

# Transform Coding in the VVC Standard

Xin Zhao, *Senior Member, IEEE*, Seung-Hwan Kim, Yin Zhao, Hilmi E. Egilmez, *Member, IEEE*,  
Moonmo Koo, Shan Liu, *Senior Member, IEEE*, Jani Lainema, Marta Karczewicz

**Abstract**—In the past decade, the development of transform coding techniques has achieved significant progress and several advanced transform tools have been adopted in the new generation Versatile Video Coding (VVC) standard. In this paper, a brief history of transform coding development during VVC standardization is presented, and the transform coding tools in the VVC standard are described in detail together with their initial design, incremental improvements and implementation aspects. To improve coding efficiency, four new transform coding techniques are introduced in VVC, which are namely Multiple Transform Selection (MTS), Low-Frequency Non-separable Secondary Transform (LFNST) and Sub-Block Transform (SBT), as well as a large (64-point) type-2 DCT. The experimental results on VVC reference software (VTM-9.0) show that average 4.5% and 3.6% overall coding gain can be achieved by the VVC transform coding tools for All Intra and Random Access configurations, respectively.

**Index Terms**—Versatile Video Coding (VVC), Joint Video Exploration Team (JVET), Transform coding, MTS, LFNST, SBT, VVC Test Model (VTM), Joint Exploration Model (JEM)

## I. INTRODUCTION

TRANSFORM coding has been an essential part of many practical video codecs for achieving a high compression ratio, and it has been successfully adopted in multiple video coding standards, e.g., H.261 [1], MPEG-1 [2], MPEG-2 [3], H.263 [4], H.264/AVC [5] and High-Efficiency Video Coding (HEVC) [6]. The development of transform coding in the past several decades was based on the traditional DCT, more specifically, type-2 DCT (DCT-2) [7], due to its reasonable tradeoff between coding performance and complexity. Under the first-order Markov conditions, which efficiently model the characteristics of natural imagery sources, it has been mathematically proved that DCT-2 approximates the optimal data-driven Karhunen-Loève transform (KLT) [8]. Fast method for implementing DCT-2 was first proposed in [9] and further developed in the following decades. In H.264/AVC, a low-complexity  $4 \times 4$  transform [10] is adopted featured by multiplier-less calculation, fixed-point and 16-bit intermediate

data representation. In addition, a secondary  $4 \times 4$  Hadamard transform can be applied in H.264/AVC to further decorrelate DC coefficients of  $4 \times 4$  transform block. In HEVC, the transform size of DCT-2 is further extended to  $8 \times 8$ ,  $16 \times 16$  and  $32 \times 32$  with fixed-point transform kernels, and the implementations can be done using either direct matrix multiplication or a partial butterfly fast method. In addition, 16-bit intermediate data representation and arithmetic is kept in the HEVC transform design [11]. Moreover, in HEVC, a  $4 \times 4$  DST-7 is applied for  $4 \times 4$  intra prediction residuals. Further technical advances on top of DCT-2 are mainly reflected by the extended transform sizes and different designs of integer DCT-2 kernels. In [12], a directional extension of conventional DCT-2 was proposed, which employs two 1-D DCTs along directional directions rather than the horizontal and vertical directions. In recent years, driven by the drastically increased traffic of multimedia communications and new systems supporting more computational power, extensive efforts have been made to seek for higher coding performance at an increased yet feasible complexity cost.

From the industrial standardization side, the Joint Video Exploration Team (JVET) was created in 2015 as a joint effort of ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11 to explore advanced video coding technologies beyond HEVC. In 2018, after receiving 23 responses to the Call for Proposal (CfP) on next-generation video coding, the Joint Video Experts Team, also known as JVET, was created that officially launched the standardization of the new generation Versatile Video Coding (VVC) [13]. After two years of development, VVC has been finalized in July 2020 with a substantial objective coding gain over its predecessor, HEVC.

This paper provides a history of the tool development in transform coding and presents how these tools are shaped into their final designs in VVC. Among all the coding tools included in VVC, the major advances in transform coding techniques can be categorized into primary transform, secondary transform and transform partitioning, which are summarized in the following three subsections.

The remainder of this paper is organized as follows. In Section II, further development of transform coding beyond HEVC is reviewed in three categories, including 1) primary transform, 2) secondary transform and 3) transform partitioning. In Section III, the transform design in VVC is described with technical details, and the encoder implementations on top of version 9.0 of the VVC reference software called VVC Test Model (VTM) are described in Section IV. Complexity analysis and experimental results are discussed in Section V and Section VI, respectively, and

X. Zhao and S. Liu are with Tencent Media Lab, Palo Alto, CA, USA.  
S.-H. Kim is with LG Electronics, San Diego, CA, USA.  
Y. Zhao is with Huawei Technologies, Hangzhou, China.  
H. E. Egilmez and M. Karczewicz are with Qualcomm Technologies, Inc., San Diego, CA, USA.  
M. Koo is with LG Electronics, Seoul, South Korea.  
J. Lainema is with is with Nokia Technologies, Tampere, Finland.  
Corresponding authors: X. Zhao (e-mail: xinzhaohao@tencent.com) and S.-H. Kim (e-mail: seunghwan3.kim@lge.com)

Section VII concludes this paper.

## II. REVIEW OF TRANSFORM CODING TOOLS BEYOND HEVC

### A. Primary Transform

The fixed transform scheme based on DCT-2 has been widely used in the past video coding standards. However, due to diverse characteristics of image/video content, it is not always efficient to use a single transform kernel for all prediction residuals. In [14], it is proposed to use sine and cosine transforms alternatively for image coding. In [15], mode-dependent directional transforms (MDDTs) are proposed for intra prediction residuals, wherein hard-coded KLTs are selected as the horizontal and vertical transform based on intra prediction mode. Moreover, with a first-order Gauss-Markov model for image pixels, it was further mathematically proved in [16] that, the optimal horizontal (vertical) transform applied on the intra prediction residual of horizontal (vertical) prediction mode is actually DST-7. In HEVC, a  $4 \times 4$  DST-7 is applied for  $4 \times 4$  intra prediction residuals, and better coding performance is achieved over DCT-2. Furthermore, to overcome the limitation of using single fixed kernel, transform signaling has been proposed, which allows multiple options of transform kernels, so that an encoder can choose a transform on a per block basis and signal such that selection in the bitstream. In such signaling schemes, transform candidates can be derived by off-line training or selected from a group of mathematically defined transforms, such as the DCT/DST families [17].

Multiple transform selection with transform signalling was proposed in [18] as known as rate-distortion optimized transform (RDOT), which applies multiple Karhunen-Loève transforms (KLTs) and the transform selection is rate-distortion optimized. Alternative schemes that apply DCT/DST as the transform candidates are also proposed in [19][20][21]. More specifically, in [20], an Enhanced Multiple Transform (EMT) scheme was proposed and included in the Joint Exploration Model, which applies intra prediction mode dependent transform sets with each set consist of multiple candidates of transform kernels. Moreover, multiple non-separable transform schemes have also been proposed in [22][23]. In [24], a row-column transform scheme is proposed, which approximates non-separable KLT using a separable transform.

### B. Secondary Transform

Secondary transform refers to an additional transform process that follows the primary transform. In H.264/AVC, a  $16 \times 16$  transform is implemented as sixteen  $4 \times 4$  transforms, and the DC coefficients of each  $4 \times 4$  transform block are combined as one  $4 \times 4$  block and further processed using a secondary Hadamard transform [5]. During the development of HEVC, a secondary rotational transform (ROT) [25] was proposed on top of DCT, with kernels featured by sparse transform matrices constructed using pre-defined Givens rotations. Note that ROT is still a separable transform and the coding gain overlaps with the DST-7. In HEVC,  $4 \times 4$  DST-7 has been adopted for intra coding with a simpler design.

Non-separable transform schemes have been proposed as a

secondary transform [26], namely Non-Separable Secondary Transform (NSST) as either a direct matrix multiply [26] or Hypercube-Givens Transform (HyGT) [27] where transform could be represented as multi-layer transforms by cascaded Givens rotations in a hypercube arrangement. The major benefit of applying non-separable transform as a secondary transform is to achieve a better tradeoff between coding efficiency and complexity. With NSST, a non-separable secondary transform is performed on the lower-frequency coefficients so that computational complexity for non-separable transform is largely reduced. Besides, HyGT provides parallel, multi-stage decompositions for non-separable transforms with lower computational complexity and memory cost, yet it introduces additional latency due to its stage-wise implementations.

### C. Transform Partitioning

Transform coding is applied to reduce the statistical dependency among residual samples. However, when the residual samples are distributed locally, applying a larger transform may create high frequencies that can be expensive for entropy coding. To address this issue, transform partitioning schemes have been proposed. In [28], adaptive block-size transforms (ABT) was proposed, and the basic idea of inter ABT is to align the block size used for transform coding of the prediction error to the block size used for motion compensation. In HEVC, a quadtree partitioning scheme has been applied for transform [29], and a spatially varying transform (SVT) was proposed in [30] to adapt the position and size of transform with localized residual samples.

## III. TRANSFORM DESIGN IN VVC

Transform design in VVC mainly includes three aspects: the primary transform, secondary transform and transform partitioning. In this section, the new transform coding tools in VVC are described with respect to each of the three aspects. In this paper, an N-point transform refers to a one-dimensional transform that can be applied on an N-point input vector, which is done using a transform matrix of size N by N.

Several design aspects of HEVC transform coding are inherited in VVC, including: 1) fixed-point operations are used and intermediate data representation and arithmetic is kept to be 16-bit, 2) the transform process can be either implemented using direct matrix multiply or a fast method, e.g., partial butterfly, 3) the transform kernels are designed by scaling the transform basis with  $64\sqrt{N}$  and rounding to the nearest integer with minor adjustment, where the norm of transform basis is 1 and  $N$  is the transform size, 4) smaller DCT-2 is part of the larger DCT-2, so all DCT-2 kernels are embedded in the  $64 \times 64$  DCT-2 transform kernel. Although up to  $128 \times 128$  coding block sizes can be applied in VVC, the transform coding is designed to be compatible with the virtual pipeline data units (VPDUs) implementation. In hardware decoding process, VPDUs are non-overlapping  $64 \times 64$  blocks and consecutive VPDUs are processed in parallel by multiple pipelines.

## A. Primary Transform

### 1) Transform Kernels

In VVC, in addition to the conventional type-2 DCT (DCT-2), alternate transform types, including type-7 DST (DST-7) and type-8 DCT (DCT-8), are employed. The basis functions of DST-7 and DCT-8 are formulated in below Equation (1) and (2), respectively.

$$T_i(j) = \sqrt{\frac{4}{2N+1}} \cdot \sin\left(\frac{\pi \cdot (2i+1) \cdot (j+1)}{2N+1}\right), \quad (1)$$

$$T_i(j) = \sqrt{\frac{4}{2N+1}} \cdot \cos\left(\frac{\pi \cdot (2i+1) \cdot (2j+1)}{4N+2}\right), \quad (2)$$

where  $N$  is the transform size,  $i = 0, 1, \dots, N-1$  refers to the element index of the output vector, and  $j = 0, 1, \dots, N-1$  refers to the element index of the input vector. For uneven distribution of residual, DST-7 and DCT-8 are usually more efficient than DCT-2 since their basis functions are more aligned with such statistics [20]. The size of DCT-2 ranges from 4-point to 64-point, and DST-7/DCT-8 ranges from 4-point to 32-point. It is noted that the transform bases of DST-7 and DCT-8 are flipped versions of each other with alternating sign changes.

