Parallel Intra Coding for HEVC on CPU plus GPU Platform

Juncheng Ma[1], Falei Luo[2], Shanshe Wang[1], Nan Zhang[3], Siwei Ma[1,4]
Email: jcma@pku.edu.cn falei.luo@vipl.ict.ac.cn sswang@jdl.ac.cn zhangnan@ccum.edu.cn swma@pku.edu.cn
[1]Institute of Digital Media & Cooperative Medianet Innovation Center, Peking University, Beijing, China
[2]Institute of Computing Technology, Chinese Academy of Sciences, Beijing, China
[3]School of Biomedical Engineering, Capital Medical University, Beijing, China
[4]Peking University Shenzhen Graduate School, Beijing, China

Abstract— **In High Efficiency Video Coding (HEVC), the intra coding performance is significantly improved due to the recursive splitting structure and up to 35 intra prediction modes. However, the computational complexity of intra coding increases largely as well. In this paper, a fast intra coding scheme is proposed based on CPU and GPU cooperation. Firstly, the intra prediction of variable blocks is performed in parallel on multi-cores GPU. Secondly, the intra prediction mode with minimum Sum of Absolute Difference (SAD) cost is selected and transmitted to the host CPU. Instead of exhaustively searching all the intra modes in Rough Mode Decision (RMD) process, the mode returned by the GPU is directly selected. Lastly, the texture gradient of each coding unit (CU) is assessed during parallel intra prediction, then used by the CPU for fast CU size decision. Experiment results show that the proposed parallel intra coding method achieves up to 62% complexity reduction with acceptable coding performance loss.**

Index Terms— **HEVC, intra prediction, GPU, gradient, CU splitting**

I. INTRODUCTION

High efficiency video coding (HEVC) is the latest video coding standard developed by the Joint Collaborative Team on Video Coding (JCT-VC), which is formed by tow standardization teams: the ISO/IEC MPEG and ITU-T VCEG [1]. With the newly introduced quadtree structure and richer intra/inter prediction modes, HEVC greatly outperforms the classic H.264/AVC standard [2] in terms of coding performance. However, the coding complexity increment that comes along makes it challenging for HEVC to be employed in real-time applications.

For intra coding, the coding unit (CU) size varies from 64x64 to 8x8, and 4x4 prediction unit (PU) is used at the largest CU depth. In rate distortion optimization (RDO) process, the transform unit (TU) can be further partitioned to smaller ones. Moreover, up to 35 intra prediction modes have to be enumerated exhaustively for the optimal one. Compared to the fixed 16x16 macroblocks and only 9 intra modes in H.264/AVC, the intra coding of HEVC is much more time-consuming [3].

To reduce the coding complexity of HEVC intra coding, many fast intra prediction methods are proposed. Shen et al. presented a fast CU size decision method for intra coding based on texture homogeneity and spatial correlations, achieving considerable coding time saving [4]. However, the texture homogeneity measurement process might cause extra complexity overhead. W. Jiang et al. considered gradient assessment for intra coding blocks to support fast mode decision with fixed threshold, which might be not suitable for all test sequences [5].

In recent years, due to the rapid development of the Graphic Processing Unit (GPU), it becomes a main trend to use General Purpose GPU (GPGPU) for parallel speedup of video coding. At the same time, the "Compute Unified Device Architecture" (CUDA), published by NVIDIA [6], makes GPU paralleling more programming-friendly. Therefore, it is possible to implement parallel intra prediction for large scale of coding blocks in the video sequence. However, for parallelization in intra prediction using graphics hardware, it is highly challenging. Specifically, there's high reconstruction dependence between the intra PU and its neighbour blocks, causing frequent synchronization when dealing current and reference samples at the same time. In this paper, an original picture based intra coding scheme is proposed using graphic hardware with the two following contributions as follows. Firstly, a parallel and fast intra prediction scheme at the GPU is proposed. Before intra coding for one slice, the best intra prediction mode with minimum SAD cost for every possible blocks of all CTUs is determined concurrently at the GPU, and transmitted back to the host CPU. For each intra block, instead of the time-consuming RMD process, the intra modes returned by GPU along with the Most Probable Modes (MPMs) are directly used for RDO decision. Secondly, a fast CU splitting and pruning algorithm is proposed based on parallel texture gradient measuring using classic Sobel operator. With asynchronous workflow and effective thread allocation, no extra computational overhead is brought in.

