Adaptive Block Coding Order for Intra Prediction in HEVC

Amin Zheng, *Student Member, IEEE*, Yuan Yuan, *Student Member, IEEE*, Jiantao Zhou, *Member, IEEE*, Yuanfang Guo, *Member, IEEE*, Haitao Yang, *Member, IEEE*, and Oscar C. Au, *Fellow, IEEE*

***Abstract*—In this paper, an adaptive block coding order for intra prediction is proposed. Modern video coding standards, including the most recent High Efficiency Video Coding (HEVC), utilize fixed scan orders in processing blocks during intra coding. However, the fixed scan orders typically result in residual blocks with noticeable edge patterns. That means the fixed scan orders cannot fully exploit the content-adaptive spatial correlations between adjacent blocks, thus the bitrate after compression tends to be large. To reduce the bitrate induced by inaccurate intra prediction, the proposed approach adaptively chooses both the block and subblock coding orders by minimizing the coding cost. Specifically, determining the block coding order is formulated as a traveling salesman problem that is solved using dynamic programming. Besides the block coding order, we also design the subblock coding order in each block with an adaptive manner. The experimental results demonstrate a Bjøntegaard-Delta-rate reduction of up to 4.4% compared with HEVC anchor.**

***Index Terms*—Adaptive block coding order (ABCO), High Efficiency Video Coding (HEVC), intra prediction, video coding.**

I. Introduction

INTRA frames are essential for a video coding system to enable random access and avoid error propagation. Moreover, the quality of intra frames closely affects the quality of subsequent inter frames. In a traditional block-based intra-coding system, a block of pixels is first predicted from the neighboring pixels in previously coded blocks, and then the prediction residual is converted into a bit stream by transform coding and entropy coding. A more accurate prediction leads to a lower residual energy, and thus a lower number of resultant bits. Therefore, a prediction scheme that generates less residual energy is highly desirable in intra coding.

In modern video coding standards, intra prediction is mostly carried out by directional extrapolation, where the predicted pixels are generated by copying the reference pixels in previously coded blocks along an angular direction. H.264/AVC [1] utilizes eight directions, and High Efficiency Video Coding (HEVC) [2] supports up to 33 directions. The prediction is performed block by block in a fixed scan order and only above and/or left neighboring reconstructed blocks are employed to generate prediction block. Our previous work [3] has shown that the prediction accuracy in HEVC is sensitive to the edge directions within blocks. The sensitivity is highly correlated with the subblock scan order. Only the edges that are approximately along the fixed Z-scan subblock coding order can be efficiently predicted.

There have been recent efforts on improving the intra-coding efficiency by changing the subblock coding order [3]–[5]. In [4], the subblock coding order in a macroblock (MB) is chosen from either original Z-scan order or inverse Z-scan order depending on the rate–distortion (RD) optimization process. The inverse Z-scan order enables the current block to utilize the below and/or right neighboring blocks for prediction. More optional subblock orders are provided in [5], where a unified extra/interpolating prediction strategy is designed to adapt all the possible orders. Our previous work [3] was proposed to deal with the subblocks with flexible sizes, since both [4] and [5] support subblocks with fixed sizes only. A common problem of these three methods is that the block coding order is always fixed, although the subblock coding orders are changed adaptively.

Changing the block coding order with fixed subblock order is allowed in flexible MB ordering (FMO) [6], [7] mode in H.264. Using FMO, MBs are classified into at most eight slice groups. The slice groups can be coded in arbitrary orders, while within a slice group, blocks are coded in default scan order. By allowing arbitrary slice group orders and nonconsecutive blocks to belong to the same slice group, FMO enables the blocks in one frame coded in a variety of orders. FMO is designed for error resilience. Using multiple slice groups spreads out burst errors, but lowers the coding efficiency.

In this paper, we propose to select both the subblock and the block coding order adaptively. Different from FMO, the purpose of changing block coding order in our approach is to maximize the coding efficiency, and the order can be changed more thoroughly by traversing all the possible orders. Specifically, we formulate the problem of choosing the block coding

Manuscript received June 28, 2015; revised September 14, 2015; accepted November 12, 2015. Date of publication November 18, 2015; date of current version October 27, 2016. This work was supported in part by the Macau Science and Technology Development Fund under Grant FDCT/009/ 2013/A1 and Grant FDCT/046/2014/A1, in part by the Research Committee, University of Macau, Macau, China, under Grant MYRG2014-00031-FST and Grant MYRG2015-00056-FST, and in part by the National Science Foundation of China under Grant 61402547. This paper was recommended by Associate Editor M. Hannuksela.

A. Zheng, Y. Yuan, and O. C. Au are with the Department of Electronic and Computer Engineering, The Hong Kong University of Science and Technology (e-mail: amzheng@connect.ust.hk; yyuanad@connect.ust.hk; eeau90@gmail.com).

J. Zhou is with the Faculty of Science and Technology, University of Macau, Macau 999078, China (e-mail: jtzhou@umac.mo).

Y. Guo is with the State Key Laboratory of Information Security, Institute of Information Engineering, Chinese Academy of Sciences, Beijing 100195, China (e-mail: eeandyguo@connect.ust.hk).

H. Yang is with the Central Research Institute, Huawei Technologies, Shenzhen 518129, China (e-mail: haitao.yang@huawei.com).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TCSVT.2015.2501738

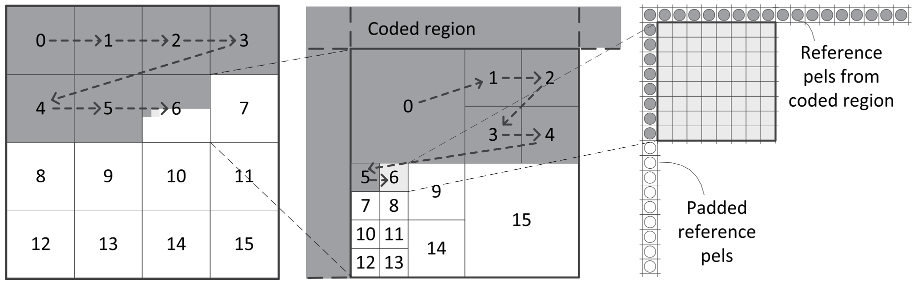


Fig. 1. Example of block partitioning and reference pixels selection in HEVC intra prediction. Left to right: CTU partitioning, CU partitioning, and reference pixels for current CU. The number denotes the processing order of blocks.

