

intra predicted to generate a prediction signal. The prediction signal is added to the respective residual signal of the first sub-partition to obtain corresponding reconstructed samples.

Thereafter, the reconstructed sample values of the first sub-partition can be available to generate a prediction of a second one of the sub-partitions. This process can be repeated sub-partition by sub-partition, until all sub-partitions from the coding block are reconstructed. In an example, all the sub-partitions share a same intra mode.

[0128] In an embodiment, the ISP coding mode is only tested with intra modes that are part of a most probable mode (MPM) list. Accordingly, if a block uses ISP, then a MPM flag can be inferred to be one. In addition, when ISP is used for a certain block, then a respective MPM list will be modified to exclude DC mode and to prioritize horizontal intra modes for the ISP horizontal split and vertical intra modes for the vertical one.

[0129] In ISP coding mode, each sub-partition can be regarded as a sub-TU, since the transform and reconstruction is performed individually for each sub-partition.

[0130] 3. Transform Coding with Expanded DCT-2 Transforms and Multiple Transform Selection (MTS)

[0131] In some embodiments, when both the height and width of a coding block is smaller than or equal to 64 samples, a transform size is always the same as a coding block size. When either the height or width of a coding block is larger than 64 samples, when performing transform or intra prediction, the coding block is further split into multiple sub-blocks, where the width and height of each sub-block is smaller than or equal to 64, and transform processing is performed on each sub-block.

[0132] 3.1 Transform Coding with Expanded DCT-2 Transforms

[0133] In some embodiments, in addition to 4-point, 8-point, 16-point and 32-point DCT-2 transforms described above, 2-point and 64-point DCT-2 transform can be used. FIGs. 13A-13E show a 64x64 transform core matrix of the 64-point DCT-2 transform.

[0134] 3.2 Explicitly Signaled Transform

[0135] In some embodiments, in addition to DCT-2 and 4x4 DST-7 transform coding, a multiple transform selection (MTS) (also known as enhanced multiple transform (EMT), or adaptive multiple transform (AMT)) can be used for residual coding of both inter and intra coded blocks. The MTS uses multiple selected transforms from discrete cosine transform (DCT) / discrete sine transform (DST) families other than the DCT-2 and 4x4 DST-7 transforms. The selection can be performed at an encoder and explicitly signaled from the encoder to a decoder.

For example, the selected transforms can include DST-7, or DCT-8 transforms. FIG. 14 shows transform basis functions of DST/DCT transforms. In some embodiments, the DST/DCT transform core matrices used in MTS are represented with 8-bit representation.

[0136] In some embodiments, MTS can be applied to CUs with both a width and height smaller than or equal to 32 samples. Whether to apply MTS or not can be controlled by a flag denoted by `mts_flag`. For example, when the `mts_flag` is equal to 0, only DCT-2 is applied to coding a residue block. When the `mts_flag` is equal to 1, which indicates MTS is applied, selected transforms can be used. For example, an index, denoted by `mts_idx`, can further be signaled using 2 bins to specify a horizontal and vertical transforms to be used.

[0137] FIG. 15 shows a table (1500) illustrating a mapping relationship between an `mts_idx` value and respective horizontal or vertical transforms. The row (1501) with the `mts_idx` having a value of -1 corresponds to a scenario where the `mts_flag` is equal to 0 (which indicates MTS is not applied), and DCT-2 transform is used. The rows (1502)-(1505) with the `mts_idx` having a value of 0, 1, 2, or 3 correspond to a scenario where the `mts_flag` is equal to 1 (which indicate MTS is applied). In the right two columns of the table (1500), 0 represents a transform type of DCT-2, 1 represents a transform type of DST-7, and 2 represents a transform type of DCT-8.

[0138] FIGs. 16A-16D show transform core matrices of a DST-7 transform type. FIGs. 17A-17D show transform core matrices of a DCT-8 transform type.

[0139] In some embodiments, MTS can be enabled or disabled using high level syntax (HLS) elements. Each of the HLS elements can be a video parameter set (VPS) syntax element, a sequence parameter set (SPS) syntax element, a picture parameter set (PPS) syntax element, a slice header syntax element, a tile header syntax element, or a tile group header syntax element, and the like. FIG. 18 shows an example of controlling usage of MTS using SPS syntax elements. As shown, an SPS syntax element, `sps_mts_enabled_flag`, can be signaled to indicate whether an MTS is enabled for a video sequence. When the MTS is enabled, two syntax elements, `sps_explicit_mts_intra_enabled_flag` and `sps_explicit_mts_inter_enabled_flag`, can be signaled to indicate whether the MTS is enabled for coding inter or intra predicted blocks, respectively.

