(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
18 July 2013 (18.07.2013)

WIPO | PCT

(10) International Publication Number
WO 2013/106710 A2

(51) International Patent Classification:
H04N 7/26 (2006.01)

(21) International Application Number:
PCT/US2013/021234

(22) International Filing Date:
11 January 2013 (11.01.2013)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:

61/586,668	13 January 2012 (13.01.2012)	US
61/588,595	19 January 2012 (19.01.2012)	US
61/597,097	9 February 2012 (09.02.2012)	US
13/738,534	10 January 2013 (10.01.2013)	US

(71) Applicant: **QUALCOMM INCORPORATED** [US/US]; ATTN: International IP Administration, 5775 Morehouse Drive, San Diego, California 92121-1714 (US).

(72) Inventors: **SEREGIN, Vadim**; 5775 Morehouse Drive, San Diego, California 92121 (US). **SOLE ROJALS, Joel**; 5775 Morehouse Drive, San Diego, California 92121 (US). **KARCZEWICZ, Marta**; 5775 Morehouse Drive, San Diego, California 92121 (US).

(74) Agent: **DAWLEY, Brian R.**; Shumaker & Sieffert, P.A., 1625 Radio Drive, Suite 300, Woodbury, Minnesota 55125 (US).

(81) Designated States *(unless otherwise indicated, for every kind of national protection available)*: AE, AG, AL, AM, AO, AT, AU, AZ, BA, BB, BG, BH, BN, BR, BW, BY, BZ, CA, CH, CL, CN, CO, CR, CU, CZ, DE, DK, DM,

[Continued on next page]

(54) Title: DETERMINING CONTEXTS FOR CODING TRANSFORM COEFFICIENT DATA IN VIDEO CODING

(57) Abstract: In one example, a device for coding video data includes a video coder configured to determine a context for coding a transform coefficient of a video block based on a region of the video block in which the transform coefficient occurs, and entropy code the transform coefficient using the determined context. The region may comprise one of a first region comprising one or more upper-left 4x4 sub-blocks of transform coefficients of the video block and a second region comprising transform coefficients of the video block outside the first region.

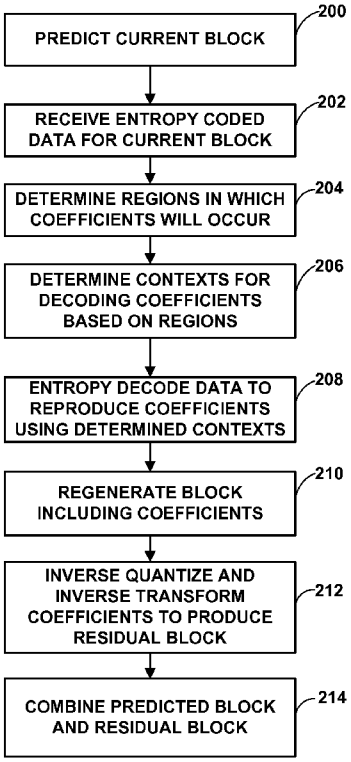


FIG. 16

WO 2013/106710 A2

DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IS, JP, KE, KG, KM, KN, KP, KR, KZ, LA, LC, LK, LR, LS, LT, LU, LY, MA, MD, ME, MG, MK, MN, MW, MX, MY, MZ, NA, NG, NI, NO, NZ, OM, PA, PE, PG, PH, PL, PT, QA, RO, RS, RU, RW, SC, SD, SE, SG, SK, SL, SM, ST, SV, SY, TH, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, ZA, ZM, ZW.

(84) **Designated States** *(unless otherwise indicated, for every kind of regional protection available)*: ARIPO (BW, GH, GM, KE, LR, LS, MW, MZ, NA, RW, SD, SL, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, RU, TJ, TM), European (AL, AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HR, HU, IE, IS, IT, LT, LU, LV, MC, MK, MT, NL, NO, PL, PT, RO, RS, SE, SI, SK, SM, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, ML, MR, NE, SN, TD, TG).

Published:

— *without international search report and to be republished upon receipt of that report (Rule 48.2(g))*

DETERMINING CONTEXTS FOR CODING TRANSFORM COEFFICIENT DATA IN VIDEO CODING

[0001] This application claims the benefit of U.S. Provisional Application Serial No. **61/586,668**, filed January 13, 2012, U.S. Provisional Application Serial No. **61/588,595**, filed January 19, 2012, and U.S. Provisional Application Serial No. **61/597,097**, filed February 9, 2012, each of which is hereby incorporated by reference in its entirety.

TECHNICAL FIELD

[0002] This disclosure relates to video coding.

BACKGROUND

[0003] Digital video capabilities can be incorporated into a wide range of devices, including digital televisions, digital direct broadcast systems, wireless broadcast systems, personal digital assistants (PDAs), laptop or desktop computers, tablet computers, e-book readers, digital cameras, digital recording devices, digital media players, video gaming devices, video game consoles, cellular or satellite radio telephones, so-called "smart phones," video teleconferencing devices, video streaming devices, and the like. Digital video devices implement video compression techniques, such as those described in the standards defined by MPEG-2, MPEG-4, ITU-T H.263, ITU-T H.264/MPEG-4, Part 10, Advanced Video Coding (AVC), the High Efficiency Video Coding (HEVC) standard presently under development, and extensions of such standards. The video devices may transmit, receive, encode, decode, and/or store digital video information more efficiently by implementing such video compression techniques.

[0004] Video compression techniques perform spatial (intra-picture) prediction and/or temporal (inter-picture) prediction to reduce or remove redundancy inherent in video sequences. For block-based video coding, a video slice (i.e., a video frame or a portion of a video frame) may be partitioned into video blocks, which may also be referred to as treeblocks, coding units (CUs) and/or coding nodes. Video blocks in an intra-coded (I) slice of a picture are encoded using spatial prediction with respect to reference samples in neighboring blocks in the same picture. Video blocks in an inter-coded (P or B) slice of a picture may use spatial prediction with respect to reference samples in neighboring blocks in the same picture or temporal prediction with respect to reference samples in

other reference pictures. Pictures may be referred to as frames, and reference pictures may be referred to a reference frames.

[0005] Spatial or temporal prediction results in a predictive block for a block to be coded. Residual data represents pixel differences between the original block to be coded and the predictive block. An inter-coded block is encoded according to a motion vector that points to a block of reference samples forming the predictive block, and the residual data indicating the difference between the coded block and the predictive block. An intra-coded block is encoded according to an intra-coding mode and the residual data. For further compression, the residual data may be transformed from the pixel domain to a transform domain, resulting in residual transform coefficients, which then may be quantized. The quantized transform coefficients, initially arranged in a two-dimensional array, may be scanned in order to produce a one-dimensional vector of transform coefficients, and entropy coding may be applied to achieve even more compression.

SUMMARY

[0006] In general, this disclosure describes techniques related to determining contexts for entropy coding, e.g., using context-adaptive binary arithmetic coding (CABAC), of video data. CABAC coding generally involves determining a context when coding binarized representations of various syntax elements. Examples of syntax elements include data for transform coefficients, such as data indicating whether the transform coefficients are significant, signs of the transform coefficients that are significant, and level values for the transform coefficients that are significant. Transform coefficients generally correspond to coefficients of a transform block, such as a transform unit (TU). This disclosure describes techniques for determining contexts for coding transform coefficients based on regions of a transform block in which the transform coefficients occur.

[0007] In one example, a method of coding video data includes determining a context for coding a transform coefficient of a video block based on a region of the video block in which the transform coefficient occurs, and entropy coding the transform coefficient using the determined context.

[0008] In another example, a device for coding video data includes a video coder configured to determine a context for coding a transform coefficient of a video block

based on a region of the video block in which the transform coefficient occurs, and entropy code the transform coefficient using the determined context.

[0009] In another example, a device for coding video data includes means for determining a context for coding a transform coefficient of a video block based on a region of the video block in which the transform coefficient occurs, and means for entropy coding the transform coefficient using the determined context.

[0010] In another example, a computer-readable storage medium has stored thereon instructions that, when executed, cause a processor to determine a context for coding a transform coefficient of a video block based on a region of the video block in which the transform coefficient occurs, and entropy code the transform coefficient using the determined context.

[0011] In another example, a method of decoding video data includes determining whether a transform coefficient of a video block is a DC transform coefficient, when the transform coefficient is determined to be the DC transform coefficient of the video block, determining a context for decoding the transform coefficient based on the transform coefficient being the DC transform coefficient without regard for a size of the video block, and entropy decoding the transform coefficient using the determined context.

[0012] In another example, a device for decoding video data includes a video decoder configured to determine whether a transform coefficient of a video block is a DC transform coefficient, when the transform coefficient is determined to be the DC transform coefficient of the video block, determine a context for decoding the transform coefficient based on the transform coefficient being the DC transform coefficient without regard for a size of the video block, and entropy decode the transform coefficient using the determined context.

[0013] In another example, a device for decoding video data includes means for determining whether a transform coefficient of a video block is a DC transform coefficient, means for determining, when the transform coefficient is determined to be the DC transform coefficient of the video block, a context for decoding the transform coefficient based on the transform coefficient being the DC transform coefficient without regard for a size of the video block, and means for entropy decoding the transform coefficient using the determined context.

[0014] In another example, a computer-readable storage medium has stored thereon instructions that, when executed, cause a processor to determine whether a transform coefficient of a video block is a DC transform coefficient, when the transform coefficient is determined to be the DC transform coefficient of the video block, determine a context for decoding the transform coefficient based on the transform coefficient being the DC transform coefficient without regard for a size of the video block, and entropy decode the transform coefficient using the determined context.

[0015] In another example, a method of encoding video data includes determining whether a transform coefficient of a video block is a DC transform coefficient, when the transform coefficient is determined to be the DC transform coefficient of the video block, determining a context for encoding the transform coefficient based on the transform coefficient being the DC transform coefficient without regard for a size of the video block, and entropy encoding the transform coefficient using the determined context.

[0016] In another example, a device for encoding video data includes a video encoder configured to determine whether a transform coefficient of a video block is a DC transform coefficient, when the transform coefficient is determined to be the DC transform coefficient of the video block, determine a context for encoding the transform coefficient based on the transform coefficient being the DC transform coefficient without regard for a size of the video block, and entropy encode the transform coefficient using the determined context.

