

Paper

Memory Bandwidth Constrained Overlapped Block Motion Compensation for Video Coding

Yoshitaka Kidani (student member)[†,††], Kyohei Unno (member)[†], Kei Kawamura (member)[†], Hiroshi Watanabe (member)[††]

Abstract Overlapped block motion compensation (OBMC) is an inter prediction tool that improves coding performance by blending reference samples of the current and neighboring block across the block boundaries. For this mechanism, OBMC increases the reference samples fetched from the external memory on hardware decoders, i.e., memory bandwidth. This is a disadvantage when introducing OBMC to video coding standards such as VVC, especially for mobile devices with limited batteries, because the extended memory bandwidth increases decoders' power consumption. In this paper, we propose a memory bandwidth constrained OBMC method with adaptive number of motion vectors and interpolation filter taps of the neighboring blocks depending on the current block sizes. Simulation results show that the proposed method achieves -0.22 % performance improvements over VVC reference software without exceeding the maximum memory bandwidth of VVC, which is comparable to the full performance of OBMC (-0.33 %), requiring 3.8 times its memory bandwidth.

Key words: video coding, memory bandwidth, inter prediction, overlapped bock motion compensation, versatile video coding

1. Introduction

Highly efficient video coding is indispensable as data traffic for video streaming services increases due to the spread of mobile devices, broadband internet access, and demand for ultra high-definition (UHD) video[1]. Video coding is developed by considering the trade-off between service requirements and encoder/decoder specifications, i.e., coding performance and complexity such as processing time and memory bandwidth. In particular, since memory bandwidth is proportional to power consumption, the memory bandwidth reduction is essential for battery-powered mobile devices[2][3][4]. Various international standards of video coding have been developed by a joint team of VCEG of ITU-T and MPEG of ISO/IEC. High efficiency video coding (HEVC) developed in 2013[5] is currently widely used for UHD video streaming over broadband networks (e.g., fixed and satellite), whereas versatile video coding (VVC) developed in 2020[6] is being considered for that over narrowband networks (e.g., terrestrial and mobile). Both HEVC and VVC are organized by the partitioning of a picture into smaller blocks, block-wise inter predictions such as motion compensation (MC), intra predictions, transforms, and in-loop filters.

In the block-wise MC of HEVC and VVC, called regular MC (RMC), predicted sample values near the block boundary are discontinued when the motion vectors of current blocks and neighboring blocks (mv_{C} and mv_{N} hereinafter) differ. Overlapped block motion compensation (OBMC) and de-blocking filters (DBF) are well-known solutions to this problem. As shown in Fig. 1, OBMC blends the predicted samples generated by mv_{N} across the block boundary into those generated by mv_{C} to reduce the block discontinuities[7][8]. In contrast, DBF directly smooths block boundaries of the reconstructed blocks generated by the predicted blocks and residual blocks[5][6]. OBMC and DBF provide additive improvements because of their different mechanisms but OBMC has not yet been adopted in HEVC and VVC due to the issue of increasing the number of reference samples, which is approximated as memory bandwidth[9][10]. We, therefore, focus on the memory bandwidth constrained OBMC method for further improvements of VVC. As that related work, uni-prediction based OBMC (i.e., OBMC using only one mv_{C} and one mv_{N})[11] was proposed but not adopted in VVC. This is because it significantly reduces the coding performance improvement with bitrate saving (coding gain hereinafter) of the bi-prediction based

Received March 30, 2022; Revised August 30, 2022; Accepted October 11, 2022
† KDDI Research, Inc. (Saitama, Japan)
†† Waseda University (Tokyo, Japan)

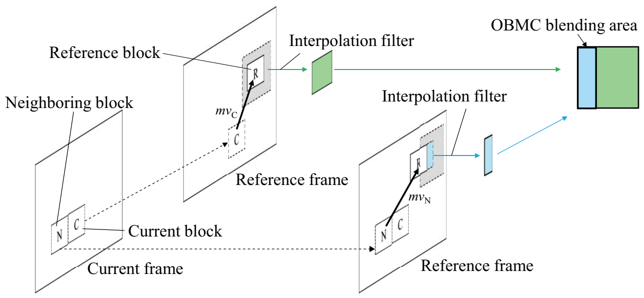


Fig. 1 A mechanism of OBMC. C, N, and R denote the current blocks, neighboring blocks, and reference blocks, respectively. Shaded areas represent the extended reference sample areas against R when interpolation filters are required, that is when mv_{C} and mv_{N} indicate non-integer-sample positions, respectively.

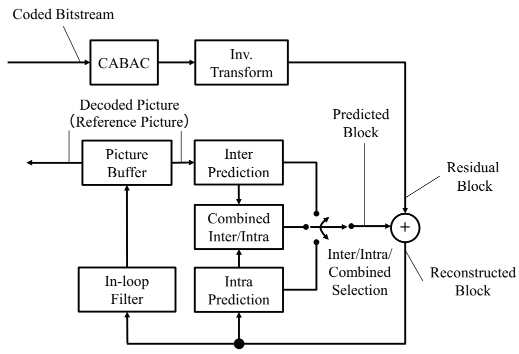


Fig. 2 An example of the architecture for the general video decoder.

OBMC (i.e., OBMC using two mv_{C} and two mv_{N})[12].

To tackle the problem, we propose a uni-prediction based OBMC method with adaptive number of motion vectors and interpolation filter taps of the neighboring blocks (n_{N} and t_{N}) depending on the current block sizes. Specifically, we generalize the proposed method as an objective function that maximizes the coding performance with n_{N} and t_{N} as variables, while not exceeding the worst-case upper limit of the memory bandwidth (M_{Wst}) on RMC and OBMC as the constraint. In this paper, we implemented the proposed method into the VVC reference software by setting the constraint as M_{Wst} of VVC (M_{WstVVC}) as an example. Simulation results show that the proposed method provides an additional coding gain over VVC reference software (-0.22 %). This gain is comparable to that of bi-prediction based OBMC (-0.33 %) requiring 3.8 times the maximum memory bandwidth of VVC, and is still greater than that of uni-prediction based OBMC (-0.12 %).

The rest of this paper is organized as follows. Memory bandwidth and related work are explained in Sec. 2. The problems are shown in Sec. 3. Section 4 presents the details of the proposed method. Section 5 describes the experimental results and discussion. Finally, we conclude the paper in Sec. 6.

2. Memory Bandwidth and Related Work

2.1 Memory Bandwidth

An architecture of the general video decoder including the portions for the inter prediction, intra prediction, and inverse transform, in-loop filter, and picture buffer, is shown in Fig. 2. The function of RMC of HEVC and VVC is included in the inter prediction and the reference samples for RMC are obtained from the picture buffer. As the video services achieve higher definition, an external memory is now commonly used for the picture buffer to store the reference picture in the hardware video decoder[4]. This throughput of the access from the external memory to the main chip on the hardware video decoder is called the memory bandwidth.

For the hardware video decoder, the memory bandwidth is designed by the M_{Wst} since it cannot be changed after manufacturing. In general, the M_{Wst} is approximated to be the maximum number of reference samples required for a predicted sample of RMC, as well as the related works[9][10]. Hence, M_{Wst} is increased when more reference samples need to be fetched with the interpolation filter than the number of samples within the prediction block. In other words, the smaller the blocks are, the larger M_{Wst} becomes. The reduction of the M_{Wst} is critical especially for the mobile devices since it saves on power consumption[2][3][4].

2.2 Factors Increasing Memory Bandwidth

The M_{Wst} for RMC is varied depending on the number of motion vectors and the interpolation filter taps of the current blocks (n_{C} and t_{C}). Regarding n_{N}, the maximum number is two, i.e., bi-prediction, and it is utilized in HEVC and VVC, for example. When n_{N} is two, the required reference samples become double as shown in Fig. 3(a) and (b). Regarding t_{N}, 8-tap and 4-tap filters are used for generating the predicted samples of the luma and chroma components, respectively. As t_{N} becomes longer, the required reference samples are increased as shown in Fig. 3(c) and (d).