The transform kernels defined in VVC are composed of 8-bit signed integers and all the primary transform kernels in HEVC, including 4-point DST-7 and DCT-2 ranging from 4-point to 32-point, are kept unchanged. The additional integer transform kernels defined in VVC are derived by scaling the floating-point transform kernel with  $64\sqrt{N}$ , where  $N$  is the transform size, and further adjusted by  $\pm 1$  after rounding. The adjustment of 64-point DCT-2 is performed in a way that all the DCT-2 kernel defined in HEVC are included, partial butterfly [11] is supported and kernel elements are optimized towards better orthogonality. The adjustment of kernel element is performed with the following criteria: 1) to align with the HEVC core transform design that smaller DCT-2 can be extracted as part of the larger DCT-2, only 32 elements can be adjusted and the remaining 33 elements are kept same as HEVC to generate smaller DCT-2 from  $4 \times 4$  to  $32 \times 32$ , 2) the orthogonality is optimized such that  $K \times K^T$  is as close as possible to the identity matrix, wherein  $K$  is the transform matrix, 3) the adjustment can be done only with offset -1 and +1 to ensure that the decorrelation capability of adjusted transform kernel approximates DCT-2 efficiently. The adjustment of DST-7/DCT-8 kernels are performed to ensure the three notable features associated with DST-7/DCT-8, as illustrated in Figure 1, including feature 1) repetitive segments  $\{b, f, i, l, o\}$  in some bases, 2) unique coefficient value in one basis and 3) mathematical relationship among coefficients in a tuple with fixed pattern in some bases, as pointed out in [32][33], are kept in the integer kernel with optimized orthogonality. Specially, for feature 3), the following formulation is supported:

$$\begin{aligned} a + j &= l \\ b + i &= m \\ c + h &= n \\ d + g &= o \\ e + f &= p \end{aligned} \quad (3)$$

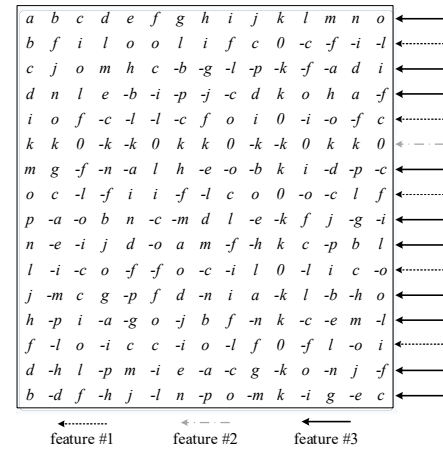


Figure 1: Illustration of three features in the  $16 \times 16$  DST-7 transform matrix of VVC.

To align the worst-case multiplications per coefficient with HEVC, for 64-point DCT-2 and 32-point DST-7/DCT-8, only the first 32 and 16 low-frequency coefficients are kept, respectively, and the high frequency coefficients are zeroed out, which is also considered in last coefficient position coding and coefficient group scanning [31]. More details will be discussed in the Section V. Moreover, based on the three features of the DST-7/DCT-8 kernels, a fast transform scheme with support of dual implementations is included in VVC [32][33]. In this way, the fast algorithm and direct matrix multiply produce identical results. Meanwhile, the fast method achieves around 50% multiplication reduction for 16-point DST-7/DCT-8 [32].

In VVC, the primary transform is specified as separable transform. Five different combinations of transform types are supported, including the conventional (DCT-2, DCT-2) and four new MTS mode combinations, i.e., (DST-7, DST-7), (DST-7, DCT-8), (DCT-8, DST-7) and (DCT-8, DCT-8). The explicit combination between DCT-2 and DST-7 (or DCT-8) with extra signalling overhead is not supported due to the limited coding gain and increased complexity for introducing extra encoder search and additional transform combinations. In VVC, DST-7 and DCT-8 can be applied to the luma blocks in several coding tools, including Multiple Transform Selection (MTS), Intra Sub-Partitioning (ISP) [34] and Sub-block Transform (SBT), which will be detailed in the following subsection related to transform type selection. For chroma coding, the potential benefits of DST-7/DCT-8 have also been studied during the development of VVC. However, since chroma components typically present smooth textures, where DCT-2 is sufficient, the coding gain versus complexity tradeoff is less beneficial.

### 2) Multiple Transform Selection

In VVC, there are two variants of Multiple Transform Selection (MTS), called explicit and implicit MTS. The explicit MTS can be applied to both intra and inter coded blocks, while the implicit MTS can be only used for intra coded blocks. In explicit MTS, the choice of DST-7/DCT-8 is indicated by

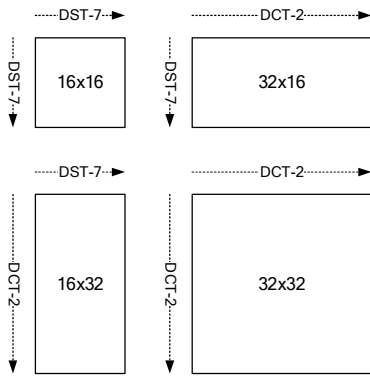


Figure 2: Examples of transform type selection for vertical and horizontal transforms based on the implicit MTS in VVC.

TABLE 1

SPECIFICATION OF TRANSFORM KERNELS DEPENDING ON THE SYNTAX

| mts_idx   | 0     | 1     | 2     | 3     | 4     |
|-----------|-------|-------|-------|-------|-------|
| trTypeHor | DCT-2 | DST-7 | DCT-8 | DST-7 | DCT-8 |
| trTypeVer | DCT-2 | DST-7 | DST-7 | DCT-8 | DCT-8 |

TABLE 2

SUMMARY OF THE COMBINATIONS OF DIFFERENT TOOLS

| Tools |                    | MIP                           | ISP |
|-------|--------------------|-------------------------------|-----|
| MTS   | Explicit intra MTS | Y                             | N   |
|       | Explicit inter MTS | N/A                           | N/A |
|       | Implicit MTS       | N                             | Y   |
| LFNST |                    | Partially (see section III-B) | Y   |
| SBT   |                    | N/A                           | N/A |

explicit signaling of the transform type. In implicit MTS, the transform type are selected based on coded information that is known to both the encoder and decoder, and transform type signalling is not needed. In the Sequence Parameter Set (SPS), there are three flags controlling MTS operation. The first is used to enable MTS itself. The second is used to select between explicit or implicit intra MTS and the last is used to enable explicit inter MTS. Thus, with the latter two flags, four MTS mode combinations need to be selected if MTS is enabled. In explicit MTS, the index `mts_idx` is signaled at the end of Coding Unit (CU) level syntax to indicate the transform type for horizontal transform (`trTypeHor`) and vertical transform (`trTypeVer`). The value of `mts_idx` ranges from 0 to 4, and the mapping to the transform type is specified in Table 1.

The MTS index, denoted as `mts_idx`, is signalled only when nonzero coefficients for luma block exist beyond the DC coefficient and nonzero coefficient is not identified outside the top-left  $16 \times 16$  coefficients region, since DST-7/DCT-8 has only an impact on the lowest  $16 \times 16$  frequency coefficients. In other words, identification of a nonzero coefficient beyond the lowest  $16 \times 16$  coefficients means DST-7/DCT-8 not being applied. In addition, when several tools are enabled, including ISP, SBT and Low-Frequency Non-Separable Transform (LFNST), `mts_idx` is not signaled [35] and transform type is inferred as either DCT-2 or a pre-defined transform type. In

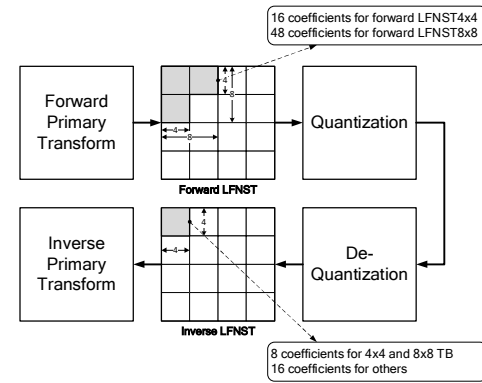


Figure 3: Illustration of the LFNST design in VVC.

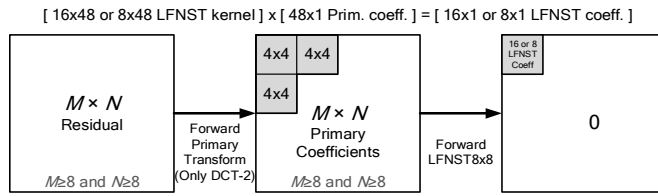
Table 2, the combinations of different tools, including MTS, LFNST, MIP, ISP and SBT, have summarized, where “Y/N” means the associated coding tools in the row and column can/cannot be combined, and “N/A” means the associated combination is not applicable.

The initial design of implicit MTS was first proposed in [36] where the transform types were derived based on the shape of the coded block. The transform type selection condition has later been simplified in [37] and led to the final design of implicit MTS in VVC. That is, DST-7 is applied as the horizontal (vertical) transform if block width (height) is smaller than 32. Otherwise, DCT-2 is used. This same rule is also used to derive transforms for ISP coded blocks. In Figure 2, examples of implicit MTS derivation are illustrated for different block sizes. The implicit MTS design in VVC can be viewed as an extension of the HEVC transform derivation for intra prediction residual, by extending the applicable block size for DST-7 from  $4 \times 4$  to  $16 \times 16$  (inclusive) and other rectangular block sizes in between. Other than the restrictions based on block sizes, implicit transform can be applied only when LFNST and Matrix-based Intra Prediction (MIP) [34] indices are set to zero [38].

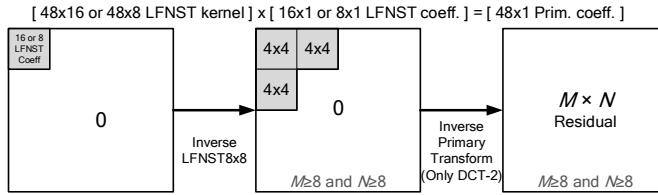
The benefits of implicit MTS are summarized as follows: (1) Although implicit MTS provides less coding gain as compared to explicit MTS, it provides significant coding gain over DCT-2 without any encoder search. This feature is appealing for simple encoder designs that cannot accommodate a complex rate-distortion search. (2) Implicit MTS provides a unified transform derivation rule for both ISP coded and non-ISP intra coded blocks. (3) Since DST-7 is not allowed for dimensions beyond 16 in implicit MTS, the built-in zeroing out operation on high-frequency coefficients (which only applies for 32-point DST-7) is avoided.

