Post-print of: Sánchez, V., et al. “Piecewise mapping in HEVC lossless intra-prediction coding” in IEEE transactions on image processing, vol. 25, issue 9 (Sep. 2016), p. 4004-4017. DOI 10.1109/TIP.2016.2571065

Piecewise Mapping in HEVC Lossless Intra-prediction Coding

Victor Sanchez, *Member, IEEE*, Francesc Aulí-Llinàs, *Senior Member, IEEE* and Joan Serra-Sagristà, *Senior Member, IEEE*

Abstract—**The lossless intra-prediction coding modality of the High Efficiency Video Coding (HEVC) standard provides high coding performance while allowing frame-by-frame basis access to the coded data. This is of interest in many professional applications such as medical imaging, automotive vision and digital preservation in libraries and archives. Various improvements to lossless intra-prediction coding have been proposed recently, most of them based on sample-wise prediction using Differential Pulse Code Modulation (DPCM). Other recent proposals aim at further reducing the energy of intra-predicted residual blocks. However, the energy reduction achieved is frequently minimal due to the difficulty of correctly predicting the sign and magnitude of residual values. In this paper, we pursue a novel approach to this energy-reduction problem using piecewise mapping (pwm) functions. Specifically, we analyze the range of values in residual blocks and apply accordingly a pwm function to map specific residual values to unique lower values. We encode appropriate parameters associated with the pwm functions at the encoder, so that the corresponding inverse pwm functions at the decoder can map values back to the same residual values. These residual values are then used to reconstruct the original signal. This mapping is, therefore, reversible and introduces no losses. We evaluate the pwm functions on 4×4 residual blocks computed after DPCM-based prediction for lossless coding of a variety of natural and screen content video sequences. Evaluation results show that the pwm functions can attain maximum bit-rate reductions of 5.54% compared to DPCM-based intra-prediction. When combined with DPCM, these functions can attain maximum bit-rate reductions of 28.33% compared to block-wise intra-prediction.**

Index Terms— **HEVC intra-prediction, lossless coding, DPCM, SAP, piecewise mapping.**

I. Introduction

Extensions and enhancements to the HEVC standard [1] are developed to support multi-view and 3D video coding [2], scalable coding [3], and coding of high bit-depth videos represented using different color formats. The latter comprises the so-called Range Extensions (RExt) [4]. An important part of RExt is the improvement of lossless coding performance. This is of special interest in professional applications such as medical imaging, automotive vision, and digital preservation in libraries and archives. Many of these applications require the compression of both video sequences and images. Therefore, improvements to lossless intra-prediction coding are highly desirable.

Intra-prediction coding in HEVC is based on block-wise spatial data prediction within the same frame. This process employs angular and planar prediction to model different directional patterns and to generate smooth sample surfaces [5]. HEVC includes a lossless coding modality that allows perfect reconstruction of the signal. This is achieved by bypassing the transform, quantization, and any other processing that produces losses [6, 7].

Recently, several improvements to intra-prediction coding have been proposed. These improvements may be broadly classified into those that employ block-wise prediction, and those that employ sample-wise prediction. Transform Skip [7], Intra-Block Copy (IntraBC) [8], Edge Mode [9] and Nearest-Neighbor (NN) intra-prediction [10], are among the most important block-wise intra-prediction improvements. While these improvements are mainly designed for lossy coding, they can also be applied for lossless coding. Transform Skip allows bypassing the transform after intra-prediction in order to avoid spreading the energy associated with sharp edges over a wide frequency range. IntraBC predicts the current block from the previously coded and reconstructed region in the same frame, similar to motion estimation/compensation in inter-prediction. Edge Mode improves coding efficiency by modeling six edge positions and selecting the intra-prediction direction that is in parallel to the edge orientation. In order to accurately predict sharp edges, NN intra-prediction selectively replaces the bi-linear interpolation used in angular intra-prediction by a nearest-neighbor interpolation. All of these improvements provide significant bit-rate reductions for videos depicting repeating patterns and sharp edges.

Improvements based on sample-wise prediction usually employ Differential Pulse Code Modulation (DPCM). Zhou *et al*. propose sample-based angular intra-prediction (SAP), which uses adjacent neighbors to perform sample-wise prediction [11]. SAP has been shown to provide important lossless bit-rate reductions compared to block-wise intra-prediction coding. Subsequent DPCM-based proposals are SAP-HV [12], SAP1 [13], and SAP-E [14]. SAP-HV applies DPCM exclusively in the pure horizontal and vertical directions. SAP1 is similar to SAP but employs a more uniform density of prediction modes in the vertical and

V. Sanchez is with the Department of Computer Science, University of Warwick, UK.

F. Aulí-Llinàs, and J. Serra-Sagristà are with the Department of Information and Communications Engineering, Universitat Autonoma de Barcelona, Spain.

This work has been supported by the EU Marie Curie CIG Programme (grant PIMCO), FEDER, the Spanish Ministry of Economy and Competitiveness (MINECO), and the Catalan Government under grants TIN2015-71126-R, TIN2012-38102-C03-03 (LIFE-VISION) and 2014SGR-691.

horizontal directions. SAP1 has been shown to improve coding efficiency over SAP and SAP-HV on gray-scale anatomical medical images [13, 15]. SAP-E employs DPCM-based prediction in all modes, including the DC mode. Specifically, SAP-E implements the DC mode as an average of two adjacent samples and replaces the PLANAR mode by an edge predictor [14, 16]. SAP-E has been shown to provide further bit-rate reductions over SAP, SAP-HV and SAP1, as tested on large color biomedical images [14].

Other DPCM-based prediction methods include sample-based weighted prediction with directional template matching (SWP2+DTM) [17], and Combined Intra-prediction (CIP) [18]. SWP2+DTM compute a weighted average of surrounding pixels to predict the current pixel. For cases in which all computed weights are zero, e.g., for sharp edges, SWP2+DTM use as a predictor the pixel that is estimated to be the most similar to the current pixel [17, 19]. CIP computes weighted prediction samples that exploit redundancies not only among neighboring blocks but also within the current block.

A number of DPCM-based methods that aim at further reducing the energy of residual signals have also been proposed. Residual DPCM (RDPCM) applies DPCM-based prediction on the residual signals in the horizontal (or vertical) direction if the block-wise horizontal (or vertical) intra-prediction mode is used [20]. A variant of RDPCM is introduced in [21] for inter-predicted residuals. This variant applies DPCM-based prediction in the horizontal or vertical direction, or no additional prediction, according to the sum of absolute differences (SAD). In [22], the authors propose applying a sample-based edge predictor on the entire residual frame, thus departing from the block-wise coding structure of HEVC. The work in [23] proposes the cross residual transform, which uses a two-step prediction process when the horizontal or vertical modes are used. The first step applies DPCM-based prediction in the horizontal or vertical direction. The second step applies DPCM-based prediction in the corresponding residual signal following an orthogonal direction. In [24, 25], the authors propose methods to improve prediction accuracy of RDPCM. This is achieved by exploiting the gradient information of neighboring samples into the prediction process. All of these methods can further reduce the energy of the residual signals if residual values are highly correlated in magnitude and sign. However, if this is not the case, they may increase their overall energy.

This paper furthers the proposals that aim at reducing the energy of residual signals after intra-prediction in HEVC lossless coding. Instead of predicting residual values, we pursue a novel approach to the energy-reduction problem using piecewise mapping (pwm) functions. Specifically, we analyze the range of values in residual blocks and accordingly apply a pwm function to map specific residual values to unique lower values. We encode appropriate parameters associated with the pwm functions at the encoder, so that the corresponding inverse pwm functions at the decoder can map values back to the same residual values. These residual values are then used to reconstruct the original signal. Since the proposed pwm functions are applied on a block-by-block basis, the block-wise encoding structure of HEVC is maintained. We evaluate the pwm functions on 4×4 residual blocks computed after DPCM-based prediction for lossless coding. Evaluation results confirm the effectiveness of the pwm functions in reducing their energy and improving lossless coding efficiency.

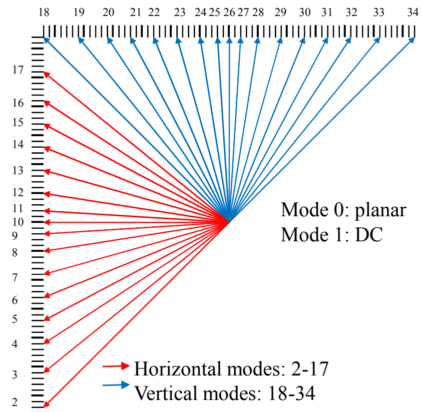

Fig. 1. Intra-prediction modes in HEVC. Angular modes are numbered 2-34.

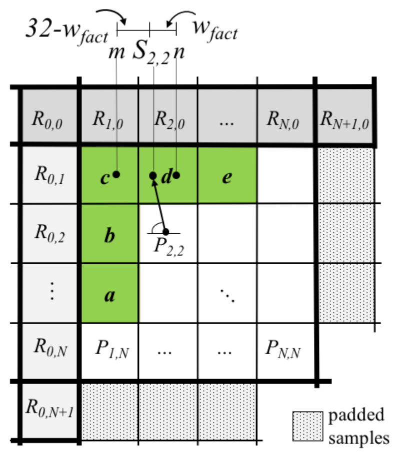

Fig. 2. Prediction principle of SAP for angular modes in HEVC. Initial reference samples are $\{R_{0,1}, R_{0,2},\ldots, R_{0,N+1}\}$ and $\{R_{0,0}, R_{1,0},\ldots, R_{N+1,0}\}$, which are located to the left and above of the current block, respectively. Samples in neighboring blocks yet to be coded are padded with available boundary samples of the current block.

The remainder of this paper is organized as follows. Section II briefly reviews DPCM-based intra-prediction in HEVC. We detail the proposed pwm functions in Section III. Evaluation results are provided in Section IV followed by conclusions in Section V.