The M_{Wst} for RMC also depends on the current block sizes. VVC diversifies the block partitioning including non-square shape, not in HEVC, so that selects the coding block size from the minimum 4×4

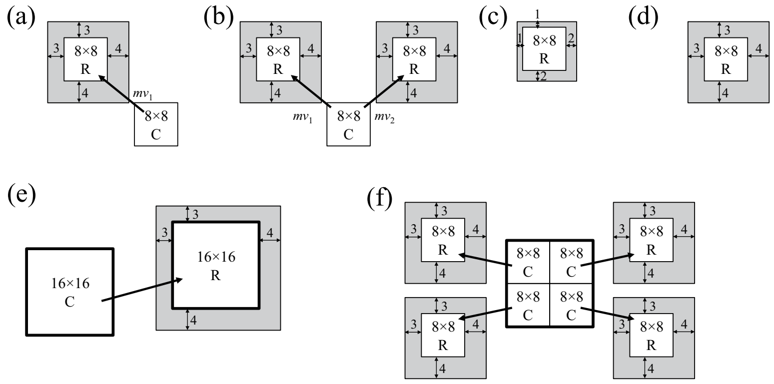


Fig. 3 Examples of the memory bandwidth required for RMC of the current blocks increased by three factors, i.e., the number of motion vectors, the number of interpolation filter taps, and the size of the current block against the size of pipeline processing for RMC. C, R, and shaded areas denote current blocks, reference blocks, and extended reference sample areas by interpolation filter against R, respectively. (a) Uni-prediction, (b) Bi-prediction, (c) 4-tap filter, (d) 8-tap filter, (e) an RMC pipeline organized by one 16 × 16 current block, and (f) an RMC pipeline organized by four 8 × 8 current blocks.

to the maximum 128 × 128 samples, for instance[6]. The extension of the block sizes increases the required internal memory, storing the reference samples from the external memory, of the main chip on the hardware video decoder[13]. The RMC of the current block is generally conducted at the subblock level to reduce the required internal memory sizes. The maximum size of the subblock is defined as 16 × 16 samples by decoder-side motion vector refinement (DMVR) in VVC. This means that each 16 × 16 predicted sample value in RMC is always the same as the non-subblock-wise RMC, and it assures the 16 × 16 sample-wise pipeline processing within RMC.

The reference samples for the pipeline processing of RMC are fetched from the external memory all at once. Hence, the more the current block consists of the multiple smaller size blocks, the more the reference samples are required as shown in Fig. 3(e) and (f). To reduce the M_{Wst}, the minimum block sizes for the uni-prediction and bi-prediction of VVC are constrained by $4 \times 8/8 \times 4$ and 8×8, respectively[6].

2. 3 Overlapped Block Motion Compensation

OBMC increases the memory bandwidth since the reference samples of neighboring blocks are required compared to those of only RMC. In addition to n_{N} and t_{N} for OBMC as in RMC, the reference samples required for OBMC are determined by the application locations and blending lines as shown in Fig. 4(a) and (b).

The original method of OBMC[7][8] does not prohibit the application of OBMC to the current block boundaries on all four sides (e.g., top, left, right, and bottom), which is sometimes called non-causal OBMC[14]. The non-causal OBMC raises another implementation issue except for the memory bandwidth. Specifically, the blending for the right and bottom side in the non-causal OBMC increases an encoding and decoding delay for block-wise processing in raster-scan order. In addition, for parallel processing, the non-causal OBMC needs to fetch ahead in the lower right blocks, but this requires storage for a large number of reference samples, which increases the internal memory size. To address the problem, the causal OBMC where OBMC can be applied only for the top and left sides is proposed[15]. In this paper, we follow the causal OBMC since we focus on the practical OBMC method.

Another method following the causal OBMC method but enabling the various size blocks including

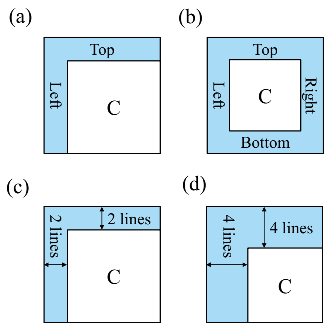


Fig. 4 Examples of the memory bandwidth required for OBMC of the current blocks increased by two factors, i.e., OBMC applicable locations and blending lines. C, and sky-blue area denote current blocks and OBMC blending area. (a) Only top and left sides, (b) all sides, (c) 2-lines, (d) 4-lines.

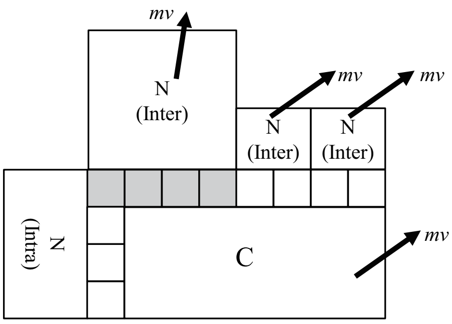


Fig. 5 An example of the SbOBMC. C and N denote the current block and neighboring blocks. Shaded and non-shaded subblocks indicate SbOBMC applied and non-applied subblocks.

non-square shaped types is proposed as shown in Fig. 5 to realize the coding performance beyond HEVC (Chen2015 hereinafter)[16]. Chen2015 newly introduces the 4×4 subblock-wise OBMC to correspond the neighboring blocks with various block sizes and prediction modes (i.e., inter or intra). The 4 × 4 subblock-wise OBMC (SbOBMC) enables a detailed applicable determination depending on the prediction modes and similarity of the motion vector of the neighboring blocks as shown in Fig. 5[16]. The SbOBMC applicable determination is conducted only when OBMC is determined to be applicable in a coding block, which is identified by a block-wise *obmc_flag* signaled from the encoder. For the signaling *obmc_flag*, the encoder calculates the rate-distortion (RD) costs with and without OBMC applied after determining whether the current block is uni-prediction or bi-prediction.

Furthermore, Chen2015 proposes adjusting the OBMC blending lines depending on the current block sizes to reduce the M_{Wst}. Specifically, Chen2015 utilizes 4 lines when the width or height of the current block is larger than 8 samples, otherwise it utilizes 2 lines. However, Chen2015 can apply OBMC even for bi-prediction current blocks having bi-prediction neighboring blocks, which increases the M_{Wst} against RMC in the small size current blocks.

To solve this problem, uni-prediction based OBMC is proposed (Lin2019 hereinafter)[11]. Lin2019 enables OBMC only for uni-prediction current blocks having a uni-prediction neighboring block. In order to maintain the OBMC application rates, Lin2019 can apply OBMC for uni-prediction current blocks having bi-prediction neighboring blocks by converting neighboring blocks from bi-prediction to uni-prediction based on the distance between the current and reference pictures. Lin2019 further proposes prohibiting the application of OBMC for $4 \times 8/8 \times 4$ blocks to reduce the M_{Wst}.

3. Problem Statement

In this section, first, we derive the formula to calculate the M_{Wst} for RMC and OBMC, and analyze the M_{Wst} of Chen2015 and Lin2019, when implemented in VVC as an example. Second, we ascertain the bottle-neck of OBMC by changing t_{N} in these two methods.