The rest of this paper is organized as follows. Section II describes the overview of intra coding techniques in HEVC. Section III details the proposed paralleling method for intra prediction and gradient measuring, along with the fast mode and CU size decision based on the GPU respectively. Experimental results and analysis are presented in Section IV. Finally Section V concludes this paper.

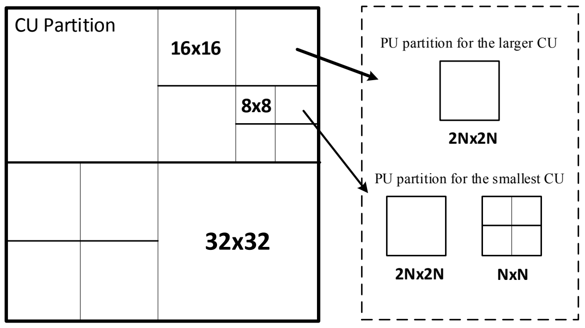

CU Partition
16x16
8x8
32x32
PU partition for the larger CU
2Nx2N
PU partition for the smallest CU
2Nx2N
NxN

Fig. 1 Example for CU and PU partition of intra CTU in HEVC

II. Overview of Intra Coding in HEVC

In HEVC, an intra slice is divided on basis of sequenced coding tree units (CTUs), which are further partitioned recursively into CUs. The maximum CU size is the same as a CTU (up to 64x64), and the minimum size is 8x8. A CU itself is considered as a PU or partitioned into 4 smaller PUs when it is the smallest available CU (8x8 as default), as shown in Fig. 1. At every CU partition depth, whether to further split the current CU depends on the rate-distortion (RD) cost of the current CU and 4 smaller CUs.

For Intra prediction mode, there're up to 35 modes in HEVC intra coding, including 33 angular prediction modes, one DC mode and one Planar mode, as shown in Fig. 2. In the HEVC reference software HM, a fast intra mode decision algorithm is adopted to reduce the complexity of the encoder. The mode decision process can be divided into two steps. The first step is called Rough Mode Decision (RMD). The 35 modes are checked using the absolute sum of Hadamard transformed difference (HAD) cost, and a sub-set (named O with K modes) with lower cost is generated. The HAD cost is described as the follow formula:

$$J_{Had} = D_{Had} + \lambda_{mode} \cdot R_{mode} \tag{1}$$

where D_{Had} indicates the HAD amplitude, λ_{mode} is the Lagrange multiplier that determines the trade-off between rate and distortion, and R_{mode} presents the bits of mode overhead. Then at most 3 Most Probable Modes (MPMs) derived from spatial and temporal neighbour blocks are added to O if not duplicated.

In the second step, the full RDO process is performed on all the candidates in O, and the intra prediction mode with minimum RD cost is selected for actual coding. The total complexity of this step depends on the number of modes in O. Actually, for different PU size and texture homogeneity, the appropriate size of O might be variable. Therefore, there's still a large space for further optimization.

III. Proposed Parallel Intra Coding Scheme

There are mainly two aspects in the proposed parallel intra coding scheme: parallel intra prediction and parallel gradient measurement. On the CPU side, the results of the two processes are used to conduct fast mode decision and fast CU size decision respectively.

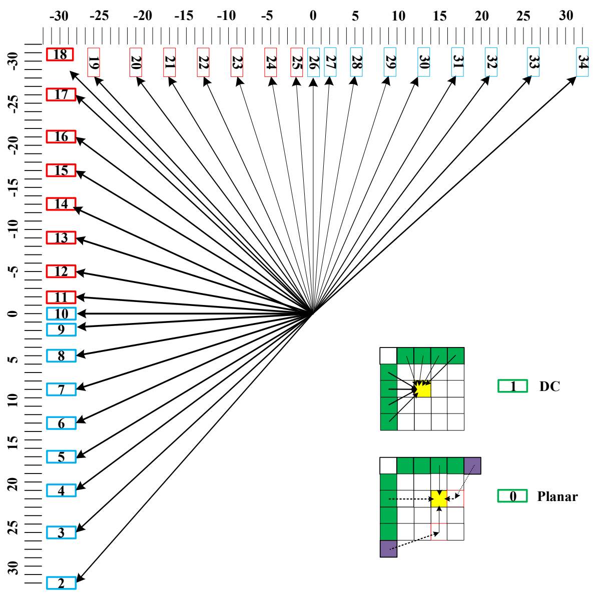


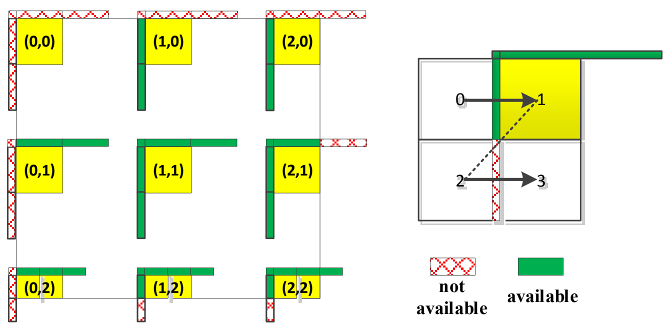
Fig. 2 35 intra prediction modes in HEVC

