Motion Vector Coding and Block Merging in the Versatile Video Coding Standard

Wei-Jung Chien, *Member, IEEE*, Li Zhang, *Senior Member, IEEE*, Martin Winken, Xiang Li, *Senior Member, IEEE*, Ru-Ling Liao, Han Gao, *Graduate Student Member, IEEE*, Chih-Wei Hsu, Hongbin Liu, and Chun-Chi Chen

(Invited Paper)

***Abstract*— This paper overviews the motion vector coding and block merging techniques in the Versatile Video Coding (VVC) standard developed by the Joint Video Experts Team (JVET). In general, inter-prediction techniques in VVC can be classified into two major groups: "whole block-based inter prediction" and "subblock-based inter prediction". In this paper, we focus on techniques for whole block-based inter prediction. As in its predecessor, High Efficiency Video Coding (HEVC), whole block-based inter prediction in VVC is represented by adaptive motion vector prediction (AMVP) mode or merge mode. Newly introduced features purely for AMVP mode include symmetric motion vector difference and adaptive motion vector resolution. The features purely for merge mode include pairwise average merge, merge with motion vector difference, combined inter-intra prediction and geometric partitioning mode. Coding tools such as history-based motion vector prediction and bidirectional prediction with coding unit weights can be applied on both AMVP mode and merge mode. This paper discusses the design principles and the implementation of the new inter-prediction methods. Using objective metrics, simulation results show that the methods overviewed in the paper can jointly achieve 6.2% and 4.7% BD-rate savings on average with the random access and low-delay configurations, respectively. Significant subjective picture quality improvements of some tools are also reported when comparing the resulting pictures at same bitrates.**

***Index Terms*— Block merging, HEVC, motion compensation, motion vector coding, versatile video coding (VVC).**

I. Introduction

THE Versatile Video Coding (VVC a.k.a. ITU-T Rec. H.266 and ISO/IEC 23090-3) is the latest international video-compression standard jointly finalized by ITU-T Video Coding Experts Group (VCEG) and ISO/IEC Motion Picture Expert Group (MPEG) in July 2020 [1], [2]. The project of this new standard was announced by the so called "Joint Video Exploration Team (JVET)" of VCEG and MPEG in October 2017 with the joint call for proposal of video compression with capability beyond High Efficiency Video Coding (HEVC) [3] and then officially launched in April 2018. At the same time, JVET has been renamed to "Joint Video Experts Team". The main goal of VVC is to address two aspects of industry needs for a future video coding standard: (a) higher coding efficiency than HEVC with over 50% bitrate reduction while keeping the same subjective quality for SDR/HDR contents with picture size at least covering from VGA to 8K × 4K and (b) broad versatility that supports efficient compression of various types of video contents and applications, such as screen contents, adaptive resolution change, omni-directional/360° videos [4].

Although the VVC standard inherits the framework of block-based hybrid coding, similar to HEVC, it adopts several highly adaptive and sophisticated coding tools. In general, VVC follows a multi-type tree structure (i.e., quadripartite, binary and/or ternary tree) to split a picture into a variety of block shapes (i.e., square, or non-square). Each block is a basic unit for signaling prediction information. Then, intra prediction and/or inter prediction operates on a block-by-block or subblock-by-subblock basis [5] within the basic unit, followed by transform and quantization processes with switchable bases for residual coding, a chain of in-loop filters (i.e., deblocking, sample adaptive offset, adaptive loop filtering [6]) for subjective quality improvement and syntax coding/parsing for transmission.

In a video signal, high temporal redundancy exists between sequential pictures. Therefore, inter prediction, targeting at reducing the temporal redundancy, makes a major contribution in the video compression capability and plays a key role in the hybrid video coding scheme. In VVC, a lot of novel coding tools are developed to further improve inter prediction. In general, those tools can be classified into two major groups, depending on whether the whole block share the same set of motion information, i.e., "whole block-based inter prediction" wherein only one set of motion information is utilized and "subblock-based inter prediction" wherein each sub-block could have its own set of motion information.

This paper covers both algorithm descriptions and performance analysis of whole block-based inter-prediction coding tools while the subblock-base inter prediction is overviewed in another overview paper of this special issue [7]. Basically,

Manuscript received August 24, 2020; revised March 24, 2021 and June 16, 2021; accepted July 17, 2021. Date of publication July 30, 2021; date of current version October 4, 2021. This article was recommended by Associate Editor J.-R. Ohm. *(Wei-Jung Chien and Li Zhang are co-first authors.) (Corresponding author: Li Zhang.)*

Wei-Jung Chien and Chun-Chi Chen are with Qualcomm Inc., San Diego, CA 92121 USA.

Li Zhang is with Bytedance Inc., San Diego, CA 92122 USA (e-mail: lizhang.idm@bytedance.com).

Martin Winken is with Fraunhofer HHI, 10587 Berlin, Germany.

Xiang Li is with Tencent, Palo Alto, CA 94306 USA.

Ru-Ling Liao is with Alibaba, Beijing 100102, China.

Han Gao is with Huawei Technologies, 80992 Munich, Germany.

Chih-Wei Hsu is with MediaTek Inc., Hsinchu 30078, Taiwan.

Hongbin Liu is with Bytedance Inc., Beijing 100190, China.

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TCSVT.2021.3101212.

Digital Object Identifier 10.1109/TCSVT.2021.3101212

the whole block-based inter-prediction coding tools include the extended adaptive motion vector prediction (AMVP) mode and block merging which are employed in the HEVC inter prediction scheme, and multiple other coding tools introduced in the VVC standardization work. Therefore, following the inter prediction scheme, the coding tools are categorized into 3 topics, AMVP, merge, and others, as follows:

- *Motion vector (MV) coding:* **MV predictor (MVP)** candidate list with 2 candidates generated based on spatial/temporal MV predictors or MVs from **history-based motion vector prediction**(HMVP) tables [8], [9]; **symmetric MV difference**(SMVD) that signals a pair of symmetric bi-prediction MV differences (MVDs) with slice-level indicated reference pictures [10], [11]; **adaptive MV resolution** (AMVR) for MV predictors and MVDs at 1) quarter- to 4- luma samples or 2) one-16th- to one- luma sample precision depending on selected motion models, i.e., 1) for the translational motion model and 2) for the affine motion model [13]–[18];
- *Block merging:* **block merging candidate list** with at most 6 candidates generated based on spatial/temporal, candidates from HMVP tables and a synthetic pair-wise motion vector predictor; **geometric partitioning mode** (GPM) that geometrically splits a block at an angle of power-of-two tangent with a boundary-shifting offset [22]–[24]; **merge mode with MV difference**(MMVD) that allows adding power-of-2 1-D offsets to either horizontal or vertical components of a selected merge candidate [22], [23]; **combined inter-intra prediction** (CIIP) to generate a prediction block with weighted combination of a planar intra predictor and the motion-compensated temporal predictor of a selected merge candidate [25], [26]; **merge estimation region** (MER) to allow independent/parallel derivation of MV prediction list and merge candidate list of coding units (CUs) inside the region [27];
- *Others:* **bidirectional prediction with coding unit weights** (BCW) to introduce non-equal weights at CU level for bi-prediction [28]–[30]; **motion vector compression and range** to encode motion fields on reference pictures by using a 10-bit mantissa-exponent representation at every 8×8 grid [31].

These coding tools introduced to inter prediction offer more encoding options for VVC to efficiently represent the motion fields of coded blocks with complex object motion. For example, the VVC standard offers quite a few motion models, including translational model, 4- and 6-parameter affine model, to accommodate more video content types with complex structures and ensures coding efficiency. The effectiveness of each individual coding tool reported during the development of VVC has justified the increase of encoding/decoding complexity [43].

This paper is organized as follows. Sections II presents the motion data coding in the HEVC standard, including the AMVP mode and merge mode. Multiple aspects of inter prediction in the VVC standard are provided in Section III. Sections IV, V and VI give the detailed description of individual coding tools and the interactions among tools. Tool-by-tool testing results are given in Section VII to explore the coding performance and analyze the implementation complexity, and Section VIII concludes this paper.

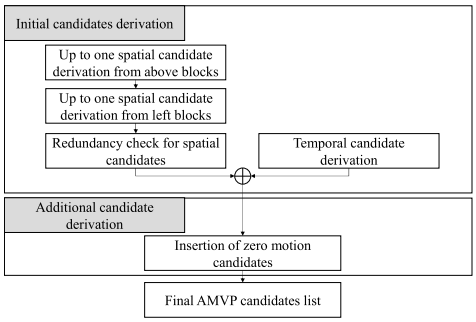


Fig. 1. AMVP candidate list construction process.

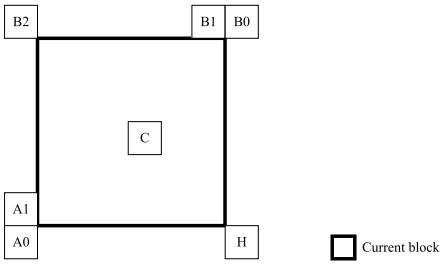


Fig. 2. Five neighboring spatial locations (above blocks: A0, A1; left blocks: B0, B1 and B2) and locations of collocated blocks for TMVP (H and C) of the current block.

II. Motion Data Coding in HEVC

In HEVC, inter prediction is represented by two modes: the AMVP mode and merge mode, wherein reference picture indices and MVDs are signaled in the former mode but not signaled in the latter one. Skip mode is a special merge mode, in which residuals are inferred to be zero and thus not signaled. In this section, we briefly review the two modes in HEVC.

AMVP mode origins from MV competition [32] wherein one of the best MVPs could be selected according to rate-distortion cost. For the AMVP mode in HEVC, motion vector predictions are used to exploit spatio-temporal correlations of MVs among prediction units (PUs). The encoder can select the best MVP from an MVP candidate list and transmit the corresponding index together with the reference picture index and MVD. The MVP candidate list for each reference picture list with up to two candidates is constructed, following the flow depicted in Fig. 1. More specifically, the MVPs from spatial neighboring PUs, left and above to the current PU, an MVP derived from temporal motion vector prediction (TMVP) and zero MVPs are added to fulfill the candidate list in order. The locations of five neighboring PUs are denoted respectively as A0, A1, B0, B1 and B2 in Fig. 2. The left spatial neighboring MVP candidate can be derived from A0 and A1 when at least one of them is inter predicted, and the above spatial neighboring MVP candidate can be derived from B0, B1 and B2 when at least one of them is inter predicted. A scaled MV, according to the temporal distances between the reference picture associated with the left MVP and the current reference picture, will be output as the left MVP if no MV refers to

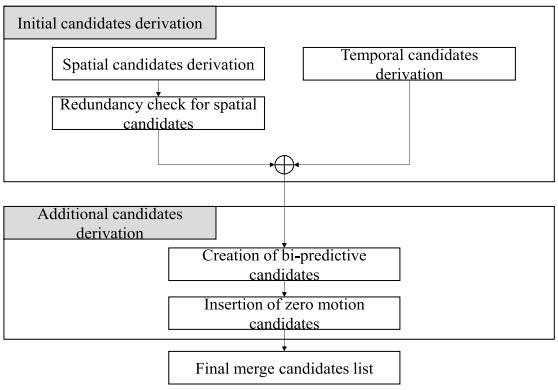


Fig. 3. Construction of a merge candidate list.