[0140] In an embodiment, an implicit MTS is applied in case the above signaling based MTS (referred to as explicit MTS) is not used. With the implicit MTS, a transform selection can be made according to a block width and height instead of based on signaling. For example, with

an implicit MTS, a DST-7 transform can be selected for a shorter side of a transform block and a DCT-2 transform can be selected for a longer side of the transform block.

[0141] 3.3 Scenarios Where Implicit Transform Selection Is Applied

[0142] In various embodiments, for certain scenarios, DST-7 and/or DCT-8 can be used without being explicitly signaled. For example, DST-7 and/or DCT-8 can be used implicitly based on information that is available for both an encoder and a corresponding decoder. These scenarios include:

[0143] 3.3.1 Intra Sub-Partitioning (ISP)

[0144] For a residual block coded with an ISP mode, a horizontal transform is selected as DST-7 as long as a block width of the residual block is greater than or equal to 4 and less than or equal to 16, and a vertical transform is selected as DST-7 as long as a block height of the residual block is greater than or equal to 4 and less than or equal to 16.

[0145] 3.3.2 Sub-Block Transform (SBT)

[0146] For an SBT mode, for a sub-TU located at the left half (or quarter) or right half (or quarter) of a current CU, the horizontal transform can be DCT-8 or DST-7, respectively. Otherwise (a sub-TU has a same width with a current CU), DCT-2 can be used. For a sub-TU located at the top half (or quarter) or bottom half (or quarter) of a current CU, the vertical transform can be DCT-8 or DST-7, respectively. Otherwise (a sub-TU has a same height with a current CU), DCT-2 can be used.

[0147] 3.3.3 MTS Disabled By HLS Elements

[0148] For example, when the `sps_mts_enabled_flag` is signaled as true, but both the `sps_explicit_mts_intra_enabled_flag` and `sps_explicit_mts_inter_enabled_flag` are signaled as false, for intra prediction residuals, a horizontal transform can be selected as DST-7 as long as a respective block width is greater than or equal to 4 and less than or equal to 16, and a vertical transform can be selected as DST-7 as long as a respective block height is greater than or equal to 4 and less than or equal to 16.

[0149] 4. Non-Separable Secondary Transform (NSST)

[0150] 4.1 Initial Design of NSST

[0151] In some embodiments, a mode-dependent non-separable secondary transform (NSST) can be applied between a forward core transform and quantization (at an encoder) and between a de-quantization and inverse core transform (at a corresponding decoder). To keep low complexity, an NSST can only be applied to low frequency coefficients after the primary

(2010). In the forward secondary transform (2012), the 64 coefficients of the 4x4 sub-blocks A/B/C/D at the top-left corner of the coefficient block (2013) is represented into a 64-length vector, and multiplied with a transform matrix of a size of 16x64 according to the equation (Eq. 3), resulting in a 16-length vector. The elements in the 16-length vector are filled back into the top-left 4x4 sub-block A of the coefficient block (2013). The coefficients in the sub-blocks B/C/D can take values of zero. The resulting coefficients after the forward secondary transform 2012 are then quantized at the step of (2014), and entropy-coded to generate coded bits in a bitstream (2016).

[0162] The coded bits can be received at a decoder side, and entropy-decoded followed by a de-quantization (2024) to generate a coefficient block (2023). An inverse secondary transform (2022) can be performed over the 16 coefficients at the top-left 4x4 sub-block E to obtain 64 coefficients that are filled back to the 4x4 sub-blocks E/F/G/H. Thereafter, the coefficients in the block (2023) after the inverse secondary transform (2022) can be processed with an inverse primary transform (2020) to obtain a recovered residual block.

[0163] The process (2100) of the FIG. 21 example is similar to the process (2000) except that fewer 48 coefficients are processed during the forward secondary transform (2012). Specifically, the 48 coefficients in the sub-blocks A/B/C are processed with a smaller transform matrix of a size of 16x48. Usage of the smaller transform matrix can reduce a memory size for storing the transform matrix, and respective computation complexity.

[0164] FIG. 22 shows an example CU-level syntax table (2200) where a syntax element `lfst_idx` indicating a selection of a LFNST kernel is signaled at the end of CU-level syntax.

[0165] 4.3 Examples of RST Computation

[0166] The main idea of a Reduced Transform (RT) is to map an N dimensional vector to an R dimensional vector in a different space, where R/N ($R < N$) is the reduction factor.

