Advanced Geometric-based Inter Prediction for Versatile Video Coding

Han Gao[*], Ru-Ling Liao[†], Kevin Reuzé[‡],
Semih Esenlik[*], Elena Alshina[*], Yan Ye[†], Jie Chen[†], Jiancong Luo[†],
Chun-Chi Chen[‡], Han Huang[‡], Wei-Jung Chien[‡], Vadim Seregin[‡], Marta Karczewicz[‡]

[*]*Huawei Technologies, Munich, Germany*
[†]*Alibaba Group, Beijing, China*
[‡]*Qualcomm Inc, San Diego, U.S.A*

Abstract

Block-based partitioning is one of the fundamental techniques in video coding. Geometric-based block partitioning is a well-studied method to enable better spatial adaptation to the signal properties. This paper introduces the most recent proposal of advanced geometric-based inter prediction (GIP) made to the state-of-the-art are video coding standard - Versatile Video Coding (VVC). Implemented in the latest test model VTM-6.0 to generalize the existing triangle partition mode (TPM) and evaluated with the Joint Video Experts Team (JVET) Common Test Conditions (CTC) sequences, the proposed advanced GIP scheme provides luma BD-rate reduction of **0.56** % for random access (RA) and **1.37** % for low-delay (LB) test cases with **2** % encoder runtime increase and negligible decoder runtime increase. Furthermore, BD-rate reductions up to **2.92** % and **3.49** % for RA and LB test cases can be achieved in the absence of multiple related VVC inter prediction tools.

1 Introduction

A block-based motion compensation is the primary method for temporal prediction in modern video coding standards. While Advanced Video Coding (AVC) [1] uses square-shaped blocks of size 8×8 or 16×16 luma samples, High Efficiency Video Coding (HEVC) [2] introduces a quad-tree partitioning structure. In combination with Asymmetric Motion Partitioning (AMP) [3], this offers more flexibility in segmenting a picture and contributes significantly to the improved coding efficiency over AVC.

Versatile Video Coding (VVC) is one of the state-of-the-art video coding standards in development and allows even higher partitioning flexibility. In VVC, the quad-tree is recursively combined with binary- and ternary-tree structures. The nested partitioning structure in VVC results in rectangular coding blocks of variable sizes that are used as the fundamental processing units of the codec. Each coding block can be coded using different block tools, signaled as coding modes to the decoder. For inter-picture prediction in particular, a coding block can be split along the main diagonals into two triangles, termed the triangular partition mode (TPM) [4], in order to allow a better adaptation to the spatial properties of motion object. However, it can

2375-0359/20/$31.00 ©2020 IEEE
DOI 10.1109/DCC47342.2020.00017

be observed that TPM still requires fine-grained rectangular partitions to approximate an object boundary sufficiently.

In this paper, we present an advanced geometric-based inter prediction (GIP) by introducing a total number of 82 geometrically splitting options for a rectangular block in VVC. Different as in TPM, the splitting of the proposed GIP is performed along a geometrically located straight line. The location of the GIP splitting line is mathematically derived from the angle and offset parameters of a specific partition. Each resulting partition is predicted through motion compensation. The two predicted partitions are combined using a blending filter, which performs a weighted averaging of the two predictions based on the distance of each sample to the split boundary. Unlike previous schemes of geometric-based block partitioning coding tool, our proposed advanced GIP is of low complexity by increasing the average encoder runtime by roughly 2 % and adding negligible additional decoder complexity. Furthermore, the proposed scheme significant reduced the memory requirements for a pre-stored blending filter implementation. Advanced GIP was proposed to JVET [5, 6] (termed *GEO*) and will be further investigated in the VVC Core Experiment on inter prediction coding technology [7].

2 Related Works

2.1 Geometric-based Coding Tool

Geometric-based block partitioning has been studied previously as a block partitioning coding tool in the context of AVC [8, 9] and during the initial standardization phase of HEVC [10, 11]. In the last unified proposal [12] for HEVC, geometric-based block partitioning provided 0.7 % of coding efficiency improvement at 17 % encoder complexity increase, utilizing 6 additional geometric-based splits compared to HM3.0. For the recent Joint Call for Proposals on Video Compression with Capabilities beyond HEVC [13], geometric-based block partitioning was again proposed on-top of the quad-tree binary-tree (QTBT) structure of JEM-7.0 [14] and was reported to provide 0.82 % of coding gain, albeit at significantly higher encoder complexity, measured at 380 % [15] as no restriction on the number of geometric splits was applied.