### B. Secondary Transform

The LFNST [39][40] is a non-separable transform which applies to the top-left low-frequency region of primary transform coefficients, as shown in Figure 3. In VVC, the LFNST can be applied for intra coded blocks that use DCT-2 as the primary transform [41]. The transform kernels defined in LFNST consists of 4 transform sets with 2 kernels per set, where each kernel is selected among  $48 \times 16$  and  $16 \times 16$  matrices



(a) forward LFNST8



(b) backward LFNST8

Figure 4: The exemplary procedure of (a) forward LFNST8 and (b) backward LFNST8 with 16x48/48x16 or sampled 8x48/48x8 matrix.

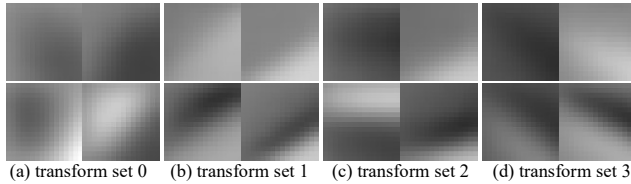


Figure 5: First basis images of 16x16 DCT-2 and LFNST combinations.

depending on transform block size. Specifically, a 48x16 kernel is applied to the top-left 8x8 region when a transform block (TB) is greater than or equal to 8x8, denoted as LFNST8, and a 16x16 kernel is applied to the top-left 4x4 region when TB width or height is 4, denoted as LFNST4. Detailed design of LFNST8 and LFNST4 is elaborated in next subsections.

### 1) LFNST computation process

The LFNST is computed in the form of matrix multiplication that is friendly to parallelism. The main idea of LFNST, also known as Reduced Secondary Transform (RST) [40], is mapping an  $N$  dimensional vector to an  $R$  dimensional vector through  $R \times N$  matrix, where  $R$  is usually less than  $N$  where  $N/R$  is the reduction factor. The reduced  $R \times N$  matrix is used as the kernel in a forward transform, which consists of  $R$  orthonormal row vectors of the  $N$  dimensional space, and its transposed matrix is used as the kernel in the inverse LFNST. The larger reduction factor indicates more complexity reduction in terms of computation and memory usage, which will be detailed in Section V.

The LFNST design in VVC employs two reduction factors to restrict the worst-case computation complexity to be 8 multiplications per sample and kernel memory to be 8KB. First, the LFNST8 has a reduction factor of 3 relative to square 48x48 matrix. With LFNST8, 48 coefficients from three pre-defined 4x4 blocks, namely Region of Interest for LFNST8 (ROI8), form the input vector that is multiplied by the 16x48 matrix to output 16 coefficients. For LFNST4, the Region of Interest, denoted as ROI4, is one single top-left 4x4 block. In addition, for 4x4 and 8x8 TBs, the LFNST design has an extra reduction

TABLE 3

THE LFNST SET SELECTION TABLE

| <i>intraPredMode</i>                   | <i>lfnstSetIdx</i> |
|--|--------------------|
| $\text{intraPredMode} < 0$             | 1                  |
| $0 \leq \text{intraPredMode} \leq 1$   | 0                  |
| $2 \leq \text{intraPredMode} \leq 12$  | 1                  |
| $13 \leq \text{intraPredMode} \leq 23$ | 2                  |
| $24 \leq \text{intraPredMode} \leq 44$ | 3                  |
| $45 \leq \text{intraPredMode} \leq 55$ | 2                  |
| $56 \leq \text{intraPredMode} \leq 80$ | 1                  |

factor of 2. That is, only the first 8 coefficients of the output vector are calculated. In this way, the worst-case multiplication count per sample is limited to be 8. It is observed that around 0.2% and 0.3% coding performance loss (i.e., BD-bitrates increase) are caused due to the reduction factor of 2 and the transform set reduction from 35 to 4, respectively.

The Figure 4(a) illustrates an example of applying forward LFNST8 with its 16x48 or sampled 8x48 matrix. When LFNST is applied, all primary transform coefficients other than ROI8 are zeroed out [42]. Then the output of LFNST8, which is a 16 or 8 coefficient vector, is further quantized and entropy coded [43]. For the case that only 8 output coefficients are generated, i.e. 4x4 or 8x8 TB, the coefficients after the 8<sup>th</sup> position along the scanning order must be zero. Therefore, for coefficient coding, the last non-zero coefficient is coded with constraint that it is within the 8<sup>th</sup> scanning position [43]. An example of applying inverse LFNST is shown in Figure 4(b). In the inverse LFNST8, given the input 16 or 8 LFNST coefficients, a 48x16 and a sampled 48x8 inverse LFNST matrix is used to perform the inverse LFNST and reconstruct the primary transform coefficients back to ROI8. Furthermore, LFNST is applied to both luma and chroma blocks for a dual tree, and LFNST indices for luma and chroma blocks are signalled separately. Otherwise, if the dual tree is disabled, a single LFNST index is signalled and used only for luma, because the LFNST is not applied to chroma in shared-tree case [44].

Regarding the interaction with ISP mode [34] in VVC, when there are multiple transform partitions in one intra coding unit, LFNST is applied to all transform blocks and a single LFNST index is signalled. The LFNST kernel was initially proposed to be 10-bit precision but it was quantized to 8-bit integer including sign. There was no noticeable coding performance impact due to this precision change.

### 2) LFNST set selection

An LFNST set indicates a group of transform kernel options that can be selected in LFNST. In VVC, four LFNST sets, denoted as *lfnstSetIdx*, are defined and the selection depends on the intra prediction mode, denoted as *intraPredMode*, as shown in Table 3. In each LFNST set, three different options of LFNST kernel are provided and the selection is indicated by an LFNST index ranging from 0 to 2. If the LFNST index is equal to 0, LFNST is not applied. Otherwise, an LFNST is applied using one of the two kernels in the LFNST set and the selection is indicated by the LFNST index.

As mentioned above, an LFNST set is selected among 4 transform sets according to *intraPredMode*, which is derived

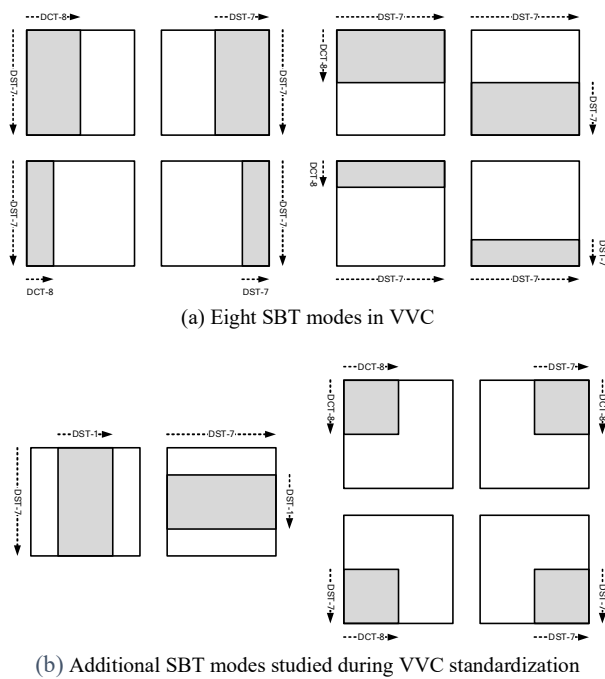


Figure 6. Sub-block Transform modes (a) The eight SBT modes in VVC and (b) Additional SBT modes studied during VVC standardization.

from an original intra prediction mode. In addition, *lfnstSetIdx* is assigned to be 1 for negative values of *intraPredMode* in Table 3, which corresponds to wide-angle intra prediction mode that is applied to cover the prediction angles beyond the diagonal prediction direction [34]. The basis images of  $16 \times 16$  DCT-2 and LFNST combination are shown in Figure 5, wherein the first basis images of four transform sets are shown in the first row and the second basis images of four transform sets are shown in the second row. Two columns of basis images are shown each transform set, corresponding to the two kernel candidates in one transform set. Here, the first and second basis refers to the basis that generates the first and second output LFNST coefficient, respectively. From Figure 5 it can be seen that, the directionality of basis images are well aligned with the associated intra prediction directionality, e.g., transform basis of transform set 1 show diagonal directionality.

There are two exceptions in the LFNST set selection process that are not defined solely by Table 3. For a Cross-Component Linear Model (CCLM) mode, the wide angle intra prediction mode of collocated luma TB at the center position of the current chroma TB is used with Table 3 to select its LFNST set [45]. If MIP is applied and both width and height of the CU is greater than or equal to 16, LFNST set 0 is selected [46].

### 3) LFNST index signalling

LFNST index can be signalled for an intra-coded coding unit only when block width and height are greater than or equal to 4. On top of that, the following extra conditions are also considered to determine the LFNST signaling.

- If a non-zero coefficient is identified in zeroed-out region, it is not needed to signal an LFNST index since LFNST is inferred to be disabled in this case.

- If a non-zero coefficient exists only for DC position or it does not exist in any relevant color components, LFNST index signaling is skipped except for ISP mode.
- If one or more of all relevant color components is coded by transform skip, an LFNST index is not signalled [47].
- For a CU coded by MIP mode with either width or height being less than 16, LFNST is not signalled.
- If either width or height of a CU is greater than maximum transform size specified in SPS, LFNST is not signalled.

When an LFNST index is not signaled, it is inferred to be zero, i.e., LFNST is not applied.

## C. Transform Partitioning

### 1) Sub-block Transform

The distribution of inter-prediction residual is different from that of intra-prediction residual. Statistically, energy of inter-prediction residual increases from the center of prediction block towards its boundaries [48]. In addition, in many cases, the inter-prediction residuals are localized at one side of the block, rather than being distributed around all block boundaries. Inspired by the SVT [30] that uses a smaller transform block to capture the localized residuals, and further considering the unique distribution of inter-prediction residual, a Sub-Block Transform (SBT) method was developed and included in VVC.

In VVC, there are eight SBT modes associated with different configurations of the size and location of transform blocks. When SBT is used, the transform block is either half or quarter size of the residual block, as illustrated by the shaded blocks in Figure 6 (a), while residual in the rest part of the residual block is not coded and regarded as zero. In SBT, the transform blocks always reside in one boundary of the residual block.

Another feature of SBT design is the adaptive transform kernel, which is selected based on the transform block position, as illustrated in Figure 6 (a). This feature contributes to near half of the SBT coding gain [49]. If the residual is not evenly distributed within the residual block, the residual energy typically increases from one side to the other. For instance, if the transform block at the left side of the residual block is selected the best mode, the right side typically contains weaker residual. In this case, DCT-8 is more efficient than DCT-2 for the horizontal transform since the basis vector decreases from left to right. For the other dimension of the transform block, DST-7 is selected based on experimental results that show marginally higher coding gain than both DCT-2 and DCT-8.