The memory bandwidth of the final inter predicted $W \times H$ sample block $M_{\mathrm{Inter}}^{W\times H}$ can be calculated depending on whether OBMC is applicable or not as

$$M_{\mathrm{Inter}}^{W\times H} = \begin{cases} M_{\mathrm{RMC}}^{W\times H} + M_{\mathrm{OBMC}}^{W\times H} & \text{if applicable,} \\ M_{\mathrm{RMC}}^{W\times H} & \text{otherwise,} \end{cases} \quad (1)$$

where W, H, $M_{\mathrm{RMC}}^{W\times H}$, and $M_{\mathrm{OBMC}}^{W\times H}$ denote the width of the current block, the height of the current block, the memory bandwidth of RMC, and the memory bandwidth of OBMC, respectively. Here, $M_{\mathrm{RMC}}^{W\times H}$ can be calculated as

$$M_{\mathrm{RMC}}^{W\times H} = (W + t_{\mathrm{C}} - 1) * (H + t_{\mathrm{C}} - 1) * n_{\mathrm{C}} * \left(\frac{P}{W * H}\right), \quad (2)$$

where P indicates the number of samples for the pipeline processing of RMC, which depends on the implementation, and 16×16 samples are provided in the VVC case as described in Sec. 2.2. $M_{\mathrm{OBMC}}^{W\times H}$ can also be calculated as

$$M_{\mathrm{OBMC}}^{W\times H} = (M_{\mathrm{OBMC_T}}^{W\times H} * \frac{W}{w} + M_{\mathrm{OBMC_L}}^{W\times H} * \frac{H}{h}) * n_{\mathrm{N}} * \left(\frac{P}{W * H}\right), \quad (3)$$

where $M_{\mathrm{OBMC_T}}^{W\times H}$, $M_{\mathrm{OBMC_L}}^{W\times H}$, w, and h represent the memory bandwidth of OBMC for the top-side block boundary, the memory bandwidth of OBMC for the left-side block boundary, the width of SbOBMC, and the height of SbOBMC, respectively. Finally, $M_{\mathrm{OBMC_T}}^{W\times H}$ can be calculated as

$$M_{\mathrm{OBMC_T}}^{W\times H} = (w + t_{\mathrm{N}} - 1) * \left[\min\left(\frac{h}{l_1}, l_2\right) + t_{\mathrm{N}} - 1\right]. \quad (4)$$

Here, l_1 and l_2 indicate OBMC blending lines for smaller size blocks and larger size blocks, respectively. $M_{\mathrm{OBMC_L}}^{W\times H}$ can also be calculated as $M_{\mathrm{OBMC_T}}^{W\times H}$.

Table 1 and Fig. 6 show the M_{Wst} of Chen2015 and Lin2019 calculated with the derived formula,

Table 1 Combination of the factors for OBMC in the conventional methods. $s_{\mathrm{min/Uni}}$ and $s_{\mathrm{min/Bi}}$ denote the minimum current block size for uni-prediction and bi-prediction of RMC.

Method	$s_{\mathrm{min/Uni}}$	$s_{\mathrm{min/Bi}}$	n_{C}	n_{N}
Chen2015	4×8	8×8	2	2
Chen2015/PhbtC	4×8	8×8	1	2
Chen2015/PhbtCN	4×8	8×8	1	1 $(2 \rightarrow 1)$
Lin2019	8×8	-	1	1 $(2 \rightarrow 1)$

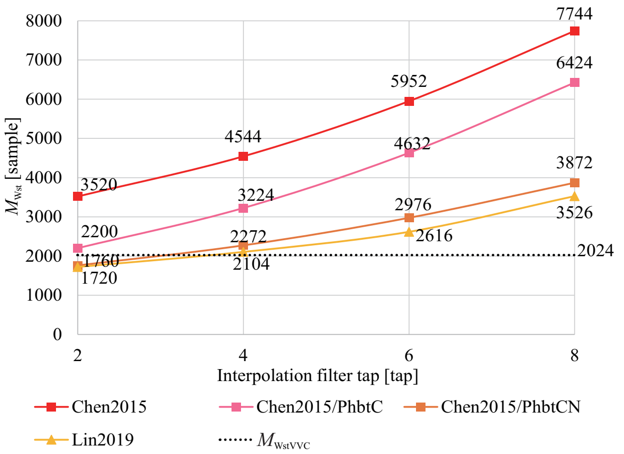


Fig. 6 Analysis of the M_{Wst} for each OBMC method using 2, 4, 6, and 8-tap interpolation filters fixedly.

including non-proposed t_{N} and considering VVC. For the detailed analysis, M_{Wst} of two additional conditions based on Chen2015 is compared as described in "Chen2015/PhbtC" and "Chen2015/PhbtCN" in Fig. 6. Chen2015/PhbtC introduces OBMC prohibition for the bi-prediction current blocks, i.e., OBMC can be applied for uni-prediction current blocks having a bi-prediction or uni-prediction neighboring block in this method. In contrast, Chen2015/PhbtCN further introduces OBMC prohibition for uni-prediction current blocks having bi-prediction neighboring blocks, i.e., OBMC can be applied only for uni-prediction current blocks having uni-prediction neighboring blocks in this method. The difference of the OBMC applicable condition for each method is shown in Table. 1. All M_{Wst} is calculated using only the luma component to simplify the comparison. The M_{WstVVC} is 2,024 $(= (16 + 7) * (4 + 7) * 2 * (16 * 16)/(4 * 16))$ reference samples, which are required in RMC for four sets of the bi-prediction $4 \times 16/16 \times 4$ current block with an 8-tap interpolation filter. Note that VVC has a 4×4 block-wise MC, i.e., affine MC, but its required memory bandwidth is constrained so as not to exceed the M_{WstVVC}[17]. Therefore, we focus on the RMC and OBMC.

From Fig. 6, it is clear that the bottle-neck is the application of OBMC for the bi-prediction current blocks having bi-prediction neighboring blocks since the M_{Wst} of Chen2015 exceeds the M_{WstVVC} by over 3.8 times ($\simeq$ 7,744/2,024) with the 8-tap filter and more than 1.7 times ($\simeq$ 3,520/2,024) even with the 2-tap filter. This is because $M_{\mathrm{OBMC}}^{W\times H}$ is purely added into the M_{WstVVC} in Chen2015. Chen2015/PhbtC still exceeds the M_{WstVVC} even with the 2-tap filter, whereas Chen2015/PhbtCN and Lin2019 with the 2-tap filter become smaller than the M_{WstVVC}. This means that uni-prediction based OBMC with the 2-tap filter can reduce the M_{Wst} lower than M_{WstVVC}. However, n_{N} and t_{N} have room to be adaptive in the larger size blocks since the required memory bandwidth becomes smaller in these blocks.

4. Proposed Method

We propose a memory bandwidth constrained OBMC method with an adaptive number of motion vectors and interpolation filter taps of the neighboring blocks depending on the current block sizes (i.e., $n_{\mathrm{N}}^{W\times H} and t_{\mathrm{N}}^{W\times H}$), with the aim of retaining the potential coding performance of OBMC while constraining $M_{\mathrm{OBMC}}^{W\times H}$. In this paper, we tried to generalize the proposed method[18] as an objective function that maximizes $M_{\mathrm{Inter}}^{W\times H}$ with the $n_{\mathrm{N}}^{W\times H}$ and $t_{\mathrm{N}}^{W\times H}$ as variables such that the constraint, $M_{\mathrm{Inter}}^{W\times H}(n_N, t_N) < M$, is satisfied. Here, M is an arbitrary memory bandwidth. This is because maximizing $n_{\mathrm{N}}^{W\times H}$ and $t_{\mathrm{N}}^{W\times H}$ contributes to increasing the coding gain of OBMC. The proposed method can be generalized as the following formula:

$$\hat{n}_{\mathrm{N}}^{W\times H}, \hat{t}_{\mathrm{N}}^{W\times H} = \underset{n_{\mathrm{N}}, t_{\mathrm{N}}}{\mathrm{argmax}} M_{\mathrm{Inter}}^{W\times H}(n_{\mathrm{N}}^{W\times H}, t_{\mathrm{N}}^{W\times H})$$
$$\text{s.t. } M_{\mathrm{Inter}}^{W\times H}(n_{\mathrm{N}}^{W\times H}, t_{\mathrm{N}}^{W\times H}) < M, \quad (5)$$

where $\hat{n}_{\mathrm{N}}^{W\times H}$ and $\hat{t}_{\mathrm{N}}^{W\times H}$ are the maximized combination of the interpolation filter taps and number of motion vectors of neighboring blocks for each current block size, $W \times H$, as an output of the formula.