II. DPCM-based Intra-prediction in HEVC

DPCM-based intra-prediction in HEVC is first introduced to all angular modes in SAP [11]. Fig. 1 depicts the prediction directions associated with these angular modes, while Fig. 2 illustrates the prediction principle of SAP for an $N\times N$ block. Specifically, two reference samples are determined based on the location of the current sample at position (x,y), denoted by $S_{x,y}$, and the prediction angle. The corresponding predicted sample, $P_{x,y}$, is then computed by interpolating the two reference samples of the set of neighbors of $S_{x,y}$ that are located at positions $g = \{a, b, c, d, e\}$:

$$P_{x,y} = ((32 - w_{fact}) * m + w_{fact} * n) >> 5 \tag{1}$$

TABLE I. PREDICTION OPERATIONS OF DPCM-BASED MODES

Mode 0			
$P_{x,y}=\begin{cases}\min(b,d) & \text{if } c\geq\max(b,d)\\ \max(b,d) & \text{if } c\leq\min(b,d)\\ b+d-c & \text{otherwise}\end{cases}$			
DC mode – mode 1			
$P_{x,y}=(b+d)>>1$			
Horizontal angular modes (SAP)			
Mode	Prediction operation	Mode	Prediction operation
2	$P_{x,y}=a$	10	$P_{x,y}=b$
3	$P_{x,y}=(26*a+6*b)>>5$	11	$P_{x,y}=(30*b+2*c)>>5$
4	$P_{x,y}=(21*a+11*b)>>5$	12	$P_{x,y}=(27*b+5*c)>>5$
5	$P_{x,y}=(17*a+15*b)>>5$	13	$P_{x,y}=(23*b+9*c)>>5$
6	$P_{x,y}=(13*a+19*b)>>5$	14	$P_{x,y}=(19*b+13*c)>>5$
7	$P_{x,y}=(9*a+23*b)>>5$	15	$P_{x,y}=(15*b+17*c)>>5$
8	$P_{x,y}=(5*a+27*b)>>5$	16	$P_{x,y}=(11*b+21*c)>>5$
9	$P_{x,y}=(2*a+30*b)>>5$	17	$P_{x,y}=(6*b+26*c)>>5$
Vertical angular modes (SAP)			
18	$P_{x,y}=c$	26	$P_{x,y}=d$
19	$P_{x,y}=(26*c+6*d)>>5$	27	$P_{x,y}=(30*d+2*e)>>5$
20	$P_{x,y}=(21*c+11*d)>>5$	28	$P_{x,y}=(27*d+5*e)>>5$
21	$P_{x,y}=(17*c+15*d)>>5$	29	$P_{x,y}=(23*d+9*e)>>5$
22	$P_{x,y}=(13*c+19*d)>>5$	30	$P_{x,y}=(19*d+13*e)>>5$
23	$P_{x,y}=(9*c+23*d)>>5$	31	$P_{x,y}=(15*d+17*e)>>5$
24	$P_{x,y}=(5*c+27*d)>>5$	32	$P_{x,y}=(11*d+21*e)>>5$
25	$P_{x,y}=(2*c+30*d)>>5$	33	$P_{x,y}=(6*d+26*e)>>5$
		34	$P_{x,y}=e$

where >> denotes a bit shift operation to the right; $\{m,n\}$ are the reference samples in g ($m \neq n$); and w_{fact} is the distance measured with 1/32 pixel accuracy between $S_{x,y}$ and n.

In order to further improve the performance of SAP, the DC mode can be implemented using DPCM by averaging the neighbors of $S_{x,y}$ located at positions $\{b, d\}$ [14, 26]. Similarly, an edge predictor can be introduced in lieu of the PLANAR mode in order to improve the performance in the presence of edges [14, 16, 27]:

$$P_{x,y}=\begin{cases}\min(b,d) & \text{if } c\geq\max(b,d)\\ \max(b,d) & \text{if } c\leq\min(b,d)\\ b+d-c & \text{otherwise}\end{cases} \tag{2}$$

This particular edge predictor, which has been successfully employed in the JPEG-LS standard [28], is capable of detecting vertical or horizontal edges very accurately. If an edge is not detected, then the prediction sample is $P_{x,y} = b + d - c$, which represents the expected smoothness of the image in the absence of edges. Table I summarizes the prediction operations of these 35 DPCM-based prediction modes. If reference samples are unavailable, e.g., reference samples located in neighboring blocks yet to be encoded, missing reference samples are padded with boundary samples of the current block [11].

III. ENERGY REDUCTION WITH PIECEWISE MAPPING

Residual blocks computed using DPCM-based intra-prediction

-2	4	-4	1	4
3	-10	1	8	-1
-3	3	8	1	-7
-5	2	-1	1	0
3	-5	-2	3	-3

energy = 342

(a)

3	-10	1	8
-3	3	8	1
-5	2	-1	1
3	-5	-2	3

Predicted block

-13	11	7	-9
6	5	-7	-8
7	-3	2	-1
-8	3	5	-6

Final residual block

energy = 791

(b)

4	-4	1	4
-10	1	8	-1
3	8	1	-7
2	-1	1	0

Predicted block

-14	5	7	-5
13	7	-7	-6
-1	-9	0	7
-7	-1	2	-3

Final residual block

energy = 792

(c)

-2	4	-4	1
3	-10	1	8
-3	3	8	1
-5	2	-1	1

Predicted block

-8	-3	12	-2
0	18	0	-15
5	-4	-7	-1
0	-4	4	-4

Final residual block

energy = 929

(d)

-9	1	**4**	-1
3	**4**	1	**-6**
2	-1	1	0
-4	-2	3	-3

energy = 205

(e)

Fig. 3. (a) Sample 4×4 residual block computed using DPCM-based prediction following a horizontal direction, and corresponding predicted and final residual blocks when DPCM-based prediction is used in the (b) horizontal, (c) vertical and (d) diagonal direction. Samples in gray represent reference residual values from the blocks above and to the left of the sample residual block. (e) Final residual block after mapping values using the lpwm function in Eq. (3) with t = 4, h = 3, v = 1 and q = -3. Values in bold and larger font represent the mapped residual values.

are expected to feature low energy values. Low-energy residual blocks tend to follow a Laplacian distribution that is highly peaked at zero. The expected large number of zero-valued residuals can then be efficiently compressed using context adaptive binary arithmetic coding (CABAC). Therefore, lossless coding efficiency may be improved if the energy of residual blocks is further reduced. Other methods that aim at reducing the energy of residual blocks use DPCM-based prediction on the residual signals [20-25]. The main challenge of predicting residual values is to correctly predict not only their magnitude, but also their sign. An incorrect sign prediction may considerably increase the overall energy of the residual block. To illustrate the challenges of predicting residual values, let us take the sample 4×4 residual block in Fig. 3(a), which has been computed using DPCM-based prediction following a horizontal direction. Note that this residual block comprises values that are correlated in magnitude, but not necessarily in sign, a common situation for

2	-2	3	-1
3	-3	2	-7
2	-4	-1	2
5	3	4	3

energy = 173
(a)

0	-2	**1**	-1
1	-3	**0**	-7
0	-4	-1	**0**
3	**1**	**2**	**1**

energy = 97
(b)

Fig. 4. (a) Sample 4×4 residual block computed using DPCM-based prediction following a horizontal direction. The bock comprises positive and negative values with no zeros. (b) Corresponding final residual block when a dpwm function with i = 2 is applied [see Eq. (5)]. Values in bold and larger font represent the mapped residual values.

10	2	3	4
3	5	2	5
6	10	8	3
7	4	6	4

energy = 518
(a)

6	**-2**	**-1**	**0**
-1	**1**	**-2**	**1**
2	**6**	**4**	**-1**
3	**0**	**2**	**0**

energy = 118
(b)

Fig. 5. (a) Sample 4×4 residual block computed using DPCM-based prediction following a horizontal direction. The block comprises only values of the same sign, excluding zeros. (b) Corresponding final residual block when a shifting operation with k = 4 is applied. Values in bold and larger font represent the residual values affected by the shifting operation.

residual signals. Fig. 3(b)-(d) illustrate the predicted and final residual blocks when DPCM-based prediction in three different directions is used. Note that in all cases, the energy of the final residual block, r^f, is higher than that of the original residual block, r. The block energy is computed as *energy* = $\sum r_{(x,y)}^2$. This increase in energy is mainly due to incorrectly predicting the sign of residual values.

Based on the above observations, instead of predicting residual values, we first analyze the range of residual values in each block, and according to this analysis, we then modify these values using a pwm function. The objective is to reduce the range of values and center it at zero, thus effectively reducing the overall energy of the block. In other words, we aim at increasing the number of residual blocks with values that tend to follow a Laplacian distribution peaked at zero. To this end, we aim at increasing the number of zero-valued residuals, while decreasing the long tails associated with the distribution. These long tails are produced by a small number of inaccurate predictions.

A. Proposed piecewise mapping functions

Let us take the same sample residual block in Fig. 3(a). Note that this block comprises residual values in the range [-10, 8], including zero-valued residuals. Also note that within this range, not all values appear in the block. For example, there are no residuals with values in the range [4, 7] or with values in the set {-4, -6, -8, -9}. Residual values can then be mapped to unique lower values within the range [4, 7] or within the set {-4, -6, -8, -9}. For example, all values greater than h = 3 can be subtracted a value t = 4; while all values lower than q = -3 can be added a value v = 1. In other words, for this sample block, we can apply the following pwm function to residual $r_{x,y}$ at position (x,y) to obtain the corresponding final residual $r^f_{x,y}$:

$$r^f_{x,y} = f\left(r_{x,y}, t, v, h, q\right) = \begin{cases} r_{x,y} - t & \text{if } r_{x,y} > h \\ r_{x,y} + v & \text{if } r_{x,y} < q \\ r_{x,y} & \text{otherwise} \end{cases} \qquad (3)$$

The final residual block for this sample block is illustrated in Fig 3(e). Note that the final residual values are now in the range [-9, 4], which is smaller and more centered at zero than the original range. Consequently, the final residual block now features a lower energy value than the original one. We call this type of function linear pwm (lpwm) function, since it modifies specific residual values in a linear fashion.

