Moving Region-based Fast Encoding Scheme for HEVC Inter Coding

Ei Ei Tun
Department of Electrical Engineering
Chulalongkorn University
Bangkok 10330, Thailand
Faculty of Computer Systems and Technologies
University of Computer Studies, Yangon
Yangon, Myanmar
eieitun@ieee.org

Htoo Maung Maung
Department of Research and Innovation
Ministry of Education
Yangon, Myanmar
htoomaung@ieee.org

Supavadee Aramvith
Multimedia Data Analytics and Processing Research Unit
Department of Electrical Engineering
Chulalongkorn University
Bangkok 10330, Thailand
supavadee@a.chula.ac.th

Yoshikazu Miyanaga
Chitose Institute of Science and Technology
Chitose, Hokkaido 066-8655, Japan
y-miyana@photon.chitose.ac.jp

***Abstract*—This paper proposes a fast inter encoding scheme for High Efficiency Video Coding (HEVC) by utilizing a moving region (MR) to enhance the video quality of our previous scheme for HEVC. Our previous scheme modeled the coding tree unit (CTU) partitioning problem as an optimization problem and solved it by a simple optimizer called a genetic algorithm (GA). To improve the quality of our previous work especially at a low bitrate, the motion information of each CTU is analyzed with the frame difference method. Next, the CTU is categorized into two groups: MR and non-MR. Then, the partitioning pattern (PP) of the current CTU is calculated by GA if the current CTU is in MR. Otherwise, PP from the collocated CTU of the previous key frame is utilized. According to the experimental results, MR-based algorithm can achieve an improved video quality compared with our previous algorithm.**

***Index Terms*—High efficiency video coding, coding tree unit, partitioning pattern, moving region, genetic algorithm**

I. Introduction

ITU-T and ISO/IEC have built a collaborative research to launch High Efficiency Video Coding (HEVC) [1] principally for the high definition (HD) and Ultra HD (UHD) videos. HEVC outperforms the encoded video with the 50% bitrate reduction under the comparable video quality of its antecedent called H.264/AVC [2]. However, the computational cost of HEVC is significantly high and not realistic for a real-time environment due to its advanced coding implementations mainly coding tree unit (CTU) partitioning as shown in Fig. 1. To carry out the fast algorithm, the earlier schemes have statistically estimated CTU partition by analyzing the original nature of CTU partitioning of HEVC such as in [3]– [5]. From a few years ago, machine learning (ML) approaches [6]– [10] focused on fast encoding due to the learning ability from huge

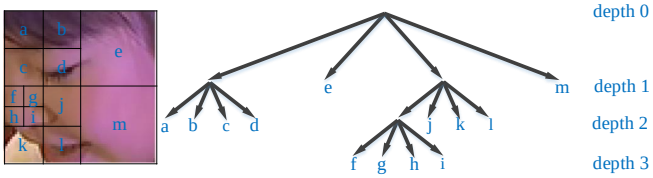

Fig. 1. Example CTU partitioning pattern of PartyScene.

and complex data amount to the optimal conclusion. To lessen the computational complexity of HEVC inter coding, we firstly presented a fast coding unit (CU) coding method by utilizing a genetic algorithm (GA) [11] for searching the partitioning pattern (PP) for each CTU. Nowadays, the temporal correlation between successive frames is huge since the frame rate of HD video is 50/60 frames per second (fps) and UHD video is up to 120 fps. Therefore, the time intervals between sucessive frames are extremely small such as 0.02, 0.03, 0.05, and 0.07 seconds for two, four, six, and eight sucessive frames of the 120 fps test sequence, respectively. Due to these insignificant time intervals, it is reasonable to share PPs of one frame to its sucessive frames under a comparable video quality. To apply for both low to the high frame rates and to accompany a group of pictures (GOP) of coding structure, we reasonably defined a sharing range as four which is the default GOP size of low delay (LD) and low delay P (LDP) configurations. Then, we utilized GA [12] to find PPs for only the important frame, named key frame, and shared to three following frames, also known as common frames, by considering the high temporal

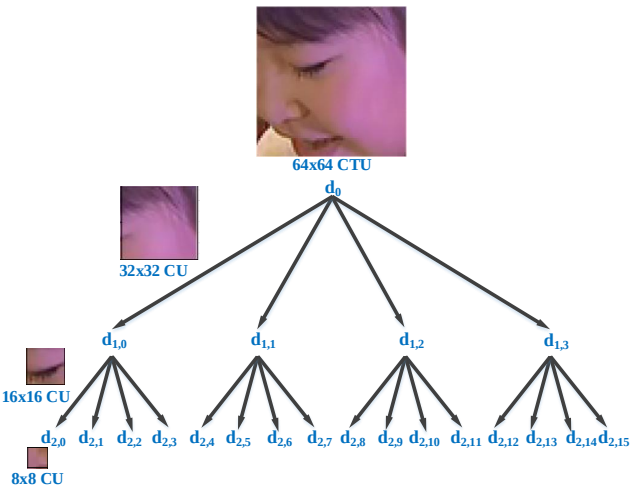

Fig. 2. Chromosome pattern of a genetic algorithm.

