# Hardware Oriented Early CU Splitting Algorithm by Coding Unit Feature Analysis for HEVC

Ryo Kuroda[1],Wen Shi[1],Tian Song[2], and Takashi Shimamoto[2]
[1]College of Systems Innovation Engineering, Graduate School of Advanced Technology and Science,
The University of Tokushima, Tokushima, 770-0814, Japan
Email: {kuroda,shiwen}@ee.tokushima-u.ac.jp
[2] Computer Systems Engineering, Institute of Technology and Science, Graduate School of Engineering,
The University of Tokushima, Tokushima, 770-0814, Japan
E-mail : {tiansong,simamoto}@ee.tokushima-u.ac.jp

**Abstract**: In this paper, a CU size early determination algorithm is proposed to efficiently reduce complexity for the hardware implementation of HEVC inter coding. The proposed algorithm can be positioned as a pre-processing for the rate-distortion optimization (RDO) which can decide the CU size by only analyzing the original pixels based on the characteristics analysis of the coding unit using sobel-filter. Simulation results show that the proposed algorithm will induce PSNR loss 0.042dB with only 1.72% bitrate increase, compared to HM9.1.

*Keywords*—**Coding Unit (CU), Coding Tree Unit (CTU), HEVC, Inter coding**

## 1. Introduction

The latest video coding standard HEVC (High Efficiency Video Coding) is recommended in 2013[1]. With the new standard approximately twice the compression performance can be achieved compared to the conventional video coding standards. However, on the other hand, the throughput is increased significantly and high calculation complexity is induced. As a trend in recent years of video applications, there are growing demands for high-definition, high-quality and mobile terminal. In particular, in a real time delivery of high quality video, a stable encoded hardware encoder is highly required. In the current reference software, in order to detect the final one best mode every possible candidate mode has to be processed. The increased coding mode significantly increased the coding complexity. Moreover, the strong correlation between the adjacent coding units make it difficult to be processed in parallel. Multiple previous works proposed many excellent algorithms to reduce the coding complexity of CU depth decision for software implementation [2-8]. There methods can be categorized to two groups as follows. (1). CU splitting and non-splitting are modeled as a classification problem [2-6]. (2). Fast CU termination and skip algorithms are proposed [7], [8]. Jiang et al. proposed a efficient algorithm which can skip redundant modes and early determinate redundant CU candidate size[2]. However, this approach can not get a steady complexity reduction rate which will increase the hardware cost. Ju et al. proposed an excellent encoder architecture [3]. In this work, hybrid algorithms are introduced to reduce the candidate modes. However, the total proposal can only reduce the number of RDO interations by 85%. Shen et al. proposed a fast inter mode decision algorithm of HEVC[4]. This method utilize the inter-level correlation of quad-tree structure and the spatiotemporal correlation. However, it is difficult to be implemented for parallel processing. Lee et al. proposed a fast CU size decision algorithm to reduce the encoder complexity for HEVC based on Bayes' theorem with complexity factor[5]. However, the performance of this method is not steady for various sequences with different character. Goswami et al. proposed a method which is designed to decide whether a CU should be decomposed into 4 sub-CUs[6]. This is based on rate distortion (RD) cost of the parent and current CUs. Shen et al. proposed a fast CU size decision algorithm [7]. In this method, CU depth range is determined and the skip mode is utilized to reduce the complexity based on the RD cost of neighboring blocks. HE et al. proposed a fast inter-mode decision algorithm for HEVC based on textural features[8]. This method determines inter-mode before taking LCU coding in HM by using textural features between video images. A sobel-operator is proposed in this paper. This algorithm determines the coding depth of final CU so as to reduce computational complexity.

As a summary, the previous works can not balance coding efficiency and encoding complexity and take account of hardware processing. In this work, we focus on the hardware efficiency together with coding complexity reduction and parallel processing. Therefore, considering hardware implementation, a pixel based CU size early decision algorithm using sobel-filter is proposed. Using the proposed CU size decision algorithm the most implementation complexity for the hardware implementation of HEVC inter coding is reduced by preprocess the exhausted recursive CU decision for each depth. The pre-decision of CU size can not only reduce the complexity of CU size decision but also provide a steady computation complexity which is the most important factor to reduce the total hardware cost.