[0017] In another example, a device for encoding video data includes means for determining whether a transform coefficient of a video block is a DC transform coefficient, means for determining, when the transform coefficient is determined to be the DC transform coefficient of the video block, a context for encoding the transform coefficient based on the transform coefficient being the DC transform coefficient without regard for a size of the video block, and means for entropy encoding the transform coefficient using the determined context.

[0018] In another example, a computer-readable storage medium has stored thereon instructions that, when executed, cause a processor to determine whether a transform coefficient of a video block is a DC transform coefficient, when the transform coefficient is determined to be the DC transform coefficient of the video block, determine a context for encoding the transform coefficient based on the transform coefficient being the DC transform coefficient without regard for a size of the video block, and entropy encode the transform coefficient using the determined context.

[0019] In another example, a method of decoding video data includes determining values for coded sub-block flags of one or more neighboring sub-blocks to a current sub-block, determining a context for decoding a transform coefficient of the current sub-block based on the values for the coded sub-block flags, and entropy decoding the transform coefficient using the determined context.

[0020] In another example, a device for decoding video data includes a video decoder configured to determine values for coded sub-block flags of one or more neighboring sub-blocks to a current sub-block, determine a context for decoding a transform coefficient of the current sub-block based on the values for the coded sub-block flags, and entropy decode the transform coefficient using the determined context.

[0021] In another example, a device for decoding video data includes means for determining values for coded sub-block flags of one or more neighboring sub-blocks to a current sub-block, means for determining a context for decoding a transform coefficient of the current sub-block based on the values for the coded sub-block flags, and means for entropy decoding the transform coefficient using the determined context.

[0022] In another example, a computer-readable storage medium has stored thereon instructions that, when executed, cause a processor to determine values for coded sub-block flags of one or more neighboring sub-blocks to a current sub-block, determine a context for decoding a transform coefficient of the current sub-block based on the values for the coded sub-block flags, and entropy decode the transform coefficient using the determined context.

[0023] In another example, a method of encoding video data includes determining values for coded sub-block flags of one or more neighboring sub-blocks to a current sub-block, determining a context for encoding a transform coefficient of the current sub-block based on the values for the coded sub-block flags, and entropy encoding the transform coefficient using the determined context.

[0024] In another example, a device for encoding video data includes a video encoder configured to determine values for coded sub-block flags of one or more neighboring sub-blocks to a current sub-block, determine a context for encoding a transform coefficient of the current sub-block based on the values for the coded sub-block flags, and entropy encode the transform coefficient using the determined context.

[0025] In another example, a device for encoding video data includes means for determining values for coded sub-block flags of one or more neighboring sub-blocks to a current sub-block, means for determining a context for encoding a transform coefficient of the current sub-block based on the values for the coded sub-block flags, and means for entropy encoding the transform coefficient using the determined context.

[0026] In another example, a computer-readable storage medium has stored thereon instructions that, when executed, cause a processor to determine values for coded sub-block flags of one or more neighboring sub-blocks to a current sub-block, determine a context for encoding a transform coefficient of the current sub-block based on the values for the coded sub-block flags, and entropy encode the transform coefficient using the determined context.

[0027] The details of one or more examples are set forth in the accompanying drawings and the description below. Other features, objects, and advantages will be apparent from the description and drawings, and from the claims.

BRIEF DESCRIPTION OF DRAWINGS

[0028] FIG. 1 is a block diagram illustrating an example video encoding and decoding system that may utilize the inter-prediction techniques described in this disclosure.

[0029] FIG. 2 is a block diagram illustrating an example video encoder that may implement the inter-prediction techniques described in this disclosure.

[0030] FIG. 3 is a block diagram illustrating an example video decoder that may implement the inter-prediction techniques described in this disclosure.

[0031] FIG. 4 is a conceptual diagram that illustrates a relation between transform coefficients in a video block and a significance map associated with the video block.

[0032] FIGS. 5A–5D are conceptual diagrams that illustrate examples of blocks of video data scanned using a zig-zag scanning order, a horizontal scanning order, a vertical scanning order, and a diagonal scanning order.

[0033] FIG. 6 is a conceptual diagram that illustrates an example video block divided into sub-blocks for transform coefficient coding.

[0034] FIG. 7 is a conceptual diagram that illustrates an example five-point support used to define a context model for a significance map of coefficients in a video block scanned using a reverse diagonal scanning order.

[0035] FIGS. 8A and 8B are conceptual diagrams that illustrate context dependency within the five-point support.

[0036] FIGS. 9A and 9B are conceptual diagrams that illustrate example divisions of a video block into two or more regions.

[0037] FIG. 10 is a conceptual diagram that illustrates example assignment of neighborhood- or position-based contexts for each region of a video block.

[0038] FIG. 11 is a conceptual diagram that illustrates example assignment of context offsets for each region of a video block.

[0039] FIG. 12 is a conceptual diagram that illustrates an example embedded division of a video block into two or more regions based on TU sizes that correlate to existing context models.

[0040] FIGS. 13A and 13B are conceptual diagrams that illustrate example divisions of a video block into two or more regions.

[0041] FIGS. 14A and 14B are conceptual diagrams that illustrate example assignment of context offsets for each region of a video block.

[0042] FIG. 15 is a flowchart illustrating an example method for encoding a current block.

[0043] FIG. 16 is a flowchart illustrating an example method for decoding a current block of video data.

DETAILED DESCRIPTION

[0044] In general, this disclosure describes techniques related to determining contexts for entropy coding, e.g., using context-adaptive binary arithmetic coding (CABAC), of video data. CABAC coding generally involves determining a context when coding binarized representations of various syntax elements. Syntax elements include, for example, data for transform coefficients, such as data indicating whether the transform coefficients are significant, signs of the transform coefficients that are significant, and level values for the transform coefficients that are significant. Transform coefficients generally correspond to coefficients of a transform block, such as a transform unit (TU).

This disclosure describes techniques for determining contexts for coding transform coefficients based on regions of a transform block in which the transform coefficients occur.

[0045] In general, in accordance with the techniques of this disclosure, a video coder may be configured to determine context for coding a transform coefficient based on a region in which the transform coefficient occurs and then entropy code the transform coefficient using the determined context. A video block may be divided into regions in a variety of ways. FIGS. 9A and 11 illustrate examples in which a video block is divided into a first region including one or more upper-left sub-blocks (e.g., 4x4 sub-blocks) and a second region including sub-blocks outside the first region. FIG. 9B illustrates an example in which a video block is divided into regions along a diagonal direction. FIG. 10 illustrates an example in which a video block is divided into quartiles, and the upper-left quartile is further divided into a first sub-region including sub-blocks of an upper-left portion of the upper-left quartile and a second sub-region including sub-blocks of the upper-left quartile external to the first sub-region. FIG. 12 illustrates an example in which a video block is divided into regions that correspond to video block sizes (e.g., 4x4, 8x8, 16x16, and 32x32). FIG. 13A illustrates an example in which a video block is divided into horizontal rectangular regions. FIG. 13B illustrates an example in which a video block is divided into vertical rectangular regions. These figures are described in greater detail below.

[0046] In various examples, a video coder may be configured to determine a context for coding a transform coefficient in various ways, e.g., based on a region in which the transform coefficient occurs. For example, a video coder may be configured to determine a context using position-based context information for some regions or neighborhood-based context information for other regions. In some examples, all transform coefficients within a particular region may be coded using the same context, determined based on the region. In other examples, contexts for transform coefficients within a region may be determined based on a context neighborhood. In still other examples, a video coder may determine an offset to be applied to a context based on the region in which a transform coefficient occurs. That is, each of the regions may be associated with a particular context offset to be applied to a context.

[0047] The techniques of this disclosure may reduce bandwidth consumption, leading to savings of bits when coding syntax elements for transform coefficients. Such syntax elements may include any or all of a significant coefficient flag (which indicates

whether a corresponding transform coefficient is significant, that is, non-zero), a sign of significant coefficients, an indication of whether a significant coefficient has an absolute value greater than 1, an indication of whether a significant coefficient with an absolute value greater than 1 has an absolute value greater than 2, and/or a remaining level value for coefficients having absolute values greater than 2.

[0048] FIG. 1 is a block diagram illustrating an example video encoding and decoding system 10 that may utilize the techniques described in this disclosure. As shown in FIG. 1, system 10 includes a source device 12 that generates encoded video data to be decoded at a later time by a destination device 14. Source device 12 and destination device 14 may comprise any of a wide range of devices, including desktop computers, notebook (i.e., laptop) computers, tablet computers, set-top boxes, telephone handsets such as so-called "smart" phones, so-called "smart" pads, televisions, cameras, display devices, digital media players, video gaming consoles, video streaming device, or the like. In some cases, source device 12 and destination device 14 may be equipped for wireless communication.

[0049] Destination device 14 may receive the encoded video data to be decoded via a link 16. Link 16 may comprise any type of medium or device capable of moving the encoded video data from source device 12 to destination device 14. In one example, link 16 may comprise a communication medium to enable source device 12 to transmit encoded video data directly to destination device 14 in real-time. The encoded video data may be modulated according to a communication standard, such as a wireless communication protocol, and transmitted to destination device 14. The communication medium may comprise any wireless or wired communication medium, such as a radio frequency (RF) spectrum or one or more physical transmission lines. The communication medium may form part of a packet-based network, such as a local area network, a wide-area network, or a global network such as the Internet. The communication medium may include routers, switches, base stations, or any other equipment that may be useful to facilitate communication from source device 12 to destination device 14.

[0050] Alternatively, encoded data may be output from output interface 22 to a storage device 34. Similarly, encoded data may be accessed from storage device 34 by input interface. Storage device 34 may include any of a variety of distributed or locally accessed data storage media such as a hard drive, Blu-ray discs, DVDs, CD-ROMs, flash memory, volatile or non-volatile memory, or any other suitable digital storage

media for storing encoded video data. In a further example, storage device 34 may correspond to a file server or another intermediate storage device that may hold the encoded video generated by source device 12. Destination device 14 may access stored video data from storage device 34 via streaming or download. The file server may be any type of server capable of storing encoded video data and transmitting that encoded video data to the destination device 14. Example file servers include a web server (e.g., for a website), an FTP server, network attached storage (NAS) devices, or a local disk drive. Destination device 14 may access the encoded video data through any standard data connection, including an Internet connection. This may include a wireless channel (e.g., a Wi-Fi connection), a wired connection (e.g., DSL, cable modem, etc.), or a combination of both that is suitable for accessing encoded video data stored on a file server. The transmission of encoded video data from storage device 34 may be a streaming transmission, a download transmission, or a combination of both.