A. Parallel Intra Prediction

Before intra slice coding, the parallel intra prediction and mode decision procedure is launched at the GPU side, while the CPU handling other encoding tasks asynchronously. After that, the results are synchronized between CPU and GPU. Finally, the GPU-returned intra prediction mode is used for fast intra mode decision. The total scheme can be described in 3 key parts as follows.

1) *Reference Samples Generation*: As mentioned in Section I, to avoid reference dependence, the intra prediction at GPU side is performed on the original frame. But the availability of reference samples is still considered to make the GPU implementation as consistent as the original CPU version. There are two main conditions deciding the sample availability: slice boundary and block reconstruction order. Specifically, a reference pixel is not available and should be padded by the nearest available pixel if it is outside the slice boundary or not reconstructed, as shown in Fig. 3. For condition 1, the slice position type for x or y component is 3, making totally 9 types. For condition 2, the Zig-zag scan order for every 4x4 blocks in the CTU is known, and there are totally 256 4x4 blocks. Since we can classify the reference availability of each PU, the reference sample offset can be generated as a table and transmitted to GPU memory, making branch-free referencing in parallel intra prediction. It's noted that, when referenced samples are fetched, the intra prediction process is the same as the HEVC standard.

2) *CUDA Thread allocation*: The parallel intra prediction is performed at max TU size level, i.e. 32x32, while 64x64 is not considered on account of high resources occupation but low usage rate. One CUDA block is assigned to exactly one 32x32 block and all its sub-blocks considering low synchronization delay. To ensure high threads utilization, a CUDA block includes 1024 threads (supported by mainstream GPU), and 4 targeted threads allocation strategies are designed for PU size varied from 32x32 to 8x8.



a) Condition 1: slice position type b) Condition 2: reconstruction order

Fig. 3 Two conditions for reference availability judgement

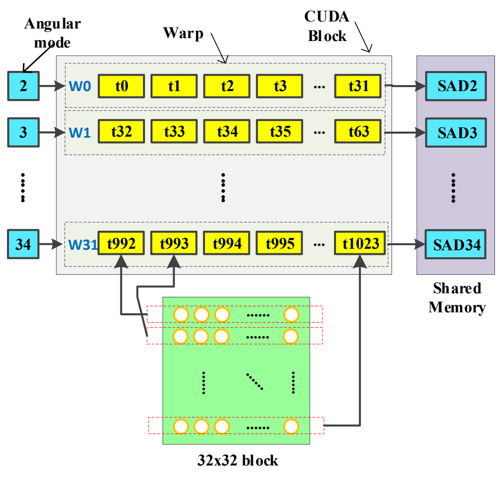


Fig. 4 CUDA thread allocation (32x32 PU and angular modes for example)

The thread allocation for each PU size includes two phases. Firstly, 32 angular mode except mode index 18 are processed according to the distribution as shown in TABLE I. For example, for 32x32 PU, 1024 threads are divided into 32 mode groups, each of which is responsible for intra prediction of a certain mode. In detail, one mode group is further divided into 1 block group, because there is only one 32x32 block in a CUDA block. And obviously, the computational load for one thread is one row of 32x32 block, i.e. 32 pixels. Thread allocation for other PU size can be referenced in TABLE I in the similar way. In the second phase, the rest 3 special modes: DC, Planar and angular mode 18, are processed by all 1024 threads. Specifically, 1024 threads are divided into 1, 4, 16, 64 block groups for 32x32 to 4x4 PU respectively. Hence only one pixel is handled by one thread. Finally, the best intra mode is selected by parallel reduction using 32 threads.

3) Fast Intra Mode Decision: After fetching the best intra prediction mode returned from GPU, the original RMD process at the CPU can be modified as follows:

- Add the GPU-returned intra mode as candidate.
- Add up to 3 MPM modes as candidates.
- For 8x8 and 4x4 PU, the signal noise and prediction error is larger than other PU sizes, so DC and Planar mode is added to keep low coding performance loss.
- Full RDO procedure for candidates as usual.

