A general framework for directional intra prediction with varying angle for video coding

Gagan Rath, Fabien Racape, Fabrice Urban, Fabrice Leleannec, Franck Galpin, Karam Naser
InterDigital Inc
975 Avenue des Champs Blancs, 35510 Cesson Sévigné, France
{firstname.lastname}@interdigital.com

Abstract— **Video coding standards such as H.264/AVC, HEVC, VVC, etc., all employ intra prediction where a directional prediction uses a single prediction direction inside a coding block. The underlying idea is that the object directionalities remain unchanged over the entire coding block. This is a simplistic model as the object directionalities over a block can change, especially if the block size is relatively large. In such a case, using a single prediction direction over a coding block will lead to smaller coding blocks after rate-distortion (R-D) optimization with side information for partitioning into smaller blocks. A better idea would be to allow the prediction direction to change gradually over the same coding block such that objects with slow changing directionalities can be modelled in larger blocks without splitting them. This would demand additional complexity and the side information required for model parameters. This paper proposes a general framework which supports this change of prediction direction inside a block and presents practical solutions for low complexity implementations. Simulation results with VVC test model (VTM 4) software show Luma BD-rate gain of 0.11% in All Intra (AI) configuration.**

Keywords—Intra prediction, coding unit, prediction mode, directional modes, prediction direction, VVC

I. Introduction

Intra prediction is a core coding tool in all video compression standards such as H.264/AVC, H.265/HEVC, and H.266/VVC. The basic idea is to exploit the spatial correlation in a frame by predicting a block of pixels based on already decoded causal neighbor blocks. For the prediction purpose, the standards define several models known as prediction modes. HEVC, for example, defines 35 prediction modes where one is a PLANAR mode, one is a DC mode, and the remaining 33 are angular modes. VVC, in addition to the PLANAR and DC modes, defines 65 regular angular modes and 28 wide angular modes. The wide angular modes are used by rectangular blocks, which is supported by VVC. The encoder selects the best prediction mode for a target block after checking the rate-distortion performance with all available modes and then encodes the mode index with a Variable Length Coding (VLC) scheme.

The basic idea behind the defined modes is to model different kinds of image content in a block. The PLANAR and DC modes aim to model slow and gradually changing intensity regions whereas the angular modes aim to model different object directionalities. Thus, in a directional prediction mode, the reference samples on top and left of a block, are repeated along the associated direction inside the block. Here the underlying assumption is that the directionalities of objects remain linear throughout the block and the intensity values do not change much along those directions. This model seems to work well, especially when the block sizes are small. When the blocks are large, as allowed in VVC, the directionality of intensities can gradually change since the content of natural images contains mostly non-straight edges. In this paper, we present a simple method to model this phenomenon. We construct a finite set of models for the change of intensity directionality inside the same block, where the directionalities correspond to adjacent prediction modes. For any directional prediction mode, the encoder tests the rate-distortion (R-D) performance with all available models with the given prediction mode in addition to the existing model of constant directionality and selects the best model. The selected model is encoded with a VLC scheme and signaled to the decoder.

The paper is organized as follows. In section II, we present a brief review of intra prediction in VVC. The interested reader is referred to [1] for details about all intra prediction tools in VVC. Section III introduces the proposed prediction models. We start with a general framework and then present some solutions that can be implemented with VVC Test model (VTM) reference code. We present the simulation results in section IV and then conclude the paper in section V.

II. Intra Prediction in VVC

As in HEVC, the intra prediction process in VVC consists of three steps: (1) reference sample generation, (2) intra sample prediction, and (3) post-processing of predicted samples. We will briefly describe these three steps below. A target block for prediction is called a coding unit (CU) in VVC. For easier reference, we will be using the terms “CU” and “block” interchangeably throughout the text.

A. Reference sample generation

The reference sample generation process is illustrated in Fig. 1. For a CU of size $H \times W$, where H and W denote its height and width, a row of $2W + 1$ decoded samples is formed from the reconstructed top, top-right, and top-left pixels. Similarly, a column of $2H + 1$ samples is formed from the reconstructed left, below-left, and top-left pixels. If some of the pixels on top or left are not available, because the current and neighbor CUs do not belong to the same slice, or the current CU is at a frame boundary, etc., then reference sample substitution is performed, where the missing samples are copied from the available sample in a clock-wise direction [1-2].