[0051] The techniques of this disclosure are not necessarily limited to wireless applications or settings. The techniques may be applied to video coding in support of any of a variety of multimedia applications, such as over-the-air television broadcasts, cable television transmissions, satellite television transmissions, streaming video transmissions, e.g., via the Internet, encoding of digital video for storage on a data storage medium, decoding of digital video stored on a data storage medium, or other applications. In some examples, system 10 may be configured to support one-way or two-way video transmission to support applications such as video streaming, video playback, video broadcasting, and/or video telephony.

[0052] In the example of FIG. 1, source device 12 includes a video source 18, video encoder 20 and an output interface 22. In some cases, output interface 22 may include a modulator/demodulator (modem) and/or a transmitter. In source device 12, video source 18 may include a source such as a video capture device, e.g., a video camera, a video archive containing previously captured video, a video feed interface to receive video from a video content provider, and/or a computer graphics system for generating computer graphics data as the source video, or a combination of such sources. As one example, if video source 18 is a video camera, source device 12 and destination device 14 may form so-called camera phones or video phones. However, the techniques described in this disclosure may be applicable to video coding in general, and may be applied to wireless and/or wired applications.

[0053] The captured, pre-captured, or computer-generated video may be encoded by video encoder 20. The encoded video data may be transmitted directly to destination device 14 via output interface 22 of source device 12. The encoded video data may also (or alternatively) be stored onto storage device 34 for later access by destination device 14 or other devices, for decoding and/or playback.

[0054] Destination device 14 includes an input interface 28, a video decoder 30, and a display device 32. In some cases, input interface 28 may include a receiver and/or a modem. Input interface 28 of destination device 14 receives the encoded video data over link 16. The encoded video data communicated over link 16, or provided on storage device 34, may include a variety of syntax elements generated by video encoder 20 for use by a video decoder, such as video decoder 30, in decoding the video data. Such syntax elements may be included with the encoded video data transmitted on a communication medium, stored on a storage medium, or stored a file server.

[0055] Display device 32 may be integrated with, or external to, destination device 14. In some examples, destination device 14 may include an integrated display device and also be configured to interface with an external display device. In other examples, destination device 14 may be a display device. In general, display device 32 displays the decoded video data to a user, and may comprise any of a variety of display devices such as a liquid crystal display (LCD), a plasma display, an organic light emitting diode (OLED) display, or another type of display device.

[0056] Video encoder 20 and video decoder 30 may operate according to a video compression standard, such as the High Efficiency Video Coding (HEVC) standard presently under development, and may conform to the HEVC Test Model (HM). Alternatively, video encoder 20 and video decoder 30 may operate according to other proprietary or industry standards, such as the ITU-T H.264 standard, alternatively referred to as MPEG-4, Part 10, Advanced Video Coding (AVC), or extensions of such standards. Extensions of standards include, for example, scalable video coding (SVC), multiview video coding (MVC), three-dimensional (3D) such as coding depth information, and the like. The techniques of this disclosure, however, are not limited to any particular coding standard or standard extension. Other examples of video compression standards include MPEG-2 and ITU-T H.263.

[0057] Although not shown in FIG. 1, in some aspects, video encoder 20 and video decoder 30 may each be integrated with an audio encoder and decoder, and may include appropriate MUX-DEMUX units, or other hardware and software, to handle encoding

of both audio and video in a common data stream or separate data streams. If applicable, in some examples, MUX-DEMUX units may conform to the ITU H.223 multiplexer protocol, or other protocols such as the user datagram protocol (UDP).

[0058] Video encoder 20 and video decoder 30 each may be implemented as any of a variety of suitable encoder circuitry, such as one or more microprocessors, digital signal processors (DSPs), application specific integrated circuits (ASICs), field programmable gate arrays (FPGAs), discrete logic, software, hardware, firmware or any combinations thereof. When the techniques are implemented partially in software, a device may store instructions for the software in a suitable, non-transitory computer-readable medium and execute the instructions in hardware using one or more processors to perform the techniques of this disclosure. Each of video encoder 20 and video decoder 30 may be included in one or more encoders or decoders, either of which may be integrated as part of a combined encoder/decoder (CODEC) in a respective device.

[0059] The JCT-VC is working on development of the HEVC standard. The HEVC standardization efforts are based on an evolving model of a video coding device referred to as the HEVC Test Model (HM). The HM presumes several additional capabilities of video coding devices relative to existing devices according to, e.g., ITU-T H.264/AVC. For example, whereas H.264 provides nine intra-prediction encoding modes, the HM may provide as many as thirty-three intra-prediction encoding modes.

[0060] In general, the working model of the HM describes that a video frame or picture may be divided into a sequence of treeblocks or largest coding units (LCU) that include both luma and chroma samples. A treeblock has a similar purpose as a macroblock of the H.264 standard. A slice includes a number of consecutive treeblocks in coding order. A video frame or picture may be partitioned into one or more slices. Each treeblock may be split into coding units (CUs) according to a quadtree. For example, a treeblock, as a root node of the quadtree, may be split into four child nodes, and each child node may in turn be a parent node and be split into another four child nodes. A final, unsplit child node, as a leaf node of the quadtree, comprises a coding node, i.e., a coded video block. Syntax data associated with a coded bitstream may define a maximum number of times a treeblock may be split, and may also define a minimum size of the coding nodes.

[0061] A CU includes a coding node and prediction units (PUs) and transform units (TUs) associated with the coding node. A size of the CU corresponds to a size of the coding node and must be square in shape. The size of the CU may range from 8x8

pixels up to the size of the treeblock with a maximum of 64x64 pixels or greater. Each CU may contain one or more PUs and one or more TUs. Syntax data associated with a CU may describe, for example, partitioning of the CU into one or more PUs. Partitioning modes may differ between whether the CU is skip or direct mode encoded, intra-prediction mode encoded, or inter-prediction mode encoded. PUs may be partitioned to be non-square in shape. Syntax data associated with a CU may also describe, for example, partitioning of the CU into one or more TUs according to a quadtree. A TU can be square or non-square in shape.

[0062] The HEVC standard allows for transformations according to TUs, which may be different for different CUs. The TUs are typically sized based on the size of PUs within a given CU defined for a partitioned LCU, although this may not always be the case. The TUs are typically the same size or smaller than the PUs. In some examples, residual samples corresponding to a CU may be subdivided into smaller units using a quadtree structure known as "residual quad tree" (RQT). The leaf nodes of the RQT may be referred to as transform units (TUs). Pixel difference values associated with the TUs may be transformed to produce transform coefficients, which may be quantized.

[0063] In general, a PU includes data related to the prediction process. For example, when the PU is intra-mode encoded, the PU may include data describing an intra-prediction mode for the PU. As another example, when the PU is inter-mode encoded, the PU may include data defining a motion vector for the PU. The data defining the motion vector for a PU may describe, for example, a horizontal component of the motion vector, a vertical component of the motion vector, a resolution for the motion vector (e.g., one-quarter pixel precision or one-eighth pixel precision), a reference picture to which the motion vector points, and/or a reference picture list for the motion vector.

[0064] In general, a TU is used for the transform and quantization processes. A given CU having one or more PUs may also include one or more TUs. Following prediction, video encoder 20 may calculate residual values corresponding to the PU. The residual values comprise pixel difference values that may be transformed into transform coefficients, quantized, and scanned using the TUs to produce serialized transform coefficients for entropy coding. This disclosure typically uses the term “video block” to refer to a coding node of a CU. In some specific cases, this disclosure may also use the term “video block” to refer to a treeblock, i.e., LCU, or a CU, which includes a coding node and PUs and TUs.

[0065] A video sequence typically includes a series of video frames or pictures. A group of pictures (GOP) generally comprises a series of one or more of the video pictures. A GOP may include syntax data in a header of the GOP, a header of one or more of the pictures, or elsewhere, that describes a number of pictures included in the GOP. Each slice of a picture may include slice syntax data that describes an encoding mode for the respective slice. Video encoder 20 typically operates on video blocks within individual video slices in order to encode the video data. A video block may correspond to a coding node within a CU. The video blocks may have fixed or varying sizes, and may differ in size according to a specified coding standard.

[0066] As an example, the HM supports prediction in various PU sizes. Assuming that the size of a particular CU is 2Nx2N, the HM supports intra-prediction in PU sizes of 2Nx2N or NxN, and inter-prediction in symmetric PU sizes of 2Nx2N, 2NxN, Nx2N, or NxN. The HM also supports asymmetric partitioning for inter-prediction in PU sizes of 2NxnU, 2NxnD, nLx2N, and nRx2N. In asymmetric partitioning, one direction of a CU is not partitioned, while the other direction is partitioned into 25% and 75%. The portion of the CU corresponding to the 25% partition is indicated by an "n" followed by an indication of "Up", "Down," "Left," or "Right." Thus, for example, "2NxnU" refers to a 2Nx2N CU that is partitioned horizontally with a 2Nx0.5N PU on top and a 2Nx1.5N PU on bottom.

[0067] In this disclosure, "NxN" and "N by N" may be used interchangeably to refer to the pixel dimensions of a video block in terms of vertical and horizontal dimensions, e.g., 16x16 pixels or 16 by 16 pixels. In general, a 16x16 block will have 16 pixels in a vertical direction (y = 16) and 16 pixels in a horizontal direction (x = 16). Likewise, an NxN block generally has N pixels in a vertical direction and N pixels in a horizontal direction, where N represents a nonnegative integer value. The pixels in a block may be arranged in rows and columns. Moreover, blocks need not necessarily have the same number of pixels in the horizontal direction as in the vertical direction. For example, blocks may comprise NxM pixels, where M is not necessarily equal to N.