TABLE I Thread Allocation for Each PU Size

PU Size	Mode group	Modes per group	Block group	Pixels per thread
32	32	1	1	32
16	32	1	4	32
8	32	1	16	32
4	16	2	64	16

B. *Parallel Gradient Measurement*

During the intra prediction, a parallel gradient measurement process is performed in another CUDA stream. Specifically, the classic Sobel operator on basis of 8x8 original blocks is adopted for gradient calculation [7]. The gradient of CU with larger size than 8 can be added up by its containing 8x8 sub-blocks. The Sobel operator includes two convolution masks of 3x3 pixels are described as a vector in formula (2)-(4):

$$\vec{D}_{i,j} = \{Dx_{i,j}, Dy_{i,j}\} \tag{2}$$

$$\begin{aligned} Dx_{i,j} = {} & p_{i+1,j-1} + 2\times p_{i+1,j} + p_{i+1,j+1} \\ & -p_{i-1,j-1} - 2\times p_{i-1,j} - p_{i-1,j+1} \end{aligned} \tag{3}$$

$$\begin{aligned} Dy_{i,j} = {} & p_{i-1,j-1} + 2\times p_{i,j-1} + p_{i+1,j-1} \\ & -p_{i-1,j+1} - 2\times p_{i,j+1} - p_{i+1,j+1} \end{aligned} \tag{4}$$

$$Grad(\vec{D}_{i,j}) = |Dx_{i,j}| + |Dy_{i,j}| \tag{5}$$

where $Dx_{i,j}$ and $Dy_{i,j}$ indicate the difference degree in horizontal and vertical directions respectively. *Grad* represents the gradient function of one pixel.

1) *CUDA Thread allocation:* In the proposed scheme, 512 threads are used as a CUDA block to process a 32x32 blocks. A thread only handles two vertically adjacent pixels, thus six of nine pixels used for the upper pixel can be reused by the lower one, making little load for memory access.

2) *Fast CU Splitting and Pruning:* Before coding each CU, the gradient (G_{cu}) is obtained from the GPU without extra time complexity. Then the fast CU decision is made as follows:

- Fast CU Splitting. If $G_{cu} > T_{high}$, then skip mode decision for current CU depth and split further.
- Fast CU Pruning. After normal intra mode decision, if $G_{cu} < T_{low}$, then terminate CU splitting and exit.

The sobel amplitude influences CU splitting behaviour in different degree for different texture homogeneity. Therefore, the two thresholds T_{high} and T_{low} are trained using the first 5 frames and applied to the rest frames of the sequence, as shown in Fig. 5. In current implementation, TH_{low} and TH_{high} are set to achieve 95% of confidence for pruning and splitting respectively. It is noted that the threshold are trained separately for different CU depth.

IV. Experimental Results

In order to evaluate the acceleration and compression performance of the proposed parallel intra coding scheme, it has been implemented into HEVC reference software HM16.2. Simulation are conducted on a desktop with Intel Xeon CPU E5-1650 plus NVIDIA GTX 780 GPU. Test sequences with different resolutions: *Traffic, Kimono, ParkScene, Cactus, BQTerrace, PartyScene, RaceHorses, BlowingBubbles* and

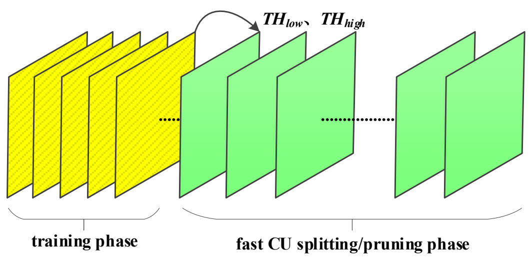


Fig. 5 Illustration of online training for gradient thresholds

BQSquare are examined with All-Intra-Main configuration. The experimental results are shown in TABLE II and TABLE III, where "Y", "U" and "V" represent the BD-rate difference of three components, and "DT" indicates the coding time change in percentage which is calculated as follows:

$$DT = \frac{AT - PT}{AT} \times 100\% \quad (6)$$

where *AT* and *PT* represent the encoding time of the anchor and proposed algorithm, respectively.

The coding performance of the proposed two optimization schemes are provided respectively as illustrated in TABLE II. It can be seen that the proposed two individual strategies can achieve 41% and 18% coding time saving on average with negligible performance loss. For parallel intra prediction, the time saving is consistent because the time of RMD process is totally hided by GPU acceleration, even for sequences with high resolution, e.g. *Traffic*, *Kimono* and *ParkScene*. For parallel gradient measurement, compared to the state-of-the-art method [5], more complexity reduction and less coding loss can be achieved due to two contributions: low-delay CUDA parallel and adaptive thresholds decision.