Note that this lpwm function allows recovering the mapped values with no loss, as long as the parameters associated with the function are signaled to the decoder. For the final residual block in Fig 3(e), the inverse lpwm function needed to recover the original residual block is as follows:

$$r'_{x,y} = f_i\left(r^f_{x,y}, t, v, h, q\right) = \begin{cases} r^f_{x,y} + t & \text{if } r^f_{x,y} > h \\ r^f_{x,y} - v & \text{if } r^f_{x,y} < q \\ r^f_{x,y} & \text{otherwise} \end{cases} \qquad (4)$$

where $r'_{x,y}$ is the recovered residual value at position (x,y).

Let us now take the sample 4×4 residual block illustrated in Fig. 4(a), which has been computed using DPCM-based prediction following a horizontal direction. This particular block contains positive and negative residuals with no zero-valued residuals. The fact that the block contains no zero-valued residuals can be exploited to map residual values to unique values close to zero, consequently reducing the overall block energy. Specifically, we can apply the following pwm function:

$$r^f_{x,y} = f\left(r_{x,y}, i, j, e_p, e_n\right) = \begin{cases} r_{x,y} - i & \text{if } r_{x,y} > 0 \text{ AND } e_p \geq e_n \\ r_{x,y} + j & \text{if } r_{x,y} < 0 \text{ AND } e_p < e_n \\ r_{x,y} & \text{otherwise} \end{cases}$$

$$i = \min(I); \;\; I = \left\{\left|\text{positive residual values}\right|\right\}$$

$$j = \min(J); \;\; J = \left\{\left|\text{negative residual values}\right|\right\} \qquad (5)$$

where $e_p = \sum_{r \in I} r^2_{(x,y)}$ and $e_n = \sum_{r \in J} r^2_{(x,y)}$ denote the energy of positive values and negative values in the residual block, respectively. The resulting final residual block after applying the function in Eq. (5) by subtracting i = 2 to all positive values, is illustrated in Fig. 4(b). We call the function in Eq. (5) dual piecewise mapping (dpwm) function, as it affects either positive or negative values. In order to recover the original residual block modified by a dpwm function, we must apply the inverse dpwm function as follows:

$$r'_{x,y} = f_i\left(r_{x,y}, i, j, e_p, e_n\right) = \begin{cases} r^f_{x,y} + i & \text{if } r^f_{x,y} \geq 0 \text{ AND } e_p \geq e_n \\ r^f_{x,y} - j & \text{if } r^f_{x,y} \leq 0 \text{ AND } e_p < e_n \\ r^f_{x,y} & \text{otherwise} \end{cases}$$

4	3	3	7
5	0	4	3
4	8	2	1
0	7	0	5

energy = 292
(a)

4	3	**-3**	**1**
5	0	**-2**	**-3**
4	8	2	1
0	7	0	5

energy = 232
(b)

-2	**-3**	**-3**	**1**
5	0	**-2**	**-3**
-4	**0**	**0**	**-1**
0	7	0	5

energy = 152
(c)

Fig. 6. (a) Sample 4×4 residual block computed using DPCM-based prediction following a horizontal direction. The block comprises values of the same sign, including zeros. (b) Corresponding final residual when a spwm function is applied on 2×2 partitions. Partitions whose values are mapped are depicted in blue. (c) Corresponding final residual when a spwm function is applied on pairs of horizontally adjacent residual values. Pairs whose values are mapped are depicted in blue.

(6).

Finally, let us consider the sample 4×4 residual block in Fig. 5(a), which has been computed using DPCM-based prediction following a horizontal direction. This block contains only values of the same sign, excluding zeros. Similarly to the previous sample blocks, we can reduce the overall energy by mapping residual values to a range of values centered at zero. For example, by subtracting each residual 2× the minimum residual value in the block, i.e., k = 2×2 = 4 in this case, the range of values is now [-2, 6], which is more centered at zero than the original range of values of [2, 10]. By applying this *shifting* operation, the overall energy of this sample residual block is reduced, as illustrated in Fig 5(b). For this example, in order to recover the original residual block, we must subtract from all final residual values 2× the minimum value in the final residual block, i.e., j = 2× (-2) = -4.

Although the previously described shifting operation can be also applied to residual blocks that comprise values of the same sign and zeros, the final residual block in this case is equal to the original residual block, as the minimum residual value is zero. However, the shifting operation can be applied to small partitions in the residual block without the need to signal the use of this operation for each partition. For example, let us take the sample 4×4 residual block in Fig. 6(a), which has been computed using DPCM-based prediction following a horizontal direction. If this block is partitioned into four 2×2 blocks, as illustrated in Fig. 6(b), the shifting operation can then reduce the energy of one of the four partitions. Based on this observation, this operation can then be applied to even smaller partitions, in an attempt to find those regions in a residual block that comprise only non-zero values of the same sign. The size of the partitions can be reduced down to a pair of adjacent residual values, as illustrated in Fig. 6(c). For example, for the sample residual block in Fig. 6(a), each residual value in a pair of horizontally adjacent residuals can then be subtracted the value $k = 2\times l$, where l denotes the minimum value in the pair. For two horizontally adjacent residuals in a residual block that comprise only values of the same sign, including zeros, this shifting operation can be defined as a pwm function:

$$\left\{r^f_{x,y}, r^f_{x+1,y}\right\} = f\left(\left\{r_{x,y}, r_{x+1,y}\right\}, k\right) = \begin{cases} \left\{r_{x,y} - k, r_{x+1,y} - k\right\} & \text{if } r_{x,y} > 0 \text{ AND } r_{x+1,y} > 0 \\ \left\{r_{x,y} + k, r_{x+1,y} + k\right\} & \text{if } r_{x,y} < 0 \text{ AND } r_{x+1,y} < 0 \\ \left\{r_{x,y}, r_{x+1,y}\right\} & \text{otherwise} \end{cases} \tag{7}$$

with $k = 2\times\min\left(\left|r_{x,y}\right|, \left|r_{x+1,y}\right|\right)$. We call the function in Eq. (7) shifting pwm (spwm) function. Note that since the spwm function is applied to blocks containing residual values of the same sign, the first condition in Eq. (7) is used for *positive* residual blocks, i.e., those with positive values and zeros; while the second condition is used for *negative* residual blocks, i.e., those with negative values and zeros. In order to recover a residual block modified by the spwm function, we must apply the inverse spwm function on the same pairs of residuals:

$$\left\{r'_{x,y}, r'_{x+1,y}\right\} = f_i\left(\left\{r^f_{x,y}, r^f_{x+1,y}\right\}, j\right) = \begin{cases} \left\{r^f_{x,y} - j, r^f_{x+1,y} - j\right\} & \text{if } r^f_{x,y} \neq 0 \text{ AND } r^f_{x+1,y} \neq 0 \\ \left\{r^f_{x,y}, r^f_{x+1,y}\right\} & \text{otherwise} \end{cases} \tag{8}$$

with $j = 2\times\min\left(r^f_{x,y}, r^f_{x+1,y}\right)$ for positive residual blocks and $j = 2\times\max\left(r^f_{x,y}, r^f_{x+1,y}\right)$ for negative residual blocks.

Note that the value of j in Eq. (8) is obtained from the final residual block, so there is no need to signal this value as side information. Signaling a single value indicating whether the spwm function was applied to a positive or negative residual block suffices.

Fig. 7 graphically illustrates the outcome of applying the pwm functions on a sample range of residual values. Indeed, the pwm functions reduce the range of values while centering it towards zero.

It is important to mention some of the similarities and differences of the proposed pwm functions with other proposed methods that also map samples to different values. Particularly, the sample adaptive offset (SAO) also maps samples by adding an offset. However, SAO maps reconstructed samples with the objective of reducing artifacts resulting from quantization errors of transform coefficients in lossy compression. SAO is an in-loop filtering technique that reduces the mean sample distortion of a region by first classifying the region samples into multiple categories with a selected classifier, obtaining an offset for each category, and then adding the offset to each sample of the category. The classifier index and the offsets of the region are coded in the bit-stream. The proposed pwm functions, differently from SAO, maps samples with the objective of reducing the energy of residual blocks to improve lossless coding efficiency.

B. Selection of piecewise mapping functions at encoder

In this work, we apply the lpwm, dpwm or spwm function to residual blocks to further reduce their energy. To this end,

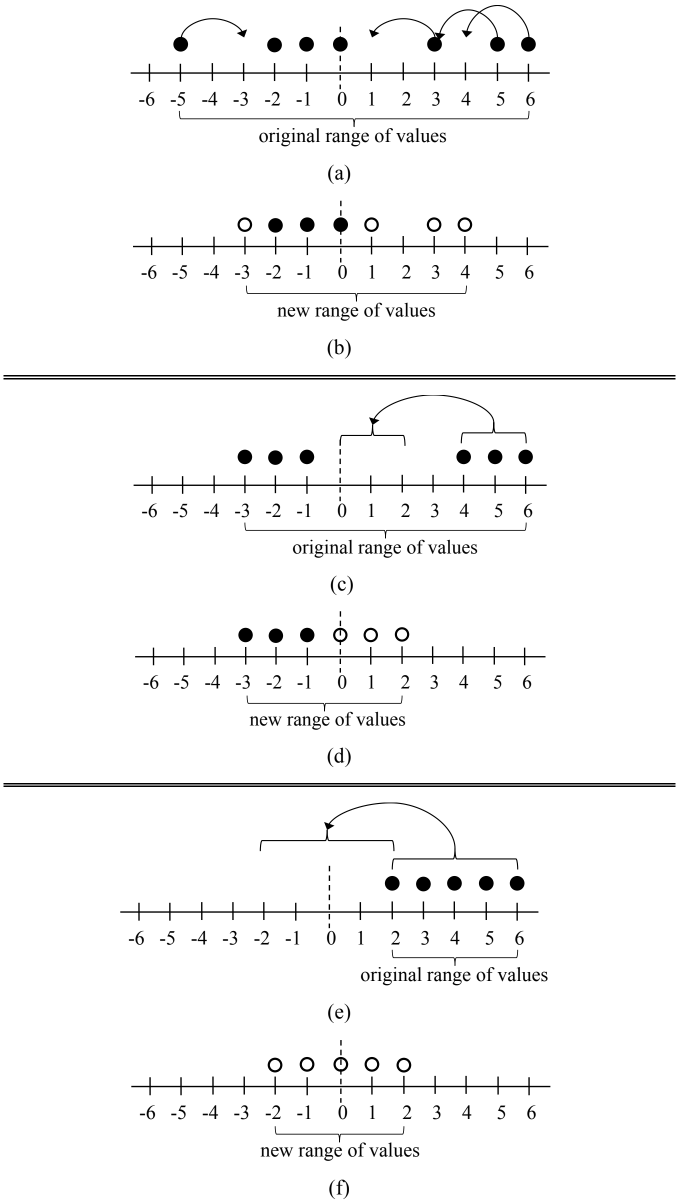

Fig. 7. Outcome of applying the (a)-(b) lpwm, the (c)-(d) dpwm, and the (e)-(f) spwm function on a sample range of residual values. Filled circles indicate values that exist in the residual block, while unfilled circles indicate mapped values.

we first classify residual blocks according to their range of values. We employ seven different categories, as tabulated in column 1 of Table II. Based on this classification process, we then apply a specific pwm function to the residual block, as tabulated in column 5 of Table II.