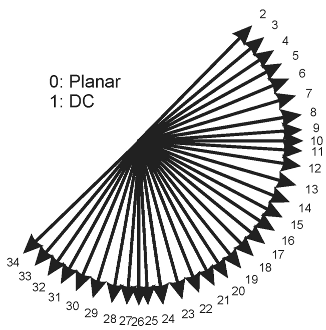


Fig. 2. Thirty-five intra-prediction modes in HEVC.

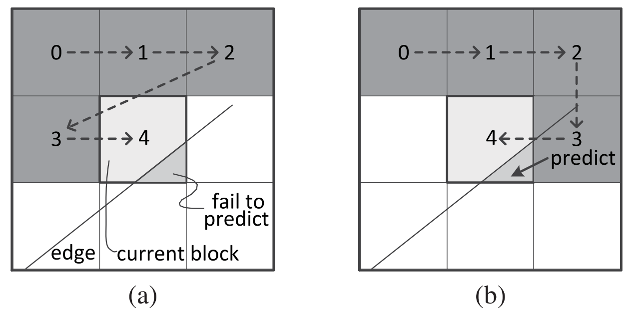


Fig. 3. Example of traditional raster scan coding order. The number denotes the processing order. (a) Blocks are processed in raster scan order. (b) Blocks are processed in a different order from (a).

order as a traveling salesman problem (TSP) that minimizes the coding cost. The TSP problem is then solved by applying dynamic programming. Experiments show that the proposed method outperforms the state-of-the-art HEVC, implying that the proposed method is capable of making a good prediction for the regions containing edges with any direction.

The rest of this paper is organized as follows. Section II presents the intra prediction of HEVC and its limitations. We introduce our block and subblock coding order selection schemes in Section III. Experimental results and analyses are given in Section IV. Section V concludes our work.

II. Intra Prediction in HEVC

Two new features of HEVC intra prediction compared with H.264 are the quadtree-based block partitioning scheme and the increased intra-prediction modes [8]. In HEVC, a frame is divided into large nonoverlapping blocks called coding tree units (CTUs). The CTUs are further partitioned into subblocks called coding units (CUs)[1] according to a quadtree-based structure. Within one frame, all the CTUs are processed in raster scan order. Within one CTU, all the CUs are processed in Z-scan order, as illustrated in Fig. 1. The CTU in the middle part of Fig. 1 is subdivided into 16 CUs with different sizes. The flexible partitioning scheme enables the encoder to better exploit the spatial correlation by choosing appropriate block sizes. A CU is further divided into one or four prediction units (PU) and each PU can have different prediction modes. HEVC intra coding supports up to 33 directional modes, along with dc mode and planar mode, as illustrated in Fig. 2. When predicting an $N \times N$ block, there are $4 \times N + 1$ reference pixels from neighboring reconstructed blocks. If the neighboring blocks are unavailable, reference pixels (padded pixels) are copied from the nearest available reference pixels, as shown in Fig. 1.

As a convention of block-based video coding standards, CTUs in HEVC are processed in raster scan order. With such a processing order, only left, left-above, above, and right-above neighboring reconstructed CTUs can be employed as references to predict the current CTU. As illustrated in Fig. 2, the prediction direction can only cover a half plane from 45° (mode 2) to 225° (mode 34). In other words, the encoder and the decoder are not able to perform prediction from angles in the other half plane. In Fig. 3(a), the current block is divided into two parts by an edge. When raster scan order is used, current block can only employ left, left-above, above, and right-above neighboring reconstructed blocks as references for prediction. Hence the gray part below the edge cannot be well predicted. However, if the coding order is approximately along the edge direction, as shown in Fig. 3(b), the gray part can be efficiently predicted from its right neighboring reconstructed block. Similar problem exists in the Z-scan CU coding order when coding one CTU. Therefore, the traditional block coding orders (raster and Z-scan) are not capable of fully exploiting the spatial correlation between adjacent blocks. This motivates us to adaptively select both CTU and CU coding orders such that the spatial redundancy can be further reduced.

III. Adaptive Block Coding Order

The basic idea of our method is to find an optimal coding order that minimizes the total coding cost of all the CTUs in one frame. We apply the well-adopted RD cost in

[1]In the rest of this paper, we denote block by CTU and subblock by CU.

our optimization

$$\mathrm{RD} = D + \lambda \times R \tag{1}$$

where D is the sum of the square difference between the original CTU and the reconstructed CTU. R is the coded bits for the current CTU, and λ is the predefined Lagrangian multiplier.

Let N be the number of CTUs in one frame. Initially each CTU is assigned one index from 1 to N in the raster scan order. We denote $\mathbf{x}^{\mathcal{V}} = [1, 2, \ldots, |\mathcal{V}|]^{\mathsf{T}} \in \mathbb{N}^{|\mathcal{V}|}$ as the vector of the initial CTU indices, where $\mathcal{V} = \{1, 2, \ldots, |\mathcal{V}|\}$ and $|\mathcal{V}| = N$ is its cardinality. We further denote $\mathbf{y}^{\mathcal{V}} \in \mathbb{N}^{|\mathcal{V}|}$ as the actual coding order of N CTUs, and the index of the ith coding CTU is $\mathbf{y}_i^{\mathcal{V}}$. Without loss of generality, we define $\mathbf{y}^{\mathcal{V}} = \mathbf{P}^{\mathcal{V}}\mathbf{x}^{\mathcal{V}}$, where $\mathbf{P}^{\mathcal{V}} \in \mathbb{N}^{|\mathcal{V}|\times|\mathcal{V}|}$ is a permutation matrix.