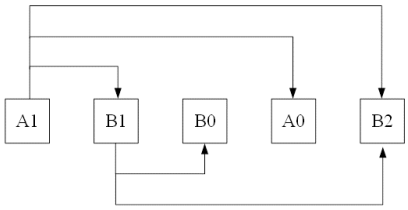


Fig. 4. Comparison pairs for spatial merge candidates.

the target reference picture in block A0 and A1. Similarly, a scaled MV will be output as the above MVP if no MV refers to the target reference picture in block B0, B1 and B2. The above spatial neighboring MVP candidate is discarded if it is identical to the left neighboring MVP candidate. The TMVP candidate is derived by scaling an MV stored at location H or C as shown in Fig. 2 in the collocated picture to the target reference picture. Location H is checked first. If no MV is available at location H, location C is checked.

With the merge mode in HEVC, motion information of the current PU can be directly inherited from spatial or temporal neighboring blocks [33]. A merge candidate list with five candidates is constructed as demonstrated in Fig. 3. Like in the AMVP mode, the encoder selects the best merge candidate from the candidate list and transmits the corresponding index, but without any reference index or MVDs.

To derive spatial merge candidates, a maximum of four merge candidates are selected among candidates located in the positions depicted in Fig. 2. The order of derivation is A1, B1, B0, A0 and B2. Position B2 is considered only when any PU of position A1, B1, B0 and A0 is not available (e.g. because it belongs to another slice or tile) or is not inter predicted. After the candidate at position A1 is added, the addition of the remaining candidates is subject to a redundancy check which ensures that candidates with same motion information are excluded from the list to improve the coding efficiency. To reduce computational complexity, only the pairs linked with an arrow in Fig. 4 are compared and a candidate is added to the list only if it passes the redundancy check. In HEVC, a CU may be partitioned into PUs, which may bring redundancy with the merge mode. Fig. 5 depicts the "second PU" partitioned from a CU by N × 2N and 2N × N pattern, respectively. When the second PU is partitioned from a CU by N × 2N, candidate at position A1 is not considered for list construction. In fact, by choosing this candidate, two PUs

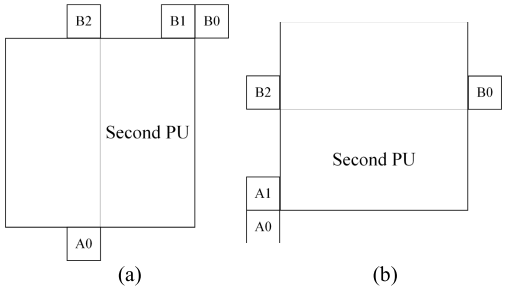


Fig. 5. The second PU partitioned by (a) N × 2N and (b) 2N × N.

will share the same motion information, which is redundant to the case when there is just only one PU in the CU. Similarly, position B1 is not considered when the second PU is partitioned from a CU by 2N × N.

In the derivation of the temporal merge candidate, the TMVP candidate is derived from MVs stored at location H or C as shown in Fig. 2 in the collocated picture, similar to the TMVP candidate for AMVP mode. For a TMVP candidate in the merge candidate list, the MVs will be scaled to the reference picture with reference index 0 in the corresponding reference picture list.

Beside spatio-temporal merge candidates, there are two additional types of merge candidates: combined bi-predictive merge candidate and zero motion candidate with (0, 0) motion vector. Combined bi-predictive merge candidates are generated by utilizing spatio-temporal merge candidates for B-slice only. The combined bi-predictive candidates are generated by combining a first MV referring to reference list 0 of a first merge candidate, and a second MV referring to reference list 1 of a second merge candidate where the first and second merge candidates are selected from available merge candidates in the merge candidate list according to a pre-defined order. The two MVs will form a new bi-predictive candidate. If the merge candidate list is not fulfilled, zero merge candidates will be appended to the list to fill it up.

III. Overview of Motion Vector Coding and Block Merging in VVC

As aforementioned, VVC supports both the whole block-based and subblock-based inter prediction tools. The whole block-based inter-prediction has been widely used for decades in former video coding standards such as HEVC. With whole block-based inter prediction, a set of motion information, which comprises MVs and reference pictures, is assigned to a block, and the motion compensation (MC) is performed on the whole block with the set of motion information. Unlike whole block-based inter prediction, subblock-based inter prediction first divides a block into subblocks, e.g. 4 × 4 or 8 × 8 subblocks, then an individual set of motion information is assigned to each subblock. Fig. 6 shows a skeleton diagram of all inter-prediction techniques in VVC.

In the following sub-sections, the newly employed whole block-based inter coding tools in VVC are provided.

A. Motion Vector Predictor From HMVP Tables

In HEVC, there are two types of MVPs, i.e., spatial MVP and temporal MVP which utilize the motion information from

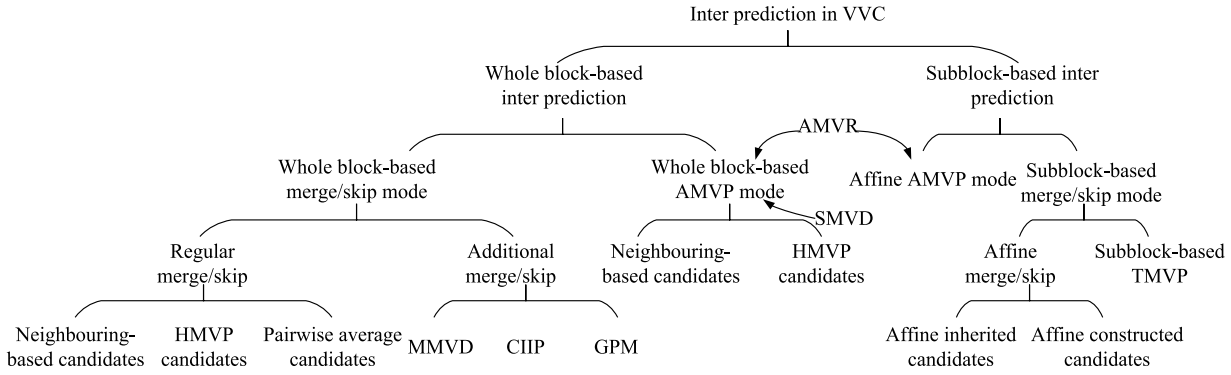


Fig. 6. Inter-prediction techniques in VVC.

spatially adjacent or temporal blocks. While in VVC, a new type of MVP, i.e., HMVP is introduced. The basic idea of HMVP is to further use previously coded MV as MVP which are associated with adjacent or non-adjacent blocks relative to current block. In order to track available HMVP candidates, a table of HMVP candidates is maintained at both encoder and decoder and updated on the fly. Whenever a new CTU row starts, the table is reset to ease parallel coding. There are up to five candidates in the HMVP table. After coding one inter predicted block which is not in sub-block mode (including affine mode) or GPM, the table is selectively updated by appending the associated motion information to the end of the table as a new HMVP candidate. A restricted first-in-first-out (FIFO) rule is applied to manage the table wherein the redundant candidate in the HMVP table is firstly removed instead of the first one. With HMVP, the motion information of previously coded blocks can be utilized for more efficient motion vector prediction, even if the coded blocks are not spatially adjacent to the current block.

The HMVP candidates can be added to the AMVP candidate list as well as the merge candidate list.

B. Motion Vector Coding

In addition to the introduction of HMVP candidate added to the AMVP mode, there are other new features for the AMVP mode in VVC. The SMVD technology sets the MVD for reference list 1 as a mirror of the MVD for reference list 0 to save the overhead for coding MVD. The AMVR technology allows MVDs of a block to be signaled in 4-, 1/2-, 1- or 4- luma sample resolutions for translational motion model, which is also adopted to further save bits of MVD. These two coding tools could be applied together for one block. Besides, to provide more precise motion compensation, the precision of MV is 1/16-luma sample in VVC instead of 4-luma sample in HEVC.

C. Block Merging

In VVC, merge/skip mode is more sophisticated than that in HEVC. First, besides the neighboring block-based merge candidates similar to those in HEVC, two new types of merge candidates are added into the merge candidate list, namely HMVP merge candidates and the pairwise average merge candidate. Second, besides the regular merge mode which is similar to merge mode in HEVC, VVC adopts three additional merge modes known as MMVD mode, CIIP mode and GPM.

The newly introduced HMVP and pairwise average merge candidates are put into the merge candidate list after the spatial or temporal neighboring block-based merge candidates. The pairwise average merge candidate in VVC replaces the combined bi-predictive merge candidates in HEVC. The pairwise average merge candidate is put after HMVP candidates and is generated by averaging the MVs of the first two available merge candidates in the merge candidate list. As in HEVC, a merge estimation region is adopted by VVC to facilitate the hardware design for the encoder.

The three additional merge/skip modes can help VVC to adapt better to varieties of video contents. MMVD serves as an intermediate motion representation between merge/skip mode and AMVP mode. An MVD index is signaled for a merge candidate and represents an MVD or a pair of MVDs that are limited to four directions and eight distances. The combination of inter-prediction and intra-prediction has been studied for many years [34]–[36]. With CIIP as adopted in VVC, only the planar mode is used to generate the intra-prediction block, and a merge candidate is used to generate the inter-prediction block. The final prediction is a weighted sum of the inter-prediction and intra-prediction blocks. MC with non-rectangular partitions also gains a lot of research attentions for years [37], [38]. With GPM in VVC, a coding block is partitioned into two parts which may be non-rectangular or asymmetric rectangular. Two inter-prediction blocks are generated with two MVs derived from two merge candidates for the two parts individually. The final prediction is a weighted sum of the two inter-prediction blocks with weighting values particularly designed for the partitioning shapes.

D. Weighting of Motion-Compensated Prediction and Motion Data Storage

In VVC, some coding tools such as BCW can be applied to both merge mode and AMVP mode. With BCW, a set of weighting value candidates can be selected for bidirectional inter prediction. The index of the selected weighting values is signaled for AMVP mode and inherited for merge mode, if allowed.