[0167] The RST matrix is an R×N matrix as follows:

$$T_{R \times N} = \begin{bmatrix} t_{11} & t_{12} & t_{13} & \dots & t_{1N} \\ t_{21} & t_{22} & t_{23} & \dots & t_{2N} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ t_{R1} & t_{R2} & t_{R3} & \dots & t_{RN} \end{bmatrix} \quad (\text{Eq. 4})$$

where the R rows of the transform are R bases of the N dimensional space. The inverse transform matrix for RT is the transpose of its forward transform.

of size 64. Matrices and offset vectors of that set or parts of these matrices and offset vectors are used for all other block-shapes.

[0181] As shown, given an 8×8 block (2610), MIP takes four averages along each axis of the boundary. The resulting eight input samples enter the matrix vector multiplication. The matrices are taken from the set S_1 . This yields 16 samples (2621) on the odd positions of the prediction block. Thus, a total of $(8 \cdot 16)/(8 \cdot 8) = 2$ multiplications per sample are performed. After adding an offset, these samples are interpolated vertically by using the reduced top boundary samples. Horizontal interpolation follows by using the original left boundary samples. The interpolation process does not require any multiplications in this case.

[0182] 5.1 Signaling of MIP Mode

[0183] In some embodiments, for each Coding Unit (CU) in intra mode, a flag indicating if an MIP mode is applied on the corresponding Prediction Unit (PU) or not can be sent in the bitstream. If an MIP mode is applied, the index predmode of the MIP mode is signaled using an MPM-list including 3 MPMs.

[0184] The derivation of the MPMs can be performed using the intra-modes of the above and the left PU as follows. There are three fixed mapping tables $\text{map_angular_to_mip}_{\text{idx}}$, $\text{idx} \in \{0,1,2\}$, and each table associates each conventional intra prediction mode $\text{predmode}_{\text{Angular}}$ with a specific MIP mode, as described in the following formula.

$$\text{predmode}_{\text{MIP}} = \text{map_angular_to_mip}[\text{predmode}_{\text{Angular}}]. \quad (\text{Eq. 5})$$

where $\text{map_angular_to_mip}$ is a fixed look-up table. The index of the mapping table is decided based on the width W and height H of PU, and in total three indices are available, as described below,

$$\text{idx}(\text{PU}) = \text{idx}(W, H) \in \{0,1,2\} \quad (\text{Eq. 6})$$

that indicates from which of the three sets the MIP parameters are to be taken above.

[0185] In some examples, to generate the MPM list for current block which is coded by MIP mode, an above MIP mode, namely $\text{mode}_{\text{MIP}}^{\text{above}}$, and a left MIP mode, namely $\text{mode}_{\text{MIP}}^{\text{left}}$, are firstly derived.

[0186] The value of $\text{mode}_{\text{MIP}}^{\text{above}}$ can be derived as follows:

- If the above PU PU_{above} is available, and it belongs to the same CTU where the current PU resides, and PU_{above} is coded by MIP using an MIP mode

[0209] In some embodiments, whether an implicit transform can be applied to an intra coded block (that is not predicted by ISP mode) depends on whether a specific coding tool is enabled as indicated by a HLS element.

[0210] 2.1 When NSST Is Enabled

[0211] In an embodiment, an implicit transform can be disabled for an intra coded block (that is not predicted by ISP mode) when a HLS element indicates NSST is enabled. For example, as described in the section II.4.1, the transform matrix T is used in the equation (Eq. 3) for the secondary transform processing. The transform matrix T can include constant elements, and be designed with an assumption of a certain statistics of targeted coefficient blocks, such as coefficient blocks resulting from an explicit transform selection. Thus, a coefficient block resulting from an implicit transform selection may not match with the NSST in terms of statistics of respective coefficients. For example, applying an NSST to results of an implicit transform may not improve coding performance. For the above reason, disabling implicit transform when NSST is enabled may be desirable.

[0212] In this disclosure, the term NSST can be used to refer to a family of non-separable secondary transform coding schemes, such as the initial design of NSST, RST, LFNST, or the like.

[0213] For example, a decoder may receive two HLS elements: one HLS element (e.g., `sps_explicit_mts_intra_enabled_flag`) indicates explicit MTS is disabled, while the other HLS element (e.g., `sps_lfnst_enabled_flag`) indicates an NSST is enabled. Accordingly, the decoder can determine not to enable an implicit transform for intra coded blocks controlled by those two HLS elements. In contrast, if one HLS element indicates explicit MTS is disabled, and the other HLS element indicates an NSST is also disabled, the decoder can determine to enable an implicit transform for respective intra coded blocks.