In the recent VVC standardization activity, geometric-based inter prediction was proposed in [16–18] as an inter prediction coding tool. The geometric-based inter prediction provides 0.22 % coding gain with 7 % encoding time increase and reported interesting subjective improvement as a better fitting of the motion objects boundaries. However, the tool is still hardly used in practice due to the large total number of splits, the memory requirement of blending filter storage and the latency between blending filter and motion storage.

2.2 Triangular Partition Mode in VVC

TPM is currently part of the working draft for VVC [4]. TPM is an inter prediction coding tool that operates using two uni-directional motion vectors, which are signaled by index values, mapping to a merge candidate list. For each motion vector, a block-based motion compensation step is performed, resulting in two intermediate prediction blocks P_i of size $w \times h$ samples with $i = 0, 1$. A final prediction block P_B

is generated by performing a blending process using integer weighting matrices W_i, containing weights in the value range of 0 to 8. This can be expressed as

$$P_{\mathrm{B}} = (W_0 \circ P_0 + W_1 \circ P_1 + 4) >> 3 \tag{1}$$

with $W_0 + W_1 = 8J_{h,w}$, where J is a matrix of ones. The integer weights of W_0 for TPM are given by the shortest Manhattan distance of each sample to the samples positioned along the main diagonals, clipped to the value range of 0 to 8 as $W_0 = \mathrm{clip}(0, 8, w_{\mathrm{TPM},0} + 4)$. For the triangle split extending from the top-left to the bottom-right sample of a block, the distance can be expressed as

$$w_{\mathrm{TPM},0}(x, y) = \frac{x}{a} - \frac{y}{b} \tag{2}$$

with two constants a and b, which depend on the aspect ratio of the coding block.

2.3 Wedge-based Prediction in AV1

The emerging open-source video compression format AV1, finalized in 2018 by the Alliance for Open Media (AOMedia) industry consortium, specifies a prediction mode named *Advanced Compound Prediction* and further distinguishes several sub-modes [19,20]. One of the inter prediction sub-modes is termed *Compound Wedge Prediction*, which defines a codebook of 16 possible wedge partitions. The codebook contains wedges that are oriented either horizontally, vertically, or oblique with slopes of ± 2 or ± 0.5, depending on the block shapes. Similar to TPM and the proposed method in this paper, two predictions are averaged using pre-computed blending weights to generate a combined predictor.

3 Advanced Geometric-based Inter Prediction

The advanced GIP mode is implemented on top of VTM-6.0 to enhance and generalize the existing TPM, such that motion vector merging and coding in both block tools are performed identically. GIP is signaled at the bottom of the merge syntax tree in the same position as TPM without additional sytnax element. In total 82 splits are supported by GIP for each possible block size $w \times h = 2^m \times 2^n$ (in terms of luma samples) with $m, n \in \{3 \dots 7\}$. The symmetric horizontal and vertical splits are excluded as they are identical to binary-tree splits. An index of GIP split mode $S_i \in \{0 \dots 81\}$ is determined by the encoder and explicitly signaled to the decoder using truncated binary coding.

3.1 Split Boundary Definition

The split boundary of the proposed GIP mode is defined as a straight line equation parameterized using an angle φ and an offset ρ based on polar coordinate system. The distance between a certain sample position and the split boundary is given in Equation (3) with x_{c} and y_{c} defining sample positions relating to the central position of the current block.

$$d(x_{\mathrm{c}}, y_{\mathrm{c}}) = x_{\mathrm{c}} \cos(\varphi) + y_{\mathrm{c}} \sin(\varphi) - \rho \tag{3}$$

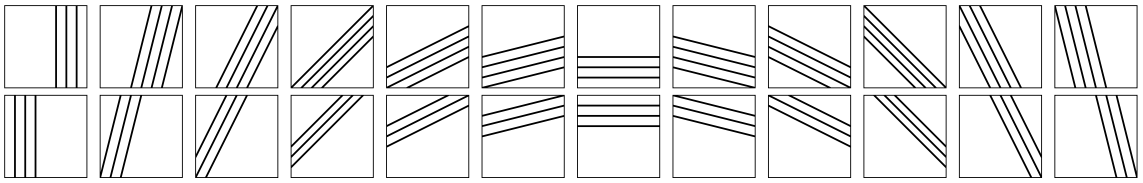

Figure 1: Examples of the proposed GIP splits grouped by identical angles φ_i.

