Analysis of Hadamard Transform Coefficient Pruning for Approximate SATD Calculation

Victor H. S. Lima*, Matheus F. Stigger*, Leonardo B. Soares†, Cláudio M. Diniz*
*Graduate Program on Electronic Engineering and Computing, Catholic University of Pelotas (UCPel), Pelotas, Brazil
†Federal Institute of Rio Grande do Sul (IFRS), Rio Grande, Brazil

***Abstract*—The recent High Efficiency Video Coding (HEVC) standard compresses video with half the number of bits compared to H.264/AVC (Advanced Video Coding) standard, at the cost of $3\times$ computational effort increase. Sum of Absolute Transformed Differences (SATD) calculation is a compute-intensive task inside video encoder. This work proposes the use of an approximation technique to reduce the computational effort of SATD calculation. SATD approximation is performed by pruning Hadamard Transform least significant coefficients. To choose the order of coefficient pruning, we analyse the magnitude of Hadamard Transform coefficients. Our experiments with real video sequences and HEVC reference software demonstrate that 11 Hadamard Transform coefficients can be pruned to calculate the approximate SATD with a compression/quality result still better than estimating the distortion with Sum of Absolute Differences, the simplest distortion metric used in HEVC encoder software. The compression/quality results on prunning 1 to 11 coefficients vary from 0.01% to 0.77% of Bjontegaard Delta Bit Rate increase, saving up to 46.84% of arithmetic operators in a SATD hardware architecture.**

***Index Terms*—SATD, Approximate Computing, Low power, Video coding**

I. Introduction

The increasing use of compute-intensive mobile applications, that requires high performance at low energy consumption levels, made the search for optimizations in video transmissions and storage a vast field of scientific research. In a digital video, a sequence of ten minutes in the Sony 4K (4096x2160 pixels) with 30 frames per second and 24 bits per pixel requires 477 GB to be stored [1]. That's why video coding is so important.

One of the most efficient video coding standards is High Efficiency Video Coding (HEVC). With HEVC it is possible to compress a video with approximately half the number of bits compared to its predecessor, the H.264/AVC (Advanced Video Coding) standard, maintaining practically the same quality characteristics of the original video [2]. Such compression gain is achieved by the introduction of new coding tools that increase the computational effort of video encoder by up to $3\times$ when compared to H.264/AVC [3].

In this context, approximate computing emerged as a promising paradigm to alleviate computational effort of applications [4]. The fundamental principle is focused on managing the trade-off between precision and quality, assuming that many applications are error-tolerant. Video coding is considered an error-tolerant application because of the limitations of the human visual system and since users can deal with quality loss in favor of real-time experience in many video applications.

Motion Estimation (ME) is the most compute-intensive task of video encoders. ME searches for the best block to be used as a prediction for each block to be encoded from a set of previously processed blocks (called reference blocks). Such choice is estimated by distortion metrics, e.g. Sum of Absolute Differences (SAD) and Sum of Absolute Transformed Differences (SATD). SAD estimates the distortion by adding the absolute values of pixel-by-pixel differences of the evaluated block and a reference block. SATD is a more refined metric, due to the use of a 2-D Hadamard Transform (HT), which provides a better distortion estimation than SAD. The use of SATD in the fractional stage of ME improves bit-rate and video quality of an HEVC encoder in 2.2% and 0.16 dB [5]. According to [3], the SATD corresponds up to 18% of the total execution time of HEVC encoder reference software (HM) [6].

This work proposes the approximation of SATD calculation through HT coefficient pruning to reduce computation effort of video encoding. We analyze the importance of each coefficient of HT in the final SATD value, and the impact on video quality and the saving of arithmetic operators when least significant HT coefficients are pruned. The analysis is conducted considering HEVC standard video encoder software and real video sequences. This analysis is used to design an approximate SATD hardware architecture. Before moving to the main contribution of this work (Section III), section II introduces background and related work discussion. Section IV presents the results and a comparison with the related work. Section V concludes the work and suggest future works.