We formulate our optimization problem as follows:

$$\begin{aligned}\min_{\mathbf{P}^{\mathcal{V}}} \quad & \sum_{i=1}^{|\mathcal{V}|}\mathrm{RD}\big(\mathbf{y}_i^{\mathcal{V}}|\mathbf{y}_{i-1}^{\mathcal{V}}, \mathbf{y}_{i-2}^{\mathcal{V}}, \ldots, \mathbf{y}_1^{\mathcal{V}}\big)\\ \text{s.t.} \quad & \mathbf{y}^{\mathcal{V}} = \mathbf{P}^{\mathcal{V}}\mathbf{x}^{\mathcal{V}}\end{aligned} \tag{2}$$

where $\mathrm{RD}(\mathbf{y}_i^{\mathcal{V}}|\mathbf{y}_{i-1}^{\mathcal{V}}, \mathbf{y}_{i-2}^{\mathcal{V}}, \ldots, \mathbf{y}_1^{\mathcal{V}})$ denotes the RD cost of CTU $\mathbf{y}_i^{\mathcal{V}}$ after CTUs $\mathbf{y}_{i-1}^{\mathcal{V}}, \mathbf{y}_{i-2}^{\mathcal{V}}, \ldots, \mathbf{y}_1^{\mathcal{V}}$ have been coded.[2] As different permutation matrices $\mathbf{P}^{\mathcal{V}}$ lead to different coding orders, the problem becomes to find an optimal $\mathbf{P}^{\mathcal{V}}$ ($\mathbf{y}^{\mathcal{V}}$) to minimize the total cost. Equation (2) is a combinatorial optimization problem and the total number of possible $P^{\mathcal{V}}$s is $N!$ Trying all $N!$ possibilities is NP hard. In the following, we reformulate (2) as a TSP and solve it by applying dynamic programming.

A. Total Cost Minimization by Dynamic Programming

Minimizing the total cost is equivalent to finding the shortest path that traverses each of N CTUs only once, with the cost of traversing one CTU $\mathbf{y}_i^{\mathcal{V}}$ being the conditional coding cost $\mathrm{RD}(\mathbf{y}_i^{\mathcal{V}}|\mathbf{y}_{i-1}^{\mathcal{V}}, \mathbf{y}_{i-2}^{\mathcal{V}}, \ldots, \mathbf{y}_1^{\mathcal{V}})$. This can be regarded as an instance of the TSP [9]. As suggested in [10], we apply dynamic programming to solve TSP. The core idea of dynamic programming is to break down the original complicated problem into simpler subproblems in a recursive manner. In terms of solving (2), the broken down simpler subproblem is to find the optimal coding order of CTUs in indices subset $\mathcal{S} \subseteq \mathcal{V}$.

We define $C(\mathcal{S})$, the minimized cost of coding all CTUs in $\mathcal{S}$, as follows:

$$C(\mathcal{S}) = \min_{j\in\mathcal{S}}\big\{C(\mathcal{S}\setminus\{j\}) + \mathrm{RD}\big(j|\hat{\mathbf{y}}_{|\mathcal{S}|-1}^{\mathcal{S}\setminus\{j\}}, \hat{\mathbf{y}}_{|\mathcal{S}|-2}^{\mathcal{S}\setminus\{j\}}, \ldots, \hat{\mathbf{y}}_1^{\mathcal{S}\setminus\{j\}}\big)\big\} \tag{3}$$

where $\hat{\mathbf{y}}^{\mathcal{S}\setminus\{j\}} = \hat{\mathbf{P}}^{\mathcal{S}\setminus\{j\}}\mathbf{x}^{\mathcal{S}\setminus\{j\}}$ denotes the optimal coding order of all the CTUs in $\mathcal{S}\setminus\{j\}$. $\mathbf{x}^{\mathcal{S}\setminus\{j\}} \in \mathbb{N}^{(|\mathcal{S}|-1)}$ is a vector whose elements are all indices in $\mathcal{S}\setminus\{j\}$ sorted in ascending order. $\hat{\mathbf{P}}^{\mathcal{S}\setminus\{j\}}$ is the calculated optimal permutation matrix for CTUs in set $\mathcal{S}\setminus\{j\}$. $\mathrm{RD}(j|\hat{\mathbf{y}}_{|\mathcal{S}|-1}^{\mathcal{S}\setminus\{j\}}, \hat{\mathbf{y}}_{|\mathcal{S}|-2}^{\mathcal{S}\setminus\{j\}}, \ldots, \hat{\mathbf{y}}_1^{\mathcal{S}\setminus\{j\}})$ denotes the RD cost of CTU j after all the other CTUs in $\mathcal{S}$ have been coded.

[2]For clear presentation, by CTU i, we actually mean CTU indexed by i.

Our dynamic programming algorithm calculates $C(\mathcal{S})$ with all possible $\mathcal{S}$ from size 1 to N. That is, we first calculate $C(\mathcal{S})$ when $|\mathcal{S}| = 1$. In that case, $C(\{j\}) = \mathrm{RD}(j|\emptyset)$ is the RD cost of CTU j when it is the first processed CTU. After solving all $C(\{j\})$ with $|\mathcal{S}| = 1$, we then use (3) to obtain $C(\mathcal{S})$ for all $\mathcal{S}$ of size 2. This process is repeated until $|\mathcal{S}| = N$. The corresponding optimal coding order of each subset $\mathcal{S}$ is recorded after $C(\mathcal{S})$ is obtained. When $|\mathcal{S}| = N$ (i.e., $\mathcal{S} = \mathcal{V}$), we get the expected minimum cost $C(\mathcal{V})$ and the corresponding optimal order $\hat{\mathbf{y}}^{\mathcal{V}}$.