To simplify the implementation, Equation (4) is derived from Equation (3) to express the distance using a sample position related to the top-left to the current block with a given size $w \times h$. Note that in the actual software implementation, an integer approximation to Equation (4) is utilized, with a scaled cosine look-up table of 3-bit precision. Including the multiplication factor of x and y in Equation (4). The total precision of $d(x, y)$ is 4 bits.

$$d(x,y) = (2x + 1 - w)\ \cos(\varphi) + (2y + 1 - h)\ \sin(\varphi) - \rho \tag{4}$$

3.2 Quantized Angle φ and Offset ρ

The values for the angle φ and the offset ρ are quantized, which has been designed to equally space the 82 geometric splits for a given block size.

The angle φ is quantized into φ_j using a pre-defined look-up table with $j \in \{0 \dots 23\}$. φ_j is non-uniformly divided 2π with fixed $\tan(\varphi_j)$ values, where $\tan(\varphi_j) \in \{0, \pm 1/4, \pm 1/2, \pm 1, \pm 2, \pm 4, \infty\}$. With such an angle φ quantization scheme, $\sin(\varphi_j)$ and $\cos(\varphi_j)$ ($\tan(\varphi_j)$ is termed as ${\sin(\varphi_j)}/{\cos(\varphi_j)}$) are quantized as shift values (n^2, $n \in \mathbb{N}_0$), the multiplications in Equation (4) are easily replaced by shift operations in pixel level.

The offset ρ is quantized into ρ_k with $k \in \{0 \dots 3\}$. In order to avoid unequally distributed split line offsets between different block sizes, the ρ_k is firstly factorized with Equation (5). The $\rho_{x,k}$ and $\rho_{y,k}$ are further coupled with the given block sizes $w \times h$ using Equation (6) and (7), where k is the index of the quantized offset ρ_k and j is the index of the quantized angle φ_j. The sign of $\rho_{x,k}$ and $\rho_{y,k}$ in Equation (6) and (7) are set as negative if $j < 12$, otherwise positive.

$$\rho_k = \rho_{x,k}\ \cos(\varphi_j) + \rho_{y,k}\ \sin(\varphi_j) \tag{5}$$

$$\rho_{x,k} = \begin{cases} 0 & j\%12 = 6 \text{ or } (j\%12 \neq 0 \text{ and } h \geq w) \\ \pm(k \times w) \gg 2 & \text{otherwise} \end{cases} \tag{6}$$

$$\rho_{y,k} = \begin{cases} \pm(k \times h) \gg 2 & j\%12 = 6 \text{ or } (j\%12 \neq 0 \text{ and } h \geq w) \\ 0 & \text{otherwise} \end{cases} \tag{7}$$

The quantized GIP splits are visualized in Figure 1, grouped by identical angles φ_j. The total number of GIP splits N_{GIP} can be calculated as $N_{\text{GIP}} = N_\varphi N_\rho - {N_\varphi}/2 - 2 = 82$ with $N_\varphi = 24$ and $N_\rho = 4$, since symmetric horizontal and vertical binary splits are excluded to avoid the emulation of regular binary-tree block partitioning.

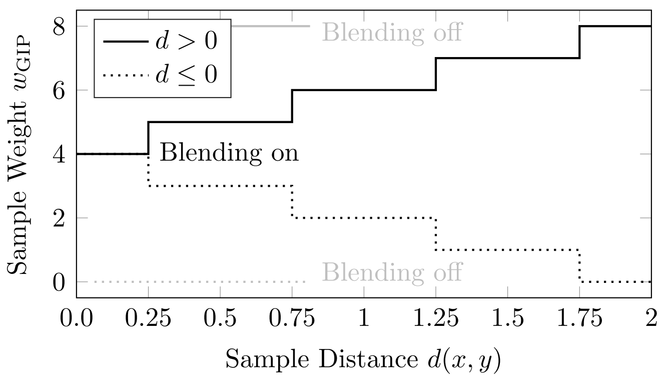

Figure 2: Mapping function $f_{\text{B}}(\cdot)$ for derivation of blending weights $w_{\text{GIP},0}(x, y)$.