For blocks that comprise a mix of positive, negative and zero-valued residuals (i.e., Z-mixed blocks), we employ a lpwm function. To reduce the number of parameters used to define the lpwm function, we use the absolute value of residuals to find those values that do not appear within the range of values of the residual block. This allows defining the lpwm function with only two parameters, t and h. For example, the sample residual block illustrated in Fig. 3(a) comprises residuals with absolute values in the set $v = \{0, 1, 2, 3, 5, 7, 8, 10\}$; no residuals with absolute values in the set $mv = \{4, 6, 9\}$ appear in the residual block. Therefore, the

TABLE II. CLASSIFICATION OF RESIDUAL BLOCKS

Block type	Values in block			Piecewise Mapping Function
	Zero	Positive	Negative	
Z	✓			no function
Z-positive	✓	✓		spwm
Z-negative	✓		✓	spwm
Z-mixed	✓	✓	✓	lpwm
NZ-positive		✓		spwm
NZ-negative			✓	spwm
NZ-mixed		✓	✓	dpwm

corresponding lpwm function may be defined with parameters $t = 1$ and $h = 3$ as follows:

$$r^f_{x,y} = f\left(r_{x,y}, t, h\right) = \begin{cases} r_{x,y} - t & \text{if } r_{x,y} > h \\ r_{x,y} + t & \text{if } r_{x,y} < -h \\ r_{x,y} & \text{otherwise} \end{cases} \tag{9}$$

with $h = fmv(v) - 1$ and $t = nv(v) - fmv(v)$, where $fmv(a)$ returns the first missing integer > 0 in array a (integers in a are arranged in ascending order) and $nv(a)$ returns the first integer $> fmv(a)$ in a.

For blocks that comprise a mix of positive and negative residuals with no zero-valued residuals (i.e., NZ-mixed blocks), we employ the dpwm function. For blocks that comprise only values of the same sign, including zeros (i.e., Z-positive, Z-negative, NZ-positive and NZ-negative blocks), we employ the spwm function. No piecewise mapping function is employed for blocks that comprise only zero-valued residuals, i.e., Z blocks.

At the encoder, the best intra-prediction mode for a Prediction Block (PB) is selected based on the final residual block obtained after applying a pwm function taking into account the associated overhead to signal parameters. In order to avoid increasing the complexity of the rate distortion optimization process, the pwm applied to a PB is selected according to the categories tabulated in Table II before this optimization process. Let P_m denote the $N\times N$ predicted block obtained by employing intra-prediction mode m, let S denote the original block, and let $r_m = S - P_m$ denote the corresponding residual block. The pwm function is applied by modifying P_m, which results in the corresponding final residual block r^f_m:

$$r^f_m = S - \left(P_m \pm \mathbf{pwm}\right) \tag{10}$$

where **pwm** represents an $N\times N$ block containing the values that need to be added (or subtracted) to each value in P_m. The final residual block r^f_m is then used by the encoder to evaluate the coding cost of mode m, taking into account the overhead associated with signaling the necessary parameters. Therefore, the complexity of the rate distortion optimization process is minimally affected, as this process evaluates r^f_m in a similar fashion as it would evaluate r_m. The only increase in complexity is due to the operations performed to compute, analyze and classify r_m according to the categories in Table II, the operations needed to apply the pwm function and any additional memory to store the values in the **pwm** block.

C. *Overhead associated with piecewise mapping functions*

Information needed to reconstruct the residual blocks whose energy values are reduced by the proposed pwm functions is signaled to the decoder as a unique *mapping* value. In this work, the level of granularity at which the mapping values are signaled is at the PU level, as the pwm functions are applied to PBs.

The unique combination of t and h values in the lpwm function is signaled by a single mapping value [see Eq. (9)]. In this work, we limit t to the range [1, 8] and h to the range [0, 6], which results in 56 distinct mapping values. The usage of the dpwm function is signaled by a mapping value representing the value of i or j [see Eq. (5)]. In this work, we limit i and j to the ranges [1, 7] and [1, 6], respectively, which results in 13 distinct mapping values. Note that by signaling the usage of the dpwm function with a value of i or j, the values of e_p and e_n are implicitly signaled. For example, applying the dpwm function with a specific value of i, implies that $e_p \geq e_n$. We signal the usage of the spwm function by a single mapping value indicating whether the function is applied to a positive or negative residual block. One of 71 different mapping values should then be signaled to the decoder for each block modified by a pwm function.

It is important to mention that the range of values of the parameters associated with the pwm functions are selected based on the assumption that intra-prediction produces residual blocks with values that tend to follow a Laplacian distribution peaked at zero, with long tails. The long tails correspond to a small number of residual values produced by inaccurate predictions.

We entropy coded mapping values using two different contexts. We first encode the flag *pwm* indicating if the residual block has been modified by a pwm function (pwm = 1) or not (pwm = 0). This flag is entropy coded within context φ_{pw}. If pwm = 0, the intra-prediction mode index, m, is entropy encoded as is currently done in HEVC. If pwm = 1, the mapping value is first compared against n = 8 most probable mapping values (MPMVs). If the mapping value is equal to one of the MPMVs, the flag mp = 1 is entropy encoded within context φ_{mp}, and the MPMV is entropy encoded using three bits with equal probability. MPMVs are common knowledge to both encoder and decoder. If the mapping mode is not equal to one of the MPMVs, the flag mp = 0 is entropy encoded within context φ_{mp}. The mapping value is then entropy coded using 6 bits with equal probability.

The entropy encoding procedure for mapping values is embodied in Algorithm 1. The mapping value, denoted by mv takes integers in the range [0, 72]; with mv = 0 signaling that no pwm function is applied to the residual block. The array MPVM[n] stores the most probable mapping values in descending order. The encodeBin(*bin*, *ctx*) procedure in lines 3, 7, 11 and 17 codes the single binary symbol *bin* within context *ctx*. The HEVCencode(*int*) procedure in line 4 codes the positive integer *int* using the current entropy coding method in HEVC for intra-prediction mode indices. The encodeBinsEP(*int*, *bins*) procedure in lines 12 and 23 codes the positive integer *int* using *bins* bits with equal probability.

Algorithm 1. Entropy coding of mapping values
Initialization: $mp \leftarrow 0$

```
 1: if mv = 0 then
 2:     pwm ← 0
 3:     encodeBin (pwm, φ_pw)
 4:     HEVCencode (m)
 5: else
 6:     pwm ← 1
 7:     encodeBin (pwm, φ_pw)
 8:     for n ϵ [1,8] do
 9:         if mv = MPMV[n] then
10:             mp ← 1
11:             encodeBin (mp, φ_mp)
12:             encodeBinsEP(n−1, 3)
13:             break
14:         end if
15:     end for
16:     if mp = 0 then
17:         encodeBin (mp, φ_mp)
18:         for n ϵ [1,8] do
19:             if mv > MPMV[n] then
20:                 mv ← (mv −1)
21:             end if
22:         end for
23:         encodeBinsEP(mv−1, 6)
24:     end if
25: end if
```

IV. Evaluation Results

This section presents two sets of evaluation experiments. The first set (Section IV.A) compares the proposed pwm functions and a number of DPCM-based methods against HEVC block-wise intra-prediction and RDPCM, which is a DPCM-based intra-prediction method standardized in HEVC. The second set (Section IV.B) compares the proposed pwm functions against the IntraBC method, which is introduced in HEVC as part of the screen content coding (SCC) extensions. In all experiments, we specifically apply the pwm functions to 4×4 residual blocks computed after DPCM-based intra-prediction. As mentioned in Section II, DPCM-based intra-prediction has been shown to provide important bit-rate reductions. Moreover, the prediction accuracy tends to increase when performed on 4×4 blocks. Therefore, the amount of zero-valued residuals is expected to increase after DPCM-based intra-prediction in these blocks. Consequently, their distribution of residual values is expected to follow a Laplacian distribution peaked at zero, with a small number of inaccurate predictions. These residual blocks are therefore well suited for the pwm functions.

All experiments are performed in lossless coding mode using only intra-prediction. All evaluated methods are implemented by modifying the HEVC reference software HM-16.6+SCM5.0 [29]. Results are provided in terms bit-rate differences, in percentage, and coding and decoding times. We specifically evaluate video sequences classified in four classes (B, F, ScreenContent and RangeExtensions) covering various resolutions in 4:2:0, 4:2:2 and 4:4:4 format. Class B sequences include camera-captured material at 8 bits-per-pixel (bpp). Class F sequences include camera-captured material and

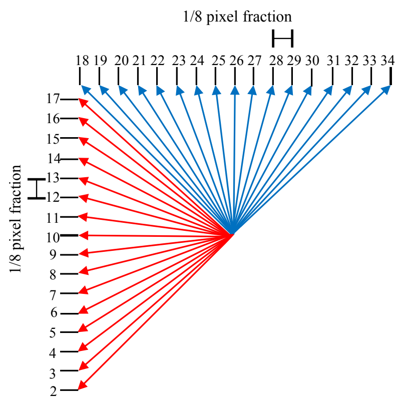

Fig. 8. Directions associated with SAP1. Eight angles are defined for each octant with an equal displacement of 1/8 pixel fraction for all modes.

computer screen content at 8 bpp. ScreenContent (SC) sequences include a wide variety of computer graphics and computer screen content at 8 bpp and 10 bpp [30]. RangeExtension (RExt) sequences include camera-captured material at 10 bpp.

