1/2-pel position filters in HEVC and AVC in comparison with ideal response.

3.2. High precision filtering operations

The H.265/HEVC interpolation process is designed to minimize the adverse effects of rounding errors and improve coding efficiency. This is done by keeping many of the intermediate values within the motion compensation process at higher precision, hence minimizing rounding errors.

In H.264/AVC interpolation filter the half-pixel samples obtained by 6-tap FIR filter are rounded to input bit-depth prior to using them for averaging the quarter-pixel samples. This creates a rounding error and reduces the precision of the interpolation filter. Instead of using a two-stage cascaded filtering process, HEVC interpolation filter computes the quarter-pixels directly. This significantly reduces the rounding error and increases the coding efficiency. Similar to reduced precision of intermediate values used within quarter-pixel interpolation, H.264/AVC motion compensation process rounds the intermediate values to obtain the bi-predictive signal. In H.264/AVC, the prediction signal of the bi-predictively coded motion blocks is obtained by averaging prediction signals from two prediction lists. For an 8-bit video, the final prediction is the average of two 8-bit precision signals, whereas in H.265/HEVC, the prediction signals are not rounded to input precision prior to averaging but kept in higher precision.

It should also be noted that in H.265/HEVC the only clipping operation is at the very end of the motion compensated process, with no clipping in intermediate stages. As there is no rounding and clipping in intermediate stages, H.265/HEVC interpolation filter allows certain implementation optimizations. Consider the case, where motion vectors of each prediction points to the same fractional position. In these cases, final prediction could be obtained by first adding two reference signals and performing interpolation and rounding once, instead of interpolating each reference block, thus saving one interpolation process

4. COMPLEXITY ANALYSIS

There are several things to consider in terms of complexity, including memory bandwidth, number of arithmetic operations (and their bit width), and storage buffer sizes.

In terms of memory bandwidth, increasing the size of the interpolation filter from 6-tap to 8-tap for luma and from bilinear to 4-tap for chroma leads to a larger support region, which in turn increases the memory bandwidth. The worst case happens when the motion vector of a block of size *N-by-M* represents a fractional-sample displacement in both horizontal and vertical directions. The worst-case increase happens for small blocks of size 4x8 or 8x4, and it corresponds to a 51.5% increase in complexity (in HEVC motion blocks of size 4x4 are not utilized and 4x8 or 8x4 blocks are used for uni-prediction only to limit the worst case memory bandwidth). In addition to increase in memory bandwidth, an increase in computational burden is also expected when going from 6-tap to 8-tap interpolation. For H.264/AVC, worst-case happens for a 4x4 motion block and a 6-tap filter for both directions, whereas in H.265/HEVC, the worst case happens for an 8x4 motion block. By counting the number of multiply and accumulate (MAC) instructions per sample required to interpolate the block, it is found that less than 20% increase over H.264/AVC is required. Regarding the storage buffer sizes, high-precision bidirectional averaging has an impact on buffer sizes as the intermediate uni-prediction values need to be stored in 16-bit buffers as opposed to 8-bit buffers.

In addition to theoretical complexity analysis using analytic derivation of worst-cases, practical implementation complexity is also considered using actual implementations. This is done by profiling both the HM reference software and an optimized decoder based on [9] using the random-

access configuration defined in the JCT-VC [10]. To limit the amount of data, tests are limited to using two quantizer settings (QP 22 and 37), and class B (1080p) sequences. Table 3 summarizes the results where two sets of results are presented for each test case. The upper row describes percentages obtained with the HM software, and the lower row those obtained with the optimized software. In this test, four sub-categories are profiled related to interpolation: i) Luma interpolation, ii) Chroma interpolation iii) Copy and iv) Bipred. The sum column represents the share of decoding time for interpolation related function of the overall total decoding is considered.

From the data obtained by profiling, several observations can be made. Overall, the motion compensation process accounts for about 35-40% of decoding time and this percentage tends to decrease as bitrate increases. Among the different functions, luma interpolation contributes most to the total interpolation time (around 55% of total interpolation time is due to luma). Chroma interpolation takes around 25% of the total; bi-prediction averaging and copying both take 10% each of the total interpolation time.