3.3 Advanced GIP Blending Process

During the GIP motion compensation process, blending filters in the form of weighting matrices W_i containing weights of 3-bit precision with $i = 0, 1$ are used to generate a final combined prediction block P_{B} as shown in Equation (1). Since the split boundaries are geometric, the integer blending weights w_{GIP} are derived through

$$w_{\text{GIP},0}(x, y) = f_{\text{B}}(d(x, y)) \tag{8}$$

using the distance $d(x, y)$ and an integer quantization function $f_{\text{B}}(\cdot)$, which is illustrated in Figure 2. One example of a generated GIP blending matrix is shown in Figure 3.

It's noted that the proposed GIP blending weights $w_{\text{GIP},0}(x, y)$ can be pre-computed and pre-stored to further reduce on-the-fly computational complexity. Based on Equation (5)-(7), the blending filters can be reused between different block sizes for a given angle φ_j as the quantized ρ_k is independent of sample position (x, y). Furthermore, it's shown in Equation (9) that the blending weights are repeated row-by-row by a phase shift when the $\tan(\varphi_j)$ is an integer value. In the case of $\tan(\varphi_j)$ smaller than 1, a $\cot(\varphi_j)$ based column-by-column phase shift is achievable with $w_{\text{GIP},0}(x, y) = w_{\text{GIP},0}(x - 1, y - \cot(\varphi_j))$. It's reported in [5, 6] that the memory requirement for the pre-stored blending weights of the proposed GIP reduced from around 3.2MBytes to 144Bytes compared with the previous design in [16–18].

$$\begin{aligned} w_{\text{GIP},0}(x, y) &= f_{\text{B}}(d(x, y) + \cos(\varphi_j)\ (2\ \tan(\varphi_j) - 2\ \tan(\varphi_j))) \\ &= f_{\text{B}}(d(x, y) + 2\ \cos(\varphi_j)\ \tan(\varphi_j) - 2\ \sin(\varphi_j)) \\ &= f_{\text{B}}((2(x - \tan(\varphi_j)) + 1 - w)\ \cos(\varphi_j) + (2(y - 1) + 1 - h)\ \sin(\varphi_j) - \rho) \\ &= f_{\text{B}}(d(x - \tan(\varphi_j), y - 1)) \\ &= w_{\text{GIP},0}(x - \tan(\varphi_j), y - 1) \end{aligned} \tag{9}$$

3.4 Latency-free GIP Motion Storage

In VVC, the motion information is stored in units equivalent to 4×4 luma samples. The motion information is used for further motion prediction and deblocking.

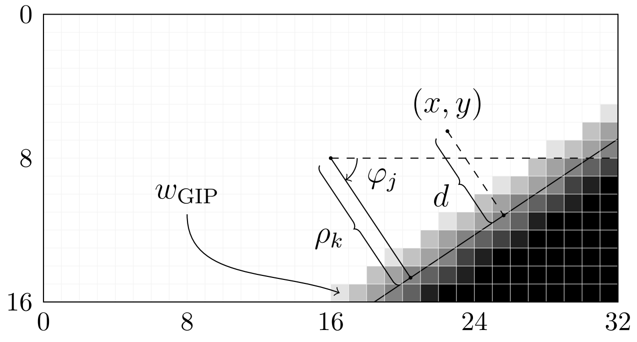

Figure 3: Exemplified generation of a blending filter W_0 using proposed GIP mode.

