Advances in Engineering Research, volume 127
3rd International Conference on Electrical, Automation and Mechanical Engineering (EAME 2018)

Intra/Inter-frame Joint WPP Coding Algorithm on Multicore Platform

Chuanwei Yin[1,2,3], Dong Hu[1,2,3] and Tao Gu[1,2,3]
[1]Education Ministry's Key Lab of Broadband Wireless Communication and Sensor Network Technology
[2]Education Ministry's Engineering Research Center of Ubiquitous Network and Heath Service
[3]Jiangsu Province's Key Lab of Image Procession and Image Communications, Nanjing University of Posts and Telecommunications, Nanjing, 210003, China

***Abstract*—Although wavefront parallel processing (WPP) proposed in the HEVC standard and various inter frame WPP algorithms can achieved comparatively high parallelism, their scalability for its parallelism is still very limited due to various dependencies introduced in spatial and temporal prediction in HEVC. In this paper, through pixel correlation analysis, establishment of CTU node model, multi-core resource allocation strategy, we proposed an intra/inter-frame joint WPP coding algorithm for multi-core platform which can significantly improve the parallelism, while achieved good results in bit rate, PSNR, and acceleration ratio. Experiments on standard HD test sequence show that the proposed algorithm can lead to up to 2x, 1.3x speed up compared with the original WPP and IWF parallelism.**

Keywords—video coding; HEVC; multicore platform; high parallelism; WPP

I. INTRODUCTION

In order to improve video coding efficiency, ISO/IEC MPEG and ITU-TVCEG have formed Joint Collaborative Team on Video Coding (JCT-VC) to develop the new video coding standard called High Efficiency Video Coding (HEVC) [1], which was finalized in early 2013. By introducing plenty of enhanced coding tools, HEVC achieves about 50% bit-rate reductions at similar mean opinion score and about 40% bit-rate reductions at similar PSNR [2] compared with the previous standard H.264/AVC [3].

However, the improvement of coding efficiency also brings great increment of computational complexity [4]. In order to cope with the high processing demands, HEVC includes several parallelization schemes, which makes multicore coding possible. In this paper, we focus on the parallelism analysis of WPP and its implementation on multi-core platforms.

Several related works focus on improving parallelism of HEVC based on WPP. Zhang [5] implemented WPP on many-core platform for real-time HEVC encoder by using ping-pang storage, data reuse and try and wait mechanism. Chi [6] presented a novel approach called Overlapped WaveFront (OWF) to enhance the parallel efficiency of WPP.

Different from their works, this paper proposes a new method to achieve intra/inter frame joint WPP. The rest of this paper is organized as follows. The II part briefly introduces the parallel strategy of WPP in HEVC. The III part analyzes WPP Parallel extensibility dependence and proposes new algorithm based on this. The IV part designs and implements the joint WPP algorithm. The experimental design ideas and results are given in Section V, and further demonstrate the analysis of the experimental results.

II. WAVEFRONT PARALLEL PROCESSING

Previous coding standards, especially H.264/AVC, have included frame-level, fragment-level, or macroblock-level parallelism [3]. However, these methods expose limitations in practical applications such as limited scalability, significant coding loss or high memory requirements [6]. In order to overcome these limitations, HEVC introduced some new parallel strategies: WPP and Tiles [1]. Both tools can subdivide each picture into multiple partitions that can be processed in parallel. When WPP is enabled, one slice is divided into several CTU lines. Compared to Slice and Tiles, this division does not destroy the dependencies at the line bound areas. HEVC performs the encoding process in raster scan order, which means that the CTU being processed needs its left, top left, top right and top right CTUs available. Therefore, transfer of at least two CTUs is performed between consecutive lines of parallel-processed CTUs. In addition, the probability of context-based Adaptive Binary Arithmetic Coding (CABAC) [7] is propagated from the second CTU of the previous row to further reduce the coding loss.