Fig. 3. Moving region result from FDM.

correlation of the current video sequences. The key frame is usually encoded by using small quantization parameter (QP) value to get a better video quality than the common frame. Thanks to the effective chromosome structure as shown in Fig. 2 and fitness function, our previous fast encoding scheme [11] achieves 16.7% and 62.5% time saving (TS) on average with a comparable video quality compared with state-of-the-art ML-based fast encoding [10] and original HEVC test model 16.5 (HM16.5), respectively, under the LDP configuration with enabled rate control (RC).

The main idea of the previously proposed scheme is to significantly reduce the computational complexity with a comparable video quality especially at a high bitrate. In current communication networks including 5G, the available bandwidth is going up. Apparently, we can increase the bitrate to be high and we can consider high bitrate in video coding. Our method gets a comparable peak signal to noise ratio (PSNR) at a high bitrate with a significant time saving. For the calculation cost for small equipment, the reduction in calculation cost should be an important issue. However, there is some quality degradation of our previous method at a low bitrate. Therefore, to increase the PSNR value of our previous encoding scheme at a low bitrate especially for the video sequence which has fast-moving objects, we analyze the motion information of each CTU and determine whether the current CTU should utilize the partitioning pattern from the collocated CTU of the key frame or not.

II. Proposed Fast Encoding Scheme

A. Moving Region Extraction

To get the moving region information of the video frame, we utilize the frame differencing method (FDM) with a global motion vector (MV) which can lower the motion effect of the camera. To quickly find a global motion vector (GMV), the gray projection method (GPM) is reasonably utilized in our proposed system. GPM is very practical for time-constrained applications because of its low computational complexity. Firstly, we calculate GMV by using GPM. Then, we predict the motion-compensated frame with GMV as the previous frame. And then, we determine the frame difference value among the current frame and the motion-compensated frame for every CTU using

$$D_{ctu}(f) = \left[F_{ctu}(f) - F_{ctu}(f-1, GMV) \right] \times w_{ctu} \quad (1)$$

where D_{ctu} is the frame difference for the specific CTU, GMV is the global motion vector, f is the current, and $(f-1)$ is the motion-compensated frames. To assign a specific weight valued based on the location of CTU, we use three heuristic values of w_{ctu} based on three specific locations of the frame such as core, middle, and boundary region. Finally, we define the MV array for the current frame by using a predefined threshold

$$MV[ctu] = \begin{cases} 1, & if\ D_{ctu}(f)/D_{avg}(f) > Th \\ 0, & otherwise \end{cases} \quad (2)$$

where $D_{avg}(f)$ is the average difference value for the current frame and Th is empirically selected as 0.75. As shown in Fig. 3, there are two regions: MR and non-MR. The black color represents the CTUs of non-MR.

B. Overall Algorithm

In [11], three main stages are necessary for encoding each CTU of the key frame. First, the rate distortion (RD) costs for 85 CUs were roughly calculated with the most common mode SKIP/MERGE if the depth of CTU is 0 and its maximum depth is 3. Next, PP for each CTU was efficiently calculated by a simple optimizer, named GA, and our innovative formulation for fitness function and effective definition for chromosome

Algorithm 1 Moving Region based Fast Encoding Scheme for the common frames

Ensure: $ppArrKFrame$
Require: $ppArrCFrames$

Step 1: Calculate Moving Vector array MV for common frame

$ctu \leftarrow 1$
for each $ctu <= numberOfCTU$ **do**
    $MV[ctu] \leftarrow findMVWithFDM(ctu)$
end for

Step 2: Searching the PPs for the moving region or utilize the PPs of the key frame for the non-moving region