To simplify the intermediate motion compensation of GIP and to avoid additional memory bandwidth, only uni-directional motion vectors are used for the GIP motion compensation process. During the GIP motion information storage, the distance between the central position of each 4×4 samples cell and the split boundary $d(x + 1.5, y + 1.5)$ is re-calculated using Equation (4), where (x, y) is the top-left sample position of 4×4 samples cell. Similar to the blending process, if the absolute value of $d(x + 1.5, y + 1.5)$ is larger than a pre-determined threshold, the respective uni-directional motion information MV_0 or MV_1 is stored for the 4×4 samples cell. Otherwise, a bi-directional motion information MV_{Bi} is assigned, which is a combination of the two uni-directional motion vectors and reference indices. It's noted that the motion storage type (either uni- or bi-directional MVs to be stored) is easily pre-stored in a form of motion storage type map for each 4×4 samples cell by involving the same features mentioned in Section 3.3 to further reduce the on-the-fly computational complexity. The proposed GIP motion storage is latency-free and independent from the blending weights, since the motion storage type is either re-calculated or pre-stored.

3.5 Encoder Search

TPM in VTM-6.0 uses a maximum number of 5 uni-directional motion vectors derived from a merge candidate list for each triangular partition. Up to 40 TPM candidates are exhaustively tested. For GIP, the same uni-directional motion vectors are inherited, resulting in a total of $82 \times 5 \times 4 = 1640$ potential GIP candidates, which is a prohibitively large number for performing a full RD-based search on the encoder side. To reduce the computational complexity of the encoder, the best matching GIP candidate is determined using the following three stages:

Stage 1: For each of the maximum of 5 uni-directional motion vectors, full-block motion-compensated prediction (MCP) is performed. The sum-of-absolute differences (SAD) between the compensated prediction and the original block is calculated as $\text{SAD}_{\text{Block},i}$ with $i = 0 \ldots 4$. Identical entries of the merge candidate list are excluded in this stage. If all motion vectors in the merge candidate list are identical, the entire GIP encoder search is aborted.

Stage 2: For each GIP split mode, the smaller split partition of the 5 full-block predictors are masked out, using a hard blending filter (rounded from regular blending filter using 0 or 8). The SAD of the smaller, masked split partition is computed and denoted as $\mathrm{SAD}_{\mathrm{Small},i,k}$ with $i = 0 \ldots 4$ and $k = 0 \ldots 81$. Since the hard blending filter is used, the SAD of the larger GIP split partition is obtained as:

$$\mathrm{SAD}_{\mathrm{Large},i,k} = \mathrm{SAD}_{\mathrm{Block},i} - \mathrm{SAD}_{\mathrm{Small},i,k} \tag{10}$$

A combined RD-cost $J_{\mathrm{Comb},k}$, including estimated motion information bits, is generated as

$$J_{\mathrm{Comb},k} = \mathrm{SAD}_{\mathrm{Small},m,k} + \mathrm{SAD}_{\mathrm{Large},n,k} + \lambda R_{\mathrm{Motion}}, \tag{11}$$

where m and n are the predictor indices giving the lowest SAD cost for each split partition and $m \neq n$. The combined GIP candidates are sorted by $J_{\mathrm{Comb},k}$. In addition, GIP candidates with $J_{\mathrm{Comb},k} > \mathrm{SAD}_{\mathrm{Block},i} + \lambda R_{\mathrm{Motion}}$ are excluded.

Stage 3: The best 25 (or less) GIP candidates from stage 2 are used to conduct the full-block blending operation according to Equation (1). The sum-of-absolute-transformed differences (SATD) is computed for each P_{B} and the 3 (or less) GIP candidates with lowest SATD cost are further tested using transform coding, sum-of-squared-differences (SSD) and CABAC-based rate estimation. Finally, the GIP candidate with the lowest overall RD-cost is selected as the determined GIP mode.

4 Experimental Results

The proposed advanced GIP coding tool has been tested and evaluated according to the JVET Common Test Conditions (CTC) [21] on a variety of different video sequences. The coding results and relative complexity measurements against the VTM-6.0 (TPM off) anchor are shown in Tables 1 and 2. In the experiment, Bjøntegaard-Delta over the bitrate (BD-rate) saving [22] is used for measuring the performance. The BD-rate saving gives the bit-rate reduction (in %) for the identical level of peak signal-to-noise ratio (PSNR). Further, the software and coding results presented in [5,6] were cross-checked according to JVET practices. It can be reported that for a random access (RA) coding configuration, average luma BD-rate reduction of 0.56 % can be achieved. The best improvements in coding efficiency for natural video sequences are observed for class C, which contains the two sequences *RaceHorsesC* and *BQMall*, where BD-rate reduction of 1.35 % and 1.87 % can be reported. These two