According to (3), when $|\mathcal{S}| = k$, k comparisons are required to get $C(\mathcal{S})$. There are $\binom{N}{k}$ kinds of $\mathcal{S}$ for $|\mathcal{S}| = k$. For calculating $C(\mathcal{S})$ with all possible $\mathcal{S}$ from size 1 to N, the total number of required comparisons as well as the required memory size is

$$1 \times \binom{N}{1} + 2 \times \binom{N}{2} + \cdots + N \times \binom{N}{N} = N \times 2^{N-1}. \tag{4}$$

We traverse all the possible CTU coding orders with a reduced complexity from $N!$ comparisons to $N \times 2^{N-1}$ comparisons. When N is reasonably small, the TSP can be efficiently solved by dynamic programming. One thing to note is that our problem is slightly different from the original TSP in [10]. In the original TSP, a salesman is required to visit once each of N different cities starting from a base city, and returning to this city. The cost is defined between two cities. The complexity is $N^2 \times 2^{N-1}$ [10]. While in our problem, the cost is defined in one CTU, not between two CTUs, and we do not need to return to the base CTU. As calculated in (4), the complexity becomes $N \times 2^{N-1}$.

In our experiments, large frames are divided into rectangular subregions. Each subregion contains $N = 5 \times 4$ CTUs and runs the dynamic programming algorithm independently. For $N = 20$, running the dynamic programming algorithm one time with given RD cost takes about 1 s in a computer with a 2.5-GHz Intel Core processor.

B. RD Cost Approximation

To solve (3), the actual RD cost of each CTU in any order must be calculated. However, it is impractical to try all the possible coding orders to get the exact RD cost for each CTU. Therefore, we propose to approximate the RD cost for CTUs in different orders.

As only eight neighboring CTUs can be employed as references to predict the current CTU in intra prediction, the RD cost of coding the current CTU depends on those neighboring CTUs which have already been coded. In our algorithm, only four-neighboring CTUs (above, below, left, and right) indexed as j^A, j^B, j^L, and j^R are considered, because the reference samples are mostly chosen from these four neighbors. The results from eight neighbors have a negligible improvement while with much more expensive complexity.

Letting $\mathcal{M}$ be the set of indices of neighboring CTUs coded before the current CTU

$$\mathcal{M} = \{j^A, j^B, j^L, j^R\} \cap \big\{\hat{\mathbf{y}}_{|\mathcal{S}|-1}^{\mathcal{S}\setminus\{j\}}, \hat{\mathbf{y}}_{|\mathcal{S}|-2}^{\mathcal{S}\setminus\{j\}}, \ldots, \hat{\mathbf{y}}_1^{\mathcal{S}\setminus\{j\}}\big\} \tag{5}$$

we approximate the RD cost in (3) as follows:

$$\mathrm{RD}(j|\mathcal{M}) \approx \mathrm{RD}\big(j|\hat{\mathbf{y}}_{|\mathcal{S}|-1}^{\mathcal{S}\setminus\{j\}}, \hat{\mathbf{y}}_{|\mathcal{S}|-2}^{\mathcal{S}\setminus\{j\}}, \ldots, \hat{\mathbf{y}}_1^{\mathcal{S}\setminus\{j\}}\big) \tag{6}$$

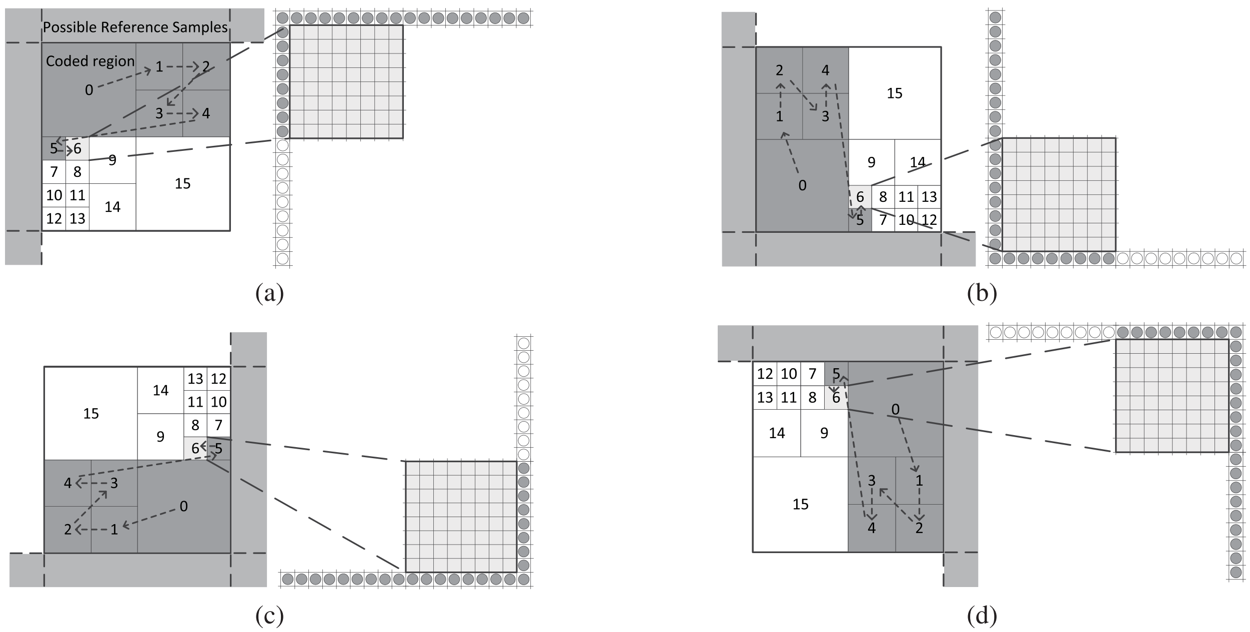


Fig. 4. Four kinds of CU coding orders. (a) Mode 0. (b) Mode 1. (c) Mode 2. (d) Mode 3.

which is the RD cost of CTU j after the CTUs in $\mathcal{M} \subseteq \{j^A, j^B, j^L, j^R\}$ have been coded. Since each of the four neighbors has either already been coded or not, there are in total $2^4 = 16$ possible $\mathcal{M}$s. When coding one CTU with proposed CU coding orders, at most two adjacent neighboring CTUs from $\mathcal{M}$ are employed as references to perform prediction. The details are given in Section III-C. Thus, the possible number of $\mathcal{M}$s is reduced from 16 to 9 as follows:

$$\mathcal{M} \in \{\emptyset, \{j^A\}, \{j^B\}, \{j^L\}, \{j^R\}, \{j^A, j^B\}, \{j^B, j^L\}, \{j^L, j^R\}, \{j^R, j^A\}\}. \tag{7}$$

Each CTU needs to be coded independently with the nine $\mathcal{M}$s, leading to nine RD cost values.