Table 3 Decoding time share of each interpolation part

Sequence	Luma	Chroma	Copy	Bipred	Sum
BasketballDrive	18.0%	7.5%	0.4%	3.5%	29.4%
QP=22	18.5%	7.4%	0.5%	2.0%	28.3%
BasketballDrive	23.8%	9.0%	1.0%	5.9%	39.7%
QP=37	30.7%	13.0%	1.6%	4.4%	49.7%
BQTerrace	18.0%	7.7%	0.8%	3.3%	29.9%
QP=22	16.0%	7.2%	1.1%	1.9%	26.1%
BQTerrace	28.0%	10.5%	2.3%	8.0%	48.8%
QP=37	36.5%	15.2%	4.3%	5.4%	61.4%
Cactus	12.9%	5.3%	3.1%	3.8%	25.1%
QP=22	11.6%	5.4%	5.1%	2.4%	24.5%
Cactus	13.9%	5.5%	8.1%	7.9%	35.4%
QP=37	17.8%	7.1%	15.3%	5.5%	45.7%
Kimono	20.8%	8.3%	0.6%	4.5%	34.4%
QP=22	22.4%	9.8%	1.0%	2.9%	36.2%
Kimono	25.0%	9.5%	1.6%	6.9%	42.9%
QP=37	32.8%	13.4%	2.7%	5.1%	54.0%
ParkScene	21.9%	9.1%	0.4%	4.4%	35.8%
QP=22	23.2%	9.7%	0.8%	2.7%	36.3%
ParkScene	26.6%	10.3%	1.7%	7.5%	46.1%
QP=37	34.2%	13.2%	2.2%	5.4%	55.0%
Average	20.9%	8.3%	2.0%	5.6%	36.8%
	24.4%	10.1%	3.5%	3.8%	41.8%

5. EXPERIMENTAL RESULTS

In this section, coding efficiency gains of the H.265/HEVC interpolation filter over H.264/AVC interpolation is studied in detail. This is done by implementing the H.264/AVC interpolation filter to H.265/HEVC test model (HM) and running both the original HM with H.265/HEVC interpolation and modified HM with H.264/AVC interpolation in different configurations. The simulations are conducted using the HM version 6.0 (HM 6.0) following the common test conditions defined in the JCT-VC group [10]. Coding tools included in the draft Main profile are enabled and a total number of 24 different sequences are coded in four quantization values (22, 27, 32, 37). These sequences are divided into different classes that represent different use-cases and video characteristics. Tests are conducted for two different prediction structures: "Random Access" and "Low Delay" The coding efficiency is measured by using the Bjontegaard-Delta bitrate measure [11]. The results are summarized in Table 4 for luma component and Table 5 for chroma component. On average, the interpolation filter of H.265/HEVC brings 4.03% coding efficiency gain for luma and 11.27% coding efficiency gain for chroma over the H.264/AVC interpolation filter. For some sequences, especially for those that contain more high frequency content, gains become very large and become more than 20%. This is mostly thanks to the longer tap-length of H.265/HEVC filter that preserves the high frequency content better than H.264/AVC interpolation filter.

Table 4 Coding efficiency results of H.265/HEVC interpolation when compared to H.264/AVC for luma

Resolution	Low Delay	Random Access
Class A (2560x1600)		-0.7%
Class B (1920x1080)	-4.0%	-2.6%
Class C (832x480)	−5.8%	−4.7%
Class D (416x240)	−7.4%	−8.2%
Class E (1280x720)	−1.5%	−0.8%
All	**−4.9%**	**−4.0%**

Table 5 Coding efficiency results of H.265/HEVC interpolation when compared to H.264/AVC for chroma

Resolution	Low Delay		Random Access	
	Cb	Cr	Cb	Cr
Class A (2560x1600)			−10.8%	−11.3%
Class B (1920x1080)	−14.7%	−16.8%	−10.8%	−12.2%
Class C (832x480)	−13.0%	−13.5%	−10.3%	−10.8%
Class D (416x240)	−20.4%	−21.7%	−15.3%	−17.1%
Class E (1280x720)	−12.4%	−13.9%		
All	**−13.3%**	**−14.3%**	**−11.7%**	**−12.8%**