The flowchart of the searching algorithm for the combination of $\hat{n}_{\mathrm{N}}^{W\times H}$ and $\hat{t}_{\mathrm{N}}^{W\times H}$ is shown in Fig. 7. Mainly, the flowchart consists of the following seven steps.

S1) Maximize the value of $n_{\mathrm{N}}^{W\times H}$ and proceed to S2

S2) Maximize the value of $t_{\mathrm{N}}^{W\times H}$ and proceed to S3

S3) Evaluate whether the constraint is satisfied with the current $n_{\mathrm{N}}^{W\times H}$ and $t_{\mathrm{N}}^{W\times H}$, and if so, determine them as $\hat{n}_{\mathrm{N}}^{W\times H}$ or $\hat{t}_{\mathrm{N}}^{W\times H}$, and proceed to S7. If not, proceed to S4

S4) Evaluate whether $t_{\mathrm{N}}^{W\times H}$ is the minimum value, and if so, proceed to S5. If not, reduce $t_{\mathrm{N}}^{W\times H}$, and return to S3

S5) Evaluate whether $n_{\mathrm{N}}^{W\times H}$ is the minimum value, and if so, proceed to S6. If not, reduce $n_{\mathrm{N}}^{W\times H}$, and return to S2

S6) Evaluate whether the constraint is satisfied with the current $n_{\mathrm{N}}^{W\times H}$ and $t_{\mathrm{N}}^{W\times H}$, and if so, determine them as $\hat{n}_{\mathrm{N}}^{W\times H}$ and $\hat{t}_{\mathrm{N}}^{W\times H}$, and proceed to S7. If not, determine that none of those that satisfy the constraints have been found, and proceed to S7 Here, the reason for prioritizing $n_{\mathrm{N}}^{W\times H}$ over $t_{\mathrm{N}}^{W\times H}$ in this search algorithm is that $n_{\mathrm{N}}^{W\times H}$ has a greater impact on coding performance. Note that whether OBMC is applied or not is finally determined by *obmc_flag* signaled by the encoder in the proposed method.

As an example, in this paper, the OBMC applicable conditions without exceeding 1.0 and 1.5 times M_{WstVVC} (the "Proposal" and "Proposal/1.5×M_{WstVVC}" hereinafter) are derived from the formula provided that these memory bandwidths are given as the arbitrary memory bandwidth M. Figure 8 shows the actually derived OBMC applicable conditions without exceeding 1.0 and 1.5 times M_{WstVVC}, i.e., $\hat{n}_{\mathrm{N}}^{W\times H}$ and $\hat{t}_{\mathrm{N}}^{W\times H}$ such that the constrains is satisfied in all sizes of the current block. In the proposed OBMC applicable conditions, 2, 4, 6, and 8-tap filters used in VVC can be selected for the $\hat{t}_{\mathrm{N}}^{W\times H}$ of the luma component. For the down-sampled chroma components, the half-tap filter of that for the luma component is utilized but the 2-tap filter is used when the luma filter is 2-tap. In both conditions, OBMC can be applied for the 4 × 8/8 × 4 uni-prediction current blocks having uni-prediction neighboring blocks, and for the other size current blocks having bi-prediction neighboring blocks, in which OBMC is prohibited in Lin2019.

The comparison analyses of $M_{\mathrm{Inter}}^{W\times H}$ for each current block size regarding the $\mathrm{RMC_{Uni}}$, $\mathrm{RMC_{Bi}}$, Chen2015, the Lin2019/2-tap, Lin2019, Proposal, and Proposal/1.5×M_{WstVVC}, are shown in Fig. 9. The Proposal can maximize $M_{\mathrm{Inter}}^{W\times H}$ (including M_{OBMC}) without exceeding the M_{WstVVC} through all size blocks as shown in Fig. 9. This is expected to maintain the potential coding gain of OBMC. The Proposal/1.5×M_{WstVVC} can also bring $M_{\mathrm{Inter}}^{W\times H}$ closer to 1.5 times M_{WstVVC} for small size blocks, but not for larger size blocks (e.g., 64×64, 64×128, and 128×128), which is the same level as the Proposal. This is expected to further preserve the potential coding gain in low-resolution sequences with many smaller size blocks. These expectations will be clarified in Sec. 5

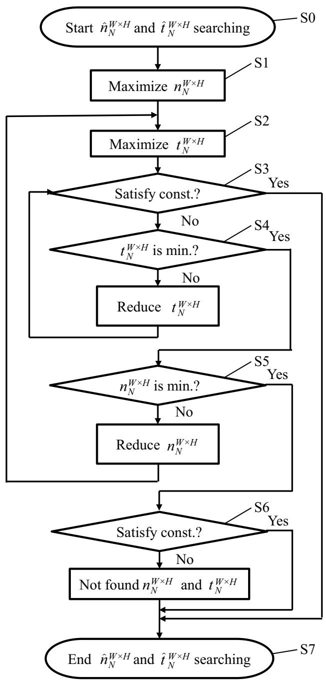


Fig. 7 The flowchart of the searching algorithm for the combination of $\hat{n}_{\mathrm{N}}^{W\times H}$ and $\hat{t}_{\mathrm{N}}^{W\times H}$ in the proposed method.

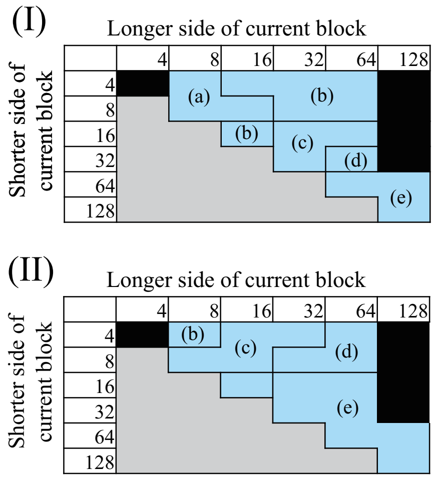


Fig. 8 The proposed OBMC applicable conditions with 1.0 and 1.5 times M_{WstVVC}: (I) = 1.0 times M_{WstVVC} and (II) = 1.5 times M_{WstVVC}. (a)–(e) indicate the combination of $\hat{n}_{\mathrm{N}}^{W\times H}$ and $\hat{t}_{\mathrm{N}}^{W\times H}$: (a) = (1, 2), (b) = (2, 2), (c) = (2, 4), (d) = (2, 6), and (e) = (2, 8). Black areas denote OBMC non-applicable area.

5. Experimental Results and Discussion

5.1 Test Conditions

1) Software Settings: The VVC reference software VTM version 10[21] (VTM-10) was used as the baseline software in our simulation experiments

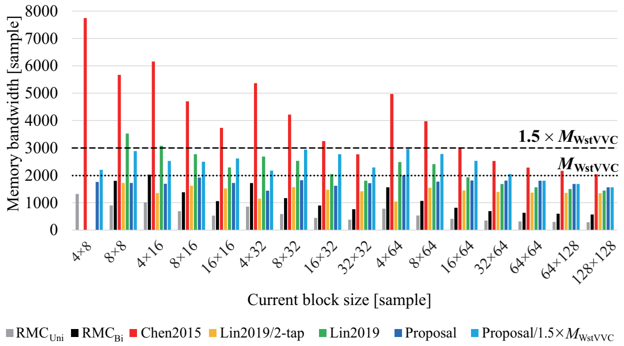


Fig. 9 Comparison analysis of the memory bandwidth versus the current block size for each method based on VVC. $\mathrm{RMC_{Uni}}$ and $\mathrm{RMC_{Bi}}$ denote RMC for uni-prediction and bi-prediction.

and the proposed method was implemented in the VTM-10. To verify the trade-offs between the coding performance and memory bandwidth, the simulations for a total of nine different methods, i.e., Chen2015, Chen2015/PhbtC, Chen2015/PhbtCN, Lin2019, Lin2019 with 2–8tap filters, the Proposal, and Proposal/1.5×M_{WstVVC} were conducted.