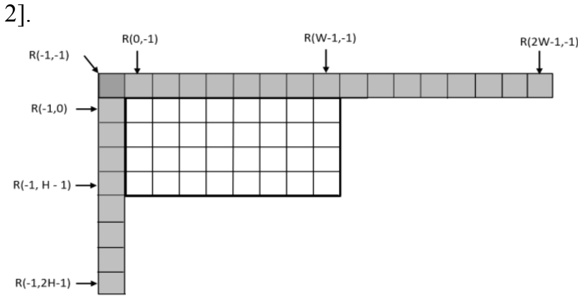

Figure 1: Example of reference samples for intra prediction.

For larger Luma CUs ($W \times H > 32$), the reference samples are filtered with the [1 2 1]/4 filter if the prediction mode is either PLANAR or a mode with integer slope (i.e., an angular mode, excluding the strictly vertical and strictly horizontal modes, where every target pixel can be mapped to a single reference sample in the prediction direction) [1].
The left and top reference arrays together are called the first reference line. VVC also supports multiple reference line (MRL) intra prediction for Luma CUs where, besides the first reference line, two other reference lines at an offset of one or three pixels from the target block position are available for prediction. The samples on those reference lines are never filtered. The parameter *multiRefidx* indicates the reference line used for the intra prediction of a CU.

B. *Intra sample prediction*

For intra prediction, VVC defines 65 regular angular modes and 28 wide angular modes, as shown in Fig. 2. The regular angular modes are indexed from mode 2 to mode 66; the wide angular modes are indexed from mode 67 to mode 80 and from mode -1 to -14. The two non-angular modes, that is, the PLANAR mode and the DC mode, are indexed as mode 0 and mode 1, respectively, as in HEVC. A square CU can have the PLANAR mode, the DC mode, or any of the 65 regular angular modes. A rectangular CU can have the PLANAR mode, the DC mode, or any one of 65 contiguous angular modes consisting of some regular angular modes and some wide angular modes. The possible wide angular modes of a rectangular CU depend on its aspect ratio and are given in [1]. In any case, the total number of angular modes used for any block size is always 65. The modes in the range [-14…-1,2 …33] are horizontal modes, and the modes in the range [34 … 80] are vertical modes. Mode 50 and mode 18 are the strictly vertical and horizontal modes, respectively, commonly referred to as VER_IDX and HOR_IDX in VVC. Note that, in Fig. 2, the wide angular modes represent only the associated directions, not the locations of the reference samples, which are only on top and left of the CU.

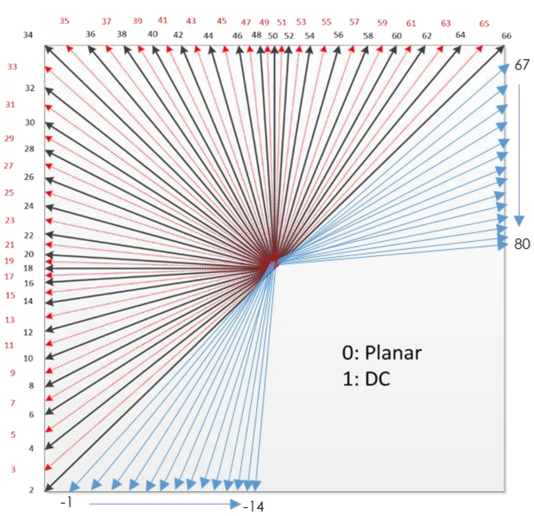

Figure 2: Prediction modes in VVC. PLANAR mode and DC mode are indexed as mode 0 and mode 1, respectively. Modes 2-66 are regular angular modes. Modes [67:80], and [-1 : -14] are wide angular modes.

Each mode is associated with an angle parameter called *predIntraAngle,* that defines the mode in terms of the offset from the VER_IDX (HOR_IDX) mode for vertical (horizontal) modes. The offset has a granularity of (1/32), i.e., the distance between two adjacent reference pixels is divided uniformly into 32 parts, and the offset is expressed as a multiple of this part. The *predintraAngle* for VER_IDX or HOR_IDX is zero. For the 65 regular angular modes, the *predIntraAngle* value lies in the range [-32, 32], as in HEVC. For the wide angular modes, the *predIntraAngle* values are all positive and lie in the range [35, 512]. The *predIntraAngle* values for all angular modes are given in [1].

If the prediction mode is VER_IDX or HOR_IDX, or a mode with integer slope (i.e., modes 2, 34, 66, 72, 76, 78, 80, -6, -10, -12, -14), then the predictor for every target sample will coincide with a reference sample. In this case, the corresponding reference sample will be copied to the target pixel as prediction. For all other angular prediction modes, the predictor samples are interpolated using the reference samples. For LUMA CUs, the predictor samples are interpolated using either a 4-tap Gaussian smoothing filter or a 4-tap cubic filter. For larger CUs ($W \times H > 32$) predicted with the first reference line, if the offset between the mode index and the VER_IDX or HOR_IDX is greater than a preset threshold based on the CU size, then the Gaussian filter is used; for all other cases the cubic filter is used for interpolation. For Chroma CUs, the predictor samples are interpolated using linear interpolation, as in HEVC.