$ctu \leftarrow 1$
$n \leftarrow 0$
for each $ctu <= numberOfCTU$ **do**
    if $MV[ctu] == 1$ **then**
        $rdCostArray \leftarrow calculateRDCostWithMerge()$
        $cost_{d_0} \leftarrow rdCostArray[n++]$
        $i \leftarrow 0$
        for each $i < 4$ **do**
            $cost_{d_{1,i}} \leftarrow rdCostArray[n++]$
            $j \leftarrow 4 * i$
            for each $j < 4 * i + 4$ **do**
                $cost_{d_{2,j}} \leftarrow rdCostArray[n++]$
                $k \leftarrow 4 * j$
                for each $k < 4 * j + 4$ **do**
                    $cost_{d_{3,k}} \leftarrow rdCostArray[n++]$
                end for
            end for
        end for
        $chro \leftarrow 0$
        for each $chro < sizeOfPopulation$ **do**
            $chroArr[chro] \leftarrow randomChromosome()$
            $d_0\ \ d_{1,0}d_{1,1}d_{1,2}d_{1,3}\ \ d_{2,0}d_{2,1}d_{2,2}d_{2,3}d_{2,4}d_{2,5}d_{2,6}d_{2,7}d_{2,8}d_{2,9}d_{2,10}d_{2,11}d_{2,12}d_{2,13}d_{2,14}d_{2,15} \leftarrow chroArr[chro]$
            $fit \leftarrow (1-d_0)cost_{d_0} + d_0\left[\sum_{i=0}^{3}(1-d_{1,i})cost_{d_{1,i}} + d_{1,i}\left(\sum_{j=4i}^{4i+3}(1-d_{2,j})cost_{d_{2,j}} + d_{2,j}\sum_{k=4j}^{4j+3}cost_{d_{3,k}}\right)\right]$
            $fitnessArr[chro] \leftarrow fit$
        end for
        $parentSize \leftarrow 5$
        $sameFitnessFrequency \leftarrow 2$
        $bestChromosome \leftarrow findBestChroWithGA(chroArr, fitnessArr, parentSize, sameFitnessFrequency)$
        $ppArrCFrames[ctu] \leftarrow bestChromosome$
    else
        $ppArrCFrames[ctu] \leftarrow ppArrKFrame[ctu]$
    end if
end for
return $ppArrCFrames$

TABLE I
SUMMARY OF NOTATIONS IN ALGORITHM 1.

Notation	Description
$ppArrCFrames$	The partitioning pattern array of the non-key frame (common frame)
$ppArrKFrame$	The partitioning pattern array of the previous key frame
ctu	The index of current CTU
$numberOfCTU$	Total number of CTUs for each frame
MV	Output motion vector array from FDM
$findMVWithFDM()$	Find MV using (1) and (2)
n	The index of RD cost array
$rdCostArray$	The RD cost array which is calculated with the most important mode (MERGE/SKIP)
$calculateRDCostWithMerge()$	Calculate the approximate RD cost after encoding MERGE/SKIP mode
$cost_{d_0}, cost_{d_{1,i}}, cost_{d_{2,j}}, cost_{d_{3,k}}$	RD cost array for depth 0, 1 and 2
$chro$	The index of the current chromosome
$sizeOfPopulation$	Total number of chromosome for each population
$chroArr$	The chromosome array for each population
$randomChromosome()$	Create a valid chromosome randomly
$d_0, d_{1,0}...d_{1,3}, d_{2,0}...d_{2,15}$	First gene, 4 and 16 genes of chromosome to decide CU partition at depth 0, 1, and 2
$fit, fitnessArr$	The fitness value of each chromosome and fitness array for each population
$parentSize$	Total number of best parent chromosomes for generating a newly better population
$sameFitnessFrequency$	The termination point of GA
$findBestChroWithGA()$	Search the best chromosome based on the fitness value of each chromosome
$bestChromosome$	The best chromosome with the best fitness value

structure of GA. Finally, the CU size was estimated by utilizing PP.

In this work, we search the partitioning pattern for each CTU of the key frame by using GA as the same as our previous work. For common frames, we design an algorithm which is described in Algorithm 1 for searching partitioning pattern of each CTU based on MR information. In this algorithm, we firstly search the MVs of the common frame and utilize the value of MV for determining whether the PP of the current CTU will be searched by GA or shared by the collocated CTU. Table I lists the notations used in the algorithm and their descriptions.

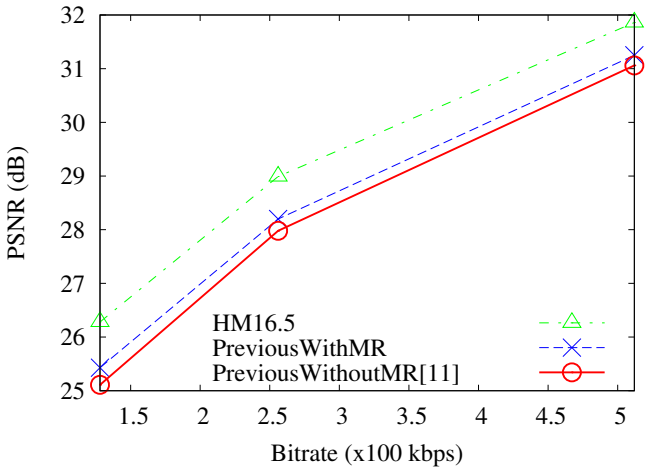

Fig. 4. Video quality improvement of the proposed system for BQMall .