2) Encoder Configurations: The coding conditions were basically followed with the VTM Common Test Condition (CTC)[22]. The random access (RA) configuration, defined in the VTM CTC and utilized for general video transmission, was used since OBMC is an inter prediction tool. The test sequences from classes A to F, as listed in RA of VTM CTC, were used. They are categorized with different resolutions, frame rates, and video content as shown in Table 2. The partial test sequences, listed as Class A(A1/A2) and B in VTM CTC, were encoded by only the first group of pictures in these sequences to reduce the encoding runtime. For each test sequence, four quantization parameter (QP) values 22, 27, 32, and 37 defined in the VTM CTC were used to generate the different rate points.

3) Evaluation Metrics: The coding performance was evaluated by the BD-rate of the luma (BDY) and two chroma (BDU, BDV) components[19][20]. The BD-rate is the evaluation index used to quantify the difference of the generated bitrate for the identical level of peak signal-to-noise ratio (PSNR) between two coding methods. The negative BD-rate values indicate the coding gain with bitrate savings. In other words, the positive value is coding performance loss. The complexity was evaluated by the relative encoding time (EncT) and decoding time (DecT) of the two coding methods measured on a homogeneous cluster PC. Note that the results of Class D and F are not included in the overall results in accordance with the VTM CTC.

5. 2 Comparison of Overall Results

The overall results of each method compared to the VTM-10 in RA configuration, which is evaluated by BDY, BDU, BDV, and DecT, are listed in Table 3. In addition, the trade-off between BDY and the M_{Wst} of each method is shown in Fig. 10.

1) Coding performance: Table 3 shows that all the methods provide the coding gain against the VTM-10. These results prove the OBMC further improves the coding performance beyond VVC. Specifically, Chen2015 (-0.33 % gain) can be assumed to attain the full coding performance of OBMC. The Proposal achieves its comparable performance (-0.22 % gain) without exceeding the M_{WstVVC}, which is still better than Lin2019/2-tap (-0.12 % gain). Proposal/1.5×M_{WstVVC} brings the further coding performance (-0.25 % gain).

Figure 10 shows that the Proposal achieves a better trade-off than those of the Chen2015 and Lin2019 series in terms of BDY and the M_{Wst}. The reason that the Proposal achieves the best trade-off is discussed as follows. First, the coding gain of the Proposal is smaller than Chen2015/PhbtC but is larger than Chen2015/PhbtCN. Comparing the three methods regarding the OBMC applicable condition, the common difference of the Proposal from the other two methods is that the shorter-tap filter is utilized for the smaller size current blocks. Here, it is clear that the shorter-tap filters do not affect the coding performance from the comparison of the Lin2019 series. As for the Chen2015/PhbtC, the other difference except for the filter is the OBMC prohibition for partial smaller size current blocks having bi-prediction neighboring blocks as in Fig. 8. On the other hand, as for the Chen2015/PhbtCN, the difference except for the filter is the OBMC application for partial larger size current blocks having bi-prediction neighboring blocks as in Fig. 8, which contributes to the coding gain of the Proposal against Chen2015/PhbtCN. In addition, a comparison of Chen2015/PhbtCN and Lin2019 shows that the OBMC application for the 4×8 size current block having uni-prediction neighboring blocks as in Fig. 8 also contributes to the coding gain of the Proposal against Lin2019.

2) Complexity: Table 3 shows that all methods increase EncT and DecT against the VTM-10. These results prove the OBMC increases the encoding and decoding runtime beyond VVC. Especially, the EncT

Table 2 Details of the VTM CTC test sequences from class A to F categorized by resolutions, frame rates, and video content.

Class	Resolutions [pixels × lines]	Frame rates [fps]	Video content
A1	3840 × 2160	30–60	Camera-captured content (Natural scene)
A2	3840 × 2160	50–60	Camera-captured content (Natural scene)
B	1920 × 1080	50–60	Camera-captured content (Natural scene)
C	832 × 480	30–60	Camera-captured content (Natural scene)
D	416 × 240	30–60	Camera-captured content (Natural scene)
F	832 × 480 ∼ 1920 × 1080	20–60	Pure screen content (SCC) and mixed SCC and camera-captured content

Table 3 Overall results of the conventional and proposed methods over VTM-10 in RA configuration, which is evaluated by BDY [%], BDU [%], BDV [%] EncT [%], and DecT [%].

Method	BDY	BDU	BDV	EncT	DecT
Chen2015	-0.33	-1.04	-0.83	105	106
Chen2015/PhbtC	-0.27	-0.57	-0.57	104	103
Chen2015/PhbtCN	-0.14	-0.57	-0.28	105	104
Lin2019	-0.12	-0.43	-0.40	105	104
Lin2019/6-tap	-0.11	-0.24	-0.35	105	104
Lin2019/4-tap	-0.10	-0.43	-0.26	105	103
Lin2019/2-tap	-0.12	-0.43	-0.26	105	103
Proposal	-0.22	-0.54	-0.55	105	103
Proposal/1.5×M_{WstVVC}	-0.25	-0.64	-0.61	105	103

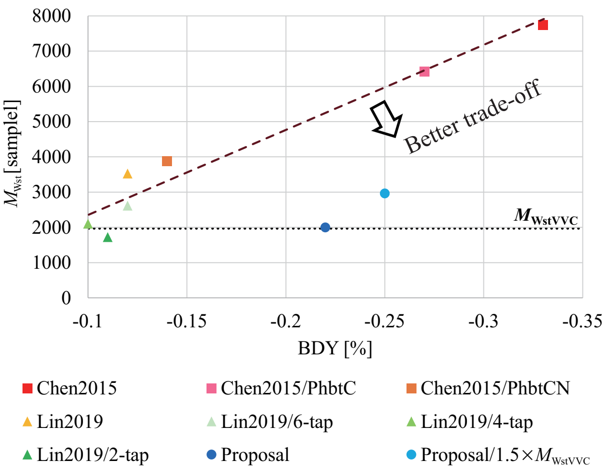


Fig. 10 The analysis of the trade-off between BDY [%] and M_{Wst} [sample] for each method.

increments of all the methods are not different, whereas the DecT increments of Chen2015 are larger than those of the other methods. This suggests that the OBMC application for the bi-prediction current block clearly further increases only DecT. Regarding EncT, this is because the number of RD cost calculations for OBMC signalling is the same between Chen2015 and Chen2015/PhbtC as described in Sec. 2. 3. The reason that DecT of Chen2015/PhbtCN is the same level as Chen2015/PhbtC even with the additional constraint of OBMC is that Chen2015/PhbtCN maintain the OBMC application rates for the uni-prediction current blocks having bi-prediction neighboring blocks, with the conversion from bi-prediction to uni-prediction as described in Sec. 2. 3.

5. 3 Comparison of Sequence-level Results

To analyze the coding gain by OBMC observed in the overall results, the sequence-level results of each method are compared with the VTM-10 as shown in Table 4.

1) Coding performance: First, from the comparison of the average gain for each class, the gain is larger for low-resolution test sequences in common with all methods. This is clear by comparing RaceHorsesC in Class C and RaceHorses in Class D which differ only in resolution. Since the selection rate of smaller size blocks is higher for low-resolution test sequences than for high-resolution test sequences, and the ratio of the OBMC applied area is higher for smaller size blocks than for larger size blocks, more coding gain by OBMC can be obtained. This characteristic is matched with the expectation as described in Sec 1 that Proposal/1.5x can provide more coding gain than the Proposal especially in low resolution sequences due to the extension of filter taps for smaller size blocks.