A. Comparisons with block-wise intra-prediction and RDPCM

In this first set of evaluation experiments, we specifically compare the following DPCM-based methods:

- RDPCM: DPCM-based intra-prediction applied on the residual signals in the horizontal (or vertical) direction if the block-wise horizontal (or vertical) mode is used. RDPCM is standardized in HEVC [20].
- SAP: DPCM-based intra-prediction applied to all angular modes. Eight angles are defined for each octant with associated displacement parameters, as shown in Fig. 1. DC and PLANAR modes are implemented using block-wise intra-prediction [11].
- SAP-HV: DPCM-based intra-prediction applied only to the pure horizontal (mode 10) and pure vertical (mode 26) directions. The rest of the modes are implemented using block-wise intra-prediction [12].
- SAP1: DPCM-based intra-prediction applied to all angular modes. Eight angles are defined for each octant with an equal displacement of 1/8 pixel fraction for all modes, as shown in Fig. 8. DC and PLANAR modes are implemented using block-wise intra-prediction [13]. Note that compared with the angular modes in SAP (see Fig. 1), the angular modes in SAP1 (see Fig. 8) are more uniformly distributed in the horizontal and vertical directions.
- SAP-E: DPCM-based intra-prediction applied to all modes (including the DC mode). Angular modes are defined using the displacement parameters of SAP. Edge predictor in Eq. (2) is used in lieu of the PLANAR mode [14].
- R-EDPCM: Edge predictor in Eq. (2) is applied to entire residual frames after block-wise intra-prediction [22].
- SAP+SWP2+DTM: DPCM-based intra-prediction applied to all angular modes. Angular modes are defined as in SAP. SWP2 algorithm is used in lieu of the PLANAR mode with DTM as the exception algorithm. DTM is also used in lieu of the DC mode [17].
- RDPCM+pwm: the proposed pwm functions applied on 4×4 residual blocks obtained after RDPCM.
- SAP-HV+pwm: the proposed pwm functions applied on 4×4 residual blocks obtained after DPCM-based prediction (mode 10 or mode 26).
- SAP-E+pwm: the proposed pwm functions applied on 4×4 residual blocks obtained after DPCM-based prediction using mode 0 [see Eq. (2)].

Coding tools introduced in SCM5.0 specifically aimed at improving screen content coding, such as palette mode, cross-component prediction, adaptive color transforms and IntraBC, are not used in this set of evaluation experiments in order to determine the coding improvements obtained exclusively by the pwm functions.

Table III summarizes the performance achieved by each method in terms of the bit-rate differences with respect to HEVC block-wise intra-prediction, in percentage. We also provide bit-rate differences with respect to RDPCM in parenthesis. For those methods using DPCM+pwm, we also provide bit-rate differences with respect to the corresponding method that uses no pwm functions. These bit-rate differences are provided in square brackets. Negative numbers indicate a decrease in bit-rate.

According to Table III, R-EDPCM attains, overall, the minimum average bit-rate reductions. Let us recall that R-EDPCM attempts to remove horizontal and vertical edges on the entire residual frame after block-wise intra-prediction. This is done without considering the associated coding cost of performing this additional prediction. Therefore, a wrong prediction in the sign of residual values can significantly increase energy values. Consequently, coding efficiency may be negatively affected. This is particularly evidenced by the low performance of R-EDPCM for SC sequences, particularly compared to RDPCM.

RDPCM and SAP-HV attain very similar performances compared to block-wise intra-prediction. This is expected, as RDPCM is mathematically identical to SAP-HV. The small performance differences are due to two important factors. First, boundaries of blocks are filtered in RDPCM before applying DPCM-based prediction. Second, SAP-HV applies DPCM-based prediction to the original signal, which leads to the selection of different prediction modes by the encoder [12, 20].

SAP and SAP1 also attain similar performances, with SAP1 performing slightly better for the majority of the test sequences. The further improvements brought about by SAP1 are mainly due to exploiting pixel correlations by using a more uniform distribution of angular modes.

SAP-E outperforms SAP and SAP+SWP2+DTM for the majority of the test sequences. Compared to SAP+SWP2+DTM, SAP-E attains further average bit-rate reductions of up to 2.84% and 3.04%, with respect to block-wise intra-prediction and RDPCM, respectively (see average results for 4:2:0 SC sequences). Although SAP+SWP2+DTM provide a powerful predictor on top of SAP, the test 4:2:0 SC sequences comprise a mix of screen content material and camera-captured material. The absence of a DC mode in SAP+SWP2+DTM affects the coding performance on the camera-captured material. A similar performance is observed for Class B sequences. Compared to SAP, SAP-E is capable of

TABLE III. LOSSLESS CODING PERFORMANCE OF VARIOUS DPCM-BASED METHODS IN TERMS OF BIT-RATE DIFFERENCES WITH RESPECT TO BLOCK-WISE INTRA-PREDICTION (AND RDPCM)