Besides coding efficiency, computational complexity and storage requirement are also intensively evaluated during standardization for video coding. To limit the required storage of MVs used for temporal prediction, VVC employs a new

algorithm to compress stored MVs, by using 10-bit mantissa-exponent representation.

IV. Motion Vector Coding in VVC

As known, inter prediction is a key part of every video coding standard. The compression ratio heavily relies on efficient representation of motion data. The motion vector coding in the new VVC standard is based on the well-established concepts of HEVC. However, there are several refinements, including the revised AMVP candidate list construction process, SMVD and AMVR, which lead to an improved coding efficiency. The detailed descriptions and theories behind those tools will be described in the following subsections.

A. Motion Vector Predictor List Generation Algorithm

The motion vector prediction algorithm of VVC is based on the AMVP of HEVC. It is applied in case of explicit motion vector signaling, i.e., for inter predicted blocks that do not use merge mode. For each motion vector, a list of exactly two motion vector predictor candidates is generated and a flag indicates which of the two is used. The following candidates are checked for availability:

- up to two spatial candidates (similar to that in HEVC),
- up to one temporal candidate (similar to that in HEVC),
- up to four HMVP candidates (new in VVC),
- zero motion vectors, if not enough other candidates are available (similar to that in HEVC).

For the spatial and the temporal (co-located) candidates, the same spatial locations as in HEVC are used. Unlike in HEVC, when the picture order count (POC) of the candidate's reference picture does not match the POC of the current reference picture, the candidate is considered unavailable in this case and the scaling of spatial candidates is removed in VVC to save the computational complexity. As in HEVC, in order to limit memory bandwidth requirements, the temporal motion vector predictor (TMVP) is restricted to only use co-located candidates that belong to the same coding tree unit (CTU) row as the current block. Also, as in HEVC, the TMVP candidate can be disabled at sequence level or at picture level. The TMVP is also not available for coding blocks of size 8×4 and 4×8 in VVC in order to improve the MV throughput since TMVP derivation process typically requires a scaling process which is relatively more complex compared to other merge candidates and MV determination process for small blocks is in a critical path for hardware implementation. In case that after checking the spatial and temporal candidates, still less than two candidates have been found, up to two candidates derived from HMVP table can be used as a new feature of VVC [8]. Finally, if there are still less than two candidates after checking HMVP, the motion vector candidate list is filled with zero motion vectors as in HEVC.

B. Symmetric Motion Vector Difference

In the bi-prediction mode, lots of bits are used to code the motion information which includes the reference picture indices, MVP indices and MVDs for reference picture list 0 and list 1. To code the motion information of bi-prediction mode in a more effective way, the SMVD technology is adopted. The SMVD mode is an inter bi-prediction mode in which part of motion information is derived with an assumption of linear motion. Therefore, the number of bits for motion information coding can be reduced. For a CU coded with a non-affine bi-prediction mode, a SMVD flag is signaled to indicate whether the SMVD mode is selected or not. When the flag is true, only the MVP indices of list 0 and list 1 and the MVD of list 0 are signaled, and other motion information (i.e., reference pictures and MVD of list 1) is not signaled but derived at the decoder side.

First, the MVD of list 1 is symmetrically derived from the MVD of list 0. That is, the MVD of list 1 $(mvdx_{L1}, mvdy_{L1})$ is symmetric to the MVD of list 0 $(mvdx_{L0}, mvdy_{L0})$, as shown in below:

$$\left(mvdx_{L1}, mvdy_{L1}\right) = \left(-mvdx_{L0}, -mvdy_{L0}\right). \quad (1)$$

Second, the reference pictures of list 0 and list 1 are determined at slice-level. These two reference pictures can only be short-term reference pictures and may be as in either case 1 or case 2, as follows.

- Case 1: Reference picture of list 0 is the nearest picture among all the pictures preceding the current picture in output order in list 0. Reference picture of list 1 is the nearest picture among all the pictures following the current picture in output order in list 1
- Case 2: Reference picture of list 0 is the nearest picture among all the pictures following the current picture in output order in list 0. Reference picture of list 1 is the nearest picture among all the pictures preceding the current picture in output order in list 1

If none of the reference pictures can be found as in case 1 or case 2, the SMVD mode is marked as unavailable and the SMVD flag is not sent.

C. Adaptive Motion Vector Resolution

The video coding standards HEVC and AVC/H.264 use a fixed motion vector resolution of quarter luma sample. However, it is a well-known fact that in order to achieve overall rate-distortion optimality, an optimum trade-off between displacement vector rate (R_d) and prediction error rate (R_e) has to be chosen [12]. In particular, at optimality the partial derivatives of the multivariate distortion rate function with respect to R_d and R_e have to be equal. Previously, this aspect has only been considered at the encoder-side in the motion estimation stage. The new video coding standard VVC allows to select the motion vector resolution at coding block level and, therefore, to trade-off bit rate versus fidelity for the signaling of the motion parameters. This is enabled by the AMVR mode which is based on ideas and concepts that have initially been described in [13]–[15], [39]. For translational inter predicted blocks, the following motion vector resolutions are which is out of scope for this paper. The AMVR mode is signaled at the coding block level if at least one component of an MVD is unequal to zero. The motion vector predictor is rounded to the given resolution such that the resulting motion vector is guaranteed to fall on a grid of the given resolution.

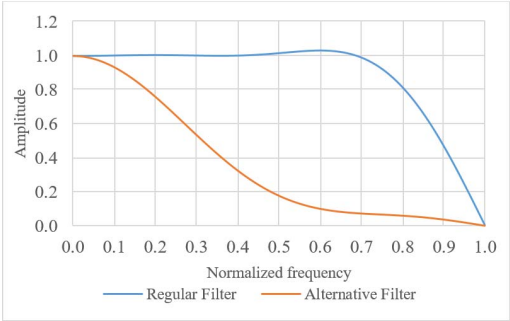


Fig. 7. Frequency responses of the half-pel interpolation filter.

Note that in case the half luma sample resolution is selected for a coding block, also an alternative luma interpolation filter is used for the half-sample position in this block. This aspect of AMVR is also known as switchable interpolation filter (SIF). The frequency responses of the regular interpolation filter and the alternative interpolation filter are shown in Fig. 7. It can be seen that the alternative filter has a strong low pass characteristic which can be beneficial for attenuating high frequency noise components. More details about SIF can be found in [16]. The application of the alternative interpolation filter is also further propagated in merge mode, i.e., if a coding block in merge mode references a neighboring block that uses the alternative interpolation filter, the referencing block will use it as well.

V. Block Merging in VVC

Block merging is an efficient coding tool in HEVC. In VVC, more merge modes are introduced for higher coding efficiency. In this section, regular merge mode is first introduced. Subsequently, new merge modes, including geometric partitioning mode, merge mode with motion vector difference, combined inter-intra prediction are discussed in order. Finally, merge estimation region for low encoding complexity is described.

A. Block Merging Candidate List Generation Algorithm

There are five types of motion vector (MV) predictor candidates in regular merge mode, i.e., spatial candidates, temporal candidates, HMVP candidates, pairwise average candidate, and zero MV candidates. The spatial and temporal candidates are the same as those in HEVC except that the order of the first two spatial candidates is swapped for higher coding efficiency. HMVP candidates, derived from an HMVP table, are inserted into merge list after spatial and temporal candidates until the merge list reaches the maximum allowed size minus one. To avoid duplicated candidates while keeping a relatively low complexity, redundancy check is applied. Only when any of the following three conditions is met, an HMVP candidate is inserted into the merge list.

- The HMVP candidate is not the last two in the HMVP table;
- The current HMVP candidate is not the same as the spatial candidates derived from A1/B1, as depicted in Fig. 2.

The pairwise average candidate is generated by averaging a pre-defined candidate pair in the existing merge candidate list. There is up to one pairwise candidate which averages the first two existing candidates in merge candidate list. The averaged motion vectors are calculated separately for each reference list. If both motion vectors are available in one list, these two motion vectors are averaged even when they point to different reference pictures; if only one motion vector is available, use the one directly; if no motion vector is available, keep this list invalid. If the merge candidate list is not full after inserting the pairwise average candidate, zero MV candidates will be added until the candidate list is full, which is the same as in HEVC.

TABLE I
Distance Tables (in Unit of Luma Samples) Used in MMVD

Index	Distance table 1	Distance table 2
0	1/4	1
1	1/2	2
2	1	4
3	2	8
4	4	16
5	8	32
6	16	64
7	32	128

B. Merge Mode With Motion Vector Difference

In addition to merge mode wherein motion information is directly derived from neighboring, historic or zero motion information, the MMVD technology [19], [20] is adopted to allow further encoding an MVD as refinement of the derived motion information. Roughly speaking, MMVD mode is between AMVP mode and merge mode, and it provides a new trade-off between accuracy of motion information and bitrate. In MMVD, one of the first two candidates in the merge candidate list is selected as base motion, and an MVD represented by a direction and a distance is encoded as refinement of the base motion. Four directions including {0, 90, 180, 270}-degrees are allowed for an MVD and a direction index is signaled to indicate the selected direction. Meanwhile, two distance tables [20], [21], each of which has eight distance entries as illustrated in Table I are designed for the MVD. The encoder can select a distance table at picture level. A distance index is signaled for an MVD to indicate the selected entry of the distance table.

Only one MVD is signaled for both unidirectional and bidirectional base motion. When the base motion is bidirectional, the signaled MVD is directly used for one reference picture list and is scaled according to POC distances from the current picture to the two reference pictures before being used for the other reference picture list. When the current picture is with shorter absolute POC distance to the reference picture in list 1 than to the reference picture in list 0, the signaled MVD is directly used for list 1, otherwise, it is directly used for list 0.

C. Geometric Partitioning Mode

The GPM [22]–[24] aims to increase the partition precision and to better fit the moving objects boundaries using a geometrical partition of a coding tree leaf node CU. Because of the more flexible partitioning and the blending process, the GPM is benefit to video contents that include rigid moving objects relative to static background or other moving objects.

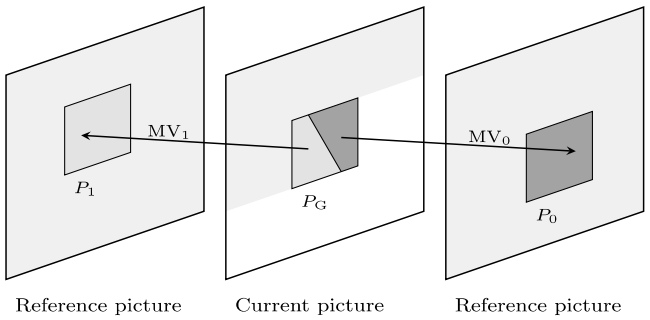


Fig. 8. Example of the prediction process of GPM; note that both predictions may originate from pictures in a same reference picture list, which is not shown in this example.