During VVC standardization, another two types of SBT modes were proposed, as shown in Figure 6 (b). Each type contributes around 0.1% extra coding gain on top of the eight SBT modes that reflect the final design in VVC. The first type includes two modes, utilizing a half-sized transform block to cover the center of the residual block [50]. The transform block is coupled with type-1 DST (DST-1) in one dimension, because the associated basis vector has the highest value in the middle [20] which fits residual that is weaker at two sides of the transform block than the center. Such a mode has smaller coding gain than each of the half-sized SBT modes with transform block locating at residual block boundaries. The

second type includes four modes, with a quarter-sized transform block to cover one corner of the residual block [50]. The transform kernel selection is optimized based on the observation that residual increases from the center to a corner. In view of that the eight modes in Figure 6 (a) already contributed to the major coding gain, the six modes in Figure 6 (b) were not adopted in VVC, to avoid a more complicated SBT design as well as the introduction of an additional type of DST/DCT transform.

In a nutshell, the SBT modes are optimized for the cases that the major residual is localized at a lateral part of the residual block. During VVC standardization, transform unit split with position-dependent transform kernels were also studied for comparison with SBT, which showed marginally coding gain over SBT (especially on top of inter MTS) but required twice extra encoding runtime than SBT [51] for RDO.

For implementation consideration, when the width or height of an SBT transform block exceeds 32 (e.g., a  $64 \times 64$  CU with any SBT mode), both horizontal and vertical transforms are forced to be DCT-2. In this way, it is aligned with the design principle in VVC that DST-7/DCT-8 is only applied to blocks no larger than  $32 \times 32$ .

## 2) Implicit transform partitioning

In VVC, the transform size can be up to 64-point, which is supported by DCT-2 only. The maximum transform size is selectable at sequence level and the transform size can be either 32 or 64-point [52]. When the coding unit width (or height) is greater than the maximum transform size, the prediction residual block will be further split horizontally (or vertically) into multiple transform blocks with width (or height) equal to maximum transform size. For one example, when the coding block size is  $128 \times 64$  and the maximum transform size is 32-point, the coding block will be split into 8 transform units arranged in 4 rows and 2 columns. For another example, when the coding block size is  $64 \times 16$  and the maximum transform size is 32-point, the coding block will be split into 2 transform units distributed horizontally with each transform unit covering a  $32 \times 16$  residual block. The implicit transform splitting is processed with always vertical splitting first if the coding block width is greater than the maximum transform size. The transform splitting is done in a way that the coding order of transform units does not cross the  $64 \times 64$  boundary. That is, all transform units within one  $64 \times 64$  block are coded before any other transform unit in the next  $64 \times 64$  block. In this way, the processing of transform units will not break down the VPDU implementation.

The signalling on maximum transform size provides the flexibility for encoder to implement up to either 32-point or 64-point transform size without limiting the coding block size. In addition, in VVC, lossless coding is achieved by enabling transform skip mode, which is supported by up to  $32 \times 32$  block size. Therefore, with the option of selecting  $32 \times 32$  as the maximum transform block size, coding block size greater than  $32 \times 32$  can be also enabled for lossless coding. Furthermore, 64-point transform in VVC is always accompanied with

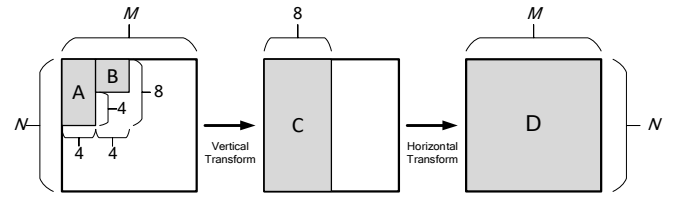


Figure 7: The non-zero coefficient block expansion of primary transform from coefficients generated by inverse LFNST.

zeroing out of the high frequencies, which may potentially create subjective quality artifacts. Maximum 32-point transform provides an alternate way of keeping large coding block without dropping any high frequency coefficients.

## IV. ENCODER IMPLEMENTATION IN VTM

An efficient encoder algorithm is important in a practical video codec. In this section, to provide some insights on the encoder side design, the implementation of transform tools in VTM is described.

### A. Encoder decision for MTS and LFNST

In this section, MTS encoder implementation in VTM-9.0 is described. The transform using DCT-2 as both horizontal and vertical transform is checked first, then Transform Skip (TS) is evaluated, and DST-7 and DCT-8 are checked in the last. To achieve a reasonable coding gain versus complexity trade-off, a subset of transform candidates is first selected based on the cost measured by the L1-norm of a transform coefficient block, then the selected candidates are further evaluated based on high-complexity RD cost measurement. Specifically, if L1-norm of a transform candidate is less than or equal to a pre-determined threshold, the candidate is included in the candidate list for next high-complexity RD cost measurement if the list is not full. All the candidates enter the list in a first-in, first-out fashion if the cost is less than or equal to the given threshold. The maximum number of entries in the candidate list can be specified default VTM configuration files.

Since LFNST is only enabled when DCT-2 is applied as the primary transform, the encoder search of LFNST and MTS is jointly optimized using following two steps.

- Step 1: A predefined reduced set of transform modes is evaluated first, which includes DCT-2, TS, DST-7 and combination of DCT-2 and LFNST. Based on the evaluation outcome, the remaining MTS modes may be skipped.
- Step 2: If the remaining MTS modes are further checked, based on the outcome, it is further determined if the encoder search is terminated after checking a certain MTS mode.

As an exception to the above encoder process, combinations of ISP and LFNST are searched in addition to the checking DCT-2 pair or transform skip with LFNST off. Moreover, as aforementioned, ISP employs implicit MTS only. These two exceptions facilitate useful early skip decisions in VTM.

### B. Sub-Block Transform (SBT)

SBT modes are searched after the regular DCT-2 transform for entire-block residual. A full search on all the eight SBT modes imposes significant encoder complexity, e.g., over 60%



TABLE 4

THE MULTIPLICATION COUNTS PER SAMPLE FOR ALL COMBINATIONS OF PRIMARY TRANSFORM KERNELS AND BLOCK SHAPES

| $M$ | $N$ | DCT-2 | DCT-2<br>w/o<br>zero-out | DST-7<br>/DCT-8 | DST-7<br>/DCT-8<br>w/o zero-out | LFNST<br>+DCT-2 |
|-----|-----|-------|--------------------------|-----------------|---------------------------------|-----------------|
| 4   | 4   | 3.0   | 3.0                      | 8.0             | 8.0                             | 11.0            |
| 4   | 8   | 4.3   | 4.3                      | 12.0            | 12.0                            | 10.9            |
| 4   | 16  | 6.9   | 6.9                      | 20.0            | 20.0                            | 6.8             |
| 4   | 32  | 12.2  | 12.2                     | 20.0            | 36.0                            | 4.8             |
| 4   | 64  | 12.2  | 22.8                     | N/A             | N/A                             | 3.8             |
| 8   | 4   | 4.3   | 4.3                      | 12.0            | 12.0                            | 10.1            |
| 8   | 8   | 5.5   | 5.5                      | 16.0            | 16.0                            | 10.8            |
| 8   | 16  | 8.1   | 8.1                      | 24.0            | 24.0                            | 10.8            |
| 8   | 32  | 13.4  | 13.4                     | 24.0            | 40.0                            | 7.8             |
| 8   | 64  | 13.4  | 24.1                     | N/A             | N/A                             | 6.3             |
| 16  | 4   | 6.9   | 6.9                      | 20.0            | 20.0                            | 5.7             |
| 16  | 8   | 8.1   | 8.1                      | 24.0            | 24.0                            | 9.7             |
| 16  | 16  | 10.8  | 10.8                     | 32.0            | 32.0                            | 6.7             |
| 16  | 32  | 16.1  | 16.1                     | 32.0            | 48.0                            | 5.2             |
| 16  | 64  | 16.0  | 26.7                     | N/A             | N/A                             | 4.4             |
| 32  | 4   | 12.2  | 12.2                     | 18.0            | 36.0                            | 3.5             |
| 32  | 8   | 13.4  | 13.4                     | 20.0            | 40.0                            | 6.2             |
| 32  | 16  | 16.1  | 16.1                     | 24.0            | 48.0                            | 4.7             |
| 32  | 32  | 21.4  | 21.4                     | 24.0            | 64.0                            | 3.9             |
| 32  | 64  | 21.4  | 32.0                     | N/A             | N/A                             | 3.5             |
| 64  | 4   | 11.4  | 22.8                     | N/A             | N/A                             | 2.4             |
| 64  | 8   | 12.0  | 24.1                     | N/A             | N/A                             | 4.4             |
| 64  | 16  | 13.4  | 26.7                     | N/A             | N/A                             | 3.7             |
| 64  | 32  | 16.0  | 32.0                     | N/A             | N/A                             | 3.3             |
| 64  | 64  | 16.0  | 42.7                     | N/A             | N/A                             | 3.1             |

encoding time increase on VTM-3.0 under Random-Access configuration. Three fast algorithms have been developed to reduce the encoding time increase to around 8%, with less than 0.1% Bjøntegaard delta bitrates (BDR) penalty [53].

Algorithm 1: Based on residual energy distribution, certain SBT modes are skipped from encoder checking. Before performing transform and quantization, the RD cost for each SBT mode is estimated using the signaling bits for the CU and the distortion of both the residual-skipped part and the residual-coded part, measured by sum of squared difference. If the estimated RD cost is already greater than the best RD cost (i.e., the minimum RD cost for the CU that is found before this prediction mode), the corresponding SBT mode is skipped further testing. In addition, at most four SBT modes are tested for any inter prediction mode.

Algorithm 2: When the prediction residual is either too large or too small, all SBT modes are skipped. More specifically, an inter prediction mode is unlikely to be chosen if its RD cost associated with full size transform exceeds the best RD cost to some extent. On the other hand, when the residual is too small, SBT is less efficient than skipping the residual due to the signaling overhead.

Algorithm 3: To decide from a large number of inter prediction modes and flexible coding tree as defined in VVC, a block may be encoded many times by different coding modes. After a residual block is encoded, the residual energy and the best transform mode among all candidates, i.e., the DCT-2 mode, inter MTS modes, and SBT mode, are recorded as a pair of historical information. When this region is encoded with another inter prediction mode, if the residual energy is similar to a recorded case, only the recorded best transform mode is tested for this prediction block. If a matching prior is not found,

TABLE 5

THE MEMORY USAGE INFORMATION OF ALL OF VVC PRIMARY TRANSFORM KERNELS (IN BYTES)

| Transform | Length |     |     |       |       | Row sum |
|-----------|--------|-----|-----|-------|-------|---------|
|           | 4      | 8   | 16  | 32    | 64    |         |
| DCT-2     | 16     | 64  | 256 | 1,024 | 2,048 | 3,408   |
| DST-7     | 16     | 64  | 256 | 512   | N/A   | 848     |
| DCT-8     | 16     | 64  | 256 | 512   | N/A   | 848     |
| Col. sum  | 48     | 192 | 768 | 2,048 | 2,048 | 5,104   |

above fast algorithms 1 and 2 are invoked and later, the best transform mode after RDO is saved for later encoder decision.