However, if only one CTU is coded, there are no available reconstructed reference pixels for prediction. Similar to [11] which employs original data rather than reconstructed ones for prediction, we also use the original data of neighboring CTUs as references to get the nine approximated RD cost values.

C. CU Coding Order

HEVC employs a Z-scan CU processing order when coding one CTU, as illustrated in Fig. 1. Such a scheme only allows left and/or above CTUs as reference samples to perform prediction, which fails to calculate RD$(j|\mathcal{M})$ when $\mathcal{M}$ contains right and/or below neighboring CTUs. Besides this, Z-scan order cannot handle certain edges, as shown in Fig. 3. Therefore, we propose three additional CU processing orders, namely, mode 1, mode 2, and mode 3 (the traditional Z-scan order is mode 0), as illustrated in Fig. 4. Taking mode 1 as an example, the CU processing order is from below to above and left to right. Left and below reconstructed CUs are employed as references to perform prediction. If left or below CUs have not been coded, the corresponding reference pixels are padded from neighboring available reconstructed pixels.

When coding one CTU with the proposed three CU orders, we flip the whole CTU together with corresponding reference pixels to make the CU order of flipped CTU same as the order in mode 0. This strategy is similar to that presented in [12]. The flipped CTU is then coded by applying the original HEVC encoder. To approximate the RD cost with one given $\mathcal{M}$, each CTU is coded with all the proposed four modes. The minimum RD cost is the expected RD$(j|\mathcal{M})$.

In summary, the final CTU coding order is obtained by (3) with the approximated RD cost. When one frame is coded with the obtained CTU order, every CTU will try the four CU modes. The mode with the minimum RD cost is the actual mode that needs to be signaled to the decoder. The CTU coding order also needs to be coded. We use the horizontal and vertical coordinates (x, y) of a CTU as the index. Only the difference of the coordinates $(\triangle x, \triangle y)$ between two sequentially processed CTUs is coded.

We solve the coding order optimization problem (2) by applying dynamic programming with updating rule (3). For traceable computation, the RD cost function in (3) is approximated by (6). Nevertheless, calculating RD$(j|\mathcal{M})$ is the most expensive part in our algorithm: RD has to be computed 9×4 times. Due to the fact that every subblock coding order is tested in calculating RD$(j|\mathcal{M})$, the final resultant coding order adaptive block coding order (ABCO) is essentially an optimal subblock-level coding order.

IV. Experiments

The proposed ABCO scheme is implemented in HEVC test model HM9.2 [13]. The calculated CTU coding order and CU coding mode for each CTU are coded as side information. Zero-order Exp-Golomb code is applied to code $(\triangle x, \triangle y)$ for the CTU coding order, and fixed length code is applied for CU coding modes. Test sequences in our experiments include all the 24 sequences in common test condition [14] along

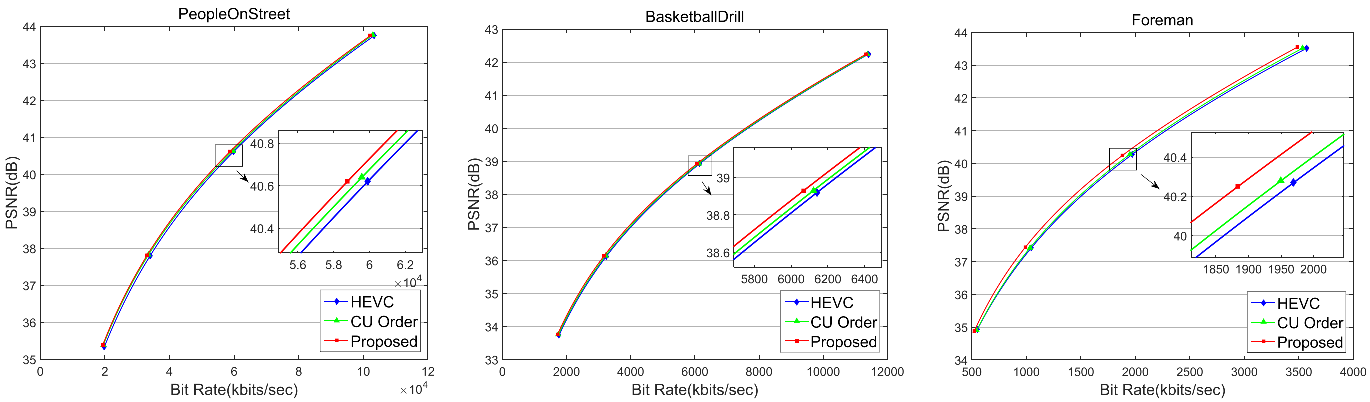


Fig. 5. RD curves of *PeopleOnStreet*, *BasketballDrill*, and *Foreman*.