TABLE III shows performance of the overall algorithm with both parallel intra prediction and parallel gradient measurement. It can be seen that the combined solution can achieve more than half complexity reduction. The maximum is 62% for *BQTerrace* and the minimum is 45% for *BQSquare* and *BlowingBubbles*, indicating that the results are stable around the average value for different kinds of sequence. On the other hand, the performance loss is negligible considering the coding complexity saving it achieves.

V. Conclusions

This paper proposes a parallel intra coding strategy by CPU-GPU cooperation for HEVC, including parallel intra prediction and parallel gradient measurement. Firstly, an effective CUDA thread allocation scheme is designed for the two processes. Secondly, based on the best intra modes and block gradients returned by the GPU, a fast mode decision and fast cu size decision algorithm is proposed respectively. Experimental results show that the proposed scheme can achieve significant complexity reduction with acceptable performance loss.

Acknowledgment

This work was supported in part by the National High-tech R&D Program of China (863 Program, 2015AA015903), National Natural Science Foundation of China (61322106, 61272255), the Scientific Research Common Program of Beijing Municipal Commission of Education (KM201310025009), and Shenzhen Peacock Plan, which are gratefully acknowledged.

TABLE II Result of Two Individual Scheme Compared to HM16.2

Sequence	**Parallel Intra Prediction**		**Parallel Gradient Measurement**	
	Y	**DT**	**Y**	**DT**
Traffic	2.01%	41%	0.77%	32%
Kimono	1.66%	44%	0.32%	10%
ParkScene	1.32%	43%	0.61%	29%
Cactus	2.24%	42%	0.09%	6%
BQTerrace	1.27%	42%	0.93%	36%
PartyScene	1.68%	40%	0.70%	7%
RaceHorses	2.00%	41%	0.76%	27%
BlowingBubbles	2.07%	39%	0.71%	11%
BQSquare	1.82%	40%	0.51%	8%
Overall	1.79%	41%	0.60%	18%

TABLE III Result of Overall Algorithm Compared to HM16.2

Sequence	**Y**	**U**	**V**	**DT**
Traffic	2.80%	0.35%	0.04%	60%
Kimono	2.03%	0.95%	1.11%	47%
ParkScene	1.93%	-0.78%	-1.03%	60%
Cactus	2.32%	0.69%	0.78%	47%
BQTerrace	2.00%	1.00%	0.38%	62%
PartyScene	2.54%	0.75%	0.87%	46%
RaceHorses	2.88%	1.35%	1.48%	58%
BlowingBubbles	2.70%	0.89%	0.97%	45%
BQSquare	2.30%	-0.09%	-0.07%	45%
Overall	2.39%	0.57%	0.50%	52%

References

[1] B. Bross, W.-J. Han, J.-R. Ohm, G. J. Sullivan, and T. Wiegand, "High efficiency video coding (HEVC) text specification draft 10," Joint Collaborative Team on Video Coding (JCT-VC) of ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11, document JCTVC-L1003, Geneva, Switzerland, Jan. 2013.

[2] Draft ITU-T Recommendation and Final Draft International Standard of Joint Video Specification ITU-T Rec. H.264/ISO/IEC 14996-10 AVC), Mar. 2003.

[3] J. Lainema, F. Bossen, W. Han, J. Min and K. Ugur, "Intra coding of the HEVC standard", IEEE transactions on Circuits System and Video Technology, vol. 22, no. 12, pp. 1792-1801, Dec. 2010.

[4] L. Shen, Z. Zhang, and Z. Liu, "Effective CU size decision for HEVC intra coding," IEEE Transactions on Image Processing, vol. 23, no. 10, pp. 4232-4241, October 2014.

[5] W. Jiang, H. J. Ma and Y. W. Chen, "Gradient based fast mode kedecision algorithm for intra prediction in HEVC", Consumer Electronics, Communications and Networks (CECNet), 2012 2nd International Conference on.

[6] NVIDIA, NVIDIA CUDA Compute Unified Device Architecture Programming Guide Version 7.0, 2015.

[7] F. Pan, X. Lin, S. Rahardja, K. P. Lim, Z. G. Li, D. Wu and S. Wu, "Fast Mode Decision Algorithm for Intra prediction in H.264/AVC Video Coding," IEEE Transaction on Circuits and Systems for Video Technology, vol. 15, no. 7, pp. 813–822, Jul. 2005.