## V. COMPLEXITY ANALYSIS

### A. Primary Transform

As mentioned in Section III-A, high-frequency DCT-2 transform coefficients are zeroed out for the transform blocks with size (width and/or height) equal to 64, so that only the 32 low frequency coefficients are retained. For example, for an  $M \times N$  transform block, when  $M$  is equal to 64, only the left 32 columns of transform coefficients are kept. Similarly, when  $N$  is equal to 64, only the top 32 rows of transform coefficients are kept. In addition, for 32-point DST-7 and DCT-8, only the 16 low frequency transform coefficients are kept and others are zeroed-out to reduce computation complexity [31]. With the zeroing-out, multiplication count is reduced to 37.5% for both 64-point DCT-2 and 32-point DST-7/DCT-8 for  $64 \times 64$  and  $32 \times 32$  transform blocks, respectively. Moreover, memory usage for 64-point DCT-2 and 32-point DST-7/DCT-8 is reduced to a half since only half of transform basis vectors are involved in the transform process. The multiplication counts per sample for all combinations of transform kernels and block shapes are summarized in Table 4.

In Table 4,  $M$ ,  $N$ ,  $m$ , and  $n$  denote TB width, TB height, width and height of top-left non-zero coefficients region, respectively. The analysis in Table 4 is based on assumption that DCT-2 is computed using a partial butterfly structure [11]. When performing inverse transform, the vertical transform is performed first with the top-left  $m$  by  $n$  non-zero coefficient region as input and the output is an  $M \times N$  block with only the left  $m \times N$  region being nonzero. Then the horizontal transform is performed, and the output is an  $M \times N$  residual block. During this transform process, only the multiplication counts associated with non-zero inputs were calculated, using the following formula, where  $N_{mul}$  is the total number of multiplication operations needed for a 2-D transform.

$$N_{mul} = \begin{cases} \frac{C_N}{N \cdot r_M \cdot r_N} + \frac{C_M}{M \cdot r_M} & \text{for DCT-2} \\ \frac{mn}{M} + m & \text{for DST-7/DCT-8} \end{cases}, \quad (4)$$

$$C_N = \frac{N^2 + 2}{3}, \quad r_N = \frac{N}{n}, \quad \text{and} \quad r_M = \frac{M}{m}, \quad (5)$$

In Equation (4) and (5),  $C_N$  corresponds to the multiplication count for a one-dimensional  $N$ -point DCT-2. Without performing zeroing-out for 32-point DST-7/DCT-8, i.e.,  $m$  equals to  $M$  and  $n$  equals to  $N$ , the worst-case multiplication counts for 32-point DST-7/DCT-8 becomes 64, as shown in Table 4, which is excessive for hardware implementation. For



TABLE 6  
CODING GAIN OF MTS AND LFNST FOR AI AND RA CONFIGURATIONS IN CTC

| Resolution            | Sequences       | All Intra     |                  |                  |               |                  |                  | Random Access |                  |                  |               |                  |                  |
|-----------------------|-----------------|---------------|------------------|------------------|---------------|------------------|------------------|---------------|------------------|------------------|---------------|------------------|------------------|
|                       |                 | MTS           |                  |                  | LFNST         |                  |                  | MTS           |                  |                  | LFNST         |                  |                  |
|                       |                 | BDR           | $\Delta T_{Enc}$ | $\Delta T_{Dec}$ | BDR           | $\Delta T_{Enc}$ | $\Delta T_{Dec}$ | BDR           | $\Delta T_{Enc}$ | $\Delta T_{Dec}$ | BDR           | $\Delta T_{Enc}$ | $\Delta T_{Dec}$ |
| Class A1              | Tango2          | -1.29%        | 116%             | 105%             | -2.75%        | 91%              | 97%              | -0.57%        | 104%             | 100%             | -1.43%        | 105%             | 99%              |
|                       | FoodMarket4     | -1.92%        | 112%             | 103%             | -2.03%        | 90%              | 99%              | -0.81%        | 104%             | 100%             | -1.07%        | 106%             | 100%             |
|                       | Campfire        | -0.84%        | 118%             | 101%             | -0.56%        | 97%              | 102%             | -0.68%        | 106%             | 100%             | -1.18%        | 107%             | 100%             |
| Class A2              | CatRobot1       | -1.43%        | 118%             | 104%             | -1.23%        | 93%              | 101%             | -0.55%        | 105%             | 99%              | -0.83%        | 105%             | 100%             |
|                       | DaylightRoad2   | -1.43%        | 120%             | 101%             | -0.76%        | 92%              | 99%              | -0.53%        | 104%             | 99%              | -0.72%        | 103%             | 99%              |
|                       | ParkRunning3    | -1.31%        | 118%             | 107%             | 0.02%         | 101%             | 106%             | -0.94%        | 110%             | 101%             | -0.07%        | 109%             | 100%             |
| Class B               | MarketPlace     | -2.05%        | 123%             | 109%             | -0.34%        | 92%              | 104%             | -1.12%        | 108%             | 105%             | -0.18%        | 106%             | 103%             |
|                       | RitualDance     | -1.77%        | 115%             | 107%             | -1.11%        | 91%              | 105%             | -1.20%        | 108%             | 105%             | -0.55%        | 108%             | 104%             |
|                       | Cactus          | -1.54%        | 121%             | 105%             | -0.97%        | 93%              | 104%             | -0.86%        | 109%             | 104%             | -0.72%        | 107%             | 102%             |
|                       | BasketballDrive | -0.85%        | 117%             | 105%             | -1.30%        | 89%              | 102%             | -0.58%        | 108%             | 106%             | -0.99%        | 107%             | 105%             |
|                       | BQTerrace       | -0.80%        | 122%             | 106%             | -0.90%        | 90%              | 106%             | -0.63%        | 108%             | 105%             | -0.77%        | 105%             | 103%             |
| Class C               | BasketballDrill | -0.66%        | 115%             | 102%             | -3.13%        | 89%              | 103%             | -0.65%        | 111%             | 102%             | -1.60%        | 108%             | 102%             |
|                       | BQMall          | -1.10%        | 118%             | 103%             | -0.69%        | 90%              | 103%             | -0.71%        | 109%             | 102%             | -0.13%        | 106%             | 101%             |
|                       | PartyScene      | -0.59%        | 121%             | 106%             | -0.38%        | 91%              | 104%             | -0.51%        | 110%             | 102%             | -0.11%        | 107%             | 102%             |
|                       | RaceHorses      | -0.88%        | 121%             | 108%             | -1.00%        | 93%              | 105%             | -0.57%        | 109%             | 103%             | -0.74%        | 106%             | 101%             |
| Class D               | BasketballPass  | -0.82%        | 116%             | 101%             | -1.18%        | 93%              | 99%              | -0.56%        | 109%             | 102%             | -0.59%        | 108%             | 102%             |
|                       | BQSquare        | -0.46%        | 121%             | 101%             | -0.44%        | 90%              | 99%              | -0.32%        | 108%             | 99%              | -0.59%        | 106%             | 99%              |
|                       | BlowingBubbles  | -0.62%        | 120%             | 99%              | -0.92%        | 92%              | 100%             | -0.43%        | 110%             | 101%             | -0.31%        | 108%             | 101%             |
|                       | RaceHorses      | -0.86%        | 121%             | 103%             | -1.49%        | 93%              | 101%             | -0.52%        | 110%             | 104%             | -0.71%        | 107%             | 102%             |
| Class E               | FourPeople      | -1.56%        | 118%             | 108%             | -1.19%        | 89%              | 105%             |               |                  |                  |               |                  |                  |
|                       | Johnny          | -1.16%        | 117%             | 106%             | -1.46%        | 87%              | 102%             |               |                  |                  |               |                  |                  |
|                       | KristenAndSara  | -1.40%        | 117%             | 105%             | -1.63%        | 88%              | 101%             |               |                  |                  |               |                  |                  |
| Class A1              |                 | -1.35%        | 115%             | 103%             | -1.78%        | 93%              | 99%              | -0.69%        | 105%             | 100%             | -1.23%        | 106%             | 99%              |
| Class A2              |                 | -1.39%        | 119%             | 104%             | -0.65%        | 96%              | 102%             | -0.68%        | 107%             | 100%             | -0.54%        | 106%             | 100%             |
| Class B               |                 | -1.40%        | 120%             | 106%             | -0.92%        | 91%              | 104%             | -0.88%        | 108%             | 105%             | -0.64%        | 107%             | 103%             |
| Class C               |                 | -0.81%        | 119%             | 105%             | -1.30%        | 91%              | 104%             | -0.61%        | 110%             | 102%             | -0.65%        | 107%             | 101%             |
| Class D               |                 | -0.69%        | 119%             | 101%             | -1.01%        | 92%              | 100%             | -0.46%        | 109%             | 102%             | -0.55%        | 107%             | 101%             |
| Class E               |                 | -1.37%        | 117%             | 106%             | -1.43%        | 88%              | 103%             |               |                  |                  |               |                  |                  |
| <b>Total Average*</b> |                 | <b>-1.25%</b> | <b>118%</b>      | <b>105%</b>      | <b>-1.19%</b> | <b>91%</b>       | <b>103%</b>      | <b>-0.73%</b> | <b>108%</b>      | <b>102%</b>      | <b>-0.74%</b> | <b>106%</b>      | <b>101%</b>      |

\*Average results do not include Class D

64-point DCT-2, by zeroing out, more than half of multiplication counts is reduced. The coding performance impact from skipping high frequencies was observed to be around 0.1% and 0.05% bit-rate increase for AI and RA, respectively [31].

The memory usage of different transform types and sizes are summarized in Table 5 measured in bytes. In both HEVC and VVC, the smaller DCT-2 kernel can be sampled from larger one. In certain implementation, with this feature, the memory usage of DCT-2 with different sizes is same to the 64-point DCT-2, i.e. 2048 bytes. Furthermore, the DCT-8 basis vector can be derived by arranging the DST-7 basis in reverse order with sign value being alternated over all basis vectors [54]. Hence, DCT-8 can be derived from DST-7 without extra memory cost.

### B. Secondary Transform

In the initial design of Reduced Secondary Transform (RST) [40], which is a previous version of LFNST, there are 35 transform sets. Each transform set is composed of 3 kernels except for Planar and DC modes that 2 kernels are defined in the transform set. In addition, for 8×8 and larger block sizes, 16×64 transform matrices are used in RST. In LFNST, total 4 transform sets are specified with 2 kernels for each transform set. The transform matrix dimensions of LFNST are 48×16 and 16×16 for LFNST8 and LFNST4, respectively. Moreover, the required worst-case number of multiplications per sample is 8,

and up to 8 non-zero LFNST coefficients are processed for 4×4 and 8×8 blocks with further reduced LFNST matrices sampled from the 8×16 LFNST4 and 8×48 LFNST8 transform matrices, respectively. By further zeroing-out primary coefficients [42] when LFNST is applied, only the first 16 or 8 LFNST coefficients are kept and all other coefficients are set as zeros. Therefore, the maximum number of final transform coefficients becomes only 16 or 8, and the latter case happens in case of 4×4 or 8×8 transform block. This zeroing-out not only reduces computation complexity but also decreases coefficient buffer storage, which will be detailed below.