Second, larger coding gain can be observed in the test sequences with various and complicated motions in a picture such as Tango2, ParkRunning3, MarketPlace, and RaceHorses. Here, Tango2 and Racehorses have several moving objects with different motions, whereas ParkRunning3 and MarketPlace have camera shakes, which easily raise the difference of the motion vectors between the blocks. In contrast, regarding the test sequences without these motions such as BQTerrace and BQSquare, a significant small coding gain or even coding loss can be observed in all the methods except for Chen2015. These tendencies are consistent with the original expected effects of OBMC as described in Sec 1. Moreover, the coding gain of Chen 2015 in BQTerrace and BQSquare suggests that the OBMC application for the bi-prediction current block can further improve the coding performance of these sequences where the bi-prediction is originally effective. Finally, the coding losses of the pure SCC such as SlideEditing and SlideShow can be observed in common with all methods. This is because OBMC overshoots the block boundaries including shape edges in SCC, and it

Table 4 Sequence-level results of Chen2015, Lin2019, the Proposal, Proposal/1.5× M_{WstVVC} over VTM-10 in RA configuration, which is evaluated by BDY [%], BDU [%], BDV [%], EncT [%], and DecT [%].

	Chen2015			Chen2015/PhbtC			Chen2015/PhbtCN			Lin2019			Proposal			Proposal/1.5× M_{WstVVC}		
Sequence	BDY	EncT	DecT	BDY	EncT	DecT	BDY	EncT	DecT	BDY	EncT	DecT	BDY	EncT	DecT	BDY	EncT	DecT
Tango2	-0.31	105	104	-0.29	104	100	-0.10	105	101	-0.18	104	102	-0.27	104	101	-0.30	104	103
FoodMarket4	0.06	104	104	-0.06	103	102	0.05	104	102	0.03	104	102	-0.07	103	102	-0.01	103	101
CampfireParty2	-0.20	104	103	-0.19	104	102	-0.13	105	102	-0.15	104	102	-0.13	104	102	-0.18	104	101
Average Class A1	**-0.15**	**105**	**104**	**-0.18**	**104**	**102**	**-0.06**	**104**	**102**	**-0.10**	**104**	**102**	**-0.16**	**104**	**102**	**-0.16**	**104**	**102**
CatRobot1	-0.33	106	105	-0.35	104	102	-0.14	105	102	-0.23	105	102	-0.16	105	102	-0.41	104	102
DaylightRoad2	-0.09	106	106	-0.29	105	103	-0.05	105	104	-0.05	105	104	-0.23	105	103	-0.26	104	104
ParkRunning3	-0.51	106	107	-0.37	104	103	-0.27	105	104	-0.19	105	103	-0.35	104	103	-0.37	105	105
Average Class A2	**-0.31**	**106**	**106**	**-0.34**	**105**	**102**	**-0.15**	**105**	**103**	**-0.16**	**105**	**103**	**-0.25**	**105**	**103**	**-0.34**	**104**	**103**
MarketPlace	-0.47	106	108	-0.35	105	104	-0.18	106	106	-0.21	105	106	-0.20	105	105	-0.32	105	103
RitualDance	-0.18	105	105	-0.19	104	103	-0.17	105	103	0.04	105	107	-0.13	104	104	-0.15	104	103
Cactus	-0.04	106	105	-0.11	105	104	0.07	105	103	-0.10	105	103	-0.15	105	104	-0.14	105	103
BasketballDrive	-0.16	105	107	-0.03	104	102	0.00	105	103	0.02	105	104	-0.19	104	102	-0.02	104	99
BQTerrace	-0.43	106	105	-0.11	104	104	-0.03	106	103	0.10	105	102	0.01	105	103	-0.10	105	102
Average Class B	**-0.25**	**106**	**106**	**-0.16**	**105**	**103**	**-0.06**	**105**	**104**	**-0.03**	**105**	**105**	**-0.13**	**105**	**104**	**-0.14**	**105**	**104**
BasketballDrill	-0.65	105	106	-0.41	104	104	-0.27	106	104	-0.24	105	102	-0.35	105	103	-0.40	105	106
BQMall	-0.46	106	107	-0.36	105	105	-0.25	106	105	-0.22	105	105	-0.35	105	105	-0.36	106	105
PartyScene	-0.60	106	108	-0.45	105	106	-0.33	106	105	-0.20	105	104	-0.29	105	105	-0.37	106	106
RaceHorses	-0.55	106	110	-0.55	105	106	-0.31	105	107	-0.23	105	105	-0.42	105	106	-0.42	105	106
Average Class C	**-0.57**	**106**	**108**	**-0.44**	**105**	**105**	**-0.29**	**106**	**105**	**-0.22**	**105**	**104**	**-0.35**	**105**	**105**	**-0.39**	**105**	**104**
BasketballPass	-0.37	106	109	-0.31	105	106	-0.17	105	106	-0.03	105	105	-0.17	105	106	-0.26	105	108
BQSquare	-0.34	105	105	-0.19	105	103	-0.09	106	103	-0.03	104	101	0.06	105	103	-0.11	104	103
BlowingBubbles	-0.63	106	107	-0.57	105	105	-0.40	106	105	-0.25	105	104	-0.40	105	105	-0.44	105	106
RaceHorsesC	-0.67	106	111	-0.63	105	108	-0.55	106	107	-0.34	106	107	-0.53	106	108	-0.59	105	109
Average Class D	**-0.50**	**106**	**108**	**-0.42**	**105**	**105**	**-0.30**	**106**	**105**	**-0.16**	**105**	**104**	**-0.26**	**105**	**105**	**-0.35**	**105**	**104**
BasketballDrillText	-0.42	104	108	-0.36	103	106	-0.23	104	105	-0.15	104	105	-0.32	104	105	-0.36	104	108
ArenaOfValor	-0.15	104	108	-0.29	104	105	-0.28	105	104	-0.06	104	105	-0.27	104	105	-0.26	104	104
SlideEditing	0.18	101	101	0.13	101	102	0.13	102	102	0.10	102	102	0.13	101	101	0.15	101	101
SlideShow	0.71	103	102	0.51	103	102	0.54	103	103	0.54	103	102	0.53	103	102	0.55	103	103
Average Class F	**0.08**	**103**	**105**	**0.00**	**103**	**104**	**0.04**	**103**	**103**	**0.11**	**103**	**103**	**0.02**	**103**	**103**	**0.02**	**105**	**104**
Overall	**-0.33**	**105**	**106**	**-0.27**	**104**	**103**	**-0.14**	**105**	**104**	**-0.12**	**105**	**104**	**-0.22**	**105**	**103**	**-0.25**	**105**	**103**

consequently degrades the objective qualities. We can avoid it with the OBMC disabling flag for the overall sequence, for example.

2) Complexity: The same tendency can be observed in DecT as the coding performance, i.e., the low-resolution test sequences have larger DecT. It demonstrates that the high OBMC applied sample rates of these test sequences increases the DecT. In contrast, EncT is at the same level for high and low resolution sequences.

5. 4 Picture-level Analysis

1) Rate-distortion curve characteristics: To analyze the coding gains and losses as discussed in Sec 5. 3, the rate-distortion curve of each method for RaceHorsesC and BQTerrace is compared as shown in Fig. 11. We selected RaceHorsesC and BQTerrace on behalf of the test sequences since they have the clearest tendencies as described in Sec. 5. 3.

First, the comparison of each method for RaceHorsesC as shown in Fig. 11(a)–(c) shows that the coding gains of all methods against VTM-10 come from the bitrate savings, not from the improvement of the objective quality. The OBMC removes the discontinuity of the block boundary and decreases the residuals, and consequently reduces the bitrate. The larger bitrate savings can be observed at QP=22 compared to those at QP=37. This is because that small quantization step of the small QP raises the selection rates of smaller size blocks, and provides the coding gain as well as the effects seen in low-resolution test sequences. In addition, no difference among the Lin2019 series with different t_{N} corresponds to the discussion in Sec. 5. 2.

Second, the comparison of each method for BQTerrace as shown in Fig. 11(d)–(e) shows that the coding gain of Chen2015 against VTM-10 also comes from the bitrate savings. In contrast, the bitrates for the other method are higher than VTM-10 at QP=37, whereas the PSNR for some of those methods is smaller than VTM-10. The signaling overhead for OBMC and the degradation of the objective quality by OBMC prohibition for the bi-prediction current block seem to affect them, respectively.