## 2. The proposed algorithm

Block division is a recursively processing by quad tree structure in HEVC as shown in Fig. 1. Comparing the case where a CTU ($64 \times 64$) is not at all divided to the case that divided all by $8{\times}8$ size, the calculation complexity induced by the recursively processing is different. In other words, it is possible to reduce the computation complexity by efficiently select the proper block size. In this work, an extended $4{\times}4$ sobel-filter is used to analysis the original pixels of a CTU ($64{\times}64$). Sobel-filter is used to estimate the edge intensity of a CTU which is defined as edge strength (ES). ES can be calculated as

$$ES = \sqrt{(ES_1)^2 + (ES_2)^2 + (ES_3)^2 + (ES_4)^2} \quad (1)$$

The definition of $ES_1 \sim ES_4$ and the sobel filter coefficients are shown in Fig. 2. $ES_1 \sim ES_4$ indicates the edge strength at horizontal, vertical, and oblique directions.

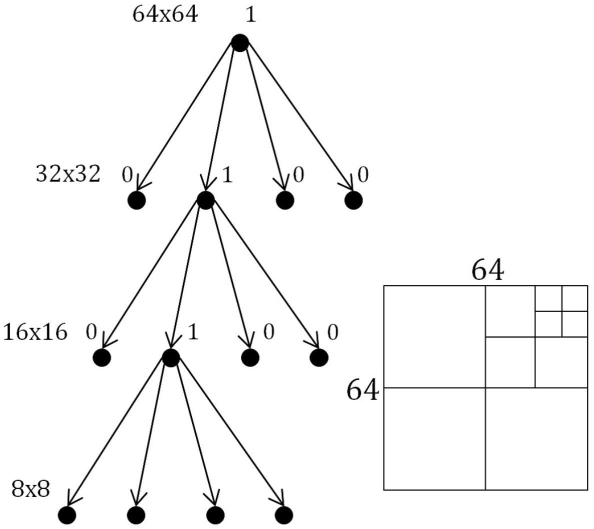
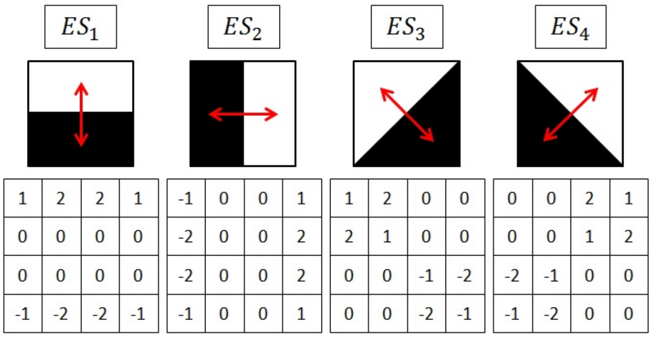


Figure 1. Quad tree structure and block splitting example



| | | | |
|---|---|---|---|
| 1 | 2 | 2 | 1 |
| 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 |
| -1 | -2 | -2 | -1 |

| | | | |
|---|---|---|---|
| -1 | 0 | 0 | 1 |
| -2 | 0 | 0 | 2 |
| -2 | 0 | 0 | 2 |
| -1 | 0 | 0 | 1 |

| | | | |
|---|---|---|---|
| 1 | 2 | 0 | 0 |
| 2 | 1 | 0 | 0 |
| 0 | 0 | -1 | -2 |
| 0 | 0 | -2 | -1 |

| | | | |
|---|---|---|---|
| 0 | 0 | 2 | 1 |
| 0 | 0 | 1 | 2 |
| -2 | -1 | 0 | 0 |
| -1 | -2 | 0 | 0 |

Figure 2. Extended 4x4 sobel filter

As indicated by previous works, a CTU with plenty edges tends to be encoded by smaller CU. On the other hand a CTU almost without edge tends to be encoded with bigger CU size. Based on this result, by calculating the ES of a CTU for each depth the optimal CU size of the CTU can be estimated. A flow chart of the proposed CU size decision algorithm is shown in Fig. 3.

In the beginning, ES is calculated in each depth of a CTU. As the results, $ES_{8\times8}$, $ES_{16\times16}$, $ES_{32\times32}$, $ES_{64\times64}$ are derived.Then, $ES_{16\times16}$, $ES_{32\times32}$, $ES_{64\times64}$ are compared with pre-defined thresholds ($TH_{16}$, $TH_{32}$, $TH_{64}$) which are generated by multiple simulations. The comparison process is a top-down decision method. When $ES_{64\times64}$ is smaller than $TH_{64}$, the CTU is encoded by $64 \times 64$. If $ES_{64\times64}$ is bigger than $TH_{64}$, the CTU should be encoded by smaller CU. Then to each CU with the size of $32 \times 32$, $ES_{32\times32}$ is compared with $TH_{32}$ to decide whether or not it need to be encoded by smaller CU. Using $ES_{16\times16}$ and $TH_{16}$, this comparison is recursively processed and the optimal CU size is determined. Process in each depth of the proposed algorithm is shown as algorithm 1.