6. CONCLUSIONS AND RELATION TO PRIOR WORK

The work presented here has focused on detail description and both theoretical worst case and average complexity assessment of motion compensation interpolation design of the H.265/HEVC standard. Experimental results show that the motion compensation interpolation filter of H.265/HEVC improves the coding efficiency by more than 4% on average and for some sequences gains reach more than 20%. The interpolation filter was designed in the JCT-VC standardization committee over a several meeting cycles by conducting several extensive experiments. These are referenced throughout the paper appropriately

7. REFERENCES

[1] Joint Video Team of ITU-T and ISO/IEC JTC 1, "Draft ITU T Recommendation and Final Draft International Standard of Joint Video Specification (ITU-T Rec. H.264 | ISO/IEC 14496-10 AVC)," Joint Video Team (JVT) of ISO/IEC MPEG and ITU-T VCEG, JVT-G050, March 2003.

[2] Sullivan, G.J.; Ohm, J.; Woo-Jin Han; Wiegand, T.; Wiegand, T.;, "Overview of the High Efficiency Video Coding (HEVC) Standard," Circuits and Systems for Video Technology, IEEE Transactions on , vol.22, no.12, pp.1649-1668, Dec. 2012

[3] T. Wedi, "Motion compensation in H.264/AVC," IEEE Trans. Circuits Syst. Video Technol., vol. 13, pp. 577—586, Jul. 2003.

[4] M. Karczewicz, C. Peisong, R. Joshi, W. Xianglin, C. Wei-Jung, R. Panchal, Y. Reznik, M. Coban and I. S. Chong, "A hybrid video coder based on extended macroblock sizes, improved interpolation, and flexible motion representation," IEEE Trans. Circuits Syst. Video Technol., vol. 20, no. 12, pp. 1698-1708, Dec. 2010.

[5] D. Marpe, H. Schwarz, S. Bosse, B. Bross, P. Helle, T. Hinz. H. Kirchhoffer, H. Lakshman, T. Nguyen, S. Oudin, M. Siekmann, K. Suhring, M. Winken and T. Wiegand, "Video compression using nested quadtree structures, leaf merging, and improved techniques for motion representation and entropy coding," IEEE Trans. Circuits Syst. Video Technol., vol. 20, no. 12, pp. 1698-1708, Dec. 2010.

[6] K. Ugur, K. Andersson, A. Fuldseth, G. Bjøntegaard, L. P. Endressen, J. Lainema, A. Hallapuro, J. Ridge, D. Rusanovskyy, C. Zhang, A. Norkin, C. Priddle, T. Russert, J. Samuelsson, R. Sjoberg and Z. Wu, "High performance, low complexity video coding and the emerging HEVC standard," IEEE Trans. Circuits Syst. Video Technol., vol. 20, no. 12, pp. 1698-1708, Dec. 2010.

[7] W. J. Han, J. Min, I. K. Kim, E. Alshina, A. Alshin, T. Lee, J. Chen, V. Seregin, S. Lee, Y. M. Hong, M. S. Cheon, N. Shlyakhov, K. McCann, T. Davies and J. H. Park, "Improved video compression efficiency through flexible unit representation and corresponding extension of coding tools," IEEE Trans. Circuits Syst. Video Technol., vol. 20, no. 12, pp. 1698-1708, Dec. 2010.

[8] T. Wiegand, G. J. Sullivan, G. Bjøntegaard, and A. Luthra, "Overview of the H.264/AVC Video Coding Standard," IEEE Trans. Circuits Syst. Video Technol., vol. 13, no. 7, pp. 560–576, July 2003.

[9] F. Bossen, "On Software Complexity," Document of Joint Collaborative Team on Video Coding, JCTVC-G757, November 2011.

[10] F. Bossen, "Common HM test conditions and software reference configurations," Document of Joint Collaborative Team on Video Coding, JCTVC-H1100, February 2012

[11] G. Bjøntegaard, "Calculation of Average PSNR Differences between RD curves," VCEG-M33, 13th VCEG Meeting, Austin, USA, April 200.