2) OBMC applied sample rates: To reveal the discussion so far, we compare Chen2015, Lin2019, and the Proposal by two types of ratio of the OBMC applied samples for each current block size as shown in Fig. 12. One is the ratio of OBMC applied samples to the inter frame samples $R_{\mathrm{OBMC/InterFrame}}$, which can compare the relative number of OBMC applied samples among the three methods, and can evaluate their effects of OBMC. The other is the ratio of OBMC applied samples to inter block samples $R_{\mathrm{OBMC/InterBlock}}$, which can estimate the effect of OBMC depending on the current block sizes. We selected the same test sequences

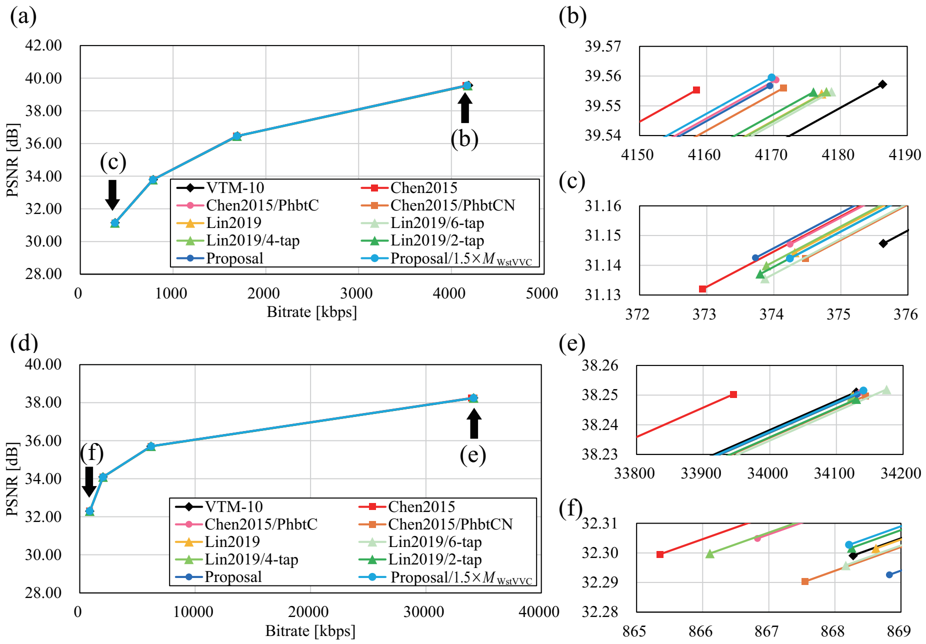


Fig. 11 Rate distortion curves of the VTM-10 and each method for RaceHorsesC and BQTerrace. (a)–(c) RaceHorsesC, (d)–(f) BQTerrace.

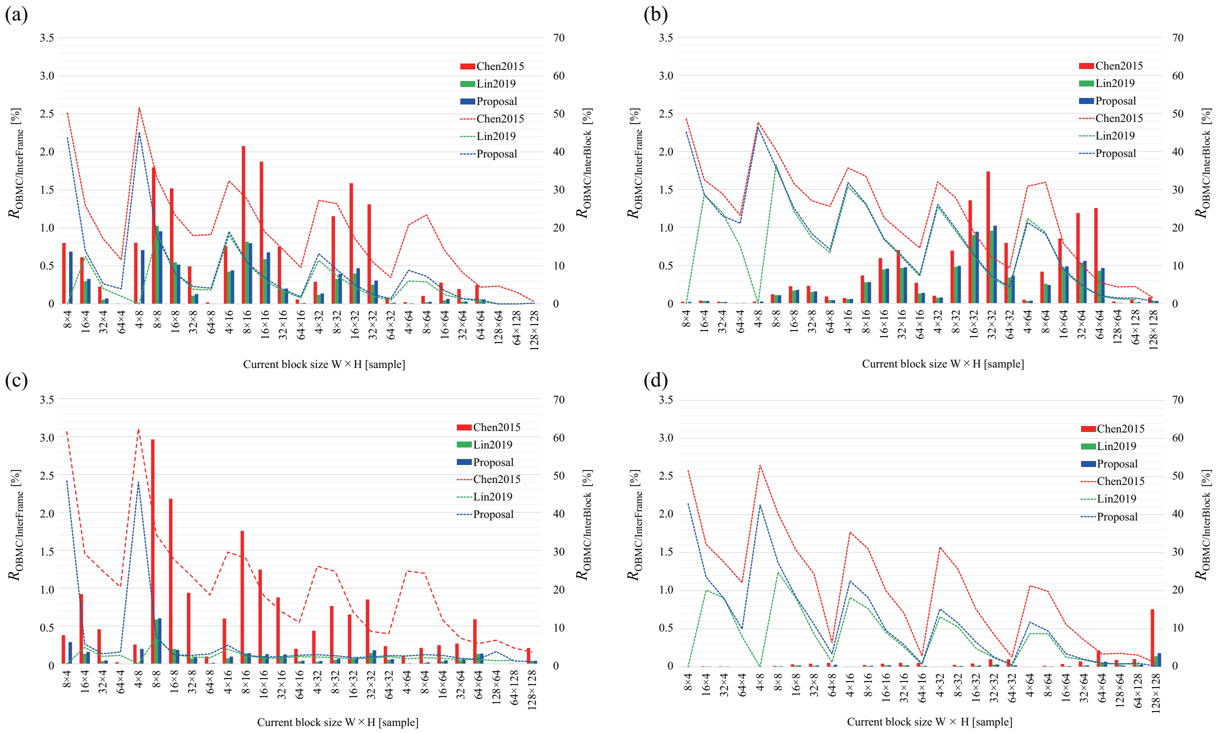


Fig. 12 Comparison analysis of Chen2015, Lin2019, and the Proposal regarding the current block sizes versus a ratio of the OBMC applied samples to the inter frame samples $R_{\mathrm{OBMC/InterFrame}}$ (shown as bar and left axis) and a ratio of those to the inter block samples $R_{\mathrm{OBMC/InterBlock}}$ (shown as dotted plot and right axis). The number of those samples is the total value generated by the RaceHorsesC/BQTerrace of QP=22/37. (a) RaceHorsesC of QP=22, (b) RaceHorsesC of QP=37, (c) BQTerrace of QP=22, and (d) BQTerrace of QP=37.

and QPs as Fig. 11.

Commonly in Fig. 12(a)–(d), the larger current block size is, the higher $R_{\mathrm{OBMC/InterBlock}}$ is, which is consistent with the discussion so far. Especially for RaceHorsesC, clear existences of OBMC applied samples can be observed in all methods at both QP=22 and QP=37. In all methods, the peaks of $R_{\mathrm{OBMC/InterBlock}}$ shift from smaller size blocks to larger size blocks at QP=22 versus QP=37, but maintain the total $R_{\mathrm{OBMC/InterBlock}}$, which is the evidence of the coding gain by OBMC in RaceHorsesC. The number of Chen2015 is significantly higher than those of Lin2019 and the Proposal, while the number of the other two methods is not so different except for $4\times8/8\times4$ current blocks. This corresponds to the reasons which provide the different coding gains for these three methods observed in RaceHorsesC of Table 4 and described in Sec. 5.3.

As for BQTerrace, the same level of $R_{\mathrm{OBMC/InterFrame}}$ as that in RaceHorsesC can be observed in Chen2015 at QP=22, while the smaller $R_{\mathrm{OBMC/InterFrame}}$ as those in RaceHorsesC can be seen in the other two methods. This is consistent with the discussion regarding the OBMC effectivity for the bi-prediction current block in this test sequence as described in Sec. 5.3. At QP=37, most of the OBMC applied samples except for 128×128 samples are dispensed, which causes no coding in the Proposal or the coding loss shown in Table 4 and described in Sec. 5.3.