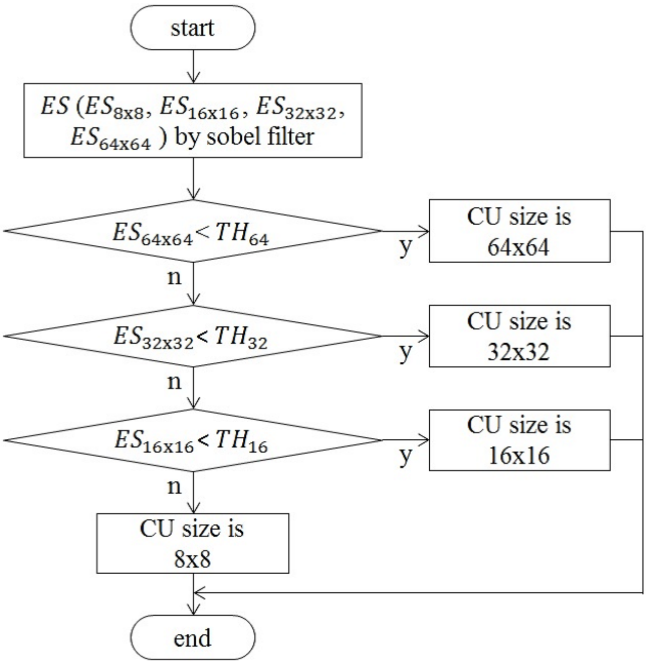


Figure 3. Block size selection process

**Algorithm 1** CTU splitting decision Algorithm

```
if $ES_{64\times64} <= TH_{64}$ then
    CU size is $64 \times 64$
else
    for $ES_{32\times32_0} : ES_{32\times32_3}$ do
        if $ES_{32\times32} <= TH_{32}$ then
            CU size is $32 \times 32$
        else
            for $ES_{16\times16_0} : ES_{16\times16_3}$ do
                if $ES_{16\times16} <= TH_{16\times16}$ then
                    CU size is $16 \times 16$
                else
                    for $ES_{8\times8_0} : ES_{8\times8_3}$ do
                        CU size is $8 \times 8$
                    end for
                end if
            end for
        end if
    end for
end if
```

## 3. Simulation Results

The proposed algorithm is implemented in the reference software HM9.1. All of the performance evaluate use Low-Delay P configuration. Several test sequences (30 frames) with picture Class B∼D are used. Due to the design motivation is ded-

Table 1. The proposed algorithm vs. HM9.1

| Class | Sequence | Δ Bitrate [%] | Δ PSNR [dB] |
|---|---|---|---|
| B(1920x1080) | BasketballDrive | -1.69 | -0.071 |
| | BQTerrace | 0.25 | -0.148 |
| | Cactus | 1.31 | -0.070 |
| | Kimono | 1.95 | -0.044 |
| | ParkScene | 3.64 | -0.086 |
| C(832x480) | BQMall | 3.51 | -0.023 |
| | PartyScene | 1.32 | -0.004 |
| | RaceHorses | 1.62 | -0.035 |
| D(416x240) | BasketallPass | 2.88 | -0.038 |
| | BlowingBubbles | 2.23 | -0.006 |
| | BQsquare | 0.55 | -0.033 |
| | RaseHorses | 2.00 | -0.008 |
| Average | | 1.72 | 0.042 |

icated to high bitrate applications the QP is set to 20. Bitrate and PSNR of proposed algorithm are compared with original reference software by defined as

$$\Delta Bitrate = \frac{Bitrate_{Proposed} - Bitrate_{HM}}{Bitrate_{HM}} \times 100\,[\%] \quad (2)$$

$$\Delta PSNR = PSNR_{Proposed} - PSNR_{HM}\,[dB] \quad (3)$$

The thresholds are set by multiple simulation results for each Class. In the case of Class D, $TH_{16} = 360$, $TH_{32} = 1100$, $TH_{64} = 4500$ are used. In the case of Class C, $TH_{16} = 360$, $TH_{32} = 1100$, $TH_{64} = 3500$ are used. In the case of Class B, $TH_{16} = 360$, $TH_{32} = 1200$, $TH_{64} = 3000$ are used. Simulation results are shown in Table 1. As shown in Table 1, the proposed algorithm did not induce significant bitrate increase. Compare to HM9.1 the average bitrate increasing is about 1.72%. The video quality loss is only 0.042 [dB] for PSNR.