C. *Post-processing*

With certain prediction modes that may create discontinuity at the CU boundary adjacent to a reference array, the predicted values of Luma CUs undergo a post-processing step called Position Dependent intra Prediction Combination (PDPC). These modes include the PLANAR mode, the DC mode, VER_IDX, HOR_IDX modes, and the angular modes with positive *predIntraAngle* values. PDPC modifies original predicted values by weighted averages with some reference samples such that the intensity change at the boundary is gradual. PDPC formulation for the applicable modes are given in [1].

For any target CU, the encoder checks all available prediction modes and selects the best mode based on the R-D performance. The best mode is signaled with a variable length coding scheme.

III. MULTI MODE PREDICTION

In our proposal, only the sample prediction step is modified. The other two steps are maintained as they are in the VVC. For convenience of understanding, we will describe the method with only vertical prediction modes in the following. The cases for horizontal prediction modes are analogous, and they can be implemented in VTM as usual by swapping the left reference array with the top reference array.

Consider the vertical prediction mode with index V, with associated *intraPredAngle* A_V. Let the *intraPredAngle* values associated with modes $V-1$ and $V+1$ be denoted as A_{V-1} and A_{V+1}, respectively. If V happens to be the minimum or maximum value of the possible modes, we will consider $V+2$ or $V-2$, for $V-1$ or $V+1$, respectively. Instead of assigning the same *intraPredAngle* value A_V to all target pixels, we can assign them values over a range $[A_{begin}, A_{end}]$ where A_{begin} and A_{end} are assigned to the first and last row of target pixels, respectively. For pixels on other rows, the assigned *predIntraAngle* values can be linearly interpolated between A_{begin} and A_{end} depending on the vertical distance of the row y. Here we consider linear interpolation for simplicity. This is illustrated in Fig. 3.

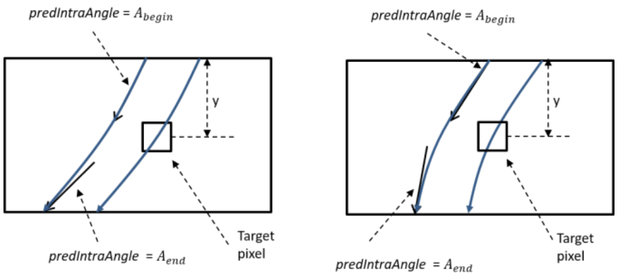

Figure 3: Varying prediction direction in a CU. Left: directions are between V (top row) and $V+1$ (bottom row); Right: directions are between V (top row) and $V-1$ (bottom row).

Various schemes can be followed for choosing the values of A_{begin} and A_{end}. For example, A_{begin} and A_{end} can be equal to $\left\lfloor\frac{A_{V-1}+A_V}{2}\right\rfloor$ and $\left\lfloor\frac{A_V+A_{V+1}}{2}\right\rfloor$, respectively, or the other way around. As another example, A_{begin} can be A_V and A_{end} can be either A_{V-1}, or A_{V+1}. These two cases are shown in Fig. 3.

In VVC, for any target pixel, the location of its predictor reference sample is determined by the horizontal offset denoted as *deltaPos*. For a target sample on row y, the offset is computed as

$$deltaPos = (1+y) * A, \tag{1}$$

where A denotes the *predIntraAngle* associated with the mode. When the mode directionality is changed as described above, the offset will be computed as:

$$deltaPos = \sum_{i=0}^{i=y} A_i, \tag{2}$$

where A_i denotes the value of the *predIntraAngle* for row i. This formulation is necessary so that the directionality model as above corresponds to object directionality in reality. Notice that, the VVC offset computation, where every row of a target block has the same *predIntraAngle*, i.e., $A_0 = A_1 = \cdots = A_{H-1} = A$, is just a special case of the above formulation.

A generalized model as above is difficult to be implemented in the current VTM software. First, it needs the *predIntraAngle* values to be computed for each row. Second, for directions having negative *predIntraAngle* values, VTM requires the inverse *predIntraAngle* values, computed as $round(2^{14}/A)$, for mapping the left reference samples onto the top reference array. Thus, implementing the generalized model in VTM would induce further complexity because of the division operations required for each row of a target block for the concerned modes.