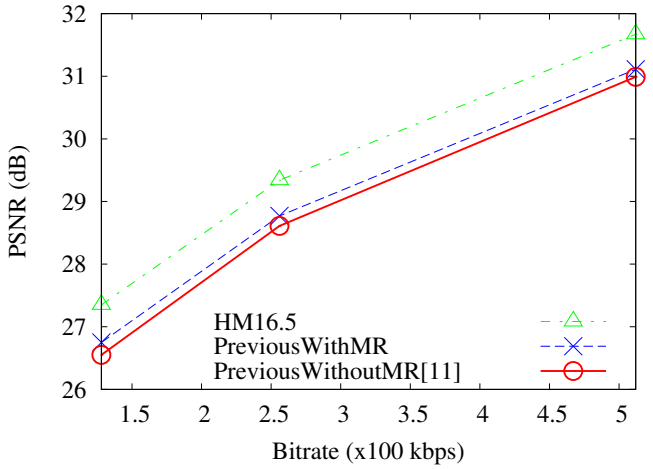

Fig. 5. Video quality improvement of the proposed system for ChinaSpeed .

III. PERFORMANCE EVALUATION

For evaluating the PSNR performance of our MR-based fast encoding scheme, we implemented our MR scheme on the previous algorithm [11]. The performance benchmarks are the original HM16.5 and the previous one without MR [11] especially for fast movement video sequences such as BQMall which includes several moving people in foreground and ChinaSpeed which is a screen recording of a driving game. Under LDP configuration with enabled RC, we can achieve a better PSNR value after combining the MR approach in our previous encoding scheme about 0.32, 0.23, and 0.19 dB at 128, 256, and 512 kbps, respectively, with the same TS as shown in Fig. 4 of BQMall. According to Fig. 5 for ChinaSpeed, it can be found that our previous algorithm

combined with MR can achieve 0.2, 0.16, and 0.12 dB quality improvement with the same TS.

IV. Conclusion

The previous fast encoding scheme has significantly achieved time saving due to the effective chromosome structure and powerful fitness function at a high bit rate. To compatible with our previous approach at a low bitrate, we designed a moving region extraction by using FDM and combined it with our GA-based fast encoding. Thanks to the simple and effective MR information, we can improve the PSNR and can achieve a similar time saving especially for video sequences that have fast-moving objects in the foreground scene.

References

[1] G. J. Sullivan, J. Ohm, W. Han, and T. Wiegand, "Overview of the high efficiency video coding (HEVC) standard," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 22, no. 12, pp. 1649–1668, Dec. 2012.

[2] T. Wiegand, G. J. Sullivan, G. Bjontegaard, and A. Luthra, "Overview of the H.264/AVC video coding standard," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 13, no. 7, pp. 560–576, 2003.

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

[4] X. Hou and Y. Xue, "Fast coding unit partitioning algorithm for HEVC," in *2014 IEEE International Conference on Consumer Electronics (ICCE)*, Jan 2014, pp. 7–10.

[5] I. Zupancic, S. G. Blasi, E. Peixoto, and E. Izquierdo, "Inter-prediction optimizations for video coding using adaptive coding unit visiting order," *IEEE Transactions on Multimedia*, vol. 18, no. 9, pp. 1677–1690, Sept 2016.

[6] X. Shen and L. Yu, "CU splitting early termination based on weighted SVM," *EURASIP Journal on Image and Video Processing*, vol. 2013, no. 1, p. 4, Jan 2013.

[7] G. Correa, P. A. Assuncao, L. V. Agostini, and L. A. da Silva Cruz, "Fast HEVC encoding decisions using data mining," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 25, no. 4, pp. 660–673, April 2015.

[8] Y. Zhang, S. Kwong, X. Wang, H. Yuan, Z. Pan, and L. Xu, "Machine learning-based coding unit depth decisions for flexible complexity allocation in high efficiency video coding," *IEEE Transactions on Image Processing*, vol. 24, no. 7, pp. 2225–2238, July 2015.

[9] A. Heindel, T. Haubner, and A. Kaup, "Fast CU split decisions for HEVC inter coding using support vector machines," in *2016 Picture Coding Symposium (PCS)*, Dec 2016, pp. 1–5.

[10] L. Zhu, Y. Zhang, S. Kwong, X. Wang, and T. Zhao, "Fuzzy SVM-based coding unit decision in HEVC," *IEEE Transactions on Broadcasting*, vol. PP, no. 99, pp. 1–14, 2017.

[11] E. E. Tun, S. Aramvith, and Y. Miyanaga, "Fast coding unit encoding scheme for HEVC using genetic algorithm," *IEEE Access*, vol. 7, pp. 68 010–68 021, May 2019.

[12] M. Mitchell, *An Introduction to Genetic Algorithms*. Cambridge, MA, USA: MIT Press, 1996.