[0068] Following intra-predictive or inter-predictive coding using the PUs of a CU, video encoder 20 may calculate residual data for the TUs of the CU. The PUs may comprise pixel data in the spatial domain (also referred to as the pixel domain) and the TUs may comprise coefficients in the transform domain following application of a transform, e.g., a discrete cosine transform (DCT), an integer transform, a wavelet transform, or a conceptually similar transform to residual video data. The residual data

may correspond to pixel differences between pixels of the unencoded picture and prediction values corresponding to the PUs. Video encoder 20 may form the TUs including the residual data for the CU, and then transform the TUs to produce transform coefficients for the CU.

[0069] Following any transforms to produce transform coefficients, video encoder 20 may perform quantization of the transform coefficients. Quantization generally refers to a process in which transform coefficients are quantized to possibly reduce the amount of data used to represent the coefficients, providing further compression. The quantization process may reduce the bit depth associated with some or all of the coefficients. For example, an *n*-bit value may be rounded down to an *m*-bit value during quantization, where *n* is greater than *m*.

[0070] In some examples, video encoder 20 and video decoder 30 may utilize a predefined scan order to scan the quantized transform coefficients to produce a serialized vector that can be entropy encoded. In other examples, video encoder 20 and video decoder 30 may perform an adaptive scan. After scanning the quantized transform coefficients to form a one-dimensional vector, or during the scan, video encoder 20 may entropy encode the one-dimensional vector, e.g., according to context adaptive variable length coding (CAVLC), context adaptive binary arithmetic coding (CABAC), syntax-based context-adaptive binary arithmetic coding (SBAC), Probability Interval Partitioning Entropy (PIPE) coding or another entropy encoding methodology. Video decoder 30 may entropy decode the coefficients, perform an inverse quantization process and an inverse transform process to reproduce residual data, and combine the residual data with predictive data to produce decoded video data. Video encoder 20 may also entropy encode syntax elements associated with the encoded video data for use by video decoder 30 in decoding the video data.

[0071] To perform CABAC, video encoder 20 and video decoder 30 may assign a context within a context model to a symbol to be coded. The context may relate to, for example, whether neighboring values of the symbol are non-zero or not. In accordance with the techniques of this disclosure, video encoder 20 and/or video decoder 30 may be configured to determine context for entropy coding (e.g., entropy encoding or entropy decoding) a transform coefficient based on a region of a video block in which the transform coefficient occurs.

[0072] Video encoder 20 and video decoder 30 may be configured with definitions of various regions for video blocks (e.g., transform units). For example, video encoder 20

and video decoder 30 may be configured with definitions of regions for various sizes of video blocks. In some examples, video encoder 20 may determine a method by which to divide a video block into regions and code data representative of how the block is to be divided. Each of the regions may be associated with a respective value and/or technique for determining context for transform coefficients occurring within the respective region.

[0073] For example, a particular region of a video block may be associated with a neighborhood-based context determination scheme, while another region of the video block may be associated with a position-based context determination scheme. As another example, a region of a video block may be associated with an offset to be applied to a context determined for transform coefficients located in that region. Different regions of the same video block may be associated with different offset values and/or different techniques for calculating context.

[0074] As one example, a video block may include two different regions: a first region including one or more sub-blocks (e.g., 4x4 transform coefficient sub-blocks) in an upper-left corner of the video block, and a second region including other sub-blocks of the video block that are not included in the first region. More specifically, video encoder 20 and video decoder 30 may determine an x- and y-coordinate of a sub-block and determine whether the sub-block is in the first region or the second region by comparing the sum of x and y to a threshold value. If the sum of x and y is less than the threshold, video encoder 20 and video decoder 30 may determine that the sub-block is in the first region, and otherwise, video encoder 20 and video decoder 30 may determine that the sub-block is in the second region. Video encoder 20 and video decoder 30 may determine context for coefficients of a video block based on whether the coefficients are in a sub-block of the first region or a sub-block of the second region.

[0075] For example, in some regions, the context may be a fixed context, in which video encoder 20 and video decoder 30 codes transform coefficients in such regions using the fixed context. That is, video encoder 20 and video decoder 30 may apply the same context to all transform coefficients in the region. Alternatively, each of the sub-blocks in the region may be associated with the same method of determining context (e.g., the fixed context method), but different sub-blocks in the region may have different fixed contexts. Video encoder 20 and video decoder 30 may determine a fixed context for a sub-block based on the position of the sub-block in the region. As still another example, fixed contexts may be assigned to individual transform coefficient

positions within the region. That is, video encoder 20 and video decoder 30 may determine context for coding a transform coefficient within the region based on a position of the transform coefficient in the video block, the sub-block, and/or the region.

[0076] As another example, in some regions, a context model may be defined according to neighboring sub-blocks. For example, video encoder 20 and video decoder 30 may be configured with sets of contexts for each sub-block within a particular region. That is, each sub-block in the region may be associated with a respective set of contexts. Video encoder 20 and video decoder 30 may select an appropriate context from the set of contexts for each transform coefficient in the respective sub-block. The set of contexts for one sub-block may be different from the set of contexts for another sub-block.

[0077] As yet another example, individual flags for each sub-block in a region may be coded representing whether there are any significant (i.e., non-zero) coefficients in the corresponding sub-block. These flags may be referred to as coded sub-block flags. Such flags may be used for selecting context for coding transform coefficients in the sub-blocks. For example, video encoder 20 and video decoder 30 may determine context for coding transform coefficients in a sub-block based on the values of the flags of one or more neighboring sub-blocks. For example, the flags may have binary values of either 0 or 1, and video encoder 20 and video decoder 30 may determine the context for coding transform coefficients in a current sub-block based on the sum of the flag values for a right-neighboring sub-block and a below-neighboring sub-block (also referred to as a bottom-neighboring sub-block). Other formulas may also be used for calculating the context for a sub-block.

[0078] Video encoder 20 and video decoder 30 may be configured to implement any or all of the techniques of this disclosure, alone or in any combination. One example combination of these techniques is that video encoder 20 and video decoder 30 may be configured to divide a transform unit into sub-blocks (e.g., 4x4 pixel sub-blocks), and then determine context for coding data of a particular transform coefficient of a sub-block based on both a position of the transform coefficient in the sub-block and based on coded block flags for one or more neighboring sub-blocks, e.g., a left-neighboring sub-block and a bottom-neighboring sub-block.

[0079] Video encoder 20 and video decoder 30 may be configured to code one or more syntax elements representative of transform coefficients using contexts determined in these various examples. Transform coefficients may include various types of syntax

elements. For example, a transform coefficient may include a significant coefficient flag (significant_coeff_flag) indicative of whether the transform coefficient has a non-zero value (i.e., is significant). If the transform coefficient is significant, the transform coefficient may include a sign value (e.g., coeff_sign_flag) indicating whether the value of the transform coefficient is greater than or less than 0 and a value indicative of whether the absolute value of the transform coefficient is greater than 1 (e.g., coeff_abs_level_greater1_flag). If the transform coefficient has an absolute value greater than 1, the transform coefficient may include a value indicative of whether the transform coefficient has an absolute value greater than 2 (e.g., coeff_abs_level_greater2_flag). If the transform coefficient has an absolute value greater than 2, the transform coefficient may include a value indicative of the absolute value of the transform coefficient minus two (e.g., coeff_abs_level_remaining).

[0080] A CABAC coder of video encoder 20 and video decoder 30 may code any or all of these values using contexts determined in accordance with the techniques of this disclosure. In addition, or in the alternative, video encoder 20 and video decoder 30 may code data indicative of a position of a last significant coefficient (e.g., last_significant_coeff_x_prefix, last_significant_coeff_x_suffix, last_significant_coeff_y_prefix, and last_significant_coeff_y_suffix) using context determined in accordance with the techniques of this disclosure.

[0081] Video encoder 20 and video decoder 30 may be configured to perform any one or more of the techniques described in this disclosure, alone or in any combination. Various techniques for determining a context for coding a transform coefficient of a video block based on a region of the video block in which the transform coefficient occurs and entropy coding the transform coefficient using the determined context are described below. Examples of such techniques are described with respect to FIGS. 9–14 below. In general, coding the transform coefficient using the determined context includes coding one or more syntax elements of the transform coefficient using the determined context. Determining the context generally includes determining a region in which the transform coefficient occurs and determining the context based on the region. For example, the region may be associated with a particular context or set of contexts, and/or associated with one or more techniques for determining the context.

[0082] FIG. 2 is a block diagram illustrating an example video encoder 20 that may implement the inter-prediction techniques described in this disclosure. Video encoder 20 may perform intra- and inter-coding of video blocks within video slices.

Intra-coding relies on spatial prediction to reduce or remove spatial redundancy in video within a given video frame or picture. Inter-coding relies on temporal prediction to reduce or remove temporal redundancy in video within adjacent frames or pictures of a video sequence. Intra-mode (I mode) may refer to any of several spatial based compression modes. Inter-modes, such as uni-directional prediction (P mode) or bi-prediction (B mode), may refer to any of several temporal-based compression modes.

[0083] In the example of FIG. 2, video encoder 20 includes a mode select unit 35, prediction processor 41, reference picture memory 64, summer 50, transform processing unit 52, quantization unit 54, and entropy encoding unit 56. Prediction processor 41 includes motion estimation unit 42, motion compensation unit 44, and intra prediction unit 46. For video block reconstruction, video encoder 20 also includes inverse quantization unit 58, inverse transform unit 60, and summer 62. A deblocking filter (not shown in FIG. 2) may also be included to filter block boundaries to remove blockiness artifacts from reconstructed video. If desired, the deblocking filter would typically filter the output of summer 62. Additional loop filters (in loop or post loop) may also be used in addition to the deblocking filter.

[0084] As shown in FIG. 2, video encoder 20 receives video data, and mode select unit 35 partitions the data into video blocks. This partitioning may also include partitioning into slices, tiles, or other larger units, as wells as video block partitioning, e.g., according to a quadtree structure of LCUs and CUs. Video encoder 20 generally illustrates the components that encode video blocks within a video slice to be encoded. The slice may be divided into multiple video blocks (and possibly into sets of video blocks referred to as tiles). Prediction processor 41 may select one of a plurality of possible coding modes, such as one of a plurality of intra coding modes or one of a plurality of inter coding modes, for the current video block based on error results (e.g., coding rate and the level of distortion). Prediction processor 41 may provide the resulting intra- or inter-coded block to summer 50 to generate residual block data and to summer 62 to reconstruct the encoded block for use as a reference picture.