When performing WPP encoding, a CTU row into which an image is divided is allocated to a core resource in a multi-core system. However, due to the inter dependence of the CTU units, only the mutual dependency can be eliminated for parallel processing. . In the CTU row-level coding, each row processed by each core resource must be delayed by two CTU units from the previous CTU row for parallel processing [8]. Therefore, this paper optimizes the WPP algorithm, fully combines the algorithm with the hardware platform, and improves the coding efficiency.

III. ANALYSIS ON PARALLELISM OF WPP

In order to describe the dependencies between CTUs, $C_{i,j,kis}$ used here to represent the k-th CTU unit of the *j*-th CTU row in the *i*-th frame in the coding order, Dep_F,inter($C_{i,j,k}$), DepF,intra($C_{i,j,k}$) represents the CTU units on which $C_{i,j,k}$ depends upon performing parallel inter-frame or intra-frame encoding using the F method.

A. Current Encoding Frame

In the CTU of the current coded frame, when WPP is

enabled, the CABAC initialization of each CTU line starts only after coding from the second CTU block of the previous CTU line [9]. This means that encoding the current CTU block waits until the left, top left, top, and top right CTU blocks are encoded, it can be expressed as:

$$Dep_{WPP,\text{int}ra}(C_{i,j,k}) = \{C_{i,j-1,k-1}, C_{i,j-1,k}, C_{i,j-i,k+1}, C_{i,j,k-1}\} \quad (1)$$

As shown in Figure I, the gray CTU block that satisfies the intra-dependency relationship can be encoded at the same time.

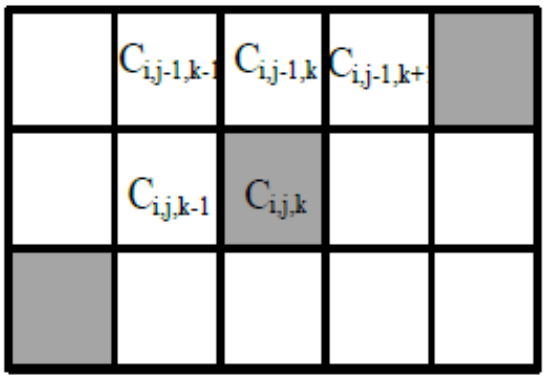

FIGURE I. CTU INTRA-FRAME DEPENDENCY RELATIONSHIP

B. *Neighboring Encoding Frame*

In continuous frames, the CTU block in the coded frame can not only consider intra-frame pixel correlation, but also can refer to inter-frame correlation information reference[10], that is, the dependency at this time comes from the motion estimation process.

As shown in Figure II, white represents a coding CTU, dark color represents a CTU being coded, and gray represents an uncoded CTU. Therefore, when the CTU block of current frame is coded, it depends on the corresponding reference area in the reference frame which is that the label area in the left drawing. The part of the area corresponding to the coded frame contains a certain motion buffer area between frames. Therefore, only when the coding of the reference area in the reference frame is completed, the corresponding block coding in the current frame can be performed.

$$Dep_{\text{int}er}(C_{i,j,k}) = \{C_{i1,j+l_h,W-1} \mid i1 \in ref(i)\} \quad (2)$$

Where W and L are the number of horizontal and vertical CTUs in a frame, and l_h is the vertical component of the motion vector.

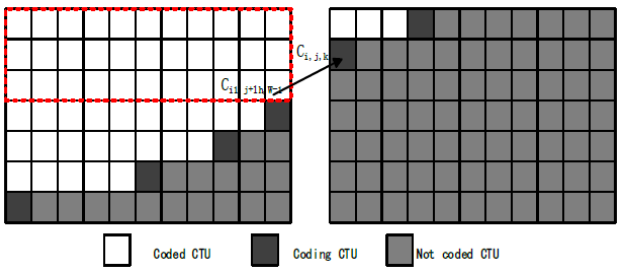

FIGURE II. CTU INTER-FRAME DEPENDENCY RELATIONSHIP

IV. IMPLEMENTATION OF JOINT WPP

By analyzing the advantages of the CTU row-level parallelism and the inter-frame and inter-frame dependency of the CTU block, an intra-frame/inter-frame joint WPP coding algorithm for optimizing the wavefront parallel processing is proposed, and based on multi-core processor hardware platform to implement this algorithm.