Table 1: Random access coding results according to JVET-CTC

Class	BD-rate change			Rel. complexity	
	Y	U	V	EncT	DecT
A1	-0.27 %	-0.53 %	-0.40 %	102 %	99 %
A2	-0.41 %	-0.60 %	-0.51 %	102 %	99 %
B	-0.34 %	-0.51 %	-0.49 %	103 %	100 %
C	-1.17 %	-1.83 %	-1.60 %	102 %	101 %
Overall	**-0.56 %**	**-0.89 %**	**-0.77 %**	**102 %**	**100 %**
D	-0.68 %	-1.37 %	-1.20 %	103 %	101 %
F	-0.55 %	-0.72 %	-0.61 %	100 %	101 %

Table 2: Low-delay coding results according to JVET-CTC

Class	**BD-rate change**			**Rel. complexity**	
	Y	U	V	EncT	DecT
B	-0.83 %	-0.96 %	-1.00 %	103 %	99 %
C	-1.71 %	-2.34 %	-2.47 %	103 %	101 %
E	-1.81 %	-2.33 %	-1.62 %	101 %	99 %
Overall	**-1.37 %**	**-1.76 %**	**-1.65 %**	**102 %**	**100 %**
D	-1.50 %	-1.79 %	-1.55 %	103 %	100 %
F	-0.91 %	-1.77 %	-1.27 %	101 %	100 %

sequences contain clearly distinctive object boundaries as overlaid visual example in Figuer 5, where GIP is more frequently used. For a low-delay (LB) coding configuration, average luma BD-rate reduction of 1.37 % are reported. The average encoder runtime increased, depending on the configuration, to 101-103 % against the VTM-6.0 (TPM off) reference. The average decoder runtime also increased marginally to 99-101 %. The proposed method also has been tested with same sequences in absence of multiple related VVC inter-prediction coding tools (all merge related tools except regular merge) compared with VTM-6.0 (related tool off) anchor. Simulation results show that average BD-rate reduction of GIP is 0.98% and 2.08 % for RA and LB configurations respectively. For certain sequences the coding-performance up to 2.92% for RA and 3.94% for LB are achieved.

As shown in Figure 4, roughly 2.3 % of all pixels are coded using TPM in the VTM-6.0 reference, averaged over all sequences and quantizer parameters (QPs) for the RA case. This compares to 3.7 % for GIP in this proposal on top of VTM-6.0(TPM off). By tendency, GIP is selected by the encoder more often for larger QPs, which was measured at 4.1 % for QP37. For the aforementioned examples *RaceHorsesC* and *BQMall*, the GIP mode usage was measured at 5.1 % and 3.2 % averaged overall QPs, which are among the highest usage values for all sequences.

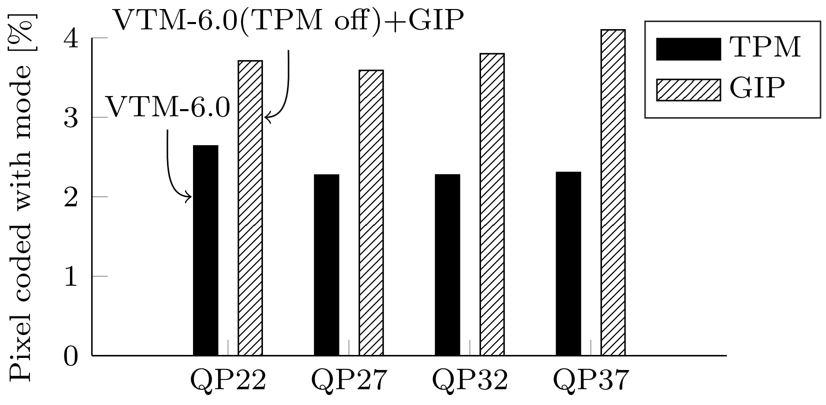

Figure 4: Percentage of pixels coded by TPM vs GIP mode for RA configuration.