[0214] As an example, FIG. 30 shows modifications (3000) to the text (2800) that correspond to a scenario where an implicit transform is enabled when an NSST is disabled. As shown, an additional restriction, “`sps_lfnst_enabled_flag` is equal to 0”, is added to the condition (iii) for deriving the variable `implicitNtsEnabled`.

[0215] 2.2 When MIP Is Enabled

[0216] Similar to the scenario where an NSST is enabled, in some embodiments, implicit transform is disabled for an intra coded block (that is not predicted by ISP mode) when a HLS element indicates an MIP is enabled. For example, applying an MIP intra coding mode to a

block may resulting in a residual block having different statistics from residual blocks coded with regular intra modes. Thus, a residual block coded with MIP may not match with implicit transform which may assume residual statistics resulting from regular intra modes. Therefore, disabling implicit transform may be desirable when an MIP is enabled.

[0217] For example, a decoder may receive two HLS elements: one HLS element (e.g., `sps_explicit_mts_intra_enabled_flag`) indicates explicit MTS is disabled, while the other HLS element (e.g., `sps_mip_enabled_flag`) indicates an MIP is enabled. Accordingly, the decoder can determine not to enable an implicit transform for intra coded blocks controlled by those two HLS elements. In contrast, if both the explicit MTS and the MIP are disabled as indicated by the two HLS elements, the decoder can determine to enable an implicit transform for respective intra coded blocks.

[0218] As an example, FIG. 31 shows modifications (3100) to the text (2800) that correspond to a scenario where an implicit transform is enabled when an MIP is disabled. As shown, an additional restriction, “`sps_mip_enabled_flag` is equal to 0”, is added to the condition (iii) for deriving the variable `implicitNtsEnabled`.

[0219] 3. Implicit Transform Enabling Based on Block Level Syntax Element indications

[0220] In some embodiments, whether an implicit transform can be applied to an intra coded block (that is not predicted by ISP mode) depends on whether MTS, NSST or MIP is applied to the intra coded block as indicated by block level syntax elements.

[0221] 3.1 When MTS And NSST Are Not Applied

[0222] In an embodiment, whether implicit transform can be applied to an intra coded block depends on whether both MTS and NSST are not applied to the intra coded block as indicated by block level syntax elements.

[0223] For example, a decoder can receive a first and second CU level (or block level) syntax elements associated with a current block that is intra coded and not predicted by ISP mode. The current block may be located at a position of coordinates [`x0`] [`y0`] within a picture.

[0224] The first CU level syntax element (e.g., `tu_mts_idx[ x0 ] [ y0 ]`) may indicate DCT-2 transforms can be used for the current block instead of transforms used in MTS (e.g., DST-7, DCT-8, DCT-4, or the like). As an example, in the table (2804), when `tu_mts_idx[ x0 ] [ y0 ]` has a value of 0, the variables of `trTypeHor` and `trTypeVer` both have a value of 0, which

MTS is not applied, but the second block level syntax element indicates the NSST is applied.

The process (3600) can proceed to (S3699), and terminates at (S3699).

[0249] IV. Computer System

[0250] The techniques described above, can be implemented as computer software using computer-readable instructions and physically stored in one or more computer-readable media. For example, FIG. 37 shows a computer system suitable for implementing certain embodiments of the disclosed subject matter.

[0251] The computer software can be coded using any suitable machine code or computer language, that may be subject to assembly, compilation, linking, or like mechanisms to create code comprising instructions that can be executed directly, or through interpretation, micro-code execution, and the like, by one or more computer central processing units (CPUs), Graphics Processing Units (GPUs), and the like.

[0252] The instructions can be executed on various types of computers or components thereof, including, for example, personal computers, tablet computers, servers, smartphones, gaming devices, internet of things devices, and the like.

[0253] The components shown in FIG. 37 for computer system are exemplary in nature and are not intended to suggest any limitation as to the scope of use or functionality of the computer software implementing embodiments of the present disclosure. Neither should the configuration of components be interpreted as having any dependency or requirement relating to any one or combination of components illustrated in the exemplary embodiment of a computer system.

[0254] Computer system may include certain human interface input devices. Such a human interface input device may be responsive to input by one or more human users through, for example, tactile input (such as: keystrokes, swipes, data glove movements), audio input (such as: voice, clapping), visual input (such as: gestures), olfactory input (not depicted). The human interface devices can also be used to capture certain media not necessarily directly related to conscious input by a human, such as audio (such as: speech, music, ambient sound), images (such as: scanned images, photographic images obtain from a still image camera), video (such as two-dimensional video, three-dimensional video including stereoscopic video).