A. *Mathematical Model of WPP*

Here, the dependency relationship between CTU blocks is described by means of parent-child nodes, as shown in Figure III, when the first CTU line is encoded in the order of raster scan, the second CTU line will begin normal encoding work after the first CTU line completes encoding the two CTU blocks, and the analogy cycle will continue. The CTU relationship is expressed as the iterative layer relationship shown in the figure. This figure shows that there is no interdependence between CTU blocks in the same iteration layer, and parallel coding schemes can be implemented.

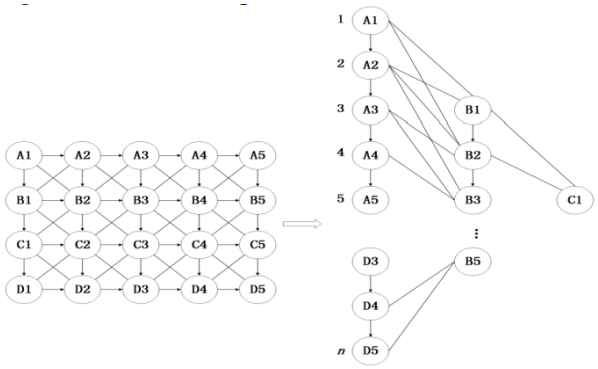

FIGURE III. CTU NODE MODEL

In order to visually describe the impact of WPP and thread resource strategies on parallel coding efficiency, the parallelism R is introduced here to measure the performance of WPP parallel coding. The formula is as follows:

$$R = \frac{N_{serial}}{N_{parallel}} \quad (3)$$

Where N_{serial} is used to represent the number of iterations required for serial encoding and $N_{parallel}$ is used to represent the number of iterations required for parallel encoding. As shown in Figure IV, this figure shows two mathematical models for the relationship between WPP and core resources. In the figure, one frame of image is divided into (W×H) CTU units according to the basic unit of CTU. According to the above-

mentioned model, the required number of iterations can be obtained as:

$$n = 2 \times (H - 1) + W \tag{4}$$

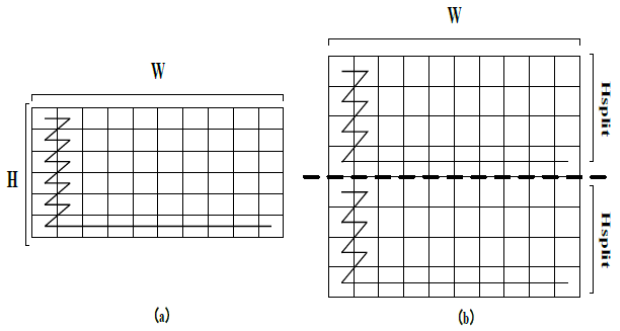

FIGURE IV. MULTICORE RESOURCE ALLOCATION MODEL

For the two relationships between the number of CTU rows for thread resources and image partitioning given in the figure above, the corresponding resource scheduling scheme will be given:

Figure IV(a) indicates that the actual operating environment has a sufficient number of core resources, $Ncore \geq H$. Each CTU line has a thread. After processing, after a CTU line is encoded, the thread resources are idle. Waiting for all CTU lines in the current frame to finish encoding will process the next frame. At this time, the average degree of parallelism is:

$$R_{aver} = W \times \frac{H}{n} \tag{5}$$

Considering the corresponding W and H restrictions, the maximum degree of parallelism is:

$$R_{\max} = \begin{cases} H & W \geq 2H - 1 \\ \dfrac{W+1}{2} & W < 2H - 1 \end{cases} \tag{6}$$