In Table 4, the total multiplication counts with both LFNST and DCT-2 being applied are reported. Given the output of LFNST, which is the top-left 48 or 16 coefficients, the number of multiplications needed for inverse primary transform is analyzed as follows. As shown in Figure 7, the left 4×8 (region A in Figure 7) and its neighboring 4×4 (region B in Figure 7) are first fed into the inverse vertical transform to generate the first 4 and the next 4 columns of output, which together form the M×N with only left 8×N block being nonzero (region C in Figure 7). Then this 8×N block is fed into the inverse horizontal transform and the output is the M×N residual block, as indicated by region D in Figure 7. In this analysis, it is assumed that the DCT-2 computation is performed using partial butterfly structure. Based on the above analysis, the following Equation (6) can be obtained for deriving the multiplication counts for DCT-2 in the case of a combination of LFNST and DCT-2,

TABLE 7  
CODING GAIN OF SBT FOR RA AND LDB CONFIGURATIONS IN CTC

| Sequences             | Random Access |                  |                  | Low Delay     |                  |                  |
|-----------------------|---------------|------------------|------------------|---------------|------------------|------------------|
|                       | BDR           | $\Delta T_{Enc}$ | $\Delta T_{Dec}$ | BDR           | $\Delta T_{Enc}$ | $\Delta T_{Dec}$ |
| Tango2                | -0.04%        | 104%             | 99%              |               |                  |                  |
| FoodMarket4           | -0.11%        | 103%             | 99%              |               |                  |                  |
| Campfire              | -0.17%        | 105%             | 99%              |               |                  |                  |
| CatRobot1             | -0.18%        | 105%             | 99%              |               |                  |                  |
| DaylightRoad2         | -0.41%        | 106%             | 100%             |               |                  |                  |
| ParkRunning3          | -0.72%        | 111%             | 100%             |               |                  |                  |
| MarketPlace           | -0.32%        | 107%             | 101%             | -0.50%        | 111%             | 99%              |
| RitualDance           | -0.36%        | 105%             | 101%             | -0.47%        | 108%             | 104%             |
| Cactus                | -0.29%        | 106%             | 100%             | -0.53%        | 110%             | 101%             |
| BasketballDrive       | -0.24%        | 106%             | 100%             | -0.51%        | 110%             | 96%              |
| BQTerrace             | -0.47%        | 110%             | 101%             | -0.82%        | 115%             | 100%             |
| BasketballDrill       | -0.19%        | 105%             | 101%             | -0.31%        | 107%             | 98%              |
| BQMall                | -0.48%        | 106%             | 101%             | -0.63%        | 110%             | 97%              |
| PartyScene            | -0.35%        | 106%             | 102%             | -0.56%        | 111%             | 96%              |
| RaceHorses            | -0.38%        | 107%             | 102%             | -0.53%        | 111%             | 100%             |
| BasketballPass        | -0.43%        | 106%             | 102%             | -0.61%        | 110%             | 98%              |
| BQSquare              | -0.21%        | 105%             | 102%             | -0.61%        | 109%             | 100%             |
| BlowingBubbles        | -0.36%        | 107%             | 102%             | -0.55%        | 113%             | 101%             |
| RaceHorses            | -0.40%        | 108%             | 102%             | -0.64%        | 112%             | 102%             |
| FourPeople            |               |                  |                  | -0.28%        | 105%             | 98%              |
| Johnny                |               |                  |                  | -0.01%        | 103%             | 100%             |
| KristenAndSara        |               |                  |                  | -0.18%        | 104%             | 99%              |
| Class A1              | -0.11%        | 104%             | 100%             |               |                  |                  |
| Class A2              | -0.44%        | 107%             | 100%             | -0.56%        | 111%             | 101%             |
| Class B               | -0.34%        | 108%             | 106%             | -0.51%        | 112%             | 113%             |
| Class C               | -0.35%        | 106%             | 103%             | -0.60%        | 112%             | 108%             |
| Class D               | -0.35%        | 107%             | 102%             | -0.16%        | 105%             | 108%             |
| Class E               |               |                  |                  | -0.20%        | 106%             | 106%             |
| <b>Total Average*</b> | <b>-0.31%</b> | <b>106%</b>      | <b>100%</b>      | <b>-0.44%</b> | <b>109%</b>      | <b>99%</b>       |

\*Average results do not include Class D

TABLE 8  
CODING GAIN OF MTS+LFNST FOR AI AND RA CONFIGURATIONS IN CTC

| Sequences             | All Intra     |                  |                  | Random Access |                  |                  |
|-----------------------|---------------|------------------|------------------|---------------|------------------|------------------|
|                       | BDR           | $\Delta T_{Enc}$ | $\Delta T_{Dec}$ | BDR           | $\Delta T_{Enc}$ | $\Delta T_{Dec}$ |
| Class A1              | -4.67%        | 163%             | 103%             | -2.54%        | 116%             | 100%             |
| Class A2              | -3.11%        | 172%             | 107%             | -1.60%        | 115%             | 101%             |
| Class B               | -3.28%        | 172%             | 107%             | -2.03%        | 117%             | 104%             |
| Class C               | -2.74%        | 170%             | 103%             | -1.54%        | 118%             | 102%             |
| Class E               | -4.20%        | 164%             | 105%             |               |                  |                  |
| <b>Total Average*</b> | <b>-3.52%</b> | <b>169%</b>      | <b>105%</b>      | <b>-1.91%</b> | <b>117%</b>      | <b>102%</b>      |

where  $C_M$ ,  $C_N$ ,  $r_M$ , and  $r_N$  are the same as in Equation (4).

$$N_{mul} = 4 \cdot \frac{C_N}{r_N} + 4 \cdot \frac{C_N}{2 \cdot r_N} + N \cdot \frac{C_M}{r_M} \quad (6)$$

It is noted that the overall complexity of LFNST and DCT-2 combination is smaller than the worst case of only primary transform being applied.

## VI. EXPERIMENTAL RESULTS

### A. Coding Efficiency

In this section, the coding performance of MTS (the explicit MTS), LFNST, SBT and 64-point DCT-2 are studied using the reference software VTM-9.0. The common test conditions (CTC), as defined by JVET for evaluating technical contributions during the standardization process of VVC, are used to perform the experiments. The test set defined in CTC includes a set of 32 video sequences, ranging from WQVGA (416×240) to 4K (3840×2160) resolutions, including four sequences with synthetic content, e.g., screen content, gaming content and mixed natural and screen content. The Bjøntegaard-delta rate (BDR) [53] is used to evaluate the coding performance. The tested quantization parameters (QP)

TABLE 9  
COMBINED CODING GAIN OF MTS, LFNST AND SBT FOR AI, RA AND LDB CONFIGURATIONS IN CTC

|                | Random Access |                  |                  | Low Delay     |                  |                  |
|----------------|---------------|------------------|------------------|---------------|------------------|------------------|
|                | BDR           | $\Delta T_{Enc}$ | $\Delta T_{Dec}$ | BDR           | $\Delta T_{Enc}$ | $\Delta T_{Dec}$ |
| Class A1       | -2.60%        | 122%             | 100%             |               |                  |                  |
| Class A2       | -1.83%        | 124%             | 100%             |               |                  |                  |
| Class B        | -2.22%        | 126%             | 104%             | -0.94%        | 113%             | 100%             |
| Class C        | -1.76%        | 127%             | 108%             | -0.84%        | 114%             | 105%             |
| Class E        |               |                  |                  | -0.42%        | 106%             | 102%             |
| <b>Average</b> | <b>-2.10%</b> | <b>125%</b>      | <b>103%</b>      | <b>-0.77%</b> | <b>112%</b>      | <b>102%</b>      |

TABLE 10  
COMBINED CODING GAIN OF 64-POINT TRANSFORM, MTS, LFNST AND SBT FOR AI, RA AND LDB CONFIGURATIONS IN CTC

|                | All Intra     | Random Access | Low Delay     |
|----------------|---------------|---------------|---------------|
| Class A1       | -7.12%        | -6.65%        |               |
| Class A2       | -4.06%        | -3.17%        |               |
| Class B        | -4.08%        | -3.41%        | -2.76%        |
| Class C        | -2.87%        | -2.04%        | -1.32%        |
| Class E        | -5.28%        |               | -2.31%        |
| <b>Average</b> | <b>-4.51%</b> | <b>-3.65%</b> | <b>-2.17%</b> |

are 22, 27, 32 and 37. The test conditions include All Intra (AI), Random Access (RA) and Low Delay B (LDB) as defined in the configuration files associated with the reference software VTM-9.0.

When evaluating the run-time difference, the following  $\Delta T$  measurement is used,

$$\Delta T = \frac{T_{Proposed}}{T_{Anchor}} \times 100\%, \quad (7)$$

where  $T_{Anchor}$  and  $T_{Proposed}$  represent the runtimes of the anchor and proposed method. This criterion is used to estimate algorithm complexity in terms of encoder and decoder software run-time, where 100% indicates no run-time difference. To evaluate the overall BDR for multiple color components, an overall PSNR value derived as the sum of weighted luma and chroma PSNR values. The weights are 6/8, 1/8 and 1/8 for the PSNR value of Y, Cb and Cr components, respectively. These set of weights have been used in JVET for tool reporting [55].

When evaluating the coding performance, i.e., BDR, the anchor used in the results is VTM-9.0 with one (or multiple) tool(s) being disabled, and the test is VTM-9.0. For example, when reporting the coding gain of MTS, the anchor is VTM-9.0 with MTS being disabled (MTS=0). Therefore, a negative BDR number indicates a bitrate reduction, i.e., coding gain.