Sequence name - bpp	Average Bit-rate Difference – %									
	RDPCM	SAP-HV	SAP	SAP1	SAP-E	SAP+SWP2 +DTM	R-EDPCM	RDPCM+ pwm*	SAP-HV+ pwm*	SAP-E+ pwm*
SC sequences – 4:4:4										
fliyingGraphics - 8 bpp	-8.22	-8.79 (-0.62)	-9.49 (-1.39)	-9.58 (-1.48)	**-13.34** **(-5.59)**	-12.14 (-4.27)	2.12 (11.26)	-8.81 (-0.65)	-11.66 (-3.75) **[-3.15]**	**-15.48** **(-7.88)** [-2.43]
programming - 8 bpp	-9.90	-9.92 (-0.02)	-9.99 (-0.10)	-10.01 (-0.13)	**-11.90** **(-2.22)**	-11.06 (-1.29)	-3.29 (7.33)	-11.19 (-1.29)	-14.09 (-4.52) **[-4.50]**	**-14.81** **(-5.32)** [-3.16]
desktop - 8 bpp	-9.31	-10.15 (-0.92)	-10.78 (-1.62)	-11.14 (-2.01)	**-16.07** **(-7.46)**	-14.75 (-6.00)	-1.75 (8.33)	-10.79 (-0.60)	-15.11 (-5.41) **[-4.53]**	**-20.27** **(-11.17)** [-4.01]
map - 8 bpp	-8.05	-8.20 (-0.16)	-10.68 (-2.86)	-10.71 (-2.89)	**-13.07** **(-5.46)**	-12.02 (-4.32)	-2.33 (6.21)	-9.57 (-1.62)	-12.09 (-4.37) **[-4.21]**	**-14.97** **(-7.50)** [-2.16]
console - 8 bpp	-9.37	-10.02 (-0.72)	-11.09 (-1.90)	-11.46 (-2.31)	**-24.06** **(-16.21)**	-16.91 (-8.32)	-8.51 (0.95)	-10.92 (-0.75)	-15.83 (-6.22) **[-5.54]**	**-28.33** **(-20.15)** [-4.71]
robot - 8 bpp	-7.20	-7.91 (-0.76)	-9.25 (-2.21)	-9.26 (-2.22)	**-10.36** **(-3.40)**	-9.46 (-2.43)	-6.77 (0.46)	-9.70 (-1.99)	-11.01 (-3.42) **[-2.67]**	**-11.97** **(-4.46)** [-1.09]
kimono - 10 bpp	-2.02	-2.03 (-0.01)	-3.79 (-1.81)	-3.83 (-1.85)	**-5.33** **(-3.39)**	-4.60 (-2.64)	-2.80 (-0.80)	-4.10 (-1.24)	-4.59 (-1.75) [-1.74]	**-8.12** **(-5.39)** **[-2.07]**
web - 8 bpp	-11.28	-13.73 (-2.76)	-13.34 (-2.33)	-13.59 (-2.61)	**-15.83** **(-5.13)**	-15.55 (-4.82)	-6.89 (4.95)	-12.36 (-1.09)	-16.34 (-5.58) [-2.90]	**-18.42** **(-7.92)** **[-2.94]**
Avg. SC – 4:4:4	-8.17	-8.84 (-0.75)	-9.79 (-1.76)	-9.95 (-1.94)	**-13.75** **(-6.11)**	-12.06 (-4.26)	-3.78 (4.84)	-9.68 (-1.16)	-12.59 (-4.38) **[-3.66]**	**-16.55** **(-8.72)** [-2.82]
SC sequences – 4:2:0										
missionControl3 - 8 bpp	-5.96	-6.69 (-0.78)	-8.76 (-2.98)	-8.88 (-3.10)	**-11.05** **(-5.41)**	-9.42 (-3.68)	-4.76 (1.28)	-8.11 (-0.96)	-10.55 (-3.59) **[-2.84]**	**-13.61** **(-6.89)** [-1.56]
slideShow - 8 bpp	-8.93	-9.59 (-0.72)	-12.69 (-4.13)	-12.74 (-4.18)	**-18.92** **(-10.97)**	-14.07 (-5.65)	-14.06 (-5.63)	-13.04 (-1.88)	-14.72 (-3.77) **[-3.07]**	**-23.17** **(-13.31)** [-2.63]
basketscreen - 8 bpp	-8.74	-9.59 (-0.94)	-11.55 (-3.08)	-11.74 (-3.30)	**-13.46** **(-5.18)**	-12.83 (-4.48)	-5.53 (3.51)	-11.26 (-2.01)	-14.19 (-5.25) **[-4.36]**	**-17.26** **(-8.64)** [-3.65]
missionControl2- 8 bpp	-4.12	-5.67 (-0.62)	-5.68 (-1.62)	-5.74 (-1.69)	**-10.00** **(-6.13)**	-5.75 (-1.70)	-5.99 (-1.95)	-5.90 (-0.14)	-7.47 (-1.80) [-1.19]	**-12.73** **(-7.39)** **[-1.33]**
Avg. SC – 4:2:0	-6.94	-7.88 (-0.76)	-9.67 (-2.95)	-9.78 (-3.07)	**-13.36** **(-6.92)**	-10.52 (-3.88)	-7.59 (-0.70)	-9.58 (-1.25)	-11.73 (-3.61) **[-2.86]**	**-16.69** **(-9.05)** [-2.29]
Class B sequences – 4:2:0										
parkScene - 8 bpp	-3.01	-3.38 (-0.38)	-4.95 (-2.00)	-4.97 (-2.02)	**-8.27** **(-5.42)**	-6.12 (-3.21)	-6.50 (-3.59)	-5.31 (-0.37)	-6.88 (-2.02) **[-1.65]**	**-11.54** **(-6.92)** [-1.59]
kimono - 8 bpp	-2.31	-2.32 (-0.01)	-4.42 (-2.17)	-4.47 (-2.22)	**-7.57** **(-5.39)**	-5.84 (-3.62)	-5.37 (-3.14)	-5.07 (-0.64)	-5.44 (-1.03) [-1.01]	**-11.43** **(-7.30)** **[-2.02]**
Avg. Class B – 4:2:0	-2.66	-2.85 (-0.20)	-4.69 (-2.08)	-4.72 (-2.12)	**-7.92** **(-5.40)**	-5.98 (-3.41)	-5.93 (-3.37)	-5.19 (-0.50)	-6.16 (-1.52) [-1.33]	**-11.48** **(-7.11)** **[-1.80]**
Class F sequences – 4:2:0										
basketDrill - 8 bpp	-1.36	-1.38 (-0.01)	-5.77 (-4.47)	-5.93 (-4.63)	**-6.38** **(-5.08)**	-6.19 (-4.89)	1.31 (2.71)	-3.33 (-0.28)	-4.22 (-1.20) [-1.18]	**-10.19** **(-7.36)** **[-2.40]**
slideEditing - 8 bpp	-7.48	-8.03 (-0.60)	-9.50 (-2.18)	-9.76 (-2.46)	**-11.80** **(-4.66)**	-11.15 (-3.97)	-0.67 (7.37)	-8.20 (-0.62)	-11.36 (-4.04) **[-3.46]**	**-13.83** **(-6.72)** [-2.15]
chinaSpeed - 8 bpp	-10.32	-11.00 (-0.76)	-13.87 (-3.96)	-14.00 (-4.10)	**-14.42** **(-4.58)**	-14.36 (-4.51)	-8.41 (2.13)	-14.22 (-2.77)	-15.95 (-4.73) **[-4.00]**	**-18.89** **(-8.06)** [-3.65]
Avg. Class F – 4:2:0	-6.39	-6.80 (-0.46)	-9.71 (-3.54)	-9.89 (-3.73)	**-10.87** **(-4.78)**	-10.57 (-4.46)	-2.59 (4.07)	-8.58 (-1.23)	-10.51 (-3.32) **[-2.88]**	**-14.30** **(-7.38)** [-2.73]
RExt sequences – 4:4:4										
EBURainFruits - 10 bpp	-3.40	-3.54 (-0.15)	-5.40 (-2.08)	-9.37 (-6.18)	**-5.44** **(-2.11)**	-7.92 (-4.68)	-7.23 (-3.96)	-5.79 (-1.62)	-6.65 (-2.52) [-2.38]	**-10.68** **(-6.73)** **[-4.71]**
BirdsCage - 10 bpp	-0.19	-0.20 (-0.01)	-0.90 (-0.71)	**-0.91** **(-0.72)**	-0.86 (-0.67)	1.00 (1.20)	0.12 (0.31)	-0.94 (-0.22)	-1.18 (-0.46) [-0.45]	**-2.68** **(-1.97)** **[-1.31]**
Avg. RExt – 4:4:4	-1.80	-1.87 (-0.08)	-3.15 (-1.39)	**-5.14** **(-3.45)**	-3.15 (-1.39)	-3.46 (-1.74)	-3.55 (-1.82)	-3.36 (-0.92)	-3.92 (-1.49) [-1.42]	**-6.68** **(-4.35)** **[-3.01]**
RExt sequences – 4:2:2										
EBUHorse - 10 bpp	-1.07	-1.10 (-0.02)	-2.18 (-1.12)	-2.20 (-1.14)	**-2.89** **(-1.83)**	-1.38 (-0.31)	-1.20 (-0.13)	-2.09 (-0.34)	-3.61 (-1.89) [-1.87]	**-5.97** **(-4.29)** **[-2.51]**
EBUWaterRocks – 10 bpp	-0.89	-0.90 (-0.01)	-1.91 (-1.02)	-1.93 (-1.04)	**-2.32** **(-1.44)**	-1.06 (-0.17)	-0.56 (0.33)	-1.93 (-0.34)	-2.91 (-1.33) [-1.32]	**-4.83** **(-3.28)** **[-1.87]**
Avg. RExt – 4:2:2	-0.98	-1.00 (-0.02)	-2.04 (-1.07)	-2.07 (-1.09)	**-2.60** **(-1.64)**	-1.22 (-0.24)	-0.88 (0.10)	-2.01 (-0.34)	-3.26 (-1.61) [-1.59]	**-5.40** **(-3.79)** **[-2.19]**

* Piecewise mapping functions are applied to 4×4 residual blocks.

Results in parenthesis indicate bit-rate differences (%) with respect to RDPCM. Results in square brackets indicate bit-rate differences [%] with respect to the corresponding method with no piecewise mapping.

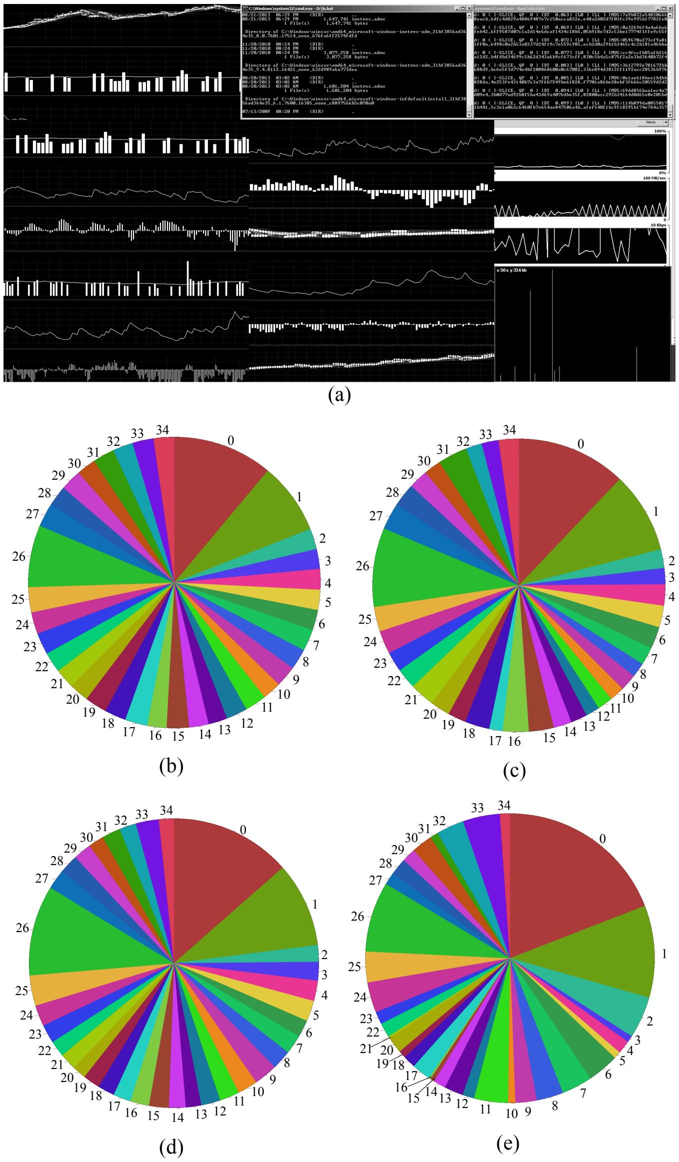

Fig. 9. (a) Red (R) component of one frame of the *console* sequence, and corresponding distribution of modes when the component is encoded using (b) SAP-HV, (c) SAP-HV+pwm, (d) SAP-E and (e) SAP-E+pwm. Each color represents the percentage of PUs predicted using a particular mode (best viewed in color). Note that in SAP-HV+pwm, mode 26 is more frequently selected compared to the case of using no piecewise mapping (SAP-HV). Similarly, mode 0 is more frequently selected in SAP-E+pwm compared to using no piecewise mapping (SAP-E).

providing further average bit-rate reductions of up to 3.96% and 4.34%, with respect to block-wise intra-prediction and RDPCM, respectively (see average results for 4:4:4 SC sequences). These results confirm the advantages of using the edge predictor in Eq. (2) and a DPCM-based DC mode.