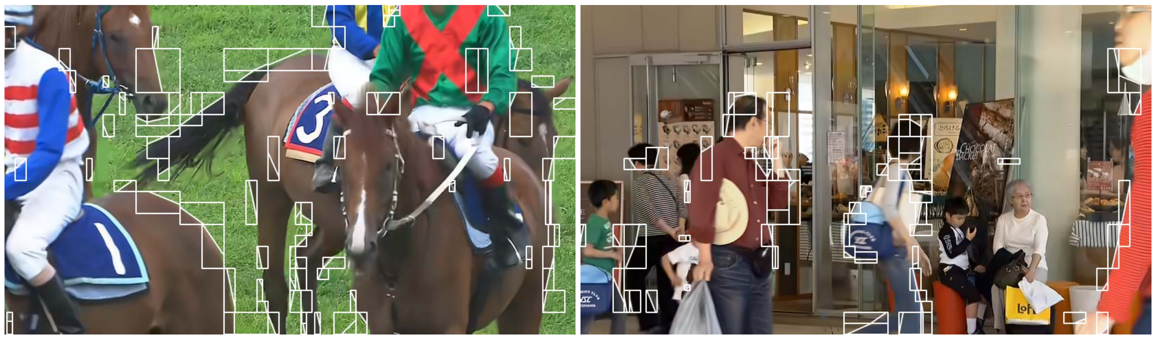

Figure 5: Overlaid visual examples of the proposed GIP for the *RaceHorsesC* and *BQMall* sequences, coded at QP37 in RA configuration.

5 Conclusion

In this paper, an advanced geometric-based inter prediction mode is introduced for VVC, which increased coding efficiency by 0.56 % and 1.37 % in terms of luma BD-rate reduction on average against VTM-6.0 (TPM off) anchor for RA and LB configurations respectively. For specific sequence and configuration, up to 3.49 % luma BD-rate redaction is achieved. The existing TPM mode is extended by 82 geometrically distributed split modes, which allows for more efficient coding of motion objects with oblique or curved boundaries using larger coding blocks. Due to shift values-based angles quantization, independent offset parameters design, latency-free motion storage and a staged estimation of the optimal GIP mode, encoder runtime is increased by only around 2 % with negligible decoding time change.

References

[1] T. Wiegand, G. J. Sullivan, G. Bjøntegaard, and A. Luthra, "Overview of the H.264/AVC Video Coding Standard," *IEEE Trans. Cir. and Sys. for Video Technol.*, vol. 13, no. 7, pp. 560–576, July 2003.

[2] G. J. Sullivan, J. Ohm, W. Han, and T. Wiegand, "Overview of the High Efficiency Video Coding (HEVC) Standard," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 22, no. 12, pp. 1649–1668, Dec. 2012.

[3] V. Sze, M. Budagavi, and G. J. Sullivan, "High Efficiency Video Coding (HEVC)," *Integrated Circuit and Systems, Algorithms and Architectures. Springer*, vol. 39, pp. 49–90, 2014.

[4] B. Bross, J. Chen, S. Liu, and Y.-K. Wang, "Versatile Video Coding (Draft 7)," Doc. JVET-P2001, JVET, Geneva, Switzerland, 16th meeting, Oct. 2019.

[5] H. Gao, S. Esenlik, E. Alshina, A. M. Kotra, B. Wang, M. Bläser, and J. Sauer, "Simplified GEO without multiplication and minimum blending mask storage (harmonization of JVET-P0107, JVET-P0264 and JVET-P0304)," Doc. JVET-P0884, JVET, Geneva, Switzerland, 16th meeting, Oct. 2019.

[6] K. Reuzé, C.-C Chen, H. Huang, W.-J Chien, V. Seregin, M. Karczewicz, R.-L Liao, J. Chen, Y. Ye, J. Luo, M. Bläser, and J. Sauer, "Simplified GEO without multiplication and minimum blending mask storage (harmonization of JVET-P0107, JVET-P0264 and JVET-P0304)," Doc. JVET-P0885, Geneva, Switzerland, 16th meeting, Oct. 2019.

[7] C.-C. Chen, R.-L. Liao, X. Xiu, and H. Yang, "Description of core experiment 4 (CE4): Inter prediction with geometric partitioning," Doc. JVET-P2024, JVET, Geneva, Switzerland, 16th meeting, Oct. 2019.