II. Background and Related Work

A. Sum of Absolute Transformed Differences (SATD)

SATD definition is shown in (1). It is calculated through the addition of the absolute values of the coefficients obtained from the 2-D Hadamard Transform (HT) of a residual block, *W*, as defined in (2). Equation (3) shows the 4×4 Hadamard Matrix (*H*). The residual block (*W*) is obtained by the pixel-by-pixel difference between a block from the current picture, to be encoded, and a candidate block to be used as prediction. Larger Hadamard matrix, e.g. 8×8 is also possible, but this work focuses on the 4×4 Hadamard matrix size.

$$SATD = \sum_{i,j} |HT_{(i,j)}| \tag{1}$$

$$HT = H \cdot W \cdot H^T \tag{2}$$

$$H = \frac{1}{2}\begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \end{bmatrix} \tag{3}$$

One can notice that the SATD algorithm has larger computational cost than the simpler SAD due the presence of the 2-D Hadamard Transform. For instance, the calculation of SATD with 4×4 Hadamard matrix size has $5.2\times$ more additions than SAD calculation [7].

B. Related Work

As previously mentioned, the SATD is a compute intensive task which demands acceleration in order to achieve real-time and energy efficiency. One practiced approach is the design of SATD hardware architecture targeting energy-efficient implementation.

In [8] an 8x8 SATD architecture was proposed. Their approach is based on vertical and horizontal HTs that are interleaved by a transposed buffer implemented with registers and multiplexers. In [9] two types of buffer are explored inside the 2-D HT: the transposed (TB-SATD) and linear (LB-SATD) buffers. The former topology reduces the number of cycles to compute SATD but increases the area, while the latter reduces the area and the penalty is the increase in the number of cycles per SATD. These works [8], [9] propose SATD architectures which consume significant amount of power and resources.

Another approach is to apply approximate computing techniques into SATD to balance the trade-off between quality and power-performance scenarios. In [7] a pruning-based algorithm is proposed to discard least significant HT coefficients for the SATD computation. The authors are focused on explore different levels of pruned architectures for ASIC implementation. The work proposes the analysis of the magnitude of each coefficient in HT to determine how much coefficients can be pruned (discarded from HT calculation) and also to decide the order the coefficients are pruned, starting from the least significant coefficient. The limitation of the work in [7] is that the magnitude of the coefficients are obtained as an average of only 64 frames of 4 video sequences.

This work extends the analysis presented in [7] by using 17 videos of the Common Test Conditions (CTC) [10] and also by analyzing all the frames of those videos. Next section details the methodology of this work.

III. Methodology

The first step of our analysis is to divide the set of 17 videos of the CTC [10] into two sets in a range of 75% for the analysis and 25% for the test/evaluation. This way, 13 videos are used for the average magnitude coefficient analysis, and 4 videos are used for testing the quality/compression when the HT coefficients are pruned. Table I shows the set of 13 videos used for average magnitude coefficient analysis and Table II shows the set of 4 videos used for quality/compression test. Note that in the two sets we have selected videos from different resolutions, frame rates and frame counts.

TABLE I
SET OF VIDEOS USED FOR AVERAGE MAGNITUDE COEFFICIENT ANALYSIS

Video sequence	Frame rate (fps)	Resolution	Frame count
BlowingBubbles	50	416x240	500
BQSquare	60	416x240	600
BasketballPass	50	416x240	500
BQMall	60	832x480	600
BasketballDrill	50	832x480	500
RaceHorsesC	30	832x480	300
Kimono	24	1920x1080	240
Cactus	50	1920x1080	500
BasketballDrive	50	1920x1080	500
BQTerrace	60	1920x1080	600
NebutaFestival	60	2560x1600	300
PeopleOnStreet	30	2560x1600	150
SteamLocomotiveTrain	60	2560x1600	300

TABLE II
SET OF VIDEOS USED FOR QUALITY/COMPRESSION ANALYSIS

Video sequence	Frame rate (fps)	Resolution	Frame count
RaceHorses	30	416x240	300
PartyScene	50	832x480	500
ParkScene	24	1920x1080	240
Traffic	30	2560x1600	150

HEVC encoder reference software, called HEVC Test Model (HM) [6], written in C++ language, was modified to calculate the average magnitude of the HT coefficients of SATD 4×4 function. Coefficients of the same position in the Hadamard matrix of all the SATD executions of one video are summed. The final result is divided by the number of SATD function executions. The result is a 4×4 matrix with the average magnitude of the HT coefficients. This analysis was done for each one of the 13 videos shown in Table I. After this analysis, the importance of each HT coefficient to the final SATD calculation are normalized and the result is shown in Figure 1. Note that the most important coefficient is located at the top-left position of the 4×4 matrix.