TABLE I

BD-RATE RESULTS VERSUS HM9.2

ΔBD-Rate[%]		CU Order	Proposed
Sequences	Classes		
Traffic	Class A	1.7	2.2
PeopleOnStreet		1.8	2.4
Nebuta		1.2	1.9
SteamLocomotive		1.1	1.8
Kimono	Class B	0.6	0.7
ParkScene		0.2	0.5
Cactus		0.4	0.5
BasketballDrive		0.4	0.7
BQTerrace		0.5	1.4
BasketballDrill	Class C	1.0	1.9
BQMall		0.4	0.7
PartyScene		0.3	0.5
RaceHorses		0.5	0.8
BasketballPass	Class D	0.2	0.4
BQSquare		0.4	0.5
BlowingBubbles		0.6	0.3
RaceHorses		0.4	0.5
FourPeople	Class E	0.8	1.5
Johnny		0.8	1.3
KristenAndSara		0.6	1.6
SlideEditing	Class F	0.9	1.5
SlideShow		0.4	0.8
BaskeballDrillText		0.9	1.7
ChinaSpeed		0.7	1.0
Foreman	CIF	1.5	4.4
Akyio		0.5	1.2
Average		0.7	1.3
Enc.time ratio		3.48	35.71
Dec.time ratio		1.01	1.01

with two CIF sequences with smaller resolution. Some test sequences are cropped since currently our method can only handle CTU with square sizes. All sequences are tested under intra-main configuration defined in [14]. The test quantization parameters (QPs) are 22, 27, 32, and 37. The performance is compared by applying the current Bjøntegaard-Delta (BD)-rate measurement techniques to compare bit rates for equal peak signal to noise ratio (PSNR) [15]. The PSNR for YUV components is summarized as a single value by the following equation [16]:

$$\mathrm{PSNR}_{\mathrm{avg}} = (6 \times \mathrm{PSNR}_Y + \mathrm{PSNR}_U + \mathrm{PSNR}_V)/8.$$

A. Rate-Distortion Performance

Table I shows the numerical results of the proposed approach. Compared with HM9.2, the proposed approach has an average of 1.3% and up to 4.4% BD-rate reduction over the intra-main condition. The improvement is significant when the sequence contains rich edges such as *Traffic*, *PeopleOnStreet*, *BasketballDrill*, and *Foreman*. To evaluate the improvement led by optimizing CTU order and CU order separately, we compare the proposed approach with our previous work [3],[3] in which only the CU coding order is changed. The improvement of the proposed approach over [3] is thus the gain brought by optimizing the CTU order, which is on average 0.6% BD-rate reduction. The overhead of additional side information plays a small part in the total bits. Particularly, it is less than 0.2% for the largest test QP, i.e., 37.

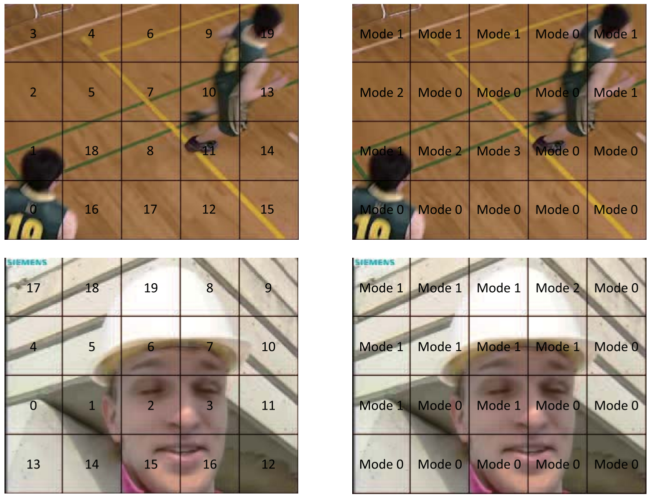
Fig. 5 plots the RD curves of *PeopleOnStreet*, *BasketballDrill*, and *Foreman*. We see that the proposed approach outperforms both our previous work and HEVC at different quantization levels, particularly at medium to small QP. When QP becomes large (i.e., the distortion also becomes large), the RD cost approximation by original frames instead of reconstructed frames in Section III-B tends to be inaccurate. In general, the objective results verify that the proposed ABCO approach compresses the edge region more efficiently than HEVC with fixed block coding order.

B. Coding Order Analysis

Fig. 6(a) shows the obtained CTU coding order for one subregion of *BasketballDrill* and *Foreman*. Fig. 6(b) gives the corresponding CU coding orders for each CTU. We observe that: 1) in most cases, the current CTU is adjacent to the previous coded one and 2) both the CTU and CU coding orders are approximately along the main edge direction. Specifically, for the below-left edges on the left part of the building, the CTUs are coded from below-left to above-right, and the corresponding CU order (mode 1) is the below-left to above-right Z-scan order. For the below-right edges on the right part of the building, the above-left to below-right block coding orders are mostly chosen.

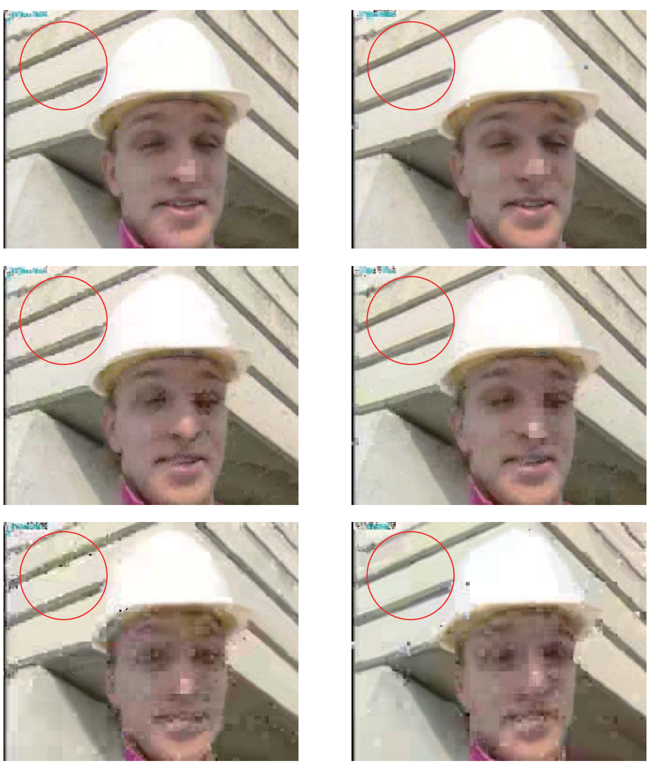
For a visual comparison, Fig. 7 gives the predicted images by HEVC and the proposed approach when QP = 22, 32, and 42. We can see that the proposed approach outperforms

[3]The implementation here is slightly different from our previous work which does not consider mode 2 in Fig. 4, while four modes are all used here to better exploit the improvement brought by optimizing the CTU order.