Furthermore, the newly design GPM algorithm significantly reduces the encoder and decoder complexity yet reserves the coding gain comparing with the prior methods proposed to HEVC. Therefore, the presented algorithm has been adopted in VVC. This section briefly presented the algorithm of GPM, for further background logic, comprehensive description, and statistical analysis, readers can refer to [24].

Fig. 8 shows an example of the prediction process of GPM. In VVC, GPM is designed for CU with size $w \times h = 2^k \times 2^l$ (in terms of luma samples) with $k, l \in \{3, \ldots, 6\}$. Moreover, GPM is disabled for a CU that has an aspect ratio larger than 4:1 or smaller than 1:4, considering narrow CUs rarely contain geometrically separated patterns. When GPM is applied, the current CU is split into two parts by a straight partitioning boundary, parameterized by an angle φ and an offset ρ. In total, 64 partitioning lines are supported and indexed by GPM partition index. Each part of the partition associates a unidirectional MV that is coded with merge mode. The GPM merge list is directly derived from the regular merge list using the parity of the indices. The GPM partition index and the two GPM merge indices are coded into the bitstream. For each part, a block-based motion compensation prediction (MCP) is performed, resulting in two intermediate prediction blocks P_0 and P_1. A GPM prediction block P_{G} is generated by performing a blending process using integer blending matrices W_0 and W_1, containing weights in the value range of [0, 8]. This can be expressed as

$$P_{\mathrm{G}} = (W_0 \circ P_0 + W_1 \circ P_1 + 4) \gg 3 \tag{2}$$

with

$$W_0 + W_1 = 8J_{w,h}, \tag{3}$$

where "$\circ$" in (2) denotes the Hadamard product and $J_{w,h}$ denotes a matrix of ones with size of the current CU. The generated GPM prediction P_{G} is subtracted from the original signal to generate residuals, which is transformed and coded into the bitstream using the regular VVC transform coding and CABAC engine.

The weights in the blending matrices of GPM are derived based on the displacement from a sample location to the partitioning boundary as shown in Fig. 9. The displacement from an arbitrary location (x_C, y_C) to the partitioning boundary is given by Hessian normal form as

$$d\,(x_C, y_C) = x_C \cos(\varphi) - y_C \sin(\varphi) + \rho, \tag{4}$$

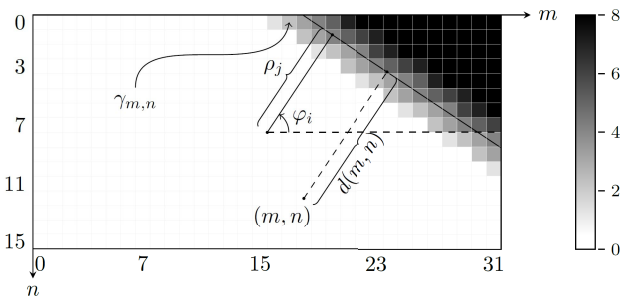


Fig. 9. An example of GPM blending matrix derivation.

where (x_C, y_C) is the coordinate relative to the CU center, φ is the anticlockwise angle from x-axis, and ρ is the displacement from the origin. The Hessian normal form contains only angle and offset parameters, which can be easily quantized and grouped to reducing the signaling cost. Equation (4) can be quantized and discretized as

$$\begin{aligned} d\,(m,n) = &\left(\left(\left(m + \rho_{x,j}\right) * 2 - w + 1\right) \cdot \mathrm{cosLut}[i]\right) \\ &+ \left(\left(\left(n + \rho_{y,j}\right) * 2 - h + 1\right) \cdot \mathrm{cosLut}[(i+8)\,\%32]\right), \end{aligned} \tag{5}$$

where (m, n) denotes integer sample locations relative to the top-left sample of the CU of size $w \times h$. In (5), the angle parameter is quantized into φ_i with fixed $\tan(\varphi_i)$ in $\{0, \pm 1/4, \pm 1/2, \pm 1, \pm 2, \infty\}$. Note that the tangent values of ± 4, which yield a near horizontal partitioning boundary, are not included, because the near horizontal partitioning of a motion field is rarely used for natural video content. The offset parameter is quantized into ρ_j that is factorized as

$$\rho_{x,j} = \begin{cases} 0, & i\%16 = 8 \text{ or } (i\%16 = 0 \text{ and } h \geq w) \\ \pm j \cdot w/8, & \text{otherwise} \end{cases} \tag{6}$$

and

$$\rho_{y,j} = \begin{cases} 0, & i\%16 = 8 \text{ or} (i\%16 = 0 \text{ and } h \geq w) \\ \pm j \cdot h/8, & \text{otherwise,} \end{cases} \tag{7}$$

where i and j denote the indices of angle parameter and offset parameter, respectively. The sign of $\rho_{x,j}$ and $\rho_{y,j}$ are set as positive if angle index $i < 16$, otherwise negative. Equations (6) and (7) couple the factorized offset parameters $\rho_{x,j}$ and $\rho_{y,j}$ with the width or height of the current CU to adapt GPM partitions to CUs with different sizes. The $\cos(\varphi)$ and $\sin(\varphi)$ values are discretized as a 4-bit precision (three bits for value and one bi for sign) look-up table cosLut[·]. Note that the displacement $d(m, n)$ in (5) can be derived without multiplication operations, because the values in cosLut[·] and the value of $w/8$ and $h/8$ in $\rho_{x,j}$ and $\rho_{y,j}$ are all shift-based values (i.e., 2^k with $k \in \mathbb{Z}^{\geq 0}$). The weights $\gamma_{m,n}$ in one of the blending matrices is given by a discrete ramp function as

$$\gamma_{m,n} = \mathrm{Clip3}\,(0, 8, (d\,(m,n) + 32 + 4) \gg 3)\,, \tag{8}$$

and the other blending matrix is easily derived from (3).

Unlike the regular bi-prediction, the GPM coded CUs contain three types of MVs. That is, each partition contains its own unidirectional MVs, and the blending area is physically predicted by bidirectional MVs from both partitions. Therefore, the MVs of GPM, which are stored for the MV prediction of the succeeding CUs, are adapted to the partitioning

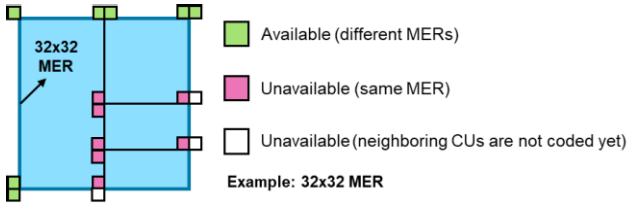


Fig. 10. Illustration of spatial merging candidate insertion with MER.

boundary. The displacement from the integer central position of a 4 × 4 motion storage unit to the partitioning boundary is re-calculated by (5). When the displacement is larger than a threshold, the unidirectional MV, which is used to predict the corresponding GPM partition, is stored depending on the sign of the displacement. Otherwise, combined bidirectional MVs from both unidirectional MVs are stored.

D. Combined Inter-Intra Prediction

Inter prediction uses the signaled motion data to reference the temporal information from different pictures to perform motion compensation which shows significant benefit in video compression. Among inter prediction, merge mode is one special inter prediction which uses simpler signaling scheme to derive the motion data based on a previously coded CU. On the other hand, intra prediction tends to provide more accurate spatial prediction when a sample is closer to the reference sample. To take the advantages of both inter-prediction merge mode and intra prediction, a new merge mode, called CIIP mode, is designed for the CU which contains at least 64 luma samples and has both CU width and CU height smaller than 128 [25], [26]. In CIIP mode, a weighted combination of inter-prediction merge mode and intra prediction is utilized for prediction as follows. The merge prediction is derived with the inter-prediction process for a regular merge mode and the intra prediction is derived with the intra-prediction process for planar mode. Then, a weighted averaging process is applied to combine both predictions. The sum of prediction weights is equal to 4 and a right-shift operation is used after adding two weighted predictions. The final prediction for CIIP, denoted as P_{CIIP}, is formed as follows.

$$P_{\text{CIIP}} = \left(W_{merge} * P_{merge} + W_{intra} * P_{intra} + 2\right) \gg 2, \quad (9)$$

wherein W_{intra} is the weight for the intra prediction, denoted as P_{intra}, W_{merge} is the weight for the merge prediction, denoted as P_{merge}, and sum of W_{merge} and W_{intra} is equal to 4. The weights for intra- and merge- predicted samples are uniform in the whole CU and decided based on the number of neighboring intra blocks. If both top and left neighboring blocks are intra-coded, W_{intra} is set as 3 which means intra prediction is preferred. Otherwise, if only one of these blocks is intra-coded, W_{intra} is set as 2 which implies identical weights are used for two predictions. Otherwise, W_{intra} is set as 1. The weights for CIIP are the same for luma and chroma components. By weighted averaging one existing merge prediction and another existing intra prediction with a simple weighting process, better coding efficiency could be achieved. Moreover, to avoid 2×N intra blocks, only the merge prediction is used for the chroma CB when the chroma CB has width smaller than 4. To indicate the usage of CIIP mode, an additional flag is conditionally signaled to indicate whether CIIP is used or not when regular merge mode is selected.

E. Merge Estimation Region

Merge estimation region (MER) was first introduced in HEVC as an implementation-friendly feature for encoders. It is used to enable parallel cost estimation of merging candidates for different CUs. MER divides a picture into equally sized and non-overlapping square regions and allows a spatial merging candidate to be added into merging candidate list only when the current CU and the neighboring CU are in different MERs, as shown in Fig. 10.

Two CUs are treated as in the same MER when the following conditions are met,

$$\begin{cases} (\text{xCb} \gg \text{Log2ParMrgLevel}) = (\text{xNb} \gg \text{Log2ParMrgLevel}) \\ (\text{yCb} \gg \text{Log2ParMrgLevel}) = (\text{yNb} \gg \text{Log2ParMrgLevel}) \end{cases}$$

wherein Log2ParMrgLevel specifies the MER size in base 2 logarithm, xCb and yCb are the coordinates of the current CU, xNb and yNb are the coordinates of the spatial neighboring CU.

In addition to spatial merging candidates, subblock-based merging candidates follow the same rules as that of spatial merging candidates.