Figure IV(b) shows that the number of core resources in the actual operating environment is limited, $N_{core} < H$. When the encoding of the first CTU line is completed, the current encoding frame has not yet been encoded. In view of this situation, the current coded frame is cut in the horizontal direction to ensure that the degree of parallelism of each branch can be consistent with the actual number of core resources. In this way, when the first CTU line of the first branch completes encoding, the core resources of the line will be transferred to the first CTU line of the next branch. Subsequent core resources will also be redirected after the current branch completes the encoding. The next branch corresponds to the task line. The corresponding degree of parallelism at this time is:

$$R_{aver} = W_{split} \times \frac{H_{split}}{n_{split}} = W_{split} \times \frac{N_{core}}{n_{split}} \tag{7}$$

$$R_{\max} = N_{core} \tag{8}$$

$$n_{split} = \begin{cases} W_{split} \times \left(\dfrac{H_{split}}{N_{core}} \right) + 2 \times (N_{core} - 1) \\ W_{split} \times \left(\dfrac{H_{split}}{N_{core} + 1} \right) + 2 \times (H_{split} \,\%\, N_{core} - 1) \end{cases} \tag{9}$$

For the two data models built for WPP, there are some defects, (a) the core resource has processed a CTU line, the core resource will be idle until the frame is encoded. (b) The core resources is After processing the CTU lines corresponding to all the branches, the core resources will still be idle after the frame encoding is completed. The above disadvantages are improved in the next part, and then propose an intra-frame/inter-frame joint WPP coding algorithm.

B. CTU Inter-frame Dependency Improvement

For the inter-frame dependency relationship analyzed in the third section, the CTU block in the current frame needs to wait for all the coding of the corresponding CTU line in the reference frame to complete. Such dependencies are too cumbersome and reduce the degree of parallelism. Therefore, this part optimizes inter-frame dependencies. For $C_{i,j,k}$ code blocks, $C_{i1,j+L_H,k+L_W}$ can be coded in the reference frame. As shown in Figure V, after the black square code in the reference frame is completed, the current frame black square can be encoded, and the dependency can be expressed as:

$$Dep_{new\_method}(C_{i,j,k}) = \left\{ \begin{array}{l} C_{i1,j1,k1} \mid i1 \in ref(i), \\ 0 \leq j_1 < j + L_H, \\ 0 \leq k_1 < k + L_W \end{array} \right\} \tag{10}$$

L_H and L_W are the vertical and horizontal components of the motion vector, which usually take the value 1.

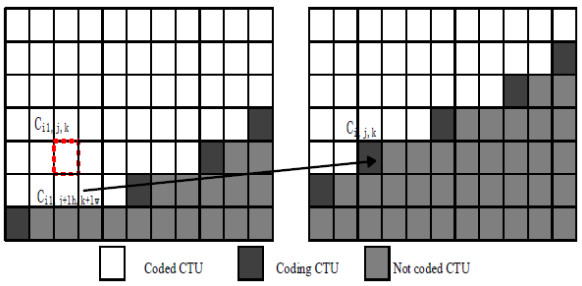

FIGURE V. OPTIMIZED CTU INTER-FRAME DEPENDENCIES

C. Improvement of WPP Mathematical Model

For the improvement of model (a): The current frame is encoded in parallel according to the WPP method. After the $C_{i,j+L_H,k+L_W}$ encoding is completed, if there are currently idle core resources, the next frame can be hung up and performed. The encoding work of $C_{i_1,j,k}$ will make full use of such spare core resources and idled core resources after encoding. Assuming that the subsequent frames have already encoded a CTU block, then, there are:

$$R_{aver} = \frac{W \times H + a}{n} \tag{11}$$

$$R_{\max} = \begin{cases} N_{core} & W \geq 2H - 1 \\ H + t & W < 2H - 1 \end{cases}, t \geq 0 \tag{12}$$