The techniques employing pwm functions achieve the maximum bit-rate reductions compared to block-wise intra-prediction and RDPCM. SAP-E+pwm attain the best performance for all of the test sequences. This is expected, as SAP-E provides the maximum bit-rate reductions among the methods not using piecewise mapping. All DPCM+pwm techniques achieve higher bit-rate reductions than their counterparts not employing the pwm functions (see results in square brackets in Table III). These bit-rate reductions are higher for SAP-HV+pwm and SAP-E+pwm than for RDPCM+pwm. Although SAP-HV and RDPCM are mathematically identical, pwm is applied in RDPCM+pwm after computing residual blocks using RDPCM. As a consequence, the rate distortion optimization process in RDPCM+pwm does not evaluate the final residuals obtained after piecewise mapping. This leads to the selection of different prediction modes by the encoder. In the case of SAP-HV+pwm, the rate distortion optimization process evaluates the final residuals after piecewise mapping. This makes the horizontal and vertical modes attractive options to be selected as the best mode. Consequently, in SAP-HV+pwm modes 10 and 26 tend to be more frequently used that in SAP-HV. A similar situation occurs in SAP-E+pwm, where mode 0 tends to be more frequently used than in SAP-E. Fig. 9 shows the distribution of modes for the depicted red (R) component of a frame of the *console* sequence. Each color represents the percentage of PUs predicted using a particular mode. The *console* sequence is the one for which SAP-HV+pwm and SAP-E+pwm attain the maximum bit-rate reductions compared to SAP-HV and SAP-E, respectively. Indeed, after applying piecewise mapping, the frequency of mode 26 increases in SAP-HV+pwm compared to SAP-HV [see Figs. 9(b) and (c)]. Notice also an increase in the frequency of mode 0. In this case, the overhead associated with piecewise mapping when applied to modes 26 and 10 makes mode 0 more cost-effective and therefore, its frequency increases. For the case of SAP-E-pwm, mode 0 is more frequently selected compared to SAP-E, as shown in Fig. 9(d) and (e).

Overall, the DPCM+pwm techniques attain a bit-rate reduction of up to 5.54% compared to the case of DPCM-based prediction using no piecewise mapping (see results for SAP-HV+pwm for *console* sequence). Compared to block-wise intra-prediction and RDPCM, DPCM+pwm techniques attain a bit-rate reduction of up to 28.33% and 20.15%, respectively (see results for SAP-E+pwm for *console* sequence).

It is important to mention that the pwm functions can be applied to all PU sizes. As mentioned before, 4×4 residual blocks are well suited for these functions because of the range of residual values generated as a consequence of a more accurate prediction. Compared to SAP-E+pwm applied to only 4×4 blocks, our evaluation results show that SAP-E+pwm applied to all PU sizes results in further average bit-rate reductions of 0.26%, 0.05%, 0.02%, 0.06%, 0.02%, and 0.03% for 4:4:4 SC, 4:2:0 SC, Class B, Class F, 4:4:4 RExt, and 4:2:2 RExt sequences, respectively. The average encoding time differences are of 1.44% 1.11%, 1.70%, 1.62%, 1.71%, and 1.65% respectively. Similar further average bit-rate reductions and encoding time differences are observed for RDPCM+pwm and SAP-HV+pwm, when applied to all PU sizes.

B. Comparisons with IntraBC

TABLE V. AVERAGE ENCODING/DECODING TIME RATIOS OF ALL EVALUATED METHODS WITH RESPECT TO BLOCK-WISE INTRA-PREDICTION (AND RDPCM)

Method	Average encoding/decoding times ratios – %					
	SC – 4:4:4	SC – 4:2:0	Class B – 4:2:0	Class F – 4:2:0	RExt – 4:4:4	RExt – 4:2:2
RDPCM	101/100	100/100	101/100	100/100	101/100	102/100
SAP-HV	96/100 (96/99)	97/100 (97/99)	98/99 (97/98.34)	97/99 (97/99)	96/99 (95/99)	97/99 (95/98)
SAP	95/99 (95/97)	96/99 (95/98)	98/99 (97/97)	96/99 (96/98)	95/98 (93/97)	96/98 (95/99)
SAP1	96/99 (95/96)	96/98 (96/98)	99/99 (98/98)	96/98 (96/98)	94/98 (93/97)	97/100 (95/90)
SAP-E	91/99 (91/94)	92/99 (92/97)	95/99 (95/98)	93/98 (93/97)	95/98 (94/98)	95/99 (93/98)
SAP+SWP2+DTM	274/193 (272/190)	273/195 (272/187)	281/195 (280/193)	266/197 (267/195)	271/198 (268/197)	260/194 (255/194)
R-EDPCM	123/112 (122/112)	129/116 (129/115)	129/113.62 (128/114)	127/117 (127/114)	120/119 (119/116)	124/117 (122/115)
IntraBC	146/98 (148/98)	150/98 (153/99)	188/100 (195/99)	168/98 (173/99)	146/97 (147/97.23)	257/99 (261/99)
RDPCM+ pwm*	104/103 (105/103)	104/104 (105/104)	105/103 (106/104)	104/103 (105/103)	107/102 (109/101)	105/102 (106/103)
SAP-HV+ pwm*	102/102 (103/103) [105/105]	103/103 (104/104) [106/105]	104/103 (105/104) [106/104]	103/102 (104/103) [107/104]	104/104 (105/104) [108/105]	102/104 (103/104) [107/105]
SAP-E+ pwm*	103/102 (104/103) [107/104]	103/102 (105/104) [108/106]	105/103 (106/105) [108/105]	103/103 (105/104) [106/106]	103/103 (104/104) [106/105]	105/103 (107/105) [109/107]

* Piecewise mapping functions are applied to 4×4 residual blocks.
Results in parenthesis indicate average encoding/decoding time ratios (%) with respect to RDPCM. Results in square brackets indicate average encoding/decoding time rations [%] with respect to the corresponding method with no piecewise mapping.

In this second set of evaluation experiments, the search range for IntraBC is set to the entire previously encoded region of the current frame. All frames are encoded using intra-prediction in lossless mode. Other coding tools introduced in the SCC extensions, such as palette mode, cross-component prediction and adaptive color transforms, are not used in order to determine the coding improvements obtained exclusively by IntraBC. Let us recall that IntraBC allows predicting PUs by using any previously encoded region as reference. Table IV tabulates average bit-rate differences of DPCM+pwm techniques with respect to IntraBC, in percentage. Since IntraBC is specifically designed to exploit the high occurrence of repeating patterns in SC sequences, this method is expected to provide the best performance for this class of sequences. The bit-rate attained by IntraBC is indeed much lower than that attained by DPCM+pwm techniques for these sequences. This comes, however, at the expense of considerably increasing encoding times, as it is later shown in Section IV.C. For sequences where repeating patterns are not commonly found, DPCM+pwm techniques outperform IntraBC. Specifically, SAP-E+pwm attains average bit-rate reductions of up to 11.48%, 6.79% and 5.38% for Class B, 4:4:4 RExt and 4:2:2 RExt sequences, respectively. For 4:2:0 SC sequences, SAP-E+pwm attains a very similar coding performance as IntraBC.

It is important to mention that the pwm functions are amenable to be used on top of other coding tools introduced in the SCC extensions. Since the pwm functions are designed to be applied to residual blocks, their application can be extended to residual blocks obtained for example after cross-component prediction in 4:4:4 sequences [6].

C. Encoding and Decoding Times

Encoding and decoding times of any DPCM-based method for intra-prediction are expected to be longer than those of block-wise intra-prediction since prediction of each pixel requires several multiplications and additions. However, a more efficient prediction usually produces more residual blocks with values that tend to follow a Laplacian distribution peaked at zero. This consequently increases the amount of zero-valued samples, thus decreasing the encoding/decoding load in CABAC. Table V tabulates the average encoding/decoding time ratios (%) for all evaluated methods with respect to block-wise intra-prediction, for each class. Average encoding/decoding time ratios are also provided with respect to RDCPM in parenthesis. For those methods using DPCM+pwm, we also provide average encoding/decoding

TABLE IV. AVERAGE BIT-RATE DIFFERENCES OF DPCM+PWM TECHNIQUES COMPARED TO INTRABC

Sequence class	Average Bit-rate Difference (%)		
	RDPCM+pwm*	SAP-HV+pwm*	SAP-E+pwm*
SC – 4:4:4	66.70	60.29	51.27
SC – 4:2:0	8.39	5.78	0.04
Class B – 4:2:0	-5.19	-6.16	-11.48
Class F – 4:2:0	5.29	2.90	-1.28
RExt – 4:4:4	-3.48	-4.03	-6.79
RExt – 4:2:2	-1.99	-3.24	-5.38

*Piecewise mapping functions are applied to 4×4 residual blocks.

time ratios with respect to the corresponding method using no pwm functions. These time ratios are provided in square brackets.

Indeed, SAP-HV, SAP, SAP1 and SAP-E provide shorter average encoding times compared to RDPCM and block-wise intra-prediction coding. Although an increase in zero-valued samples also happens in SAP+SWP2+DTM, R-EDPCM, and DPCM+pwm, the amount of extra operations to be performed in these methods results in longer encoding/decoding times. The increase in encoding time in RDPCM is mainly due to the fact that DPCM-based prediction is performed as an extra coding step after block-wise intra-prediction. Note that the increase in encoding times is highest in IntraBC and SAP+SWP2+DTM. Long encoding times are expected for IntraBC, as this method performs a thorough search within the previously encoded region of the frame. Long encoding times are also expected for SAP+SWP2+DTM, as this method requires multiple multiplications and additions to be done in a causal neighborhood for each pixel to be predicted [15]. It is reported in [17] that SWP, a variant of SWP2 that does not employ DTM, can achieve average encoding/decoding times of 100.3%/89.2% in the All Intra-main profile for natural sequences if optimizations to the encoding process are introduced, such as the use of look-up tables to reduce the number of calculations. In these tests, no optimizations to the encoding process are employed in SAP+SWP2+DTM, and all required calculations are performed for each casual neighborhood.

The increase in encoding/decoding times of the DPCM+pwm techniques are relatively small as these techniques do not test several prediction modes on residual blocks, but rather, apply a specific pwm function to each residual block according to their range of residual values (see Table II).

D. Reconstruction structure of residuals

The application of the inverse pwm functions at the decoder can be parallelized to recover the residual blocks required to invert the DPCM-based prediction. Specifically, the inverse pwm functions can be applied on separate processing threads at the block level, as the application of these functions does not depend on the reconstruction of other blocks. Their application only depends on the mapping values signaled to the decoder. Additional memory is, therefore, needed at the decoder to store the parameters associated with the inverse pwm function. Once the residual blocks are reconstructed, the original signal can then be recovered by applying DPCM-based reconstruction. Note that some DPCM-based modes may require that samples be decoded sequentially and be readily available for the prediction and reconstruction of subsequent samples. In the case of RDPCM+pwm and SAP-HV+pwm, it is possible to apply a separate processing thread on rows or columns after reconstruction using the inverse pwm functions. This constitutes one of the main benefits provided by RDPCM and SAP-HV.