When updating HMVP table, a constraint is imposed to break the dependency between different CUs within one MER in order to allow parallel processing. That is, HMVP table is not updated until the last CU located at the bottom-right of a MER satisfies the following conditions,

$$\begin{cases} ((\text{xCb} + \text{cbWidth}) \gg \text{Log2ParMrgLevel}) \\ \quad > (\text{xCb} \gg \text{Log2ParMrgLevel}) \\ ((\text{yCb} + \text{cbHeight}) \gg \text{Log2ParMrgLevel}) \\ \quad > (\text{yCb} \gg \text{Log2ParMrgLevel}) \end{cases} \quad (10)$$

wherein cbWidth and cbHeight are width and height of the current CU.

Moreover, encoder-only binary tree (BT) and ternary tree (TT) split constraints are needed where the general rule for MER is that any CU not smaller than MER size should contain one or multiple complete MERs and any CU smaller than MER size should locate within one MER entirely. The following shows the detailed constraint: When either width or height of current CU is larger than MER, the following applies:

- If cbHeight <= R, then disallow horizontal BT split for current CU;
- If cbWidth <= R, then disallow vertical BT split for current CU;
- If cbHeight <= 2 * R, then disallow horizontal TT split for current CU;
- If cbWidth <= 2 * R, then disallow vertical TT split for current CU;

wherein R is equal to 1 ≪ Log2ParMrgLevel.

The MER size can be adaptively decided and signaled as sps_log2_parallel_merge_level_minus2 in the sequence

parameter set, wherein Log2ParMrgLevel is equal to 2 +sps_log2_parallel_merge_level_minus2.

VI. Others

A. Bidirectional Prediction With Coding Unit Weights

The BCW technology (a. k. a. generalized bi-prediction) is a syntax shortcut of weighted bi-prediction (WP) to predict a block by weighted-averaging two motion-compensated prediction blocks. Unlike WP which indicates weights at slice level respectively for all reference pictures, BCW signals the use of weight at CU level by using an index (denoted as *wIdx*) pointing to where the selected weight is located in a list of pre-defined candidate weights. In general, this list pre-defines 5 candidate weights (i.e., {−2, 3, 4, 5, 10}/8) to be selected for reference pictures in reference list 1, where −2/8 and 10/8 are used to reduce negatively correlated noises between prediction blocks of bi-prediction. The list may be reduced to {3, 4, 5}/8 when there are forward and backward reference pictures in both reference lists to achieve better trade-off between performance and complexity [45]. Since unit-gain constraint is applied, once the weight (denoted as W pointed to by *wIdx*) corresponding to reference list 1 is determined, the weight corresponding to the other reference list is 1–W. Specifically, each luma/chroma prediction sample of BCW is computed as follows:

$$P_{BCW} = (8(1 - W) * P0 + 8W * P1 + 4) \gg 3, \quad (11)$$

where P_{BCW} is the final prediction of a current-block sample and $P0$ and $P1$ are prediction samples pointed to by the motion vectors respectively from list 0 and list 1 reference pictures. Note that BCW enables only for bi-predicted CUs with at least 256 luma samples and WP being turned off.

The notion of BCW is also extended to affine AMVP modes. Same as aforementioned, BCW signals a *wIdx* for the whole affine CU, and the corresponding weights of the signaled index are applied to subblock motion compensation [29].

The use of *wIdx* is buffered for subsequent CUs in the same frame to perform spatial motion merging, either for regular or for affine merge mode. When a spatial neighboring merge candidate is bi-predicted and the current CU selects this candidate, all the reference indices and motion vectors (or control-point motion vectors in the case of inherited affine merge mode) including its *wIdx* are inherited by the current CU. The only exception that the *wIdx* is not inherited occurs when the current CU has CIIP flag enabled. In the case of constructed affine merge mode, the *wIdx* is simply inherited from the one associated with above-left control-point motion vectors (or above-right control-point motion vectors when above-left ones are not used) [30]. It is noted that when the inferred *wIdx* points to a non-0.5 weight, decoder-side motion vector refinement and bidirectional optical flow are both turned off.

B. Motion Vector Compression and Range

In VVC, the MV precision is increased from quarter luma sample in HEVC to 1/16-luma sample to provide a more precise motion compensation. With the increase of the MV precision, the bit depth of MV is also extended by 2 bits in order to allow the same MV range as that in HEVC. Thus, in VVC, 18 bits are used for one MV component with the range from -2^{17} to $2^{17}-1$.

In terms of temporal motion storage, the motion field compression is performed in two aspects. First, the temporal motion vectors are stored at 8×8 granularity instead of 16×16 granularity in HEVC. Second, each component of a temporal MV is represented using a 6-bit signed mantissa plus a 4-bit exponent to further reduce the temporal motion storage. The benefit of using the mantissa plus exponent format is that this representation effectively quantizes larger MVs more coarsely while maintaining higher precision for smaller MVs. The conversion between an 18-bit MV representation and 10-bit mantissa-exponent representation is not analogous to IEEE 754 but an approximation that allows for efficient implementations using only additions and shifts, as shown below:

$$\begin{aligned} sign &= mv \gg 17, scale = \lfloor \log_2((mv \oplus sign)|31) \rfloor - 5, \\ val &= (mv + ((1 \ll scale) \gg 1)) \gg scale, \\ exp &= \begin{cases} scale + ((val \oplus sign) \ll 5) & scale \geq 0 \\ 0 & others, \end{cases} \\ mant &= \begin{cases} (val\&31)|(sign \ll 5) & scale \geq 0 \\ mv & others, \end{cases} \end{aligned} \quad (12)$$

and the inverse is obtained as:

$$mv' = \begin{cases} mant & exp = 0 \\ (mant \oplus 32) \ll (exp - 1) & others, \end{cases} \quad (13)$$

where $\oplus$ represents bitwise XOR operation, | is bitwise OR operation, & is bitwise AND operation, mv and mv are the value of an MV component before and after compression, respectively. With this compression, number of bits for temporal MVs stored in a 16×16 luma block are reduced from 288 bits (i.e. $2 \times 2 \times 18 \times 4$) to 160 bits. As compared to HEVC in which 64 bits (i.e. $2 \times 2 \times 16 \times 1$) are needed, 96 bits are increased for a 16×16 luma block in a reference picture.

VII. Experimental Results

In this section, the inter-prediction coding regarding MV coding and block merging in VVC is evaluated. The JVET common test conditions [40] were used to measure the coding performance with bit-rate savings in terms of the Bjøntegaard Delta (BD) rate [41]. To calculate the BD-rates, four rate points were generated by using quantization parameters (QPs) 22, 27, 32, and 37 and piece-wise cubic interpolation [40] was used. Weighted combined PSNRs of YUV components (using a weighting factor of 6 for the luma component and 1 for the two chroma components) were used to consider chroma fidelity in the BD-rates calculation. Note, the positive ΔBD-rate in tables below represents the bitrate increases in the bitstreams while maintaining the same PSNR for the reconstructed video due to disabling the coding tools. The less than 100% encoding and decoding time indicates that the encoder and decoder speed are faster than anchor.

The video sequences in the common test conditions (CTC) are categorized into six classes (Class A-F) covering

TABLE II

OVERALL PERFORMANCE OF WHOLE BLOCK-BASED INTER TOOLS IN VVC

	RA			LDB		
Sequences	ΔBD-Rate	Enc. Time	Dec. Time	ΔBD-Rate	Enc. Time	Dec. Time
Class A1	6.0%	48%	99%	-	-	-
Class A2	6.7%	44%	98%	-	-	-
Class B	6.2%	51%	101%	4.8%	45%	96%
Class C	5.8%	51%	101%	5.3%	44%	95%
Class E	-	-	-	3.6%	52%	105%
Averages	6.2%	49%	100%	4.7%	46%	100%
Class D	4.7%	50%	100%	4.5%	48%	95%

resolutions from UHD (Class A1/A2, 3840 × 2160) to WQVGA (Class D, 416 × 240) and frame rates from 60 frame per second to 20 frame per seconds. While Classes A-D consist of natural camera captured material, Class F includes computer-generated contents and mixed natural video and computer-generated contents. The average BD-rate saving only measures Class A-C, targeting higher resolution natural camera applications.

VTM-9.0 reference software [42] with the Main10 profile settings were used to evaluate the VVC inter-prediction coding performance. Two configurations, random access (RA) and low delay with B slices (LDB), were utilized to simulate common video applications. In the RA configuration, which targets video broadcasting applications, an Intra Random Access Pictures (IRAP) frame was inserted approximately every one second. In LDB configuration, which targets real time video applications, decoding order and output picture order need to be the same.

A. Performance Analysis of VVC Inter-Prediction Coding Tools in VTM

The MV coding and block merging techniques mentioned in Section III were tested to demonstrate their overall performance impact relative to VTM anchors. Table II summarizes the coding performance by disabling all coding tools described in Section III, in terms of ΔBD-rate and runtime ratio of encoder and decoder, where a 100% in Enc/Dec Time represents the test has the same software runtime as VTM anchor. The positive ΔBD-rate represents the bitrate increases in the bitstreams while maintaining the same PSNR for the reconstructed video due to disabling the coding tools. The less than 100% encoding and decoding time indicates that the encoder and decoder speed are faster than anchor. Component-wise coding performance (by disabling individual tool) was also reported in Table III and Table IV respectively for RA and LDB to profile individual contribution to overall coding gain.

As reported in Table II, the overall ΔBD-rate is slightly larger in RA (4.7%−6.7%) than in LDB (3.6%−5.3%). Basically, better coding gain is observable from large-resolution sequences (e.g., 4.7% for WQVGA and 6.7% for UHD in RA). According to Table III and Table IV, it is attributed to tools, such as AMVR, which are more adaptive to resolution changes and texture-complexity variation, while others appear relatively invariant to video content types.