100,00%	53,68%	55,31%	38,73%
31,50%	14,81%	20,71%	16,32%
58,10%	25,93%	36,50%	27,52%
39,15%	19,37%	26,99%	21,23%

Fig. 1. Importance of each coefficient of the 4×4 Hardamard Transform for the SATD Calculation.

The importance of the HT coefficients shown in Figure 1 were used to define the order in which the HT coefficient pruning is conducted. Our coefficient pruning algorithm is very simple. First, we have ordered the coefficients by the increasing importance obtained with our average coefficient magnitude analysis shown in Figure 1. Then, we prune the HT coefficients from 1 to 15 starting from the least important coefficients, in a increasing order of importance, and compared with the precise version of SATD with no coefficient pruning. The coefficient pruning is done by changing the value of the HT coefficient to zero.

IV. Results and Comparison with related work

The four evaluated videos (Table II) are encoded with HM software using four Quantization Parameters (QP), $QP = 22, 27, 32, 37$ with Random Access configuration. All the remaining HM parameters are kept as default. The bit rate (BR) and Peak Signal-to-Noise Ratio (PSNR) values of the precise and each approximate solution were obtained to calculate the Bjontegaard Delta Bit Rate (BD-BR) [11] values. BD-BR is an average bit rate difference (in percentage) of the approximate solution (with coefficient pruning) compared to the precise SATD solution when considering the four QP points. Positive and small BD-BR values represent a small drop in video quality of the approximate solution compared with the precise one.

Table III presents the compression/quality results of the four evaluated videos in terms of BD-BR. See Table II for the information of the evaluated videos. We have also evaluated the computation with SAD compared to the approximate SATD since it is a lower bound for the number of pruned coefficients. Since SAD computation is simpler than SATD, there is no reason to use an approximate version of SATD, that is more complex than SAD, if it results in a worse BD-BR result than SAD compared to the precise SATD. Observing Table III we conclude that SATD can be approximated by pruning up to 11 coefficients and the impact in BD-BR increase in still lower than using SAD. This behaviour also repeats for three evaluated videos (RaceHorses, ParkScene and Traffic). The PartyScene video will benefit for SATD approximation if we prune up to 9 coefficients, because the BD-BR results of pruning more coefficients will be worse than using SAD, that has simpler computation. This result shows that the BD-BR results on coefficient pruning depend on video content.

TABLE III
Compression/quality results (BD-BR) for the 4 evaluated videos and average BD-BR results

Pruned Coeff.	BD-BR				
	RaceHorses	PartyScene	ParkScene	Traffic	Avg.
1	0.06%	0.05%	-0.04%	-0.02%	0.01%
2	0.10%	0.06%	0.04%	-1.06%	-0.22%
3	0.12%	0.10%	0.00%	-0.02%	0.05%
4	-0.10%	0.16%	0.04%	0.04%	0.03%
5	2,11%	0.21%	0.09%	-0.01%	0.60%
6	0.37%	0.29%	0.05%	-0.03%	0.17%
7	0.58%	0.41%	0.09%	0.03%	0.28%
8	0.83%	0.53%	0.18%	0.07%	0.40%
9	0.93%	0.67%	0.15%	0.13%	0.47%
10	1.25%	0.82%	0.23%	0.16%	0.61%
11	1.52%	0.96%	0.36%	0.24%	0.77%
12	1.91%	1.27%	0.44%	0.29%	0.98%
13	2,10%	1.41%	0.52%	0.30%	1.08%
14	2,65%	1.69%	0.73%	0.60%	1.42%
15	3,40%	2,15%	0.85%	0.69%	1.77%
SAD	1.79%	0.72%	0.37%	0.28%	0.79%

We have also analyzed the impact of approximating SATD by pruning HT coefficients on the saving of arithmetic operators in a SATD hardware architecture. Figure 2 shows a SATD hardware architecture with 4×4 HT and Sum of Absolute Values (SAV) module. The figure show examples of the number of adders/subtractors that can be discarted from the architecture when one, six and ten coefficients are pruned, considering our magnitude coefficient analysis. For example, if one cofficient is pruned, only two adder/subtractor can be discarded from the architecture; if six coefficients are pruned, it is possible to eliminate sixteen adder/subtractors, and so on.