We notice that, if A_{begin} and A_{end} are consecutive integers, then interpolating the *predIntraAngle* values in a block will assign A_{begin} to half of the rows and A_{end} to the other half. This is because *predIntraAngle* can have integer values only. Thus, if A_{begin} and A_{end} are associated with modes V and $V-1$ or $V+1$, this is equivalent to assigning the mode V to one half of the block and the mode $V-1$ or $V+1$ to the other half. Starting with this example, we can now define the change of directionality in terms of mode indexes. We will assume that a target block can have two prediction modes V_{begin} and V_{end}, where V_{begin} can be equal to the original prediction mode V, and V_{end} can be equal to $V+1$ or $V-1$, or the other way around. We will assume that V_{begin} and V_{end} are applied over non-overlapping partitions of the coding block, where the partitions are obtained with horizontal splits for vertical modes, and with vertical splits for horizontal modes. We can have flexibility by not restricting the split to be at the center of the block but also at either one-fourth or three-fourth of the block as well. Examples of mode and partition combinations are shown in Fig. 4.

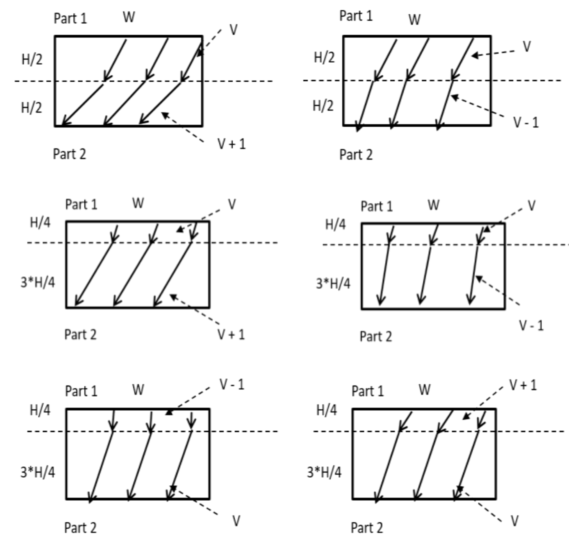

Figure 4: Varying prediction direction with two modes. The split can be symmetric or asymmetric, with modes mapped to different partitions.

With two prediction modes, we can also have a higher number of partitions. Fig. 5 illustrates cases with three and four partitions.

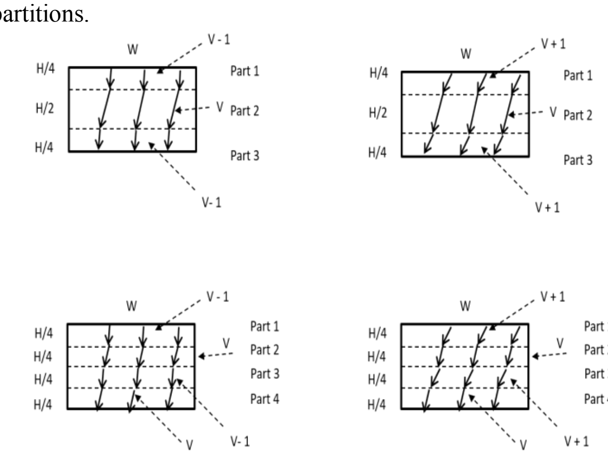

Figure 5: Varying prediction direction with two modes over multiple partitions

In any case, the offset of the predictor for a target sample, i.e., *deltaPos*, will be computed as in Eqn.(2), where the A_i's are equal to one of the two *predIntraAngle* values associated with the two modes.

A. Signalling with mode partitioning

In a practical encoding scenario, the number of candidate models for the directionality change will be finite. The encoder will check the R-D performance of each model and encode the best model with a suitable encoding scheme. Each candidate model would need additional complexity as the encoder has to check the R-D performance. Therefore, to keep the complexity low, in this paper, we will consider the simple case with three candidates: (1) no partition, (2) two partitions with modes V and $V-1$ in the top (left) and bottom (right) partitions when V is a vertical (horizontal) mode (3) two partitions with modes V and $V+1$ in the top (left) and bottom (right) partitions when V is a vertical (horizontal) mode. Alternatively, we can also consider the same splits with the modes swapped between the two partitions [3]. The three candidates can be coded with 0, 10, 11, in that order where the bits can be context-encoded.