TABLE III

CODING PERFORMANCE ON RA CONFIGURATION, Δ BD-RATE (%)

Sequences	HMVP/ Pairwise	SMVD	AMVR	GPM	MMVD	CIIP	BCW
Class A1							
Tango2	1.1	0.3	3.2	0.8	0.6	0.1	0.4
FoodMarket4	0.8	0.3	1.5	0.4	0.5	0.1	0.5
Campfire	0.4	0.1	1.2	0.2	0.2	0.2	0.4
Class A2							
CatRobot	0.8	0.3	1.6	0.7	0.5	0.1	0.6
DaylightRoad	1.5	0.3	2.4	0.4	0.9	0.2	0.2
ParkRunning	1.1	0.3	1.1	0.5	0.5	0.2	0.3
Class B							
MarketPlace	0.9	0.2	1.3	0.5	0.5	0.1	0.5
RitualDance	1.1	0.3	1.8	0.6	0.4	0.3	0.3
Cactus	0.4	0.2	1.1	0.7	0.3	0.5	0.4
BasketballDrive	1.2	0.4	2.5	0.4	0.9	0.1	0.6
BQTerrace	1.5	0.2	1.1	0.3	0.8	0.0	0.9
Class C							
BasketballDrill	1.0	0.2	1.7	1.3	0.1	0.3	0.2
BQMall	1.0	0.2	0.9	2.4	0.6	0.1	0.3
PartyScene	0.6	0.2	0.8	0.7	0.6	0.3	0.3
RaceHorses	1.2	0.1	1.7	1.6	0.4	0.2	0.3
Class D							
BasketballPass	0.9	0.2	1.7	1.0	0.4	0.2	0.1
BQSquare	0.7	0.1	0.6	0.1	1.8	0.1	0.2
BlowingBubbles	0.5	0.2	0.5	0.8	0.5	0.3	0.2
RaceHorses	1.0	0.1	1.2	1.5	0.4	0.2	0.1
Averages							
Class A1	0.8	0.2	2.0	0.5	0.4	0.1	0.5
Class A2	1.1	0.3	1.7	0.6	0.6	0.2	0.4
Class B	1.0	0.3	1.6	0.5	0.6	0.2	0.5
Class C	0.9	0.2	1.3	1.5	0.4	0.2	0.2
Average (A, B, C)	1.0	0.2	1.6	0.8	0.5	0.2	0.4
Class D	0.8	0.1	1.0	0.8	0.8	0.2	0.2

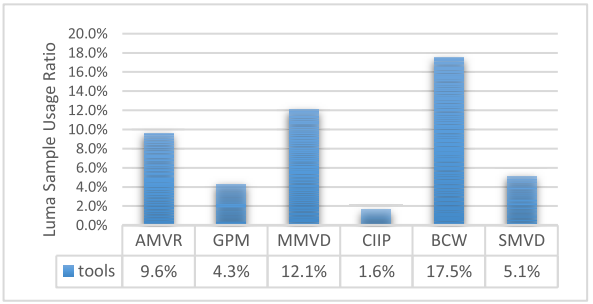
When Table II is compared with Table III and Table IV, it is observed that these tools can cooperate to reach nearly synergy effect. It is obvious that one tool would compete with another since they share partially but not completely the same source that boosts coding performance. For example, SMVD and MMVD share the same common ground in motion field representation but their MVDs are signaled differently at finer or coarser granularity based on their respective underlying motion models. The experiment results confirm the performance-wise overlap among tools is modest.

The encoding time is approximately doubled, mainly due to extra rate-distortion evaluation for mode selection (e.g., AMVR, BCW, MMVD) at the encoder [43]. The variation of decoding time is relatively minor since the extra computational complexity introduced (e.g. blending for GPM, BCW and CIIP) is nearly negligible when compared with interpolation process of motion compensation.

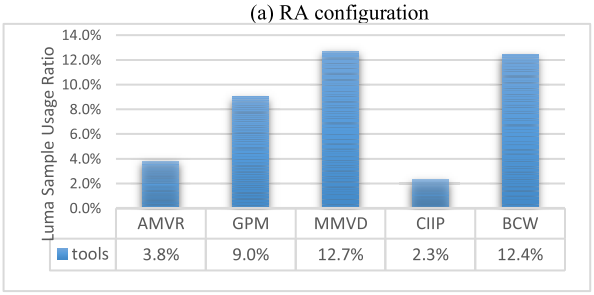
Among all, AMVR and GPM are the two most performing tools, each of which could deliver up to 1.6% of ΔBD-rate on average. Detailed analyses will be provided in later sections.

B. Luma Samples Coverage of VVC Inter-Prediction Coding Tools in VTM

The effectiveness of the MV coding and block merging techniques were further justified by using the luma samples coverage of each coding tool. Fig. 11 (a) and (b) illustrate the average percentage of inter predicted samples in different coding modes for RA and LDB, respectively. Over 40% (and



(a) RA configuration



(b) LDB configuration

Fig. 11. Luminance sample coverage of inter coding tools in VTM-9.0.

TABLE IV
CODING PERFORMANCE ON LOW DELAY B CONFIGURATION, Δ BD-RATE (%)

Sequences	HMVP/ Pairwise	AMVR	GPM	MMVD	CIIP	BCW
Class B						
MarketPlace	0.5	0.5	0.8	0.7	0.5	0.2
RitualDance	0.5	0.8	1.0	0.6	0.8	0.0
Cactus	0.1	0.5	1.5	0.0	0.8	0.4
BasketballDrive	0.6	1.2	1.1	1.0	0.5	0.3
BQTerrace	1.2	0.7	0.5	0.5	0.3	0.8
Class C						
BasketballDrill	0.4	0.9	2.6	0.3	0.6	0.3
BQMall	0.5	0.6	2.8	0.7	0.3	0.1
PartyScene	0.3	0.4	1.2	0.5	0.4	0.1
RaceHorses	0.6	0.9	1.8	0.7	0.3	0.2
Class D						
BasketballPass	0.3	0.9	1.9	0.8	0.4	0.1
BQSquare	0.6	0.6	0.8	0.4	0.3	0.7
BlowingBubbles	0.4	0.4	2.0	0.4	0.5	0.0
RaceHorses	0.4	1.0	2.4	1.0	0.2	0.0
Class E						
FourPeople	-0.1	0.4	2.3	0.2	0.4	0.1
Johnny	0.3	0.2	1.8	0.1	0.5	0.5
KristenAndSara	0.2	0.4	2.3	0.1	0.3	0.3
Averages						
Class B	0.6	0.7	1.0	0.6	0.6	0.3
Class C	0.5	0.7	2.1	0.5	0.4	0.2
Class E	0.1	0.3	1.9	0.2	0.4	0.3
Average (B, C, E)	0.4	0.6	1.6	0.5	0.5	0.3
Class D	0.4	0.7	1.8	0.7	0.3	0.2

up to 50% for RA) of the samples were coded with at least one of the VVC inter coding tools (not including HMVP and pairwise merge candidates). Note that one luma sample may be coded with multiple tools (e.g., being coded with BCW and another inter coding tools), so the luma sample may be counted multiple times accordingly. And since SMVD is disallowed for LDB configuration, it is not depicted in Fig. 11 (b). These numbers roughly indicate VVC could achieve a more effective trade-off between the accuracy of the motion field representation and the required overhead. Thus, almost half

TABLE V
LUMA SAMPLE COVERAGE OF AMVR AND SIF WHEN USING VTM-9.0

Class	AMVR								SIF
	all	translational				affine			
		all	1-Pel	4-Pel	1/2-Pel	all	1/16-Pel	1-Pel	
A1	8.0%	6.8%	3.4%	1.5%	1.9%	1.2%	0.1%	1.0%	7.2%
A2	7.7%	4.9%	3.0%	0.5%	1.4%	2.9%	0.5%	2.3%	4.2%
B	5.7%	4.2%	2.3%	0.6%	1.3%	1.6%	0.5%	1.1%	4.4%
C	5.1%	3.6%	2.5%	0.4%	0.7%	1.6%	0.8%	0.8%	1.2%
D	4.4%	3.1%	2.2%	0.3%	0.6%	1.4%	0.7%	0.7%	1.1%
F	2.7%	1.9%	1.3%	0.4%	0.2%	0.9%	0.6%	0.3%	0.3%
Avg.	**5.4%**	**3.9%**	**2.4%**	**0.6%**	**1.0%**	**1.5%**	**0.5%**	**1.0%**	**2.9%**

of the luma samples in testing sequences that were coded by using HEVC-based methods (e.g., quarter luma sample MVD without precision adaptivity, rectangular-only prediction unit with equal weights, regular merge without offsetting) are now replaced by using the VVC-based MV coding and block-merging techniques.

C. AMVR

In Table V, the various AMVR modes are analyzed based on the luma sample usage. The numbers give the percentage of all luma samples which are encoded using the given AMVR mode when using VTM-9.0 under the CTC. In the heading, "x-Luma sample" corresponds to an MV resolution of x luma samples. It can be seen that AMVR is used more often for higher resolution sequences. The same applies to SIF. Note that due to merge mode, the percentage for SIF is higher than that for the half sample AMVR mode.

In Table VI, the AMVR modes are analyzed using tool-off tests. Again, the CTC are used. For the results in the first column, AMVR has been disabled completely, i.e., both in the translational and the affine variant. The resulting BD-rate values are positive, indicating a coding loss caused by disabling AMVR. In Test 1, only translational AMVR with a motion vector resolution of one luma sample ("full-sample AMVR") has been enabled. A small, positive BD-rate number indicates smaller coding loss. In other words, the difference between the values in the first and the second column shows the coding gain of full-sample AMVR. In Test 2, both full-sample and 4-sample AMVR are enabled. In particular for the high-resolution sequences in Class A1, the coding gain has been further improved. In Test 3, translational AMVR with all supported motion vector resolutions (i.e., full-sample, 4-sample, and half- sample), including SIF, has been enabled, only affine AMVR is still disabled. Again, the positive BD-rate values have been further reduced, especially for the high-resolution classes of sequences, indicating a higher coding gain. In the last test ("SIF off"), only the switchable interpolation filter has been disabled, showing an average coding loss of 0.3%.

D. GPM

As shown in Table III (fourth column) and Table IV (third column), an average BD-rate reduction of 0.8% and 1.6% for RA and LDB configurations can be achieved by GPM. In some sequences (e.g., *BQMall, RaceHorses BasketballDrill*, and *KristenAndSara*), the coding performance was notably better

TABLE VI

YUV BD-RATE RESULTS FOR VARIOUS AMVR CONFIGS (TEST 1: ONLY TRANSLATIONAL AMVR WITH 1-SAMPLE RESOLUTION; TEST 2: LIKE 1, BUT ADDITIONALLY ALSO 4-SAMPLE RESOLUTION; TEST 3: LIKE 2, BUT ADDITIONALLY ALSO 1/2-SAMPLE RESOLUTION)

Class	AMVR off	Test 1	Test 2	Test 3	SIF off
A1	1.96%	0.93%	0.64%	0.27%	0.35%
A2	1.72%	0.80%	0.73%	0.34%	0.24%
B	1.57%	0.78%	0.68%	0.22%	0.38%
C	1.27%	0.52%	0.47%	0.21%	0.20%
Avg.	**1.60%**	**0.74%**	**0.63%**	**0.25%**	**0.30%**
D	0.99%	0.42%	0.38%	0.20%	0.11%
F	0.97%	0.34%	0.26%	0.17%	0.07%

TABLE VII

YUV BD-RATE RESULTS FOR VARIOUS MER SIZES

(A) ON RANDOM ACCESS CONFIGURATION

MER Size	RA-YUV	Y	U	V	EncT	DecT
8	1.13%	1.05%	1.45%	1.51%	81%	99%
16	2.61%	2.49%	3.00%	3.15%	63%	100%
32	2.44%	2.38%	2.57%	2.73%	69%	101%
64	2.34%	2.32%	2.33%	2.54%	106%	101%
128	3.89%	3.87%	3.88%	4.08%	110%	100%