Table IV shows the results of operators (adders/subtractors) saved for each approximate version and SAD, compared to precise SATD version. Note that we have shown only the approximate versions from 1 to 11 pruned coefficients, because in our compression/quality results the maximum number of pruned coefficients that present BD-BR results still lower than SAD is with 11 coefficients, on average of the four videos evaluated. The approximate SATD version with 11 coefficients saves 46.84% of the arithmetic operators and has a BD-BR increase of only 0.77%.

TABLE IV
Average BD-BR

Pruned coefficients	Number of saved operators	% of saved operators	Average BD-BR
1	2	2.53%	0.01%
2	5	6.33%	-0.22%
3	7	8.86%	0.05%
4	10	12.66%	0.03%
5	14	17.72%	0.60%
6	16	20.25%	0.17%
7	18	22.78%	0.28%
8	22	27.85%	0.40%
9	30	37.97%	0.47%
10	35	44,30%	0.61%
11	37	46.84%	0.77%
SAD	64	81.01%	0.79%

Table V compares the methodology proposed in this work with the work in [7]. As we can see, by using the proposed methodology we can prune 11 HT coefficients, saving up to 46,84% of arithmetic operators of the SATD architecture. By

TABLE V
Comparison with related work

Number of pruned coefficient	This work		[7]	
	Number of saved operators	% of saved operators	Number of saved operators	% of saved operators
1	2	2,53%	2	2,53%
2	5	6,33%	4	5,06%
3	7	8,86%	6	7,59%
4	10	12,66%	10	12,66%
5	14	17,72%	12	15,19%
6	16	20,25%	14	17,72%
7	18	22,78%	16	20,25%
8	22	27,85%	20	25,32%
9	30	37,97%	22	27,85%
10	35	44,30%	30	37,97%
11	37	46,84%	-	-