[0255] Input human interface devices may include one or more of (only one of each depicted): keyboard (3701), mouse (3702), trackpad (3703), touch screen (3710), data-glove (not shown), joystick (3705), microphone (3706), scanner (3707), camera (3708).

[0256] Computer system may also include certain human interface output devices. Such human interface output devices may be stimulating the senses of one or more human users through, for example, tactile output, sound, light, and smell/taste. Such human interface output devices may include tactile output devices (for example tactile feedback by the touch-screen (3710), data-glove (not shown), or joystick (3705), but there can also be tactile feedback devices that do not serve as input devices), audio output devices (such as: speakers (3709), headphones (not depicted)), visual output devices (such as screens (3710) to include CRT screens, LCD screens, plasma screens, OLED screens, each with or without touch-screen input capability, each with or without tactile feedback capability—some of which may be capable to output two dimensional visual output or more than three dimensional output through means such as stereographic output; virtual-reality glasses (not depicted), holographic displays and smoke tanks (not depicted)), and printers (not depicted).

[0257] Computer system can also include human accessible storage devices and their associated media such as optical media including CD/DVD ROM/RW (3720) with CD/DVD or the like media (3721), thumb-drive (3722), removable hard drive or solid state drive (3723), legacy magnetic media such as tape and floppy disc (not depicted), specialized ROM/ASIC/PLD based devices such as security dongles (not depicted), and the like.

[0258] Those skilled in the art should also understand that term “computer readable media” as used in connection with the presently disclosed subject matter does not encompass transmission media, carrier waves, or other transitory signals.

[0259] Computer system can also include an interface (3754) to one or more communication networks (3755). Networks can for example be wireless, wireline, optical. Networks can further be local, wide-area, metropolitan, vehicular and industrial, real-time, delay-tolerant, and so on. Examples of networks include local area networks such as Ethernet, wireless LANs, cellular networks to include GSM, 3G, 4G, 5G, LTE and the like, TV wireline or wireless wide area digital networks to include cable TV, satellite TV, and terrestrial broadcast TV, vehicular and industrial to include CANBus, and so forth. Certain networks commonly require external network interface adapters that attached to certain general purpose data ports or peripheral buses (3749) (such as, for example USB ports of the computer system); others are commonly integrated into the core of the computer system by attachment to a system bus as described below (for example Ethernet interface into a PC computer system or cellular network interface into a smartphone computer system). Using any of these networks, computer system

can communicate with other entities. Such communication can be uni-directional, receive only (for example, broadcast TV), uni-directional send-only (for example CANbus to certain CANbus devices), or bi-directional, for example to other computer systems using local or wide area digital networks. Certain protocols and protocol stacks can be used on each of those networks and network interfaces as described above.

[0260] Aforementioned human interface devices, human-accessible storage devices, and network interfaces can be attached to a core (3740) of the computer system.

[0261] The core (3740) can include one or more Central Processing Units (CPU) (3741), Graphics Processing Units (GPU) (3742), specialized programmable processing units in the form of Field Programmable Gate Areas (FPGA) (3743), hardware accelerators for certain tasks (3744), and so forth. These devices, along with Read-only memory (ROM) (3745), Random-access memory (3746), internal mass storage such as internal non-user accessible hard drives, SSDs, and the like (3747), may be connected through a system bus (3748). In some computer systems, the system bus (3748) can be accessible in the form of one or more physical plugs to enable extensions by additional CPUs, GPU, and the like. The peripheral devices can be attached either directly to the core's system bus (3748), or through a peripheral bus (3749). Architectures for a peripheral bus include PCI, USB, and the like.

[0262] CPUs (3741), GPUs (3742), FPGAs (3743), and accelerators (3744) can execute certain instructions that, in combination, can make up the aforementioned computer code. That computer code can be stored in ROM (3745) or RAM (3746). Transitional data can be also be stored in RAM (3746), whereas permanent data can be stored for example, in the internal mass storage (3747). Fast storage and retrieve to any of the memory devices can be enabled through the use of cache memory, that can be closely associated with one or more CPU (3741), GPU (3742), mass storage (3747), ROM (3745), RAM (3746), and the like.

[0263] The computer readable media can have computer code thereon for performing various computer-implemented operations. The media and computer code can be those specially designed and constructed for the purposes of the present disclosure, or they can be of the kind well known and available to those having skill in the computer software arts.

[0264] As an example and not by way of limitation, the computer system having architecture, and specifically the core (3740) can provide functionality as a result of processor(s) (including CPUs, GPUs, FPGA, accelerators, and the like) executing software embodied in one or more tangible, computer-readable media. Such computer-readable media