(B) ON LOW DELAY B CONFIGURATION

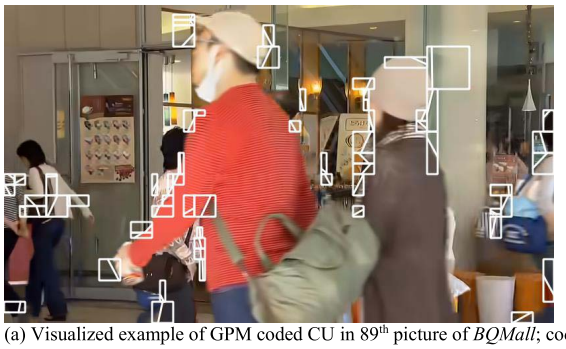
MER Size	LDB-YUV	Y	U	V	EncT	DecT
8	1.37%	1.26%	1.61%	2.00%	82%	100%
16	3.19%	3.01%	3.84%	4.00%	64%	100%
32	2.86%	2.76%	3.20%	3.33%	69%	102%
64	2.76%	2.78%	2.53%	2.81%	103%	102%
128	4.64%	4.67%	4.38%	4.61%	106%	100%

than the average for both RA and LDB configurations. The particularly favorable results of these sequences were mainly attributed to clearly distinctive motion field and moving object boundaries contained in these sequences. Another observation is that the performance of GPM for LDB was generally better than that for RA. The reason is that the reference pictures are typically closer to the current picture in LDB than in RA, which yields shorter MVs that are more easily to be coded using merge mode in both partitions of GPM. Therefore, GPM is more often selected in LDB than in RA.

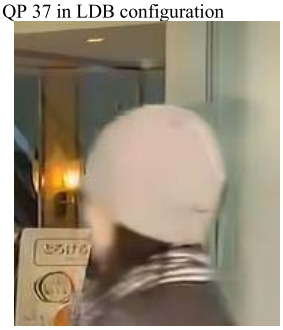
In addition to the improved coding performance and the low complexity, GPM also enhanced the visual quality. For the sequences containing motion of rigid objects, the GPM was used predominantly for coding of the moving objects boundaries as an example shown in Fig. 12(a). Therefore, sharp and clear edges of moving objects were visible in the coded sequences with GPM instead of the serrated and blurred moving object boundaries caused by rectangular partitions in the coded sequences without GPM. This phenomenon was shown as still picture example in Fig. 12(b) and (c).

E. MER

MER is an important feature for a commercial hardware encoder because it can effectively reduce pipeline latency which is introduced by deriving merge candidates depending on the MVs of the spatial neighbors of the current CU. By using MER, merge mode decision of all CUs inside the MER region can be performed at the same time without



(a) Visualized example of GPM coded CU in 89th picture of *BQMall*; coded at QP 37 in LDB configuration



(b) VTM-9.0 without GPM

(c) VTM-9.0 with GPM

Fig. 12. Visual impression of GPM.

waiting for each other. When exerting MER at the encoder, the performance on VTM-9.0 are shown in Table VII. Y with MER size equal to 8×8, 16×16, 32×32, 64×64 and 128×128, which corresponds to Log2ParMrgLevel 3, 4, 5, 6, 7. Since encoder-only BT and TT split constraints are also applied, the valid CU partitions for encoding may differ from MER size to MER size, which results in encoder run time variations, e.g. 20% run time reduction for 8×8 MER region, 30% reduction for 16×16 and 32×32 cases, and 10% increase for 64×64 and 128×128 cases, respectively. In a practical commercial hardware encoder architecture for VVC, 32×32 or 64×64 MER regions are activated in most cases. There is one alternative encoder-only design [44] that simply disables inter merge modes in CUs smaller than MER where the sequential processing problem in merge mode can be avoided as well. Compared with the alternative encoder-only design, MER method shows much less coding efficiency loss in both 32×32 and 64×64 cases.

VIII. CONCLUSION

The motion vector coding and block merging tools in the VVC which enhance and extend the main concept of inter coding in HEVC are introduced in this paper. The technical details and design philosophy are presented and illustrated. Thanks to those advanced coding tools, significant objective and subjective improvements have been demonstrated when compressing various video contents and under different use cases. In addition, the complexity of the tools was investigated and carefully optimized during the standardization process in JVET which will be definitely helpful for the success of VVC.

ACKNOWLEDGMENT

The authors would like to greatly thank all the JVET experts who have contributed to the VVC inter mode coding.

REFERENCES

[1] B. Bross, J. Chen, S. Liu, and Y.-K. Wang, *Versatile Video Coding (Draft 10)*, document JVET-S2001 ITU-T/ISO/IEC, Joint Video Experts Team, Jul. 2020.

[2] J. Chen, Y. Ye, and S. Kim, *Algorithm Description for Versatile Video Coding and Test Model 10 (VTM 10)*, document JVET-S2002 ITU-T/ISO/IEC, Joint Video Experts Team, Jul. 2020.

[3] A. Segall, V. Baroncini, J. Boyce, J. Chen, and T. Suzuki, *Joint Call for Proposals on Video Compression with Capability beyond HEVC*, document JVET-H1002 ITU-T/ISO/IEC, Joint Video Exploration Team, Oct. 2017.

[4] *Requirements for a Future Video Coding Standard V5*, document N17074 ISO/IEC JTC 1/SC 29/WG 11, MPEG, Jul. 2017.

[5] W.-J. Chien, Y. Chen, J. Chen, L. Zhang, M. Karczewicz, and X. Li, "Sub-block motion derivation for merge mode in HEVC," *Proc. SPIE Appl. Digit. Image Process.*, vol. 9971, Sep. 2016, Art. no. 99711K.

[6] M. Karczewicz, L. Zhang, W.-J. Chien, and X. Li, "Geometry transformation-based adaptive in-loop filter," in *Proc. Picture Coding Symp. (PCS)*, Nuremberg, Germany, 2016, pp. 1–5, doi: 10.1109/PCS.2016.7906346.

[7] H. Yang *et al.*, "Subblock-based motion derivation and inter prediction refinement in versatile video coding standard," *IEEE Trans. Circuits Syst. Video Technol.*, early access, Jul. 27, 2021, doi: 10.1109/TCSVT.2021.3100744.

[8] L. Zhang, K. Zhang, H. Liu, Y. Wang, P. Zhao, and D. Hong, *CE4: History-based Motion Vector Prediction (Test 4.4.7)*, document JVET-L0266 ITU-T/ISO/IEC, Joint Video Experts Team, Macao, CN, USA, Oct. 2018.

[9] L. Zhang *et al.*, "History-based motion vector prediction in versatile video coding," in *Proc. Data Compress. Conf. (DCC)*, Snowbird, UT, USA, Mar. 2019, pp. 43–52, doi: 10.1109/DCC.2019.00012.

[10] H. Chen, H. Yang, and J. Chen, *Symmetrical Mode for Bi-Prediction*, document JVET-J0063 ITU-T/ISO/IEC, Joint Video Experts Team, San Diego, CA, USA, Apr. 2018.

[11] J. Luo and Y. He, *CE4-related: Simplified Symmetric MVD based on CE4.4.3*, document JVET-M0444 ITU-T/ISO/IEC, Joint Video Experts Team, Marrakech, MA, USA, Jan. 2019.

[12] B. Girod, "Rate-constrained motion estimation," in *Proc. SPIE*, vol. 2308, Chicago, IL, USA, Sep. 1994, pp. 1026–1034.

[13] J. Chen *et al.*, *Further Improvements to HMKTA-1.0*, document VCEG-AZ07, Warsaw, PL, USA, Jun. 2015.

[14] J. Chen, W.-J. Chien, N. Hu, V. Seregin, M. Karczewicz, and X. Li, *Enhanced Motion Vector Difference Coding*, document JVET-D0123 ITU-T/ISO/IEC, Joint Video Exploration Team, Chengdu, China, Oct. 2016.

[15] A. Henkel *et al.*, *Non-CE4: Switched Half-pel Interpolation Filter*, document JVET-N0309 ITU-T/ISO/IEC, Joint Video Experts Team, Geneva, China, Mar. 2019.

[16] A. Henkel *et al.*, "Alternative half-sample interpolation filters for versatile video coding," in *Proc. ICASSP*, Barcelona, Spain, May 2020, pp. 2053–2057.

[17] H. Liu, L. Zhang, and K. Zhang, *CE2: Adaptive Motion Vector Resolution for Affine Inter Mode (Test 2.1.2)*, document JVET-M0246 ITU-T/ISO/IEC, Joint Video Experts Team, Jan. 2019.

[18] H. Liu *et al.*, "Adaptive motion vector resolution for affine-inter mode coding," in *Proc. Picture Coding Symp. (PCS)*, Ningbo, China, Nov. 2019, pp. 1–4, doi: 10.1109/PCS48520.2019.8954531.

[19] S. Jeong *et al.*, "Merge mode with motion vector difference," in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, Abu Dhabi, United Arab Emirates, Oct. 2020, pp. 1157–1160.

[20] S. Jeong, M. W. Park, Y. Piao, M. Park, and K. Choi, *CE4 Ultimate Motion Vector Expression (Test 4.5.4)*, document JVET-L0054 ITU-T/ISO/IEC, Joint Video Experts Team, Macao, China, Oct. 2018.

[21] H. Liu *et al.*, *AHG11: MMVD without Fractional Distances for SCC*, document JVET-M0255 ITU-T/ISO/IEC, Joint Video Experts Team, Marrakech, MA, Jan. 2019.

[22] M. Bläser *et al.*, "Low-complexity geometric inter-prediction for versatile video coding," in *Proc. Picture Coding Sym. (PCS)*, Ningbo, China, 2019, pp. 1–5.

[23] H. Gao *et al.*, "Advanced geometric-based inter prediction for versatile video coding," in *Proc. Data Compress. Conf. (DCC)*, Snowbird, UT, USA, Mar. 2020, pp. 93–102.

[24] H. Gao, S. Esenlik, E. Alshina, and E. Steinbach, "Geometric partitioning mode in versatile video coding: Algorithm review and analysis," *IEEE Trans. Circuits Syst. Video Technol.*, early access, Nov. 24, 2020, doi: 10.1109/TCSVT.2020.3040291.

[25] M.-S. Chiang, C.-W. Hsu, Y.-W. Huang, and S.-M. Lei, *CE10.1.1: Multi-hypothesis Prediction for Improving AMVP Mode, Skip or Merge Mode, and Intra Mode*, document JVET-L0100 ITU-T/ISO/IEC, Joint Video Experts Team, Oct. 2018.