Where t is the number of redundant core resources, for model (b) improvement: Since the number of core resources is limit, the image frames are divided according to the above, and the WPP algorithm is continuously used for parallel processing within the branches. The core resources are processed in the order of the branches. After the core resources in the previous branches have processed the current CTU line, they move to the next branch for processing. After the last branch, when the processed CTU lines are processed, the core resource turns to the next position, one frame of coding. Assume that CTUs have already been encoded in subsequent frames. There are:

$$R_{aver} = W_{split} \times \frac{(H_{split} + y)}{n_{split}} = W_{split} \times \frac{(N_{core} + y)}{n_{split}} \tag{13}$$

$$R_{\max} = N_{core} \tag{14}$$

From the formulae obtained above, the parallelism of the improved algorithm can be significantly improved compared to the unmodified algorithm. At the same time, from the point of view of multi-core processors, it is found that even if the many-core resources are in surplus or in deficit, the improved model makes full use of idle nuclear resources, which significantly improves the efficiency of video coding based on multi-core processors.

D. Algorithm Flow Framework

The intra/inter-frame joint WPP coding algorithm is mainly to make full use of the many-core resources provided by the multi-core processor, establish the node mathematical model based on the wavefront parallel processing, and utilize the inter-dependency of the frames to encode the current coding frame. The idle core resources that have been encoded are fully utilized to handle the frame encoding that satisfies the inter-frame dependency relationship and improves the efficiency of video encoding. The actual steps are as follows:

(1) Open the main thread of the coding work, perform corresponding space application and create a thread resource pool to generate a corresponding encoder

(2) The encoder reads the command line parameters entered by the user and parses them. The parameters are passed into the encoder and the encoder is configured according to the parameters.

(3) Read the number of CPUs of the multi-core processor through the interface to create a corresponding number of threads, and bind the threads to the corresponding TILE core for subsequent parallel encoding processing.

(4) The encoder begins reading the video sequence frame by frame and adds it to the frame-level queue. When the queue capacity reaches the maximum value, the encoder stops reading; otherwise, it continues reading. After the frame is full, a frame Encoder task will be allocated for each frame. Each task has its own memory. All frame Encoder share a thread pool.

(5) Each frame Encoder acquires the frame type and related dependencies of this frame, and starts the real coding work.

(6) After the image is divided into CTU lines, an idle thread is called from the thread pool to encode the CTU line and join the task queue. All CTU lines in the image call idle threads in the thread pool until there are no idle threads. If there is a new CTU row in the task queue, it is in a waiting state. After the CTU line is encoded, it is not necessary to wait for the frame to be completely coded. The thread resource is returned directly to the thread pool for use by other CTU lines in the queue.

(7) According to the frame type information obtained in (5), a multi-line parallel encoding may be performed for a CTU line in an I-frame while satisfying an intra-frame dependency relationship, and for non-I-frames, not only intra-frame parallel coding is performed, but In the time domain, parallel coding in the time domain is required after the inter-frames satisfy the dependency relationship.

(8) After the encoding of one frame is completed, the corresponding stream information will be output, and the new frame will be read again, and a new frame Encoder task will be created until all the frames are coded. Finally destroy the corresponding thread resources and space.

Consequently, the proposed scheme can be divided into two task levels, as shown in Figure VI. In the figure, FT is a frame-level task and CT is a CTU line-level task. When the video data is loaded, it is divided into a frame-level task queue. Then, each frame-level task applies for several thread resources to the resource pool, and generates a CTU row-level task queue for processing. After a frame is processed, the thread resources of the frame are released and the next frame level is continued until all frame levels are reached. The task is completed and finally the encoded data stream is output.