[0085] Intra prediction unit 46 within prediction processor 41 may perform intra-predictive coding of the current video block relative to one or more neighboring blocks in the same frame or slice as the current block to be coded to provide spatial compression. Motion estimation unit 42 and motion compensation unit 44 within prediction processor 41 perform inter-predictive coding of the current video block

relative to one or more predictive blocks in one or more reference pictures to provide temporal compression.

[0086] Motion estimation unit 42 may be configured to determine the inter-prediction mode for a video slice according to a predetermined pattern for a video sequence. The predetermined pattern may designate video slices in the sequence as P slices, B slices or GPB slices. Motion estimation unit 42 and motion compensation unit 44 may be highly integrated, but are illustrated separately for conceptual purposes. Motion estimation, performed by motion estimation unit 42, is the process of generating motion vectors, which estimate motion for video blocks. A motion vector, for example, may indicate the displacement of a PU of a video block within a current video frame or picture relative to a predictive block within a reference picture.

[0087] A predictive block is a block that is found to closely match the PU of the video block to be coded in terms of pixel difference, which may be determined by sum of absolute difference (SAD), sum of square difference (SSD), or other difference metrics. In some examples, video encoder 20 may calculate values for sub-integer pixel positions of reference pictures stored in reference picture memory 64. For example, video encoder 20 may interpolate values of one-quarter pixel positions, one-eighth pixel positions, or other fractional pixel positions of the reference picture. Therefore, motion estimation unit 42 may perform a motion search relative to the full pixel positions and fractional pixel positions and output a motion vector with fractional pixel precision.

[0088] Motion estimation unit 42 calculates a motion vector for a PU of a video block in an inter-coded slice by comparing the position of the PU to the position of a predictive block of a reference picture. The reference picture may be selected from a first reference picture list (List 0) or a second reference picture list (List 1), each of which identify one or more reference pictures stored in reference picture memory 64. Motion estimation unit 42 sends the calculated motion vector to entropy encoding unit 56 and motion compensation unit 44.

[0089] Motion compensation, performed by motion compensation unit 44, may involve fetching or generating the predictive block based on the motion vector determined by motion estimation, possibly performing interpolations to sub-pixel precision. Upon receiving the motion vector for the PU of the current video block, motion compensation unit 44 may locate the predictive block to which the motion vector points in one of the reference picture lists. Motion compensation unit 44 may also generate syntax elements

associated with the video blocks and the video slice for use by video decoder 30 in decoding the video blocks of the video slice.

[0090] Intra prediction unit 46 may intra-predict a current block, as an alternative to the inter-prediction performed by motion estimation unit 42 and motion compensation unit 44, as described above. In particular, intra prediction unit 46 may determine an intra-prediction mode to use to encode a current block. In some examples, intra prediction unit 46 may encode a current block using various intra-prediction modes, e.g., during separate encoding passes, and intra prediction unit 46 (or mode select unit 35, in some examples) may select an appropriate intra-prediction mode to use from the tested modes. For example, intra prediction unit 46 may calculate rate-distortion values using a rate-distortion analysis for the various tested intra-prediction modes, and select the intra-prediction mode having the best rate-distortion characteristics among the tested modes. Rate-distortion analysis generally determines an amount of distortion (or error) between an encoded block and an original, unencoded block that was encoded to produce the encoded block, as well as a bit rate (that is, a number of bits) used to produce the encoded block. Intra prediction unit 46 may calculate ratios from the distortions and rates for the various encoded blocks to determine which intra-prediction mode exhibits the best rate-distortion value for the block.

[0091] In any case, after selecting an intra-prediction mode for a block, intra prediction unit 46 may provide information indicative of the selected intra-prediction mode for the block to entropy encoding unit 56. Entropy encoding unit 56 may encode the information indicating the selected intra-prediction mode in accordance with the techniques of this disclosure. Video encoder 20 may include in the transmitted bitstream configuration data, which may include a plurality of intra-prediction mode index tables and a plurality of modified intra-prediction mode index tables (also referred to as codeword mapping tables), definitions of encoding contexts for various blocks, and indications of a most probable intra-prediction mode, an intra-prediction mode index table, and a modified intra-prediction mode index table to use for each of the contexts.

[0092] After prediction processor 41 generates the predictive block for the current video block via either inter-prediction or intra-prediction, video encoder 20 forms a residual video block by subtracting the predictive block from the current video block. Summer 50 represents the unit that performs this calculation. The residual video data in the residual block may be included in one or more TUs and applied to transform processing

unit 52. Transform processing unit 52 generally converts the residual video data from a pixel domain to a transform domain, such as a frequency domain. Transform processing unit 52 may transform the residual video data into residual transform coefficients using a transform, such as a discrete cosine transform (DCT) or a conceptually similar transform. Alternatively, transform processing unit 52 may apply a 2-dimensional (2-D) transform (in both the horizontal and vertical direction) to the residual data in the TUs.

[0093] Transform processing unit 52 may send the resulting transform coefficients to quantization unit 54. Quantization unit 54 quantizes the transform coefficients to further reduce the bit rate. The quantization process may reduce the bit depth associated with some or all of the coefficients. The degree of quantization may be modified by adjusting a quantization parameter.

[0094] Following quantization, entropy encoding unit 56 entropy encodes the quantized transform coefficients. For example, entropy encoding unit 56 may perform context adaptive variable length coding (CAVLC), context adaptive binary arithmetic coding (CABAC), syntax-based context-adaptive binary arithmetic coding (SBAC), probability interval partitioning entropy (PIPE) coding or another entropy encoding methodology or technique. Such entropy encoding generally includes scanning the quantized transform coefficients (generally referred to herein simply as "transform coefficients" for brevity) one or more times, and entropy coding syntax elements for the transform coefficients during each scan, such as syntax elements indicating whether corresponding transform coefficients are significant, have an absolute value greater than 1 or 2, the absolute value (or a portion thereof, e.g., a portion greater than 2) and sign of significant coefficients.

[0095] In accordance with the techniques of this disclosure, entropy encoding unit 56 may determine a context for coding (that is, entropy encoding) a transform coefficient of a video block (e.g., a transform unit) based on a region of the video block in which the transform coefficient occurs. For example, during the scan, entropy encoding unit 56 may determine a position of the transform coefficient in the video block, and determine in which region the position occurs. In addition, entropy encoding unit 56 may include configuration data defining regions for a video block.

[0096] For example, entropy encoding unit 56 may be configured with a threshold value. In this example, entropy encoding unit 56 may determine whether x- and y-coordinates defining the position of the transform coefficient have a sum (that is, x+y)

that is greater than the threshold value. A first region, in this example, corresponds to transform coefficients for which the sum of the x- and y-coordinate values is less than the threshold value, and a second region corresponds to transform coefficients for which the sum of the x- and y-coordinate values is greater than or equal to the threshold value. Alternatively, multiple threshold values may be used to define multiple regions. An example of regions defined in this manner is shown in FIG. 9B, which is described in greater detail below.

[0097] As another example, entropy encoding unit 56 may be configured to determine the position of a sub-block, including the transform coefficient, in the video block. A sub-block may correspond to a 4x4 transform coefficient sub-block. That is, a video block may include a plurality of non-overlapping sub-blocks, each having the same size, e.g., 4x4 transform coefficients. To determine a region for a sub-block, entropy encoding unit 56 may compare the sum of an x- and y-coordinate of the sub-block (e.g., a particular transform coefficient of the sub-block, such as an upper-left transform coefficient of the sub-block) to the threshold value. Whether the sum of the x- and y-coordinates is less than the threshold value or not may be indicative of whether the transform coefficients of the sub-block are included in a first region or a second region.

[0098] For example, let Cij represent the position of a sub-block having an upper-left transform coefficient at position (i, j), where x=i and y=j. Further, let T define the threshold value. Entropy encoding unit 56 may determine a region in which transform coefficients of the sub-block occur using the following pseudocode:

(i+j<T) ? region1 : region2.

[0099] In this example, when i+j is less than T (that is, the sum of the x- and y-coordinates of the sub-block is less than the threshold value), entropy encoding unit 56 determines that all transform coefficients of the sub-block occur in region 1, whereas when i+j is greater than or equal to T (that is, the sum of the x- and y-coordinates of the sub-block is greater than or equal to the threshold value), entropy encoding unit 56 determines that all transform coefficients of the sub-block occur in region 2. These and other examples of regions are described in greater detail below with respect to FIGS. 9–14.

[0100] Entropy encoding unit 56 may be configured to determine contexts based on regions in various ways. For example, entropy encoding unit 56 may determine context for coding a transform coefficient, based on the region in which the transform

coefficient occurs, using the location of the transform coefficient in the video block or the position of the 4x4 sub-block in which the transform coefficient occurs.

[0101] Alternatively, a context model may be defined according to neighboring 4x4 sub-blocks. For example, entropy encoding unit 56 may assign to each 4x4 sub-block a respective set of available contexts, and select one of the contexts for the current transform coefficient to be coded in the sub-block, e.g., based on a position of the transform coefficient in the sub-block. The sets of contexts may be assigned to respective sub-blocks, such that each sub-block may have a different set of available contexts. As still another example, entropy encoding unit 56 may calculate a context as ctx= Right4x4SubBlockFlag + Bottom4x4SubBlockFlag. In this case, Right4x4SubBlockFlag represents a coded sub-block flag for a right-neighboring sub-block, while Bottom4x4SubBlockFlag represents a coded sub-block flag for a bottom-neighboring coded sub-block flag.

[0102] In some examples, entropy encoding unit 56 may apply an offset to the determined context for entropy encoding a transform coefficient, and may further determine the offset to apply based on the region in which the transform coefficient occurs. That is, entropy encoding unit 56 may calculate a base context in the same general manner for coefficients of two or more regions, but different regions may have different corresponding offset values. Thus, entropy encoding unit 56 may apply the offset to the calculated context value based on the offset to which the region is mapped (that is, the offset with which the region is associated).