We finish this section with comments about the overhead associated with signaling mapping values to the decoder. For the results tabulated in Table III, this overhead represents an average of 2.03%, 1.87% and 1.97% increase in bit-rate for RDPCM+pwm, SAP-HV+pwm and SAP-E+pwm, respectively, compared to their counterparts not employing piecewise mapping. This shows that the pwm functions are very effective in reducing the energy of 4×4 blocks. The increase in bit-rate, as expected, is due to the large range of mapping values. More powerful coding techniques can be designed to reduce this overhead, such as those that predict mapping values based on previously used values.

V. CONCLUSIONS

In this paper, we presented a novel approach to reduce the energy of residual blocks in HEVC intra-prediction for lossless coding. The approach employs piecewise mapping (pwm) functions to map residual values to unique lower values. Piecewise mapping is applied on a block-by-block basis according to the range of values present in the residual block. The main objective is to improve lossless coding efficiency by increasing the number of residual blocks that comprise values that follow a Laplacian distribution peaked at zero. All associated parameters with the pwm functions are encoded and signaled to the decoder, so that mapped residual blocks are recovered with no loss.

We evaluated the performance of the pwm functions on 4×4 residual blocks computed by DPCM-based prediction. Evaluation results over a wide range of natural, screen content and range-extension sequences showed that piecewise mapping can attain maximum bit-rate reductions of 5.54% compared to DPCM-based prediction. When used in conjunction with DPCM-based prediction, the pwm functions can attain maximum bit-rate reductions of 28.33% over block-wise intra-prediction coding. Compared to IntraBC, piecewise mapping in conjunction with DPCM-based prediction was able to attain maximum bit-rate reductions of 11.48% for natural sequences. Evaluation results also indicated that the increase in encoding and decoding times incurred by the pwm functions is minimal with respect to block-wise intra-prediction coding and RDPCM.

The proposed pwm functions can be easily applied to residuals computed using inter-prediction. This may be achieved by following the same approach as the one followed for intra-predicted residuals. These extensions are part of our future work. Our future work also includes reducing the overhead associated with pwm functions by improving the encoding of parameters needed to reconstruct residuals after piecewise mapping.

REFERENCES

[1] G.J. Sullivan, J.R. Ohm, W.J. 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.

[2] G. Tech, Y. Chen, K. Muller, J.-R. Ohm, A. Vetro, and Y.-K. Wang, "Overview of the Multiview and 3D Extensions of High Efficiency Video Coding," *IEEE Transactions on Circuits and Systems for Video Technology,* vol. 26, no. 1, pp. 35-49, Jan. 2016.

[3] J.M. Boyce, Y. Ye, J. Chen, and A.K. Ramasubramonian, "Overview of SHVC: Scalable Extensions of the High Efficiency Video Coding Standard," *IEEE Transactions on Circuits and Systems for Video Technology,* vol. 26, no. 1, pp. 20-34, Jan. 2016.

[4] D. Flynn, D. Marpe, M. Naccari, T. Nguyen, C. Rosewarne, K. Sharman, J. Sole, and J. Xu, "Overview of the Range Extensions for the HEVC Standard: Tools, Profiles, and Performance", *IEEE Trans. on*

Circuits and Systems for Video Technology, vol. 26, no. 1, pp. 4-19, Jan. 2016.

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

[6] X. Jizheng, R. Joshi, and R.A. Cohen, "Overview of the Emerging HEVC Screen Content Coding Extension," *IEEE Transactions on Circuits and Systems for Video Technology,* vol. 26, no. 1, pp. 50-62, Jan. 2016.

[7] M. Mrak and J. Xu, "Improving screen content coding in HEVC by transform skipping," in *Proc. of the 20th European Signal Processing Conference (EUSIPCO)*, 2012, pp. 1209-1213.

[8] M. Budagavi and D.-K. Kwon, "Video coding using intra motion compensation," Joint Collaborative Team on Video Coding (JCT-VC), Doc. JCTVC-M0350, Incheon, Korea, Apr. 2013.

[9] S. Hu, R. Cohen, A. Vetro, and C.C. J. Kuo, "Screen content coding for HEVC using edge modes," in *Proc. 2013 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2013, pp. 1714-1718.

[10] H. Chen, A. Saxena and F. Fernandes, "Nearest-neighbor Intra Prediction for Screen Content Video Coding," in *Proc. 2014 IEEE Int. Conference on Image Processing (ICIP2014)*, pp. 3151-3155, Oct. 2014.

[11] M. Zhou, W. Gao, M. Jiang, and H. Yu, "HEVC lossless coding and improvements," *IEEE Trans. Circuits and Systems for Video Tech.*, vol. 22, no. 12, pp. 1839-1843, Dec. 2012.

[12] M. Zhou and M. Budagavi, "RCE2: Experimental results on Test 3 and Test 4," Joint Collaborative Team on Video Coding (JCT-VC), Doc. JCTVC-M0056, Incheon, Korea, Nov. 2013.

[13] V. Sanchez, J. Bartrina-Rapesta, F. Aulí-Llinàs, and J. Serra-Sagristà, "Improvements to HEVC Intra Coding for Lossless Medical Image Compression," in *Proc. 2014 Data Compression Conference (DCC)*, March 2014, pp. 423.

[14] V. Sanchez, F. Aulí-Llinàs, J. Bartrina-Rapesta and J. Serra-Sagristà, "HEVC-based Lossless Compression of Whole Slide Pathology Images," in *Proc. 2014 IEEE Global Conference on Signal and Information Processing (GlobalSIP)*, pp. 452-456, Dec. 2014.

[15] V. Sanchez and J. Bartrina-Rapesta, "Lossless Compression of Medical Images based on HEVC Intra Coding," *Proc. 2014 IEEE International Conference on Acoustics, Speech and Signal Processing* (*ICASSP2014*), pp. 6622-6626, Florence, Italy, May 2014.

[16] V. Sanchez, "Lossless screen content coding in HEVC based on sample-wise median and edge prediction," *Proc. 2015 IEEE Int. Conference on Image Processing* (*ICIP2015*), pp. 4604-4608, Sept. 2015.

[17] E. Wige, G. Yammine, P. Amon, A. Hutter, and A. Kaup, "Sample-based weighted prediction with directional template matching for HEVC lossless coding," in *Proc. 2013 Picture Coding Symposium*, pp. 305-308, Dec. 2013.

[18] A. Gabriellini, D. Flynn, M. Mrak and T.J. Davies, "Combined intra prediction for high efficiency video coding," *IEEE Journal of Selected Topics in Signal Processing*, vol. 5, no. 7, pp. 1282 - 1289, 2011.

[19] E. Wige, G. Yammine, P. Amon, A. Hutter, and A. Kaup, "Pixel-based averaging predictor for HEVC lossless coding," *Proc. 2013 IEEE International Conference on Image Processing (ICIP2013)*, pp. 1806-1810, Sept. 2013.

[20] S. Lee, I.-K. Kim, and C. Kim, "RCE2: Test 1 Residual DPCM for HEVC lossless coding," Joint Collaborative Team on Video Coding (JCT-VC), Doc. JCTVC-M0079, Incheon, Korea, Nov. 2013.

[21] M. Naccari, S. G. Blasi, M. Mrak, and E. Izquierdo, "Improving Inter-Prediction in HEVC with Residual DPCM for Lossless Screen Content Coding," in *Proc. 2013 Picture Coding Symposium (PCS)*, Dec. 2013, pp. 361-364.

[22] Y. H. Tan, C. Yeo and Z. Li. "Residual DPCM for lossless coding in HEVC," in *Proc. IEEE Int. Conf. Acoustics, Speech Signal Processing (ICASSP)*, pp. 2021–2025, May 2013.

[23] S.-W. Hong, J. H. Kwak, Y.-L. Lee, "Cross residual transform for lossless intra-coding for HEVC," *Signal Processing: Image Communication*, vol. 28, no. 10, pp. 1335-1341, Nov. 2013.

[24] G. Jeon, K. Kim, and J. Jeong, "Improved Residual DPCM for HEVC Lossless Coding," *Proc. 2014 Conf. on Graphics, Patterns and Images* (*SIBGRAPI2014*), pp. 95-102, 2014.

[25] K. Kim, G. Jeon, and J. Jeong. "Improvement of Implicit Residual DPCM for HEVC," *Proc. 2014 IEEE Int. Conf. Signal-Image Technology and Internet-Based Systems* (*SITIS2014*), pp. 652-658, 2014.

[26] K. Kim, G. Jeon, and J. Jeong, "Piecewise DC prediction in HEVC," Signal Processing: Image Communication, vol. 29, no. 9, pp. 945-950, Oct. 2014.

[27] V. Sanchez, "Fast Intra-prediction for Lossless Coding of Screen Content in HEVC," *Proc. 2015 IEEE Global Conference on Signal and Information Processing* (*GlobalSIP2015*), Orlando, Florida, USA, December 2015, *in-press*.

[28] M. Weinberger, G. Seroussi, and G. Sapiro, "The LOCO-I lossless image compression algorithm: Principles and standardization into JPEG-LS," *IEEE Trans. Image Process.*, vol. 9, no. 8, pp. 1309-1324, Aug. 2000.

[29] HM16.6+SCM-5.0 software. [Online]. Available: https://hevc.hhi.fraunhofer.de/svn/svn_HEVCSoftware/tags/HM-16.6+SCM-5.0rc1/

[30] H. Yu, R. Cohen, K. Rapaka, and J. Xu, "Common test conditions for screen content coding," Joint Collaborative Team on Video Coding (JCT-VC), Doc. JCTVC-T1015, Geneva, CH, Feb. 2015.