Another comparison with a excellent previous work which is proposed for software implementation is shown in Table 2. From the comparison results we can find that the proposed work show almost the same performance than previous work. Both PSNR and bitrate loss are very tiny and completely acceptable.

The most important feature of the proposed algorithm is on the hardware friendly feature. Compare with the previous work [2], the recursive selection of CU size is unnecessary. It can significantly reduce the clock cycles for one CTU. It will lead to small area implementation and low power consumption.

The pre-decision of the CU size not only significantly reduce the recursive process of RDO but also fixed the computation complexity. A fixed computation complexity can help to design a low cost hardware because the hardware don't has to provide redundant hardware in the worst case. Furthermore, the pre-decision process can be completely processed in parallel to the coding process which can help to achieve fast inter encoding.

Figure 4. Block division by HM9.1(BasketballDrive)

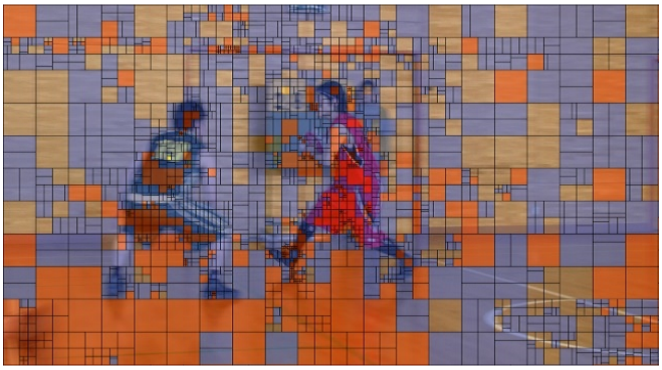


Figure 5. Block division by the proposed algorithm (BasketballDrive)

Fig 4 and 5 show the block division results of the same frame of BasketballDrive of HM9.1 and proposed algorithm. The block size selected by HM9.1 performs fine splitting when texture is not complicated and there is little motion in adjacent frames. On the other hands, the proposed algorithm selects bigger blocks at the part that has complex textures. From the simulation results, it is clear that in this case the proposed algorithm can save more bits by selecting bigger block size. The proposed method has a weak point in the case of some other sequences, such as the ParkScene. Figure 6 and 7 show block division of the same frame of HM9.1 and the proposed algorithm. A appropriate larger block size is selected according to the texture by HM9.1 regardless of the texture. However, the proposed algorithm selected small block size and it induces decrease of PSNR and bitrate gain. From these results, some other method may be introduced to the proposed algorithm to make it work well even in some rare case. By further eliminating redundant splitting the coding efficiency can be improved by comparing the motion density between frames.

## 4. Conclusion

In this paper, we propose a CU size early decision algorithm based on the characteristics of the CTU to reduce computing redundancy of recursive processing. As a result, the proposed algorithm taking into consideration the hardware processing succeeded in suppressing bitrate increasing of 1.72%,

Table 2. The proposed algorithm vs. previous work

| | | proposed algorithm | | previous work[2] | |
|---|---|---|---|---|---|
| Class | Sequence | Δ Bitrate [%] | Δ PSNR [dB] | BDBR [%] | BDPSNR [dB] |
| B | BasketballDrive | -1.69 | -0.071 | 1.63 | -0.040 |
| | BQTerrace | 0.25 | -0.148 | 0.72 | -0.014 |
| | Cactus | 1.31 | -0.070 | 1.59 | -0.034 |
| | Kimono | 1.95 | -0.044 | 0.69 | -0.023 |
| | ParkScene | 3.64 | -0.086 | 0.78 | -0.024 |
| C | BQMall | 3.51 | -0.023 | 2.27 | -0.089 |
| | PartyScene | 1.32 | -0.004 | 0.41 | -0.017 |
| | RaceHorses | 1.62 | -0.035 | 0.71 | -0.028 |
| D | BasketallPass | 2.88 | -0.038 | 2.30 | -0.107 |
| | BlowingBubbles | 2.23 | -0.006 | 0.54 | -0.020 |
| | BQsquare | 0.55 | -0.033 | 0.08 | -0.003 |
| | RaseHorses | 2.00 | -0.008 | 0.43 | -0.019 |
| Average | | 1.72 | -0.042 | 1.01 | -0.035 |