B. Interaction with ISP

VVC also supports intra prediction with sub-partitions (ISP), in which intra-predicted Luma blocks can be split vertically or horizontally into 2 or 4 sub-partitions depending on the block dimensions [4]. Blocks of size 4×8 or 8×4 can be split into two sub-partitions only, whereas blocks of larger size can be split into 4 sub-partitions. The minimum size for a sub-partition is 16 pixels. For any target CU, the encoder tests the R-D performance for no-split, horizontal and vertical splits and encodes a parameter called *ispMode* to indicate the selected split. In case the CU is split, all sub-partitions use the same prediction mode, like the case with no split. The sub-partitions are encoded and decoded sequentially with the block transform based on the sub-partitions size. The sub-partitions are processed sequentially at the decoder so that a sub-partition benefits from the availability of the decoded pixels in the previous sub-partition.

The proposed method of multiple prediction modes can be still used with ISP. In one case, the proposed method can be restricted to the blocks which are not split by ISP; in this case, the transform size will correspond to the size of the CU. Alternatively, it can be included on top of ISP, that is, the prediction modes in different partitions in ISP can be modelled as in our proposal, provided that the split type matches with the directionality of the prediction mode. In this case, the transform is performed at sub-partition level, as done in ISP. As the ISP splits the block into smaller sub-blocks which are likely to get better prediction because of the proximity of reference samples, we propose to use the first method.

IV. Experimental results

We implemented the proposed method within the VTM 4.0 codec in All Intra (AI) configuration using the common test conditions (CTC) as specified in [5]. For directionality models, we used the mode split at one fourth the dimension where the top part is assigned the mode $V-1$ or $V+1$ and the bottom part is assigned the current mode V (Fig. 4, last case). These two models with no mode split were signaled with bits: 0 (no partition), 10 (top part $V-1$ and bottom part V), and 11 (top part is $V+1$ and bottom part is V). The mode split method was applied to blocks which used the second or the fourth reference lines for prediction. As the CUs using these reference lines are not split in ISP, there was no clash of mode split with ISP partition. Table 1 shows the resulting BD-rate performance when the method is applied to Luma blocks only. An overall BD-rate gain of 0.11% for Luma over the VTM is observed. The encoding and decoding times are indicative as the simulations were performed on a heterogenous CPU cluster at different times with other concurrent jobs. We observe that the proposed method is more effective for lower resolution sequences (class B, C, E). Lower gains for very high resolution sequences (class A1 and A2) is not surprising as the directionalities of objects in blocks are better preserved with higher resolution. Higher encoding time is expected as the encoder has to perform additional searches for the best directionality model; however, the overall decoding time is about the same as the VTM anchor.

Table 1: BD-rate performance of the proposed method over VTM 4.0

	All Intra Main10				
	Over VTM-4.0				
	Y	U	V	EncT	DecT
Class A1	0.00%	0.06%	0.01%	111%	101%
Class A2	-0.08%	0.03%	-0.05%	108%	99%
Class B	-0.16%	-0.12%	-0.07%	108%	100%
Class C	-0.12%	-0.10%	-0.03%	109%	101%
Class E	-0.17%	-0.07%	-0.12%	111%	102%
Overall	-0.11%	-0.05%	-0.05%	109%	100%
Class D	0.01%	-0.09%	-0.18%	108%	100%
Class F	-0.07%	-0.01%	-0.15%	106%	103%

V. Conclusion

In this paper, we presented a method for intra prediction with varying angle to be used in VVC. The method aims to model the change of object directionalities inside a coding unit. First, we presented a general framework and then presented a simpler method where the directionality change can be modelled with mode indices rather than the angle parameter. Simulation results obtained with JVET test sequences demonstrate that the method does produce some gains over the VTM anchors, which have to be weighted by the added complexity. Future research work will involve relaxation of split constraints, higher number of modes and partitions used in a CU, etc.

References

[1] B. Bross, J. Chen, S. Liu, Y. -K. Wang, “Versatile Video Coding (Draft 7),” JVET-P2001-vE, 16th Meeting, Geneva, CH, 1-11 Oct. 2019.

[2] V. Sze, M. Budagavi, G. J. Sullivan, Eds, “High Efficiency Video Coding (HEVC), Algorithms and Architectures,” Springer Publishing, Switzerland, 2014.

[3] G. Rath, F. Urban, and F. Racape, “Non-CE3 : Directional intra prediction with varying angle,” JVET-N0371, 14th Meeting, Geneva, CH, 19-27 March 2019.

[4] S. De-Luxán-Hernández, et al, “CE3 : Intra Sub-Partitions Coding Mode (Tests 1.1.1 and 1.1.2),” JVET-M0102-v5, 13th Meeting, Marrakech, MA, 9-18 Ja. 2019.

[5] F. Bossen, J. Boyce, X. Li, V. Seregin, K. Suhring, “JVET common conditions and software reference configurations for SDT video,” JVET-N1010, 14th Meeting, Geneva, CH, 19-27 March 2019.