(a) (b)

Fig. 6. Coding order on the first frame of *BasketballDrill* (one subregion) and *Foreman* at QP = 32. (a) CTU coding order. (b) CU coding order.



(a) (b)

Fig. 7. Intra-predicted images of the first frame of *Foreman*. Top to bottom: QP = 22, 32, and 42. (a) HEVC intra prediction. (b) Proposed intra prediction.

HEVC in predicting edges over a wide range of QP, especially the edges inside the red circles.

C. Computational Complexity and Parallelism

Table I shows that the encoding running time ratio of the proposed approach is on average 35.71 times the one of HEVC, which is approximately nine times than that of [3]. The decoding time of both our methods is almost identical to that of HEVC. The extra encoding computation mainly comes from the RD cost approximation part. This is because each CTU is coded under 9×4 CU coding orders.

One attempt of reducing the encoding complexity is to adopt parallelism techniques, as there is no dependency among CTUs when approximating the RD cost. It is possible to approximate all the RD cost in parallel and combine their results. From the analyses and the experimental results, we find that both the optimal CTU and the optimal CU coding order are mostly achieved along the dominant edge direction. Thus, another suboptimal attempt is to select the CTU and the CU coding order based on the detected dominant edge direction in the image.

In terms of parallelism, there are three ways to enable parallel processing in HEVC: 1) slices; 2) tiles; and 3) wavefront parallel processing (WPP) [2]. Each of them may have benefits in particular application contexts. The proposed approach is compatible with slices and tiles, which both split frame into independently decodable regions with integer number of CTUs. Each region can run the proposed approach independently. Unfortunately, the proposed approach does not support WPP, because WPP is based on that only above and/or left neighboring CTUs can be employed as references to perform prediction.

V. Conclusion

We present an improved intra-prediction scheme with ABCO. In contrast to HEVC intra coding with fixed block coding orders, the proposed scheme is capable of making better predictions in edge regions by adaptively selecting both block and subblock coding orders. The experimental results demonstrate that the proposed scheme achieves an average of 1.3% and up to 4.4% bit saving compared with HEVC (HM9.2).

References

[1] *Advanced Video Coding for Generic Audiovisual Services*, ISO/IEC Standard 14496-10, 2005.

[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] A. Zheng, O. C. Au, Y. Yuan, H. Yang, J. Pang, and Y. Ling, "Intra prediction with adaptive CU processing order in HEVC," in *Proc. IEEE Int. Conf. Image Process.*, Oct. 2014, pp. 3724–3728.

[4] T. Shiodera, A. Tanizawa, and T. Chujoh, "Block based extra/inter-polating prediction for intra coding," in *Proc. IEEE Int. Conf. Image Process.*, Sep./Oct. 2007, pp. VI-445–VI-448.

[5] I. Matsuda, Y. Ohtake, S. Mochizuki, H. Fukai, and S. Itoh, "Interpolative intra prediction by adapting processing order in block-based image coding," in *Proc. IEEE Int. Conf. Image Process.*, Sep. 2013, pp. 1646–1650.

[6] S. Wenger and M. Horowitz, *FMO-101*, document JVT-D063, JVT, Klagenfurt, Austria, Jul. 2002.

[7] P. Lambert, W. De Neve, Y. Dhondt, and R. Van de Walle, "Flexible macroblock ordering in H.264/AVC," *J. Vis. Commun. Image Represent.*, vol. 17, no. 2, pp. 358–375, Apr. 2006.

[8] J. Lainema, F. Bossen, W.-J. Han, J. Min, and K. Ugur, "Intra coding of the HEVC standard," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 22, no. 12, pp. 1792–1801, Dec. 2012.

[9] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to Algorithms*. Cambridge, MA, USA: MIT Press, 2001.

[10] R. Bellman, "Dynamic programming treatment of the travelling salesman problem," *J. ACM*, vol. 9, no. 1, pp. 61–63, 1962.

[11] T. A. da Fonseca, Y. Liu, and R. L. de Queiroz, "Open-loop prediction in H.264/AVC for high definition sequences," in *Proc. 25th Simpósio Brasileiro Telecomunicações*, 2007, pp. 1–4.

[12] T. Murakami, I. Karube, H. Ito, and M. Takahashi, "Adaptive picture flipping coding for enhancing H.264/AVC," in *Proc. 26th Picture Coding Symp.*, Nov. 2007, pp. 1–4.

[13] *HEVC Test Model*. [Online]. Available: https://hevc.hhi.fraunhofer.de/svn/svn_HEVCSoftware/tags/

[14] F. Bossen, *Common Test Conditions and Software Reference Configurations*, document JCTVC-K1100, JCT-VC, Shanghai, China, Oct. 2012.

[15] G. Bjontegaard, *Calculation of Average PSNR Differences Between RD-Curves*, document Rec. VCEG-M33, ITU-T, Apr. 2001, pp. 1–9.

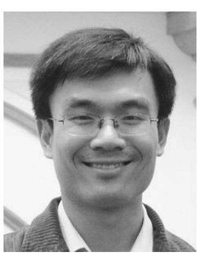
[16] D. Hoang, *PSNR Computation on R'G'B' Color System*, document JCTVC-D040, JCT-VC, Turin, Italy, Jan. 2011.