6. Conclusion

In this paper, we proposed the memory bandwidth constrained OBMC method. The proposed method is generalized as the objective function with the constraint that maximizes the coding performance with the number of motion vectors and interpolation filter taps of the neighboring blocks depending on the current block sizes. The constraint is not to exceed an arbitrary memory bandwidth and we set the worst-case upper-limit of the memory bandwidth of VVC as an example. Simulation results showed that the proposed method achieves an additional coding gain (-0.22 %) over VVC reference software. This gain is comparable to the full performance of bi-prediction based OBMC (-0.33 %) which requires 3.8 times the memory bandwidth of VVC, and is still better than that of the conventional uni-prediction based OBMC (-0.12 %).

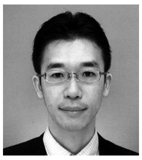
Acknowledgment This work was supported by Ministry of Internal Affairs and Communications (MIC) of Japan (Grant no. JPJ000595).

References

1) Cisco Systems. Cisco Annual Internet Report (2018-2023) White Paper. (Mar. 2020)
2) M. Budagavi and M. Zhou, "Video Coding Using Compressed Reference Frames," 2008 IEEE International Conference on Acoustics, Speech and Signal Processing pp. 1165-1168 (Mar. 2008)
3) M. E. Sinangil, A. P. Chandrakasan, V. Sze and M. Zhou, "Memory Cost vs. Coding Efficiency Trade-offs for HEVC Motion Estimation Engine," 2012 19th IEEE International Conference on Image Processing, pp. 1533-1536 (Sep. 2012)
4) R. Hashimoto and S. Mochizuki, "AHG5: How to Use the Software to Evaluate Memory Bandwidth," in ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11, San Diego, JVET-J0090 (Apr. 2018)
5) Recommendation ITU-T H.265, "High Efficiency Video Coding," in Edition 1.0 (Apr. 2013)
6) Recommendation ITU-T H.266, "Versatile Video Coding," in Edition 1.0 (Aug. 2020)
7) H. Watanabe and S. Singhal "Windowed Motion Compensation", Proc. SPIE 1605, Visual Communications and Image Processing '91: Visual Communication (Nov. 1991)
8) S. Nogaki and M. Ohta, "An Overlapped Block Motion Compensation for High Quality Motion Picture Coding," Proc. 1992 IEEE International Symposium on Circuits and Systems, San Diego, CA, USA, pp.184-187 Vol.1 (May 1992)
9) P. Lai, S. Liu and S. Lei, "Combined Temporal and Inter-layer Prediction for Scalable Video Coding using HEVC," 2013 Picture Coding Symposium, pp. 117-120 (2013)
10) Y.-W. Chen and X. Wang, "AHG5: Reducing VVC Worst-case Memory Bandwidth by Restricting Bi-directional 4x4 Inter CUs/Sub-blocks," in ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11, Macao, JVET-L0104 (Oct. 2018)
11) Z.-Y. Lin, T.-D. Chuang, C.-Y. Chen, C.-W. Hsu, Z.-Y. Lin, Y.-C. Lin, Y.-W. Huang, S.-M. Lei, X. Xiu, and Y. He, "CE10.2.1: Uni-prediction-based CU-boundary-only OBMC," in ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11, Marrakech, JVET-M0178 (Jan. 2019)
12) G. Sullivan and J.-R. Ohm, "Meeting Report of the 13th Meeting of the Joint Video Experts Team (JVET), Marrakech, MA, 9–18 January 2019," in ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11, Marrakech, JVET-M1000 (Jan. 2019)
13) H. Gao, X. Chen, S. Esenlik, J. Chen and E. Steinbach, "Decoder-side Motion Vector Refinement in VVC: Algorithm and Hardware Implementation Considerations," in IEEE Transactions on Circuits and Systems for Video Technology, Vol. 31, No. 8, pp. 3197-3211 (Aug. 2021)
14) Y. Chen and D. Mukherjee, "Variable Block-size Overlapped Block Motion Compensation in the Next Generation Open-source Video Codec," 2017 IEEE International Conference on Image Processing, pp. 938-942 (Sep. 2017)
15) Y. Liao, A. Leontaris, and A. M. Tourapis, "A Low Complexity Architecture for Video Coding with Overlapped Block Motion Compensation," 2010 IEEE International Conference on Image Processing, pp. 2041-2044 (Dec. 2010)
16) J. Chen, Y. Chen, M. Karczewicz, X. Li, H. Liu, L. Zhang, and X. Zhao, "Coding Tools Investigation for Next Generation Video Coding based on HEVC," Proc. SPIE, Vol.9599 (Sep. 2015)
17) H. Yang et al., "Subblock-Based Motion Derivation and Inter Prediction Refinement in the Versatile Video Coding Standard," in IEEE Transactions on Circuits and Systems for Video Technology, Vol. 31, No. 10, pp. 3862-3877 (Oct. 2021)
18) Y. Kidani, K. Kawamura, K. Unno and S. Naito, "Block-size Dependent Overlapped Block Motion Compensation," 2020 IEEE International Conference on Image Processing, pp. 1191-1195 (Sep. 2020)
19) G. Bjøntegaard, "Calculation of Average PSNR Differences between RD-curves," in ITU-T SG 16 WP 3, Texas, VCEG-M33 (Apr. 2001)
20) K. Andersson et al., "Summary Information on BD-rate Experiment Evaluation Practices,"in Joint Video Experts Team (JVET) of ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11, Brussels, JVET-Q2016 (Jan. 2020)
21) J. Chen, Y. Ye, and S. Kim, "Algorithm Description for Versatile Video Coding and Test Model 10 (VTM 10)," in ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11, by teleconference, JVET-S2002 (Jun. 2020)
22) F. Bossen, J. Boyce, X. Li, V. Seregin, and K. Sühring, "VTM Common Test Conditions and Software Reference Configurations for SDR Video," in Joint Video Experts Team (JVET) of ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29, by teleconference, JVET-T2010 (Oct. 2020)

Yoshitaka Kidani received B.E. and M.E. degrees in Engineering from Nagoya University, in 2010 and 2012. He joined KDDI Corporation in 2012. He is currently a research engineer at KDDI Research, Inc. He has been involved with the development of VVC standards under JVET. His research interests include video coding and multimedia distribution. He is a member of IEICE.

Kyohei Unno received B.E., M.E., and Ph.D. degrees in Engineering from Tokyo University of Science, in 2009, 2011, and 2021. He joined KDDI Research, Inc. in 2018. He is currently a research engineer in KDDI Research, Inc. He has been involved with the development of VVC and G-PCC standards under JVET and MPEG. His research interests include video coding and point cloud coding. He is a member of IEICE.



Kei Kawamura received B.E., M.Sc., and Ph.D. degrees in Global Information and Telecommunication Studies from Waseda University, Japan, in 2004, 2005, and 2013, respectively. He joined KDDI in 2010. He has been involved with the development of HEVC and VVC standards under JCT-VC and JVET. He is currently engaged in the research and development of a video coding system at KDDI Research, Inc. His research interests include image and video processing, video coding, and multimedia distribution. He is a member of a steering committee of PCSJ/IMPS. He is a member of IEEE and IEICE.

Hiroshi Watanabe received B.E., M.E. and Ph.D. degrees in Engineering from Hokkaido University in 1980, 1982, and 1985, respectively.He joined Nippon Telegraph and Telephone Corporation (NTT) in 1985, and was engaged in research and development of image and video coding systems at NTT Human Interface Labs. and NTT Cyber Space Labs. until 2000. He has also been involved with the development of JPEG, MPEG standards under JTC 1/SC 29. He is currently a professor at the Department of Communications and Computer Engineering, School of Fundamental Science and Engineering, Waseda University. His research interests include object recognition, deep learning, image processing, video coding, and multimedia distribution. He is a member of IEEE, IPSJ, IEICE, and IIEEJ.