[8] S. Kondo and H. Sasai, "A motion compensation technique using sliced blocks in hybrid video coding," in *IEEE International Conference on Image Processing 2005*, Sept 2005, vol. 2, pp. II–305–8.

[9] O. Divorra Escoda, P. Yin, C. Dai, and X. Li, "Geometry-adaptive block partitioning for video coding," in *2007 IEEE International Conference on Acoustics, Speech and Signal Processing - ICASSP '07*, Apr. 2007, vol. 1, pp. I–657–I–660.

[10] M. Karczewicz, P. Chen, R. L. Joshi, X. Wang, W.-J. Chien, R. Panchal, Y. Reznik, M. Coban, and I. S. Chong, "A hybrid video coder based on extended macroblock sizes, improved interpolation, and flexible motion representation," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 20, no. 12, pp. 1698–1708, Dec. 2010.

[11] P. Bordes, P. Chen, I.-K. Kim, L. Guo, H. Yu, and X. Zheng, "CE2: Unified solution of flexible motion partitioning," Tech. Rep. JCTVC-E374, JCT-VC, Geneva, Switzerland, 5th meeting, Mar. 2011.

[12] X. Zheng, H. Yu, S. Li, Y. He, and P. Bordes, "CE2: Non-rectangular motion partitioning," Tech. Rep. JCTVC-F415, JCT-VC, Torino, Itlay, 6th meeting, July 2011.

[13] A. Segall, V. Baroncini, J. Boyce, J. Chen, and T. Suzuki, "Joint Call for Proposals on Video Compression with Capability beyond HEVC," Tech. Rep. JVET-H1002, Macao, CN, 8th meeting, Oct. 2017.

[14] M. Bläser, J. Sauer, and M. Wien, "Description of SDR and 360° video coding technology proposal by RWTH Aachen University," Tech. Rep. JVET-J0023, San Diego, USA, 10th meeting, Apr. 2018.

[15] M. Bläser, J. Schneider, J. Sauer, and M. Wien, "Geometry-based Partitioning for Predictive Video Coding with Transform Adaptation," in *2018 Picture Coding Symposium (PCS)*, June 2018, pp. 134–138.

[16] S. Esenlik, H. Gao, A. Filippov, V. Rufitskiy, A. M. Kotra, B. Wang, E. Alshina, M. Bläser, and J. Sauer, "Non-CE4: Geometrical partitioning for inter blocks," Doc. JVET-O0489, JVET, Gothenburg, Sweden, 15th meeting, July 2019.

[17] S. Esenlik, H. Gao, B. Wang, A. M. Kotra, Z. Zhao, E. Alshina, M. Bläser, and J. Sauer, "Non-CE4: Adaptive blending filtering for TPM," Doc. JVET-O0513, JVET, Gothenburg, Sweden, 15th meeting, July 2019.

[18] M. Bläse, H. Gao, S. Esenlik, E. Alshina, Z. Zhao, C. Rohlfing, and E. Steinbach, "Low-complexity geometric inter-prediction for versatile video coding," in *2019 Picture Coding Symposium (PCS)*, IEEE, Ed., Ningbo, 2019.

[19] O. de Rivaz and J. Haughton, "AV1 Bitstream & Decoding Process Specification," Tech. Rep., The Alliance for Open Media, 2019.

[20] Y. Chen, D. Murherjee, J. Han, A. Grange, Y. Xu, Z. Liu, S. Parker, C. Chen, H. Su, U. Joshi, C. Chiang, Y. Wang, P. Wilkins, J. Bankoski, L. Trudeau, N. Egge, J. Valin, T. Davies, S. Midtskogen, A. Norkin, and P. de Rivaz, "An Overview of Core Coding Tools in the AV1 Video Codec," in *2018 Picture Coding Symposium (PCS)*, June 2018, pp. 41–45.

[21] F. Bossen, J. Boyce, X. Li, V. Seregin, and K. Sühring, "JVET common test conditions and software reference configurations for SDR video," Doc. JVET-N1010, JVET, Geneva, Switzerland, 14th meeting, Mar. 2019.

[22] G. Bjøntegaard, "Calculation of average PSNR differences between RD-curves," Tech. Rep. VCEG-M33, ITU-T SG16/Q6 VCEG, Austin, USA, 2001.