[0103] Entropy encoding unit 56 may determine whether a transform coefficient is a DC (direct current) transform coefficient (typically presented in the upper-left corner of the transform block), and select the context for coding the transform coefficient based on the region in which the transform coefficient occurs as well as whether the transform coefficient is the DC transform coefficient or not. For example, entropy encoding unit 56 may determine contexts for transform coefficients using shared contexts for dedicated positions. That is, the shared context may comprise the same context that is applied to all transform coefficients occurring at a particular position, e.g., an upper-left corner of a sub-block. Thus, the shared context may further include an indication of a particular context to be applied when coding a DC transform coefficient, as opposed to non-DC transform coefficients occurring at the upper-left position of other sub-blocks.

[0104] Additionally or alternatively, shared context may comprise shared contexts among different sizes of blocks for transform coefficients occurring at particular

positions of the blocks. For example, entropy encoding unit 56 may be configured to apply the same context when coding DC transform coefficients of video blocks (e.g., TUs) of any size, e.g., 4x4, 8x8, 16x16, or the like. That is, entropy encoding unit 56 may include data that maps the DC transform coefficient, for blocks of any size, to the same context data for coding the DC transform coefficient. In other words, entropy encoding unit 56 may be configured to code the DC transform coefficient using a context determined for the DC transform coefficient, without regard for a size of the current video block being coded. Typically, the DC transform coefficient is the upper-left coefficient of the video block.

[0105] Following the entropy encoding by entropy encoding unit 56, the encoded bitstream may be transmitted to video decoder 30, or archived for later transmission or retrieval by video decoder 30. Entropy encoding unit 56 may also entropy encode motion vectors, intra-mode indications, and the other syntax elements for the current video slice being coded.

[0106] Inverse quantization unit 58 and inverse transform unit 60 apply inverse quantization and inverse transformation, respectively, to reconstruct the residual block in the pixel domain for later use as a reference block of a reference picture. Motion compensation unit 44 may calculate a reference block by adding the residual block to a predictive block of one of the reference pictures within one of the reference picture lists. Motion compensation unit 44 may also apply one or more interpolation filters to the reconstructed residual block to calculate sub-integer pixel values for use in motion estimation. Summer 62 adds the reconstructed residual block to the motion compensated prediction block produced by motion compensation unit 44 to produce a reference block for storage in reference picture memory 64. The reference block may be used by motion estimation unit 42 and motion compensation unit 44 as a reference block to inter-predict a block in a subsequent video frame or picture.

[0107] In this manner, video encoder 20 represents an example of a video coder configured to determine a context for coding a transform coefficient of a video block based on a region of the video block in which the transform coefficient occurs, and entropy code the transform coefficient using the determined context. The region may comprise one of a first region comprising one or more upper-left 4x4 sub-blocks of transform coefficients of the video block and a second region comprising transform coefficients of the video block outside the first region.

[0108] FIG. 3 is a block diagram illustrating an example video decoder 30 that may implement the inter-prediction techniques described in this disclosure. In the example of FIG. 3, video decoder 30 includes an entropy decoding unit 80, prediction processor 81, inverse quantization unit 86, inverse transformation unit 88, summer 90, and reference picture memory 92. Prediction processor 81 includes motion compensation unit 82 and intra prediction unit 84. Video decoder 30 may, in some examples, perform a decoding pass generally reciprocal to the encoding pass described with respect to video encoder 20 from FIG. 2.

[0109] During the decoding process, video decoder 30 receives an encoded video bitstream that represents video blocks of an encoded video slice and associated syntax elements from video encoder 20. Entropy decoding unit 80 of video decoder 30 entropy decodes the bitstream to generate quantized coefficients, motion vectors, and other syntax elements. Entropy decoding unit 80 forwards the motion vectors, intra-mode indications, and other prediction-related syntax elements to prediction processor 81. Entropy decoding unit 80 forwards quantized coefficients, in the form of a block (e.g., a TU) to inverse quantization unit 86. Video decoder 30 may receive the syntax elements at the video slice level and/or the video block level.

[0110] In particular, in accordance with the techniques of this disclosure, entropy decoding unit 80 may determine context for entropy decoding transform coefficients based on a region of a block in which the transform coefficients occur. Specifically, entropy decoding unit 80 may determine the context based on a region of the block in which the transform coefficient will occur once the transform coefficient is positioned within the block. Entropy decoding unit 80 may be configured to determine the regions as explained with respect to FIGS. 9–14 below, or other such regions. For example, as shown in FIG. 9A, entropy decoding unit 80 may be configured to determine whether a transform coefficient will occur in a first region including one or more sub-blocks in an upper-left corner of the block, or a second region including sub-blocks outside the first region, and determine the context based on whether the transform coefficient will occur in the first region or the second region.

[0111] Likewise, entropy decoding unit 80 may determine the context based on the region, in that entropy decoding unit 80 may be configured with one or more various techniques for calculating or determining the context associated with coefficients in each region. That is, each region may be associated with one or more techniques for calculating or determining context. For example, a region may be associated with a

context that is shared among one or more transform coefficients. As another example, a region may be associated with contexts that are shared among sub-blocks of the region. As still another example, a region may be associated with an offset value to be applied to a context value calculated for a transform coefficient in the region. Entropy decoding unit 80 may be configured to determine the context for decoding a transform coefficient using these or other techniques as described herein, based on the region in which the transform coefficient occurs. Entropy decoding unit 80 may then entropy decode the transform coefficient using the determined context.

[0112] Additionally or alternatively, shared context may comprise shared contexts among different sizes of blocks for transform coefficients occurring at particular positions of the blocks. For example, entropy decoding unit 80 may be configured to apply the same context when coding DC transform coefficients of video blocks (e.g., TUs) of any size, e.g., 4x4, 8x8, 16x16, or the like. That is, entropy decoding unit 80 may include data that maps the DC transform coefficient, for blocks of any size, to the same context data for coding the DC transform coefficient. In other words, entropy decoding unit 80 may be configured to code the DC transform coefficient using a context determined for the DC transform coefficient, without regard for a size of the current video block being coded. Typically, the DC transform coefficient is the upper-left coefficient of the video block.

[0113] When the video slice is coded as an intra-coded (I) slice, intra prediction unit 84 of prediction processor 81 may generate prediction data for a video block of the current video slice based on a signaled intra prediction mode and data from previously decoded blocks of the current frame or picture. When the video frame is coded as an inter-coded (i.e., B, P or GPB) slice, motion compensation unit 82 of prediction processor 81 produces predictive blocks for a video block of the current video slice based on the motion vectors and other syntax elements received from entropy decoding unit 80. The predictive blocks may be produced from one of the reference pictures within one of the reference picture lists. Video decoder 30 may construct the reference frame lists, List 0 and List 1, using default construction techniques based on reference pictures stored in reference picture memory 92.

[0114] Motion compensation unit 82 determines prediction information for a video block of the current video slice by parsing the motion vectors and other syntax elements, and uses the prediction information to produce the predictive blocks for the current video block being decoded. For example, motion compensation unit 82 uses some of

the received syntax elements to determine a prediction mode (e.g., intra- or inter-prediction) used to code the video blocks of the video slice, an inter-prediction slice type (e.g., B slice, P slice, or GPB slice), construction information for one or more of the reference picture lists for the slice, motion vectors for each inter-encoded video block of the slice, inter-prediction status for each inter-coded video block of the slice, and other information to decode the video blocks in the current video slice.

[0115] Motion compensation unit 82 may also perform interpolation based on interpolation filters. Motion compensation unit 82 may use interpolation filters as used by video encoder 20 during encoding of the video blocks to calculate interpolated values for sub-integer pixels of reference blocks. In this case, motion compensation unit 82 may determine the interpolation filters used by video encoder 20 from the received syntax elements and use the interpolation filters to produce predictive blocks.

[0116] Inverse quantization unit 86 inverse quantizes, i.e., de-quantizes, the quantized transform coefficients provided in the bitstream and decoded by entropy decoding unit 80. The inverse quantization process may include use of a quantization parameter calculated by video encoder 20 for each video block in the video slice to determine a degree of quantization and, likewise, a degree of inverse quantization that should be applied. Inverse transform unit 88 applies an inverse transform, e.g., an inverse DCT, an inverse integer transform, or a conceptually similar inverse transform process, to the transform coefficients in order to produce residual blocks in the pixel domain.

[0117] In some cases, inverse transform unit 88 may apply a two-dimensional (2-D) inverse transform (in both the horizontal and vertical direction) to the coefficients. According to the techniques of this disclosure, inverse transform unit 88 may instead apply a horizontal one-dimensional (1-D) inverse transform, a vertical 1-D inverse transform, or no transform to the residual data in each of the TUs. The type of transform applied to the residual data at video encoder 20 may be signaled to video decoder 30 to apply an appropriate type of inverse transform to the transform coefficients.

[0118] After motion compensation unit 82 generates the predictive block for the current video block based on the motion vectors and other syntax elements, video decoder 30 forms a decoded video block by summing the residual blocks from inverse transform unit 88 with the corresponding predictive blocks generated by motion compensation unit 82. Summer 90 represents the component or components that perform this summation operation. If desired, a deblocking filter may also be applied to filter the

decoded blocks in order to remove blockiness artifacts. Other loop filters (either in the coding loop or after the coding loop) may also be used to smooth pixel transitions, or otherwise improve the video quality. The decoded video blocks in a given frame or picture are then stored in reference picture memory 92, which stores reference pictures used for subsequent motion compensation. Reference picture memory 92 also stores decoded video for later presentation on a display device, such as display device 32 of FIG. 1.

[0119] In this manner, video decoder 30 represents an example of a video coder configured to determine a context for coding a transform coefficient of a video block based on a region of the video block in which the transform coefficient occurs, and entropy code the transform coefficient using the determined context. The region may comprise one of a first region comprising one or more upper-left 4x4 sub-blocks of transform coefficients of the video block and a second region comprising transform coefficients of the video block outside the first region.