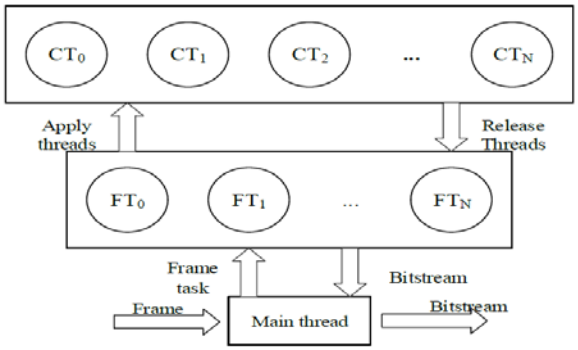

FIGURE VI. TWO-LEVEL PRIORITY TASK QUEUE

V. EXPERIMENTAL RESULTS

A. Experimental Design

Based on the TILE-Gx36 multi-core platform, the algorithm of this paper was experimented. At the same time, two sets of comparison experiments are set up. The first one is the WPP algorithm that does not take into account the inter-frame correlation and the second one is the IFW algorithm that takes into account the inter-frame correlation. Since the experiment involves the dependency of the CTU in the time domain, the coding frame type is divided into two cases: IPPPP and IBBBBP. The indicators used for evaluating the experimental performance are bit rate, PSNR, and acceleration ratio; with the increase in the number of threads, the trend of the parallel speedup of the three algorithms is compared. The video sequence used in this paper is 1600p Traffic, 1080p BasketballDrive and 720p Fourpeople.

B. Experimental Results

From the experimental data (Table I), we can get line charts of each algorithm which show the relationship of speedup and the number of threads (See Figure VII-IX).

TABLE I COMPARISON RESULTS OF THREE ALGORITHMS

Frame type	Sequence name	resolution	WPP			IFW			Intra/Inter-frame joint WPP		
			PSNR (dB)	Bit rate (bps)	Speed up	PSNR (dB)	Bit rate (bps)	Speed up	PSNR (dB)	Bit rate (bps)	Speed up
IPPPP	Traffic	1600p	31.422	3111.93	3.56	31.462	3246.68	6.3	31.367	3265.43	8.92
	BasketballDrive	1080p	33.086	3098.74	3.37	33.103	3138.55	6.14	33.066	3098.01	7.79
	Fourpeople	720p	23.167	712.09	2.39	23.298	796.34	5.69	23.112	811.48	7.01
IBBBP	Traffic	1600p	32.087	3243.55	5.45	32.118	3395.56	7.43	32.036	3403.01	9.22
	BasketballDrive	1080p	35.126	3122.88	4.39	35.139	3209.08	6.98	35.196	3122.21	8.14
	Fourpeople	720p	26.253	792.03	3.09	26.233	819.11	5.03	26.247	823.89	7.98