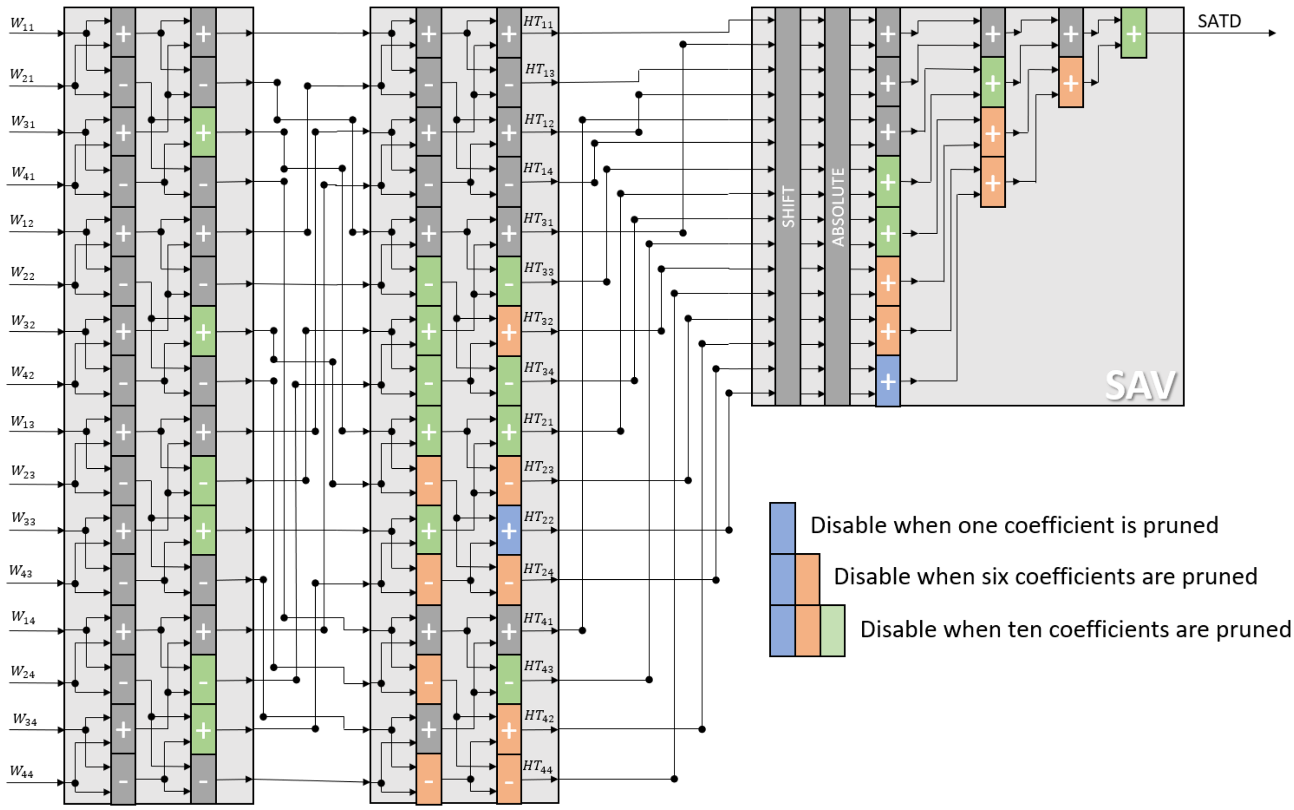

Fig. 2. Block Diagram of the SATD Hardware Architecture with 4×4 HT.

using the methodology presented in [7] we can prune up to 10 coefficients, saving up to 37,97% of arithmetic operators. Both works consider the comparison with BD-BR obtained with SAD calculation to manage the maximum number of coefficients the algorithm can prune.

V. Conclusions and Future Works

This work proposed an analysis of SATD approximation based on coefficient pruning of Hadamard Transform. HEVC reference software was modified to obtain the average magnitude of 4×4 Hadamard Transform coefficients with 13 videos. This magnitude analysis was used to determine the order of coefficients to be pruned. The analysis on four video sequences show that on average 11 Hadamard Transform coefficients can be pruned to calculate the approximate SATD with the a compression/quality result still better than estimating the distortion with SAD. Overall, the compression/quality results on pruning 1 to 11 coefficients vary from 0.01% to 0.77% of BD-BR increase, saving up to 46.84% of arithmetic operators in a SATD hardware architecture. As future works, we aim to implement the SATD approximate architecture in standard cells flow and obtain area and power results.

Acknowledgment

The authors would like to thank CNPq for the financial support to this research.

References

[1] C. M. Diniz, "Dedicated and reconfigurable hardware accelerators for High Efficiency Video Coding standard." Tese de Doutorado, Universidade Federal do Rio Grande do Sul, 2015.

[2] G. J. Sullivan, J. R. Ohm, W. J. Han, and T. Wiegand, "Overview of the high efficiency video coding (HEVC) standard," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 22, no. 12, pp. 1649–1668, Dec 2012.

[3] J. Vanne, M. Viitanen, T. D. Hamalainen, and A. Hallapuro, "Comparative rate-distortion-complexity analysis of hevc and avc video codecs," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 22, no. 12, pp. 1885–1898, Dec 2012.

[4] J. Han and M. Orshansky, "Approximate computing: An emerging paradigma for energy-efficient design," in *18th IEEE European Test Symposium (ETS)*. IEEE, 2013, pp. 1–6.

[5] G. Correa, P. Assuncao, L. Agostini, and L. A. da Silva Cruz, "Performance and computational complexity assessment of high-efficiency video encoders," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 22, no. 12, pp. 1899–1909, Dec 2012.

[6] "High Efficiency Video Coding Test Model (HM) v. 16.0," http://HEVC.hhi.fraunhofer.de.

[7] L. Soares, "A novel pruned-based algorithm for energy-efficient SATD operation in the hevc coding." *Symposium on Integrated Circuits and Systems Design (SBCCI)*, 2016.

[8] B. Silveira, C. Diniz, M. B. Fonseca, and E. Costa, "SATD hardware architecture based on 8x8 hadamard transform for hevc encoder," in *Proc. IEEE International Conference on Electronics, Circuits, and Systems*, Dec 2015, pp. 576–579.

[9] I. Seidel, A. B. Bräscher, J. L. Güntzel, and L. Agostini, "Energy-efficient SATD for beyond hevc," in *2016 IEEE International Symposium on Circuits and Systems (ISCAS)*, May 2016, pp. 802–805.

[10] F. Bossen, "Common test conditions and software configurations," in *JCT-VC document no. L1100*, Jan 2013.

[11] G. Bjontegaard, "Calcuation of average PSNR differences between RD-curves," in *Doc. VCEG-M33 ITU-T Q6/16*, Apr 2001.