[0120] FIG. 4 is a conceptual diagram that illustrates a relation between transform coefficients in a video block and a significance map associated with the video block. As illustrated in FIG. 4, the significance map includes a "1" to indicate each instance of a significant coefficient value, i.e., a value greater than zero, in the video block. The significance map may be signaled in a bitstream that is decodable by a video decoder, such as video decoder 30, to determine the location of the significant, i.e., greater than zero, coefficients in the video block to be decoded. More specifically, a position of a last non-zero coefficient within the video block may be signaled in the bitstream. The positional of the last non-zero coefficient in the video block depends on the scanning order used for the video block. Additional syntax elements may be signaled to indicate the other significant coefficients relative to the last non-zero coefficient according to a known or knowable scanning order.

[0121] FIGS. 5A–5D are conceptual diagrams that illustrate examples of blocks of video data scanned using a zig-zag scanning order, a horizontal scanning order, a vertical scanning order, and a diagonal scanning order. As shown in FIGS. 5A–5D, an 8x8 block of video data, e.g., a TU of a CU, may include sixty-four transform coefficients in corresponding block positions, denoted with circles. In this example, blocks 100, 102, 104 and 106 each have a size of 8x8 and, therefore, include sixty-four transform coefficients generated using prediction techniques previously described.

[0122] According to the techniques described in this disclosure, the sixty-four transform coefficients in each of blocks 100, 102, 104 and 106 may have been transformed, or may be inverse transformed, using one of a 2-D transform, a horizontal 1-D transform, and a vertical 1-D transform, or the transform coefficients may not be transformed at all. Whether transformed or not, the coefficients in each of video blocks 100, 102, 104 and 106 are scanned in preparation for entropy coding using one of the zig-zag scanning order, the horizontal scanning order, the vertical scanning order, and the diagonal scanning order.

[0123] As shown in FIG. 5A, the scanning order associated with block 100 is the zig-zag scanning order. The zig-zag scanning order causes a video coder, such as video encoder 20 or video decoder 30, to scan the quantized transform coefficients of block 100 in a diagonal manner as indicated by the arrows in FIG. 5A. Similarly in FIG. 5D, the diagonal scanning order causes a video coder to scan the quantized transform coefficients of block 106 in a diagonal manner as indicated by the arrows in FIG. 5D. As shown in FIGS. 5B and 5C, the scanning orders associated with blocks 102 and 104 are the horizontal scanning order and the vertical scanning order, respectively. The horizontal scanning order causes a video coder to scan quantized transform coefficients of block 102 in a horizontal line-by-line, or "raster" manner, while the vertical scanning order causes a video coder to scan the quantized transform coefficients of block 104 in a vertical line-by-line, or "rotated raster" manner, also as indicated by the arrows in FIGS. 5B and 5C.

[0124] In other examples, as described above, a block may have a size that is smaller or larger than the size of blocks 100, 102, 104 and 106, and may include more or fewer quantized transform coefficients and corresponding block positions. In these examples, a scanning order associated with a particular block may causes a video coder to scan the quantized transform coefficients of the block in a substantially similar manner as shown in the examples of 8x8 blocks of FIGS. 5A–5D, e.g., a 4x4 block or a 16x16 block, may be scanned following any of the scanning orders previously described.

[0125] Although the direction of scans in FIGS. 5A–5D generally is shown as proceeding from low-frequency coefficients to high-frequency coefficients, in other examples, video encoder 20 and video decoder 30 may be configured to perform an inverse scan order, in which the scan may proceed from the high-frequency coefficients to the low-frequency coefficients. That is, video encoder 20 and video decoder 30 may scan the coefficients in the reverse order of that shown in FIGS. 5A–5D.

[0126] FIG. 6 is a conceptual diagram that illustrates an example video block 110 divided into sub-blocks for transform coefficient coding. In the current HM, a sub-block concept is used for transform coefficient coding. A video coder may sub-divide any transform unit (TU) that is larger than a determined sub-block size into sub-blocks. For example, video block 110 is divided into four 4x4 sub-blocks.

[0127] In the illustrated example of FIG. 6, the video coder divides video block 110 into 4x4 sub-blocks. In other examples, the video coder may divide video blocks into sub-blocks of other sizes, e.g., 8x8, 16x16, and the like. If the video coder uses the same sub-block size for all TUs of a frame or slice, gains may be achieved in a hardware implementation due to the uniformity achieved with the sub-block sizes. For example, all processing may be split in such sub-blocks, regardless of the TU size. A uniform sub-block size is not necessary, however, to carry out the techniques of this disclosure.

[0128] For coefficient coding, a video coder may scan each 4x4 sub-block of video block 110 using a diagonal scanning order, as shown on FIG. 6. In some examples, the video coder may use a unified scan for scanning transform coefficients of each sub-block. In this case, the same scan order is used for significance information, i.e., a significance map, coefficient levels, sign, and the like. In a first example, as shown in FIG. 6, the video coder may scan the transform coefficients using a diagonal scan. In another example, the video coder may scan the transform coefficients in an order that is opposite of that shown in FIG. 6, e.g., a reverse diagonal scan that begins in the lower right corner and proceeds to the upper left corner. In other examples, the video coder may scan the transform coefficients using a zig-zag, horizontal, or vertical scan. Other scanning directions/orientations are also possible.

[0129] For ease of explanation, this disclosure describes sub-blocks of a video block as being 4x4 sub-blocks. The techniques of this disclosure, however, may also be applied with respect to sub-blocks of different sizes, e.g., 8x8, 16x16, and the like. For every 4x4 block a *significant_coeffgroup_flag* is coded, and if there is at least one nonzero coefficient in the sub-block this flag is set to one, otherwise it is equal to zero. If *significant_coeffgroup_flag* is nonzero for a given sub-block, the 4x4 sub-block is scanned in the backward diagonal order and *significant_coeff_flag* is coded for every coefficient of the sub-block to indicate the significance of the coefficient. The group of these flags may be referred to as a significance map for the video block. In some example, instead of explicitly signaling the significance map, the *significant_coeffgroup_flag* may be implicitly derived using neighboring 4x4 sub-block

flags, or when the 4x4 sub-block contains the last coefficient or a DC coefficient. Absolute values of the coefficients are also coded, i.e., coefficient levels.

[0130] Although the direction of the scan in FIG. 6 is generally shown as proceeding from low-frequency coefficients to high-frequency coefficients, in other examples, video encoder 20 and video decoder 30 may be configured to perform an inverse scan order, in which the scan may proceed from the high-frequency coefficients to the low-frequency coefficients. That is, video encoder 20 and video decoder 30 may scan the coefficients in the reverse order of that shown in FIG. 6.

[0131] FIG. 7 is a conceptual diagram that illustrates an example five-point support neighborhood used to define a context model for selection of contexts for a significance map of coefficients in a video block 112 scanned using a reverse diagonal scanning order. As noted above, for context-adaptive coding, transform coefficients may be coded based on a context model that describes probabilities of the transform coefficient having a value of 0 or a value of 1. With respect to significance map coding, the context model describes the probabilities of whether a particular transform coefficient is significant, i.e., non-zero.

[0132] For the significance map coding, a five-point support *S* may be used to define a context model to code the significance map of the transform coefficients of video block 112. The five-point support may be referred to as a "context support neighborhood," or simply a "support neighborhood." That is, a video coder may look to the support to determine the probability of the significance of a current position being one or zero. The context support neighborhood defines the neighboring coefficients (e.g., which may include significance information) that may be used as contexts for coding a current coefficient. According to some examples of this disclosure, the context support neighborhood may be different for different coefficient positions within a block or sub-block.

[0133] In the example shown in FIG. 7, the five-point support *S* is represented by a dot surrounded by a square, relative to a current or "target" position represented by a dot surrounded by a circle. Context model *Ctx* (equation (1) below) may be defined as a sum of the significant flags in every point of the support, where a significance flag may be set to "1" if the corresponding transform coefficient is nonzero, and set to "0" otherwise.

$$Ctx = \sum_{p \in S} (coef_p != 0) \quad (1)$$

Accordingly, the significance flag count can be less or equal to the support cardinality. The value of ctx is not necessarily the raw context value, but may be applied to a base context value, in the form of an offset, to derive the context to be used to code data for a particular coefficient.

[0134] However, the support *S* shown in FIG. 7 may not be suitable when calculating context for more than one transform coefficient (e.g., significance information associated with the transform coefficient) in parallel (referred to as "parallel significance context calculation" or simply "parallel context calculation"). For example, using the support *S* shown in FIG. 7 may impede the ability of the video coder to calculate contexts for significance information in parallel, because all data in the support *S* must be available (e.g., already coded) for enabling parallel calculation of contexts. In some instances, as described below with respect to FIG. 8A, a coder may be forced to wait for a support element in support *S* to finish coding before determining the context for another support element in support *S*. This delay reduces the ability of the video coder to efficiently process significance information.

[0135] FIGS. 8A and 8B are conceptual diagrams that illustrate context dependency within the five-point support. For example, to calculate a significance context for the circled position, it may be necessary to parse the significance flag of the position within the support *S* depicted by a diamond (shown in FIG. 8A). Such parsing may introduce a delay if there is a requirement to calculate significance contexts of two coefficients in parallel, because the diamond is positioned immediately before the circled element in scanning order. That is, the context of the circled position cannot be calculated at the same time as the position marked by a diamond, because the circled position depends on the position marked by the diamond, and therefore, the position marked by a diamond must be coded prior to determining the context for the circled position.

[0136] To resolve this dependency, certain elements may be removed from support *S*, making the support with a so called "hole" (non-filled dot surrounded by a triangle, shown in FIG. 8B). For example, the significance flag in the hole is skipped and not taken into account for the context calculation (i.e., assumed to be zero). Accordingly, there is no need to parse the significance flag in the hole position. The 5-point support shape depends on the position to allow for better parallel processing.

[0137] FIGS. 9A and 9B are conceptual diagrams that illustrate example divisions of a video block into two or more regions. In the current HM, neighborhood context modeling is used for TU sizes greater than 8x8 (that is, 16x16, 32x32 and the non-square transform sizes 16x4, 4x16, 32x8 and 8x32) with the 5-point support. However, context modeling with the 5-point support may increase the complexity of the context calculations in the larger block sizes. Region R1 of FIG. 9A represents an example of a region including one or more upper-left 4x4 sub-blocks of transform coefficients of a video block, while region R2 of FIG. 9A represents an example of a region including transform coefficients of the video block outside region R1. FIG. 9A also represents an example in which a plurality of regions comprise respective sets of one or more sub-blocks.