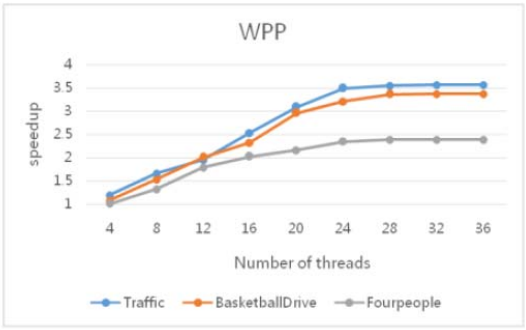

FIGURE VII. WPP SPEEDUP

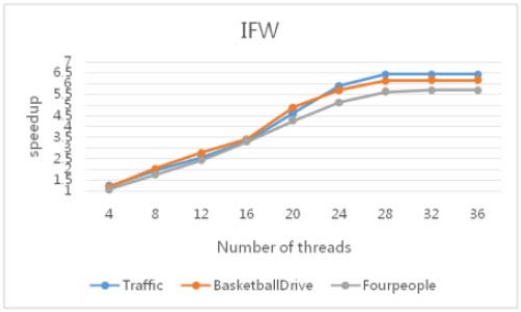

FIGURE VIII. IFW SPEEDUP

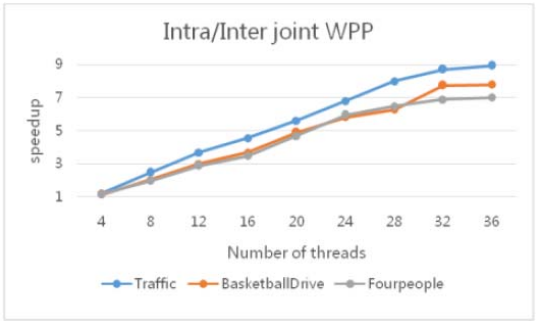

FIGURE IX. JOINT WPP SPEEDUP

C. Analysis of Results

From the line chart, it can be seen that the parallel speedup ratio increase gradually in the process of increasing thread resources of WPP, IFW, and Intra/frame joint WPP coding algorithm. Eventually, there will be peaks, but they will appear as peaks. The number of corresponding threads is different, and WPP first appeared at the peak, mainly due to taking into account the parallelism between frames; IFW will be earlier than the intra/inter-join WPP coding scheme, mainly due to the long waiting time to satisfy the dependencies in the time domain. This also means that based on the existing many-core platform, the intra-frame/inter-frame joint WPP coding scheme has higher thread resource utilization than WPP and IFW without significantly affecting the coding quality.

VI. Conclusion

In this paper, we firstly introduced the parallel strategy of WPP. Then we analyze the maximum and average degree of parallelism that WPP can theoretically achieve by establishing a mathematical model. Finally, the improved joint WPP algorithm is implemented on the TILE-Gx36 multi-core platform. Experiments on standard HD test sequences show that the proposed algorithm can lead to up to 2x, 1.3x speed up compared with the original WPP and IWF parallelism. Therefore, the Intra/Inter-frame Joint WPP Coding Algorithm can significantly improve the performance of video encoding.

References

[1] G. Sullivan, J. Ohm, W.-J. Han, and T. Wiegand, "Overview of the highefficiency video coding (hevc) standard," Circuits and Systems for VideoTechnology, IEEE Transactions on, vol. 22, no. 12, pp. 1649–1668, Dec2012

[2] J. Ohm, G. Sullivan, H. Schwarz, T. K. Tan, and T. Wiegand, "Comparison of the coding efficiency of video coding standards #x2014;including high efficiency video coding (hevc)," Circuits and Systems for VideoTechnology, IEEE Transactions on, vol. 22, no. 12, pp. 1669–1684, Dec2012.

[3] T. Wiegand, G. Sullivan, G. Bjontegaard, and A. Luthra, "Overview ofthe h.264/avc video coding standard," Circuits and Systems for VideoTechnology, IEEE Transactions on, vol. 13, no. 7, pp. 560–576, July2003.

[4] F. Bossen, B. Bross, K. Suhring, and D. Flynn, "Hevc complexityand implementation analysis," Circuits & Systems for Video TechnologyIEEE Transactions on, vol. 22, no. 12, pp. 1685 – 1696, 2012.

[5] S. Zhang, X. Zhang, and Z. Gao, "Implementation and improvement of wavefront parallel processing for hevc encoding on many-core platform," in Multimedia and Expo Workshops (ICMEW), 2014 IEEE International Conference on, 2014, pp. 1–6.

[6] C. C. Chi, M. Alvarez-Mesa, B. Juurlink, G. Clare, F. Henry, S. Pateux, and T. Schierl, "Parallel scalability and efficiency of hevc parallelization approaches," Circuits & Systems for Video Technology IEEE Transactions on, vol. 22, no. 12, pp. 1827–1838, 2012.

[7] V. Sze and M. Budagavi,"High throughput cabac entropy coding in hevc ," Circuits & Systems for Video Technology IEEE Transactions on, vol. 22, no. 12, pp. 1778 – 1791, 2012.

[8] F. Henry and S. Pateux, Wavefront Parallel Processing,document JCTVC-E196, Mar. 2011.

[9] C. Yan, Y. Zhang, F. Dai, X. Wang, L. Li, and Q. Dai, "Parallel deblocking filter for HEVC on many-core processor," Electron. Lett.,vol. 50, no. 5, pp. 367–368, Feb. 2014.

[10] 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.