[26] L. Pham Van, G. Van der Auwera, A. K. Ramasubramonian, V. Seregin, and M. Karczewicz, *CE10: CIIP With Position-Independent Weights (Test CE10-1.1)*, document JVET-N0302 ITU-T/ISO/IEC, Joint Video Experts Team, Mar. 2019.

[27] H. Huang *et al.*, *AHG16: Merge Estimation Region With Constraint in HMVP Update*, document JVET-Q0297 ITU-T/ISO/IEC, Joint Video Experts Team, Jan. 2020.

[28] C.-C. Chen, X. Xiu, Y. He, and Y. Ye, "Generalized bi-prediction method for future video coding," in *Proc. Picture Coding Symp. (PCS)*, Dec. 2016, pp. 1–5.

[29] Y.-C. Su *et al.*, *CE4-related: Generalized Bi-prediction Improvements Combined from JVET-L0197 and JVET-L0296*, document JVET-L0646 ITU-T/ISO/IEC, Joint Video Experts Team, Oct. 2018.

[30] N. Park, J. Nam, H. Jang, J. Lim, and S. Kim, *Non-CE4: Simplifications on BCW Index Derivation Process*, document JVET-O0366 ITU-T/ISO/IEC, Joint Video Experts Team Jul. 2019.

[31] F. Bossen, K. Misra, and A. Segall, *Non-CE4: On Temporal Motion Buffer Compression*, document JVET-M0512 ITU-T/ISO/IEC, Joint Video Experts Team, Jan. 2019.

[32] G. Laroche, J. Jung, and B. Pesquet-Popescu, "RD optimized coding for motion vector predictor selection," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 18, no. 9, pp. 1247–1257, Sep. 2008.

[33] P. Helle *et al.*, "Block merging for quadtree-based partitioning in HEVC," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 22, no. 12, pp. 1720–1731, Dec. 2012.

[34] K. Andersson, *Combined Intra Inter Prediction Coding Mode*, document ITU-T SG.16 Q.6 VCEG-AD11, Oct. 2006.

[35] K. Zhang, S. Ma, D. Zhao, and W. Gao, "Directional residue prediction with motion alignment for video coding," *Electron. Lett.*, vol. 44, no. 19, pp. 1124–1126, Sep. 2008.

[36] K. Zhang, L. Zhang, W.-J. Chien, and M. Karczewicz, "Intra-prediction mode propagation for video coding," *IEEE J. Emerg. Sel. Topics Circuits Syst.*, vol. 9, no. 1, pp. 110–121, Mar. 2019.

[37] O. Divorra Escoda, P. Yin, C. Dai, and X. Li, "Geometry-adaptive block partitioning for video coding," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, Apr. 2007, vol. 1, pp. 657–660.

[38] M. Karczewicz *et al.*, "A hybrid video coder based on extended macroblock sizes, improved interpolation, and flexible motion representation," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 20, no. 12, pp. 1698–1708, Dec. 2010.

[39] X. Li, J. Sole, and M. Karczewicz, *Adaptive MV Precision for Screen Content Coding*, document JCTVC-P0283, Joint Collaborative Team on Video Coding, Jan. 2014.

[40] F. Bossen, J. Boyce, X. Li, V. Seregin, and K. Sühring, *JVET Common Test Conditions and Software Reference Configurations for SDR Video*, document JVET-N1010, Geneva, China, Mar. 2019.

[41] G. Bjøntegaard, *Calculation of Average PSNR Differences Between RD-Curves*, document VCEG-M33 ITU-T VCEG Meeting, Austin, TX, USA, 2001.

[42] *Versatile Video Coding Test Model (VTM) 9.0*. Accessed: Jun. 2020. [Online]. Available: https://vcgit.hhi.fraunhofer.de/jvet/VVCSoftware_VTM

[43] W.-J. Chen *et al.*, *JVET AHG Report: Tool Reporting Procedure and Testing (AHG13)*, document JVET-S0013 ITU-T/ISO/IEC, Joint Video Experts Team, Jun. 2020.

[44] Y.-L. Hsiao *et al.*, *AHG16: On Merge Estimation Region for VVC*, document JVET-Q0185 ITU-T/ISO/IEC, Joint Video Experts Team, Jan. 2020.

[45] E. Alshina *et al.*, *Exploration Experiments on Coding Tools Report*, document JVET-D0010 ITU-T/ISO/IEC, Joint Video Experts Team, Oct. 2016.

Wei-Jung Chien (Member, IEEE) received the B.S. and M.S. degrees in electronics engineering from the National Chiao Tung University, Hsinchu, Taiwan, in 1995 and 1999, respectively, and the Ph.D. degree in electrical engineering from Arizona State University, Tempe, AZ, USA, in 2009. Since 2008, he has been actively involved in the development of HEVC and VVC standard that was jointly issued by ITU-T VCEG and ISO/IEC MPEG. Since 2009, he has been a Senior Engineer with Qualcomm Technologies Inc., San Diego, CA, USA. He is currently the director of engineering. His current research interests include areas of image and video processing and image and video coding.

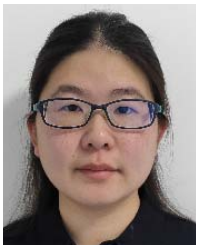
Li Zhang (Senior Member, IEEE) received the Ph.D. degree in computer science from the Institute of Computing Technology, Chinese Academy of Sciences, Beijing, China, in 2009.

From 2009 to 2011, she held a post-doctoral position at the Institute of Digital Media, Peking University, Beijing. From 2011 to 2018, she was a Senior Staff Engineer with Multimedia R&D and Standards Group, Qualcomm, Inc., San Diego, CA, USA. She was a Software Coordinator for Audio and Video Coding Standard (AVS) and the 3D extensions of High Efficiency Video Coding (HEVC). She has been an Active Contributor to the Versatile Video Coding (VVC), Advanced AVS, the IEEE 1857, 3D Video (3DV) coding extensions of H.264/AVC, HEVC, and HEVC screen content coding extensions. During the development of those video coding standards, she co-chaired several *ad hoc* groups and core experiments. She is currently the Head of Advanced Video Group, Bytedance Inc., San Diego. She has authored more than 450 standardization contributions, more than 170 granted U.S. patents, and more than 70 technical articles in related book chapters, journals, and proceedings in image/video coding and video processing. Her research interests include 2D/3D image/video coding, video processing, and transmission. She has been appointed as an Editor of AVS and the Main Editor of the Software Test Model for 3DV Standards.

Martin Winken received the Dipl.-Ing. and Dr.-Ing. degrees from the Technical University of Berlin, Germany, in 2006 and 2015, respectively. Since 2005, he has been active in video coding standardization. He is currently a Research Associate with Fraunhofer HHI, Berlin.

Xiang Li (Senior Member, IEEE) received the B.Sc. and M.Sc. degrees in electronic engineering from Tsinghua University, Beijing, China, and the Dr.-Ing. degree in electrical, electronic and communication engineering from the University of Erlangen-Nuremberg, Germany.

He is currently a Senior Principal Researcher and the Head of video coding standards at Tencent's Media Lab. Before joining Tencent, he was with Qualcomm, MediaTek, the Institute of Communications Engineering, RWTH Aachen University, and Siemens. He has been working in the field of video compression for years. He is an active contributor to international video coding standards. He served as the chair and the co-chair in a number of *ad hoc* groups, core experiments, including the Co-Chair of JEM Reference Software and VVC Reference Software. He has published over 50 journals and conference papers, more than 300 standard contributions, and holds more than 240 U.S. granted and pending patents. His research interests include video coding and processing. He is a Co-Editor of *MPEG-5 EVC*.



Ru-Ling Liao received the B.S. and M.S. degrees in computer science and engineering from the National Chao Tung University, Hsinchu, Taiwan, in 2013 and 2016, respectively.

She has been participated in the development of VVC standard since 2016. After 2019, she joined XG Laboratory, Alibaba Inc., Beijing, China. She is currently a senior engineer. Her research interests include video processing and neural network-based video compression technology.

Han Gao (Graduate Student Member, IEEE) received the B.Sc. degree in electrical engineering from Xidian University, Xi'an, China, in 2012, and the M.Sc. degree in electrical engineering and information technology from the Technical University of Munich (TUM), Munich, Germany, in 2015, where he is currently pursuing the Ph.D. degree. In 2016, he joined the Chair of Media Technology, TUM, and Audiovisual Technology Laboratory, Huawei Technologies, Munich. Since 2016, he has been actively involved in the development of the VVC standard that was jointly issued by ITU-T VCEG and ISO/IEC MPEG. His research focuses on image and video processing, traditional and neural network-based video compression, and coding.

Chih-Wei Hsu received the B.S. degree in electrical engineering and the M.S. degree in electronics engineering from the National Taiwan University (NTU), Taipei, Taiwan, in June 2001 and June 2003, respectively.

He joined MediaTek Inc., Hsinchu, Taiwan, in October 2003, and is currently a Senior Manager of the Multimedia Technology Development Division. From 2003 to 2010, he researched and developed video encoding algorithms, hardware architectures, and related IC designs for consumer electronics products. He started attending international video coding standard meetings held by ITU-T VCEG and ISO/IEC MPEG in 2011 and is an active contributor. During the development of VVC, he has made several technical contributions and was the coordinator of the core experiments on combined and multi-hypothesis prediction. His current research interests include image and video coding, image and video processing, and related hardware architectures.

Hongbin Liu received the B.S., M.S., and Ph.D. degrees in computer science and technology from Harbin Institute of Technology, Harbin, China, in 2005, 2007, and 2011, respectively.

He is currently a Video Coding Engineer at Bytedance Inc., Beijing. He was actively participating in the research of 3D-HEVC and VVC standards. His current research interest includes video coding and processing.

Chun-Chi Chen received the Ph.D. degree in computer science and engineering from the National Chao Tung University, Hsinchu, Taiwan, in 2017. Since 2011, he has been participating in the video coding standardization process of ISO/IEC MPEG. In 2014–2015, he has been an active contributor to HEVC-SCC and has co-chaired several core experiments. During his internship with InterDigital, San Diego, in 2015–2016, he initiated works on generalized bi-prediction which has been included as a coding tool in VVC. After 2018, he focused on inter-prediction techniques and conducted quarterly service works as a core experiment coordinator of VVC. He is currently a Senior Engineer with Multimedia R&D and Standards Group, Qualcomm Technologies, Inc., San Diego, CA, USA. He has authored/coauthored more than 70 technical papers and video standards contributions. His research interests include image/video compression and screen content coding.