The coding performance and software run-time of MTS and LFNST are reported in Table 6. For AI configuration, the average coding gains of MTS and LFNST are -1.25% and -1.19%, respectively. For RA configuration, the average gains of MTS and LFNST are -0.73% and -0.74%, respectively. The encoding run-time is approximately reduced by 10% when LFNST is enabled. One of main reasons is that, two different MTS encoding procedures are applied depending on the enabling of LFNST, and heavier MTS encoding procedure is applied when LFNST is turned off, which is detailed in Section IV. The combined coding gains of MTS and LFNST are also shown in Table 8, and it is noted that the combined coding gain of MTS and LFNST is greater than the sum of individual

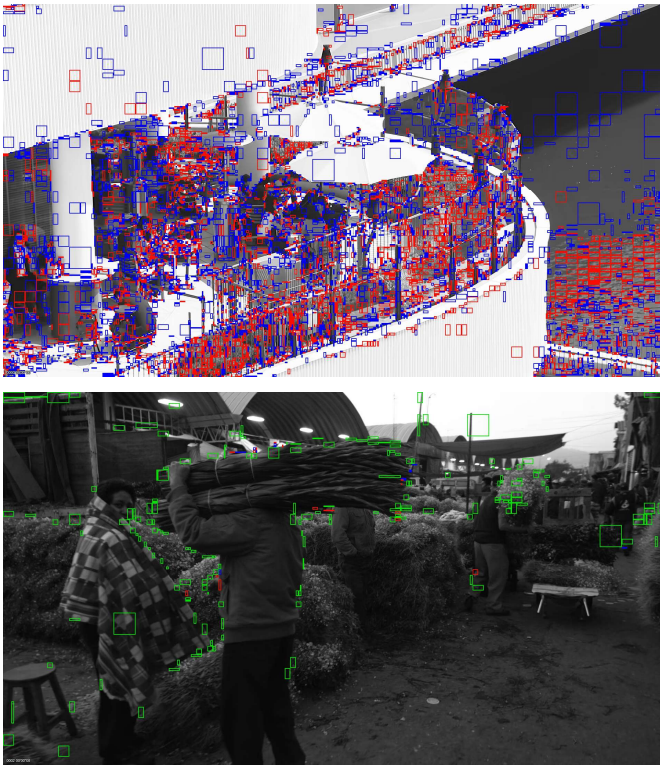


Figure 8: Reconstructed picture of BQTerrace (top) and MarketPlace (bottom) with coding blocks selecting MTS, LFNST and SBT highlighted in red, blue and green, respectively.

coding gains, i.e., -3.52% and -1.91% for AI and RA configurations, respectively. This indicates that MTS and LFNST has some overlap on the coding gain, which is expected since both coding tools introduce additional transform kernels in a similar fashion as competing transform modes.

The results for SBT are reported in Table 7, the average gains for RA and LDB are -0.31% and -0.44%, respectively. With more advanced inter prediction and block partition in VVC, the inter prediction residue is greatly reduced, especially for RA, which leaves less room for SBT as well as other transform tools like inter MTS. Compared with inter MTS, SBT showed 0.3% higher coding gain in LDB with 40% of encoding runtime on VTM-3.0 [50], and achieved the best tradeoff between coding gain and encoding complexity among all transform tools for inter residual during VVC standardization.

The combined coding gains of MTS, LFNST and SBT, are reported in Table 9. Only RA and LDB results are reported since SBT only has an impact for those two test configurations. The combined coding gains of all transform coding techniques, including 64-point transform, MTS, LFNST and SBT, are also reported in Table 10, and -4.5%, -3.6% and -2.2% are achieved for AI, RA and LDB configurations, respectively. The run-time impact of 64-point transform is very minor and not shown in this paper due to page limitations.

### B. Tool Analysis

In Figure 8, the coding blocks using MTS, LFNST and SBT are highlighted using red, blue and green boxes, respectively. The above one is the first reconstructed picture (Intra frame) of BQTerrace (1920×1080), and the bottom one is the third

reconstructed picture (B frame) of MarketPlace (1920×1080). Those pictures are reconstructed using bitstream coded by VTM-9.0 following the CTC under RA configuration using QP 37 (BQTerrace) and 27 (MarketPlace). It is observed that MTS and LFNST coded blocks cover a considerable percentage of blocks in Intra frame coding. However, for blocks with smooth or vertical texture patterns (top-left part of the picture), neither MTS nor LFNST is used frequently. These blocks typically show little specific residual patterns, where separable transform using DCT-2 seems to be efficient. For Inter coding, it is observed that SBT is more frequently used along the moving object boundaries. This represents a typical case where SBT can be helpful, since the residual can be more frequently locally distributed within one coding block.

From Table 6 and Table 7, it is noticed that MTS and SBT contribute relatively consistent coding gain across different video content and resolutions, while the coding gain of LFNST is higher for an input sequence with rich directional texture patterns (BasketballDrill) of which coding gain can go up to 3% for AI coding configuration. For SBT, from Table 7, it is noted that the coding gain is consistent across different resolutions, and for sequences with complex motion, and frequent object occlusions (ParkRunning3), the RA coding gain of SBT is peaked at 0.7%.

## VII. CONCLUSIONS

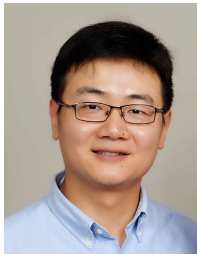
The transform coding design for VVC has been described in this paper, including new primary transform kernel types with explicit and implicit selection schemes, explicitly signalled non-separable secondary transform with reduced kernels and sub-block transform partitioning. Besides these new transform tools, VVC also extended several aspects of the HEVC transform design, including 64-point DCT-2 and transform for rectangular block shape. Moreover, normative zeroing-out schemes are applied in VVC for 64-point DCT-2, 32-point DST-7/DCT-8 and LFNST to reduce complexity. Experimental results show significant coding gains contributed by MTS, LFNST, SBT and 64-point DCT-2, especially for intra coding with up to 7.1% BD rate reduction for 4K video contents.

## REFERENCES

- [1] *Video Codec for Audiovisual Services at p x 64 kbit/s*, ITU-T Rec. H.261, Dec, 1990.
- [2] *Information Technology-Coding of Moving Pictures and Associated Audio for Digital Storage Media at up to About 1.5 Mbit/s: Video*, ISO/IEC 11172-2 MPEG-1, 1993.
- [3] *Information Technology-Generic Coding of Moving Pictures and Associated Audio Information: Video*, ITU-T Rec. H.262 and ISO/IEC 13818-2 MPEG-2, 1995.
- [4] *Video Coding for Low Bitrate Communication*, ITU-T Rec. H.263 Version 1, 1995, Version 2, Sep. 1997.
- [5] T. Wiegand, G. J. Sullivan, G. Bjontegaard, and A. Luthra, "Overview of the H.264/AVC video coding standard", in *IEEE Trans. Circuits Systems Video Technol.*, vol.13, no.7, pp.560-576, Jul. 2003.
- [6] G. J. Sullivan, J. Ohm, W. J. Han, and T. Wiegand, "Overview of the High Efficiency Video Coding (HEVC) standard," in *IEEE Trans. Circuits and Syst. for Video Technol.*, vol. 22, no. 12, pp. 1649–1668, 2012.
- [7] N. Ahmed, T. Natarajan and K. R. Rao, "Discrete Cosine Transform," in *IEEE Transactions on Computers*, vol. C-23, no. 1, pp. 90-93, Jan. 1974.
- [8] K. Karhunen, "Über lineare Methoden in der Wahrscheinlichkeitsrechnung", *Ann. Acad. Sci. Fennicae. Ser. A. I. Math.-Phys.*, pp. 1-79, 1947.