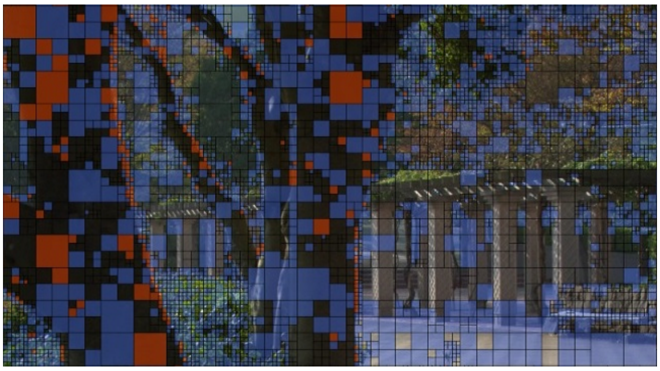


Figure 6. Block division by HM9.1 (ParkScene)

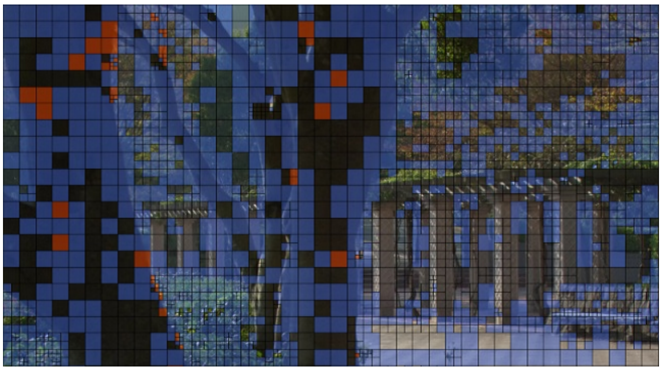


Figure 7. Block division by the proposed algorithm (ParkScene)

the deterioration of the video quality in 0.042[dB] for PSNR. However, in some sequences, including complex textures like ParkScene deterioration increases and PSNR of the bit rate are larger. In the future, further improvement will be discussed by adaptively selecting the threshold for different sequences.

## References

[1] G. J. Sullivan, J.-R. Ohm, W.-J. Han, and T. Wiegand, "Overview of the high efficiency video coding (HEVC) standards," IEEE Trans. Circuit Syst. Video Technol., vol.22, no.12, pp.1649-1668, 2012.

[2] X. T. Jiang, T. Song, T. Shimamoto, W. Shi and L. S. Wang, "High efficiency CU depth prediction algorithm for high resolution applications of HEVC," IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences, vol.E98-A, no.12, pp.2528-2536, Dec. 2015.

[3] C. C. Ju, T. M. Liu, K. B. Lee, Y. C. Chang, H. L. Chou, C. M. Wang, T. H. Wu, H. M. Lin, Y. H. Huang, C. Y. Cheng, T. A. Lin, C. C. Chen, Y. K. Lin, M. H. Chiu, W. C. Li, S. J. Wang, Y. C. Lai, P. Chao, C. D. Chien, M. J. Hu, P. H. Wang, F. C. Yeh, Y. C. Huang, S. H. Chuang, L. F. Chen, H. Y. Lin, M. L. Wu, C. H. Chen, R. Chen, H. Y. Hsu, K. Jou, "18.6 A 0.5nJ/pixel 4K H.265/HEVC codec LSI for multi-format smartphone applications," 2015 IEEE International Solid-State Circuits Conference (ISSCC), Feb. 2015.

[4] L. Shen, Z. Zhang, and Z. Liu, "Adaptive inter-mode decision for HEVC jointly utilizing inter-level and spatiotemporal correlations," IEEE Trans. Circuit Syst. Video Technol., vol.24, no.10, pp.1709-1722, Oct. 2014.

[5] J. Lee, S. Kim, K. Lim, and S. Lee, "A fast CU size decision algorithm for HEVC," IEEE Trans. Circuit Syst. Video Technol., vol.25, no.3, pp.411-421,March 2015.

[6] K. Goswami, B.-G. Kim, D.Jun, S.-H. Jung, and J.S. Choi, "Early coding unit-splitting termination algorithm for high efficiency video coding (HEVC)," Etri. J., vol.36, no.3, pp407-417, June 2014.

[7] L. Shen, Z. Liu, X. Zhang, W. Zhao, and Z. Zhang, "An effective CU size decision method for HEVC encoders," IEEE Trans. Multimedia, vol.15, no.2, pp.465-470, Feb. 2013.

[8] J. He, X. He, X. Li, and L. Qing, "Fast Inter-Mode Decision Algorithm for High-Efficiency Video Coding Based on Textural Features," Journal of Communications Vol. 9, No. 5, May 2014.