Amin Zheng (S'13) received the B.S. degree from the Department of Electronic and Information Engineering, Huazhong University of Science and Technology, Wuhan, China, in 2012. He is currently pursuing the Ph.D. degree in electronic and computer engineering with The Hong Kong University of Science and Technology, Hong Kong.

His current research interests include image/video compression, restoration, and machine learning.

Yuan Yuan (S'13) received the B.S. degree from the Department of Electronic and Information Engineering, Huazhong University of Science and Technology, Wuhan, China, in 2012. She is currently pursuing the Ph.D. degree in electronic and computer engineering with The Hong Kong University of Science and Technology, Hong Kong.

Her current research interests include image/video processing, machine learning, and computer vision.



Jiantao Zhou (M'11) received the B.Eng. degree from the Department of Electronic Engineering, Dalian University of Technology, Dalian, China, in 2002, the M.Phil. degree from the Department of Radio Engineering, Southeast University, Nanjing, China, in 2005, and the Ph.D. degree from the Department of Electronic and Computer Engineering, The Hong Kong University of Science and Technology, Hong Kong, in 2009.

He held various research positions at the University of Illinois at Urbana–Champaign, Champaign, The Hong Kong University of Science and Technology, Hong Kong, and the McMaster University, Hamilton, ON, Canada. He is currently an Assistant Professor with the Department of Computer and Information Science, Faculty of Science and Technology, University of Macau, Macau, China. He holds three granted U.S. patents and two granted Chinese patents. His current research interests include multimedia security and forensics, and high-fidelity image compression.

Dr. Zhou was a co-author of a paper that received the best paper award at the IEEE Pacific-Rim Conference on Multimedia in 2007.

Yuanfang Guo (S'11–M'15) received the B.S. and Ph.D. degrees from The Hong Kong University of Science and Technology, Hong Kong, in 2009 and 2015, respectively.

He is currently an Assistant Professor with the State Key Laboratory of Information Security, Institute of Information Engineering, Chinese Academy of Sciences, Beijing, China. His current research interests include image/video watermarking, data hiding, forensics, compression, restoration, and enhancement.

Haitao Yang (M'14) received the B.S. and Ph.D. degrees in electronics engineering from Xidian University, Xi'an, China, in 2004 and 2009, respectively.

He was involved in the project of Multiview Video Coding and 3D video coding from 2006 to 2009. Since 2009, he has been involved in the project of High Efficiency Video Coding standardization, focusing on the research of the spatial transform and the coefficient scanning. He was with The Hong Kong University of Science and Technology, Hong Kong, from 2009 to 2010, where he was involved in video compression as a Post-Doctoral Research Associate. In 2010, he joined Huawei Technologies, Shenzhen, China. He is currently a Principle Engineer with the Media Technology Laboratory, Corporate Research Department of Huawei, where he is involved in compression and processing algorithms for video communication.

Oscar C. Au (S'87–M'90–SM'01–F'11) received the B.A.Sc. degree from the University of Toronto, Toronto, ON, Canada, in 1986, and the M.A. and Ph.D. degrees from Princeton University, Princeton, NJ, USA, in 1988 and 1991, respectively.

He spent one year as a Post-Doctoral Fellow with Princeton University. He joined The Hong Kong University of Science and Technology (HKUST), Hong Kong, as an Assistant Professor in 1992. He is/was a Professor of Department of Electronic and Computer Engineering, the Director of the Multimedia Technology Research Center, and Director of Computer Engineering at HKUST. His main research contributions are on video/image coding and processing, watermarking/light weight encryption, and speech/audio processing. He has authored over 60 technical journal papers, 350 conference papers, and 70 contributions to international standards. He has over 20 granted U.S. patents and is applying for more than 70 on his signal processing techniques. His current research interests include fast motion estimation for H.261/3/4/5, MPEG-1/2/4, and AVS, optimal and fast suboptimal rate control, mode decision, transcoding, denoising, deinterlacing, post-processing, multiview coding, view interpolation, depth estimation, 3DTV, scalable video coding, distributed video coding, subpixel rendering, JPEG/JPEG2000, HDR imaging, compressive sensing, halftone image data hiding, GPU processing, and software-hardware co-design.

Dr. Au is a fellow of HKIE and a BoG member of APSIPA. He is a member of six other technical committees, such as the IEEE CAS VSPC TC, DSP TC, the IEEE SPS IVMSP TC, IFS TC, and the IEEE ComSoc MMC TC. He received five best paper awards, such as SiPS 2007, PCM 2007, MMSP 2012, ICIP 2013, and MMSP 2013. His fast motion estimation algorithms were accepted into the ISO/IEC 14496-7 MPEG-4 international video coding standard and the China AVS-M standard. His light-weight encryption and error resilience algorithms are accepted into the China AVS standard. He was the Chair of Screen Content Coding AdHoc Group in JCTVC for HEVC. He has performed forensic investigation and stood as an expert witness in Hong Kong courts many times. He is/was Associate Editor of eight journals, such as the IEEE Transactions on Circuits and Systems for Video Technology, the IEEE Transactions on Image Processing, the IEEE Transactions on Circuits and Systems I, the *Journal of Visual Communication and Image Representation*, the Journal of Signal Processing Systems, *TSIP*, the *Journal of Micromechanics and Microengineering*, and the *Journal of Financial Intermediation*. He is/was Chair of three technical committees, such as the IEEE CAS MSA TC, the IEEE SPS MMSP TC, and the APSIPA IVM TC. He served on two steering committees, such as the IEEE Transactions on Multimedia, and the IEEE International Conference on Multimedia and Expo. He also served on organizing committee of many conferences, including ISCAS 1997, ICASSP 2003, ISO/IEC 71st MPEG in 2005, and ICIP 2010. He was/will be the General Chair of several conferences, such as PCM 2007, ICME 2010, PV 2010, APSIPA ASC 2015, and ICME 2017. He was the IEEE Distinguished Lecturer (DL) in 2009 and 2010, and APSIPA DL in 2013 and 2014, and has been a keynote speaker multiple times.