- [9] W.-H. Chen, C. H. Smith and S. Fralick, "A fast computational algorithm for the Discrete Cosine Transform," in *IEEE Transactions on Communications*, vol. 25, no. 9, pp. 1004-1009, Sep. 1977.
- [10] H. S. Malvar, et al., "Low-complexity transform and quantization in H.264/AVC," in *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 13, no. 7, pp. 598-603, July 2003.
- [11] M. Budagavi, et al., "Core Transform Design in the High Efficiency Video Coding (HEVC) Standard," in *IEEE Journal of Selected Topics in Signal Processing*, vol. 7, no. 6, pp. 1029-1041, Dec. 2013.
- [12] B. Zeng and J. Fu, "Directional Discrete Cosine Transforms—A New Framework for Image Coding," in *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 18, no. 3, pp. 305-313, March 2008.
- [13] B. Bross, Y.-K. Wang, Y. Ye, S. Liu, G. J. Sullivan, and J.-R. Ohm, "Overview of the Versatile Video Coding (VVC) Standard and its Applications," preprint submitted to *IEEE Trans. Circuits and Syst. for Video Tech.*, December 2020.
- [14] K. Rose, A. Heiman and I. Dinstein, "DCT/DST alternate-transform image coding," in *IEEE Transactions on Communications*, vol. 38, no. 1, pp. 94-101, Jan. 1990.
- [15] Y. Ye and M. Karczewicz, "Improved H.264 intra coding based on bi-directional intra prediction, directional transform, and adaptive coefficient scanning," in *Proc. 15th IEEE International Conference on Image Processing*, San Diego, CA, USA, 2008, pp. 2116-2119.
- [16] J. Han, A. Saxena, V. Melkote and K. Rose, "Jointly optimized spatial prediction and block transform for video and image coding," in *IEEE Trans. Image Process.*, vol. 21, no. 3, pp. 1874-1884, Apr. 2012.
- [17] A. K. Jain, "A Sinusoidal Family of Unitary Transforms," in *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. PAMI-1, no. 4, pp. 356-365, Oct. 1979.
- [18] X. Zhao, L. Zhang, S. Ma, and W. Gao, "Rate-Distortion Optimized Transform," *ISO/IEC JTC1/SC29/WG11*, m16926, Xi'an, CN, Oct. 2009.
- [19] S.-C. Lim, et al., "Rate-distortion optimized adaptive transform coding," *Opt. Eng.*, vol. 48, no. 8, pp. 087004-1-087004-14, Aug. 2009.
- [20] X. Zhao, J. Chen, M. Karczewicz, A. Said, and V. Seregin, "Joint separable and non-separable transforms for next-generation video coding," in *IEEE Transactions on Image Processing*, vol. 27, no. 5, pp. 2514-2525, May 2018.
- [21] T. Biatek, et al., "Transform Competition for Temporal Prediction in Video Coding," in *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 29, no. 3, pp. 815-826, Mar. 2019.
- [22] A. Arrufat, P. Philippe, and O. Déforges, "Non-separable mode dependent transforms for intra coding in HEVC," in *IEEE Proc. on Visual Communications and Image Processing*, pp. 61-64, 2014.
- [23] S. Takamura and A. Shimizu, "On Intra Coding Using Mode Dependent 2D-KLT," in *Proc. Picture Coding Symposium*, pp. 137-140, 2013.
- [24] H. E. Egilmez, et al., "Row-column transforms: Low-complexity approximation of optimal non-separable transforms," in *Proc. 2016 IEEE International Conference on Image Processing (ICIP)*, Phoenix, AZ, 2016, pp. 2385-2389.
- [25] A. Saxena, and F. C. Fernandes. "On secondary transforms for prediction residual," in *Proc. IEEE International Conference on Image Processing*, pp. 2489-2492, 2012.
- [26] X. Zhao, et al., "NSST: Non-Separable Secondary Transforms for Next Generation Video Coding," in *Proc. Picture Coding Symposium*, Nuremberg, 2016, pp. 1-5.
- [27] A. Said, et al., "Highly efficient non-separable transforms for next generation video coding," in *Proc. Picture Coding Symposium (PCS)*, Nuremberg, 2016, pp. 1-5.
- [28] M. Wien, "Variable block-size transforms for H.264/AVC," in *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 13, no. 7, pp. 604-613, July 2003.
- [29] I. Kim, J. Min, T. Lee, W. Han and J. Park, "Block Partitioning Structure in the HEVC Standard," in *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 22, no. 12, pp. 1697-1706, Dec. 2012.
- [30] C. Zhang, et al., "Video Coding Using Spatially Varying Transform," in *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 21, no. 2, pp. 127-140, Feb. 2011.
- [31] M. Koo, M. Salehifar, J. Lim, S. Kim, "CE6-related: 32 point MTS based on skipping high frequency coefficients," *Joint Video Experts Team of ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11*, JVET-M0297, Marrakech, MA, 9-18 Jan. 2019.
- [32] X. Zhao, et al., "CE6: Fast DST-7/DCT-8 with dual implementation support (Test 6.2.3)," *Joint Video Experts Team of ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11*, JVET-M0497, Marrakech, MA, 2019.
- [33] Z. Zhang, et al., "Fast DST-7/DCT-8 with Dual Implementation Support for Versatile Video Coding," in *IEEE Transactions on Circuits and Systems for Video Technology*, Accepted.
- [34] J. Pfaff, et al., "Intra Prediction and Mode Coding in VVC", submitted to *IEEE Transactions on Circuits and Systems for Video Technology*.
- [35] S. De-Luxán-Hernández et al., "Non-CE6: Combination of JVET-P0196 and JVET-P0392 on applying LFNST for ISP Blocks and simplifying the transform signalling," *Joint Video Experts Team of ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11*, JVET-P1026, Geneva, CH, 2019.
- [36] J. Lainema, "CE6-related: Shape adaptive transform selection," *Joint Video Experts Team of ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11*, JVET-L0134, Macao, CN, 2018.
- [37] H. Gao, et al., "Non-CE6: Combined Test of JVET-N0172/JVET-N0375/JVET-N0419/JVET-N0420 on Unification of Implicit Transform Selection," *Joint Video Experts Team of ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11*, JVET-N0866, Geneva, CH, 2019.
- [38] K. Naser, F. Le Léannec, T. Poirier, "Non-CE6 Interaction between Implicit MTS and LFNST and MIP," *Joint Video Experts Team of ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11*, JVET-O0529, Gothenburg, SE, 2019.
- [39] M. Koo, et al., "Low Frequency Non-Separable Transform (LFNST)," in *Proc. Picture Coding Symposium (PCS)*, Ningbo, China, 2019, pp. 1-5.
- [40] M. Koo, et al., "CE6: Reduced Secondary Transform (RST) (CE6-3.1)," *Joint Video Experts Team of ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11*, JVET-N0193, Geneva, CH, 2019.
- [41] M.-S. Chiang, et al., "CE6-related: Simplifications for LFNST," *Joint Video Experts Team (JVET) of ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11*, JVET-O0292, Gothenburg, SE, 3-12 July 2019.
- [42] M. Siekmann, H. Schwarz, D. Marpe, T. Wiegand, "CE6-2.1: Simplification of Low Frequency Non-Separable Transform" *Joint Video Experts Team of ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11*, JVET-O0094, Gothenburg, SE, 2019.
- [43] H. Schwarz, et al., "Quantization and Entropy Coding in the Versatile Video Coding (VVC) Standard", submitted to *IEEE Transactions on Circuits and Systems for Video Technology*.
- [44] H. E. Egilmez, et al., "Combination of JVET-Q0106, JVET-0686 and JVET-Q0133 for LFNST Signaling, Latency Reduction and Scaling Process," *Joint Video Experts Team of ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11*, JVET-P0784, Brussels, BE, 2020.
- [45] Z. Zhang, et al., "Non-CE6: On LFNST transform set selection for a CCLM coded block," *Joint Video Experts Team of ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11*, JVET-O0219, Gothenburg, SE, 2019.
- [46] J. Pfaff et al., "Non-CE3: Simplifications of MIP," *Joint Video Experts Team of ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11*, JVET-O0925, Gothenburg, SE, 2019.
- [47] B. Heng, et al., "AHG16: Combination of LFNST with transform skip," *Joint Video Experts Team of ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11*, JVET-Q0106, Brussels, BE, 2020.
- [48] J. An, X. Zhao, X. Guo, S. Lei, "Non-CE7: Boundary-Dependent Transform for Inter-Predicted Residue", *Joint Collaborative Team on Video Coding (JCT-VC) of ITU-T SG16 WP3 and ISO/IEC JTC1/SC29/WG11*, JCTVC-G281, Geneva, CH., 2011.
- [49] Y. Zhao, H. Yang, J. Chen, "CE6: Spatially Varying Transform (Test 6.1.12.1)," *Joint Video Experts Team (JVET) of ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11*, JVET-K0139, Ljubljana, SI, 2018.
- [50] Y. Zhao, H. Gao, H. Yang, J. Chen, "CE6: Sub-block transform for inter blocks (Test 6.4.1)," *Joint Video Experts Team of ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11*, JVET-M0140, Marrakech, MA, 2019.
- [51] Y. Zhao, et al., "CE6: RQT-like sub-block transform for inter blocks (Test 6.4.4)," *Joint Video Experts Team of ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11*, JVET-M0141, Marrakech, MA, 2019.
- [52] X. Zhao, X. Li, S. Liu, "Non-CE6: Configurable maximum transform size in VVC," *Joint Video Experts Team of ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11*, JVET-O0545, Gothenburg, SE, 2019.
- [53] G. Bjontegaard, "Improvement of BD-PSNR model," *ITU-T SG16/Q6*, Doc. VCEG-A111, Berlin, Germany, 2008.
- [54] M. Koo, M. Salehifar, J. Lim, S. Kim, "CE 6-1.11: AMT replacement and restriction," *Joint Video Experts Team of ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11*, JVET-K0096, Ljubljana, SI, 2018.
- [55] W.-J. Chien et al., "JVET AHG report: Tool reporting procedure (AHG13)," *Joint Video Experts Team of ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11*, JVET-S0013, 2020.





**Xin Zhao** (M'18-SM'21) received the B.S. degree in electronic engineering from Tsinghua University, Beijing, China, and the Ph.D. degree in computer applications from Institute of Computing Technology, Chinese Academy of Sciences, Beijing, China. In 2017, he joined Tencent, Palo Alto, CA, USA, where he is currently a Principal Researcher and Manager of Multimedia Standards in Tencent Media Lab. Before 2017, he was a Staff Engineer with Qualcomm, San Diego, CA, USA. He has been actively contributing to the development of multiple video standards for over 10 years, such as HEVC, 3D extensions to H.264/AVC and HEVC and VVC. He has contributed over 200 standard proposals and published over 30 journal and conference papers. His research interests include image and video coding, video processing, and transmission.



**Seung-Hwan Kim** received the Ph.D. degree from Gwangju Institute of Science Technology (GIST) in 2008. In 2009, he was in University of Southern California (USC) as a postdoctoral research fellow. In 2011, he joined Sharp Labs of America where he has been active in video coding standardization in ITU-T and ISO/IEC for development of High-Efficiency Video

Coding (HEVC) and its extension projects including JVET (Joint Video Experts Team). From 2016, he has been with LGE US where he is an assistant vice president and has lead Versatile Video Coding (VVC) activity as a head of video codec part. His main research interests are image and video signal processing, data compression, pattern recognition, and machine learning.



**Yin Zhao** received the B.S. and Ph.D. degrees, both from Zhejiang University, Hangzhou, China, in 2008 and 2013, respectively. In 2013, he joined Huawei Technologies Hangzhou, China, where he is currently a Technical Expert. Since 2017, he has been actively participating in the development of several video coding standards, including the H.266/VVC,

AVS3 and MPEG-5 EVC standards. He designed and developed the Sub-block Transform and Local Dual Tree in H.266/VVC. He has been one of the software coordinators of AVS3 test model. His research interests include video coding, quality assessment and image processing.



**Hilmi E. Egilmez** (S'10-M'18) received his B.S. and M.Sc. degrees in Electrical and Electronics Engineering from Koc University, Istanbul, Turkey in 2010 and 2012, respectively. In 2017, he received his Ph.D. in Electrical Engineering from the University of Southern California, Los Angeles, USA. He is currently with

Qualcomm Technologies Inc., San Diego, USA. His research interests are in the areas of signal processing, machine learning and image/video coding.



**Moonmo Koo** received the B.S. and Ph.D. degrees in electrical and computer engineering from Seoul National University, Seoul, South Korea, in 2001 and 2011, respectively. He joined LG Electronics Inc. in 2011, and since then he did projects for depth estimation and 3D view synthesis hardware designs, developed video coding algorithms, and has been participating in video coding standardization in ITU-T and ISO/IEC for development of Versatile Video Coding (VVC). His research interests include image/video compression, machine learning based video coding, and HW/SW system design for video applications.



**Shan Liu** (M'01-SM'11) received the B.Eng. degree in electronic engineering from Tsinghua University, the M.S. and Ph.D. degrees in electrical engineering from the University of Southern California, respectively. She is now a Tencent Distinguished Scientist and General Manager of Tencent Media Lab. She was formerly Director of Media Technology Division at MediaTek USA. She was also formerly with MERL, Sony and IBM. She has been actively contributing to international standards since the last decade and served co-Editor of HEVC SCC and the emerging VVC. She has numerous technical contributions adopted into various standards, such as HEVC, VVC, OMAF, DASH and PCC, etc. Technologies and products developed by her and under her leadership have served several hundred million users. Dr. Liu holds more than 150 granted US and global patents and has published more than 80 journal and conference papers. Her research interests include audio-visual, high volume, immersive and emerging media compression, intelligence, transport, and systems.



**Jani Lainema** received his M.Sc. degree in computer science from the Tampere University of Technology, Finland in 1996. He joined the Visual Communications Laboratory of Nokia Research Center in 1996. Since then he has contributed to the designs of ITU-T's and MPEG's video coding standards as well as to the evolution of different multimedia service standards in 3GPP, DVB and DLNA. He is a Bell Labs Distinguished Member of Technical Staff and working currently as a Distinguished Scientist, Visual Media at Nokia Technologies, Tampere, Finland. His research interests include video, image and graphics coding and communications.



**Marta Karczewicz** received her M.Sc. and Ph.D. (with honors) degrees from Tampere University of Technology, Finland, in 1994 and 1997, respectively. From 1996 to 2006, she was with Nokia where she managed the “Visual Technologies” Competence Area and in 2004 she became Head of Nokia Research Center, Dallas, TX, USA. In 2006 she joined Qualcomm, where she is currently a Vice President, Technology, in the Multimedia R&D group. Since 1998, she has been an active participant in H.264/AVC, HEVC and VVC video codecs development within MPEG and ITU-T standardization forums. She holds over 300 granted US and over 100 granted European patents on video compression. She

is also one of the three finalists for the 2019 European Inventor Award in the “Lifetime Achievement” category. She is an author of 15 journal and over 40 conference papers including 4 invited papers to two special issues of the IEEE Transactions on Circuits and Systems for Video Technology on AVC and on HEVC. She received 1996-1997 EURASIP’s best paper award in Image Communication category.