[0138] In accordance with the techniques described in this disclosure, a video coder, such as video encoder 20 or video decoder 30, may divide a video block into regions *R* (e.g., as shown in FIGS. 9A and 9B) and use different context assignment procedures for each of the different regions. For example, some regions may use fixed or position-based context and some regions may use neighborhood-based context. As illustrated in FIG. 9A, the regions may be based on 4x4 sub-blocks such that entire sub-blocks are included in one region or another. Also, the division into the regions may be flexible in some examples. As illustrated in FIG. 9B, the video block may be divided into regions in the diagonal direction such that portions of sub-blocks may be included in two different regions. In other examples, the division might be dependent on the coefficient positions or the position of the 4x4 sub-block containing this coefficient.

[0139] In some examples, context may be defined according to the coefficient position in the video block, or according to the position of the 4x4 sub-block that contains this coefficient. Alternatively, the context model might be defined according to the neighbor 4x4 sub-blocks. For example, every coefficient within same 4x4 sub-block can use one or several contexts, coefficients of the next 4x4 sub-block can use also one or several contexts. However, contexts of one 4x4 sub-block might be different from previous 4x4 sub-block based contexts. Alternatively, contexts might be calculated as $Ctx = Right4x4SubBlockFlag + Bottom4x4SubBlockFlag$, or similar formulas depending on the neighborhood. Again, the *Right4x4SubBlockFlag* may represent a coded sub-block flag for a right-neighboring sub-block (e.g., indicating whether the right-neighboring, 4x4 sub-block includes at least one non-zero coefficient), and the *Bottom4x4SubBlockFlag* may represent a coded sub-block flag for a right-neighboring

sub-block (e.g., indicating whether the bottom-neighboring, 4x4 sub-block includes at least one non-zero coefficient).

[0140] FIG. 10 is a conceptual diagram that illustrates example assignment of neighborhood- or position-based contexts for each region of a video block. As illustrated in FIG. 10, hybrid type of contexts might be used as well, for example, for some regions contexts could be neighborhood based and for some regions of the same video block it can be fixed or position based. A potential advantage of the position-based approach is that it is not necessary to calculate context in a coefficient-wise manner. Instead, a video coder may calculate context once for all coefficients in a region, such that all coefficients in the region have the same context. FIG. 10 represents an example in which a plurality of regions comprises a respective set of one or more sub-blocks.

[0141] For a coefficient with coordinates (x, y), regions can be defined according to the coefficient position. For example, if the condition (x + y >= threshold) is true, then the video coder may determine that the corresponding coefficient occurs within region R2; otherwise, if the condition is not true, the video coder determines that the corresponding coefficient occurs within region R1. Similarly, coordinates can be assigned to regions based on 4x4 sub-blocks. For the sub-block with (X, Y) coordinates, regions can be defined according to the 4x4 sub-block position. For example, if the condition (X + Y >= Threshold) is true, than the video coder may determine that the corresponding coefficient occurs within region R2; otherwise, the video coder may determine that the corresponding coefficient occurs within region R1. The threshold may be fixed to some predefined value, such as an integer number equal to 4, 5, 6, 7 or 8, or may dependent on the video block, e.g., TU, size.

[0142] In this manner, FIG. 10 represents an example in which a video coder may be configured to determine context for coding a transform coefficient, based on a region in which the transform coefficient occurs, using one of position-based context information and neighborhood-based context information based on the region. In particular, if a transform coefficient is in a first region, the video coder may use a first context determination approach to determine the context for coding the transform coefficient. If a transform coefficient is in a second region, the video coder may use a second context determination approach to determine the context for coding the transform coefficient, where the second context determination approach is different from the first context determination approach and the first region is different from the second region. In an

example, the first and second regions do not overlap. Again, examples of the first and second context determination approaches include the use of position-based context information and neighborhood-based context information.

[0143] FIG. 11 is a conceptual diagram that illustrates example assignment of context offsets for each region of a video block. The context model may be separate for the different regions, but still use the same method for context calculation. In other words, a video coder may be configured with one method for calculating context for coding a transform coefficient, but may include different context models, determined based on a region in which the transform coefficient occurs.

[0144] For example, the context may be calculated based on neighborhood, but for different regions it uses an offset. The offset for each region may be fixed or dependent on one or more of the video block size, the coefficient position in the video block or sub-block, and the sub-block position in the video block. Region R1 of FIG. 11 represents another example of a region including one or more upper-left 4x4 sub-blocks of transform coefficients of a video block, while region R2 of FIG. 11 represents another example of a region including transform coefficients of the video block outside region R1. FIG. 11 also an example in which a plurality of regions comprise respective sets of one or more sub-blocks.

[0145] With offset, the context may be calculated according to equation (2).

$$Ctx = offset(region) + \sum_{p \in S} (coef_p != 0) \tag{2}$$

Alternatively, the video coder may calculate the context according to a function using Ctx as an input, for example, Ctx = (Ctx + 1) >> 1.

[0146] One example of the region-based offsets is shown on FIG. 11, where regions R1 and R2 are defined based on 4x4 sub-blocks and offsets are different for regions R1 and R2. Offset values *offset1* and *offset2* could be any integer numbers, for example, offset1 = 0, offset2 = 3. In other example, other divisions into regions are also possible, and divisions into more than two regions are also possible.

[0147] FIG. 12 is a conceptual diagram that illustrates an example embedded division of a video block into two or more regions based on TU sizes that correlate to existing context models. Since there are several sizes of TU in current HM (4x4, 8x8, 16x16 and 32x32), division of the larger blocks can be done along smaller TU sizes using an embedded style of division, as illustrated in FIG. 12. For the embedded division, the

method of context calculation may be shared and the context model itself may be shared.

[0148] For example, for a TU size 32x32, in region R1, representing a 4x4 TU, the context calculation may use the same method for context calculation as for an actual TU of size 4x4. In addition, a context model may be shared between the TU of size 4x4 and R1 of the TU of size 32x32, or an offset may be applied to the context model for the TU of size 4x4. As for R2, the context calculation method may be shared between a TU of size 8x8 and R2 of the TU of size 32x32. R3 represents a 16x16 TU region, while R4 represents a 32x32 TU region. A potential advantage of this method is that the same units may be used for the context calculations, and additional correlation between embedded regions and TUs can be taken into account.

[0149] Alternatively, using embedded style division, some significance map context models may be shared for dedicated positions among all TUs or some group of TUs. For example, a context model, corresponding to DC coefficients, may be shared among all TUs with sizes from 4x4 to 32x32. As another example, a context model, related to high frequency coefficients, may be shared between all TUs. In these cases, region R1, representing a 4x4 TU, in the TU of size 32x32 may use the same context model for DC coefficients and/or high frequency coefficients as TUs having any of sizes 4x4, 8x8, 16x16, 32x32, and the like.

[0150] As a further example, instead of sharing among all TUs, a context model of the coefficients described above (e.g., DC and/or high frequency coefficients) may be shared among only a subset or group of all the TUs. For example, the context model of the coefficient may be shared among only two sizes of TUs, such as 4x4 and 8x8 TUs. In this case, region R1, representing a 4x4 TU, in the TU of size 32x32 may use the same context model for DC coefficients and/or high frequency coefficients as TUs having size 4x4 and 8x8.

[0151] In this manner, the example of FIG. 12 represents an example in which a video coder, such as video encoder 20 or video decoder 30, may be configured to determine a region in which a transform coefficient occurs from a plurality of regions of a video block, wherein each of the regions corresponds to a respective one of a plurality of transform unit (TU) sizes, and wherein the video coder determines the context by selecting a context that is shared between the region and a TU having the same size as the region.

[0152] FIG. 12 also represents an example in which a video coder, such as video encoder 20 or video decoder 30, may be configured to determining a region in which a transform coefficient occurs from a plurality of regions of a video block, wherein each of the regions corresponds to a respective one of a plurality of transform unit (TU) sizes, and wherein to determine the context, the video coder selects a shared context for dedicated positions of transform coefficients between two or more TUs of different sizes, wherein the region has the same size as one of the two or more TUs of different sizes. The shared context for the dedicated positions of transform coefficients may comprise a context for one of DC coefficients and high frequency coefficients shared between the two or more TUs of different sizes. Additionally or alternatively, the shared context for the dedicated positions of transform coefficients may comprise a shared context between a first TU having a size of 4x4 transform coefficients and a second TU having a size of 8x8 transform coefficients.

[0153] FIGS. 13A and 13B are conceptual diagrams that illustrate example divisions of a video block into two or more regions. In a similar manner as described above with respect to examples where regions are based on square, e.g., 4x4, sub-blocks, the techniques of this disclosure also describe a classification method to divide a video block, e.g., a TU, into two or more regions based on rectangular shaped sub-blocks. For example, 2x8 and 8x2 sub-blocks can be used for an 8x8 video block depending on the coefficients scan as shown on FIGS. 13A and 13B. In this example, a video coder applies a horizontal scan for the coefficients in the block shown in FIG. 13A and a vertical scan to the block shown in FIG. 13B. In the examples illustrated in FIGS. 13A and 13B, one square block represents one single coefficient, and the size of the entire video block is 8x8.

[0154] According to the techniques of this disclosure, the video block may be divided into different rectangular regions, e.g., R1, R2, R3, and R4. Each of the different rectangular regions may have a different context assignment. For example, for some regions, a fixed context may be used. These regions may be formed based on rectangular (for example 2x8 or 8x2) sub-blocks, described above and shown in FIGS. 13A and 13B. For example, context could be defined according to the coefficient position in the video block, or according to the position of the rectangular sub-block that contains this coefficient.

[0155] Alternatively, the context model might be defined according to the neighbor rectangular shaped sub-blocks. For example, every coefficient within the same

rectangular sub-block can use one or several contexts. In addition, coefficients of the neighboring rectangular sub-block can also use one or several contexts. However, contexts of one rectangular sub-block may be different from previous rectangular sub-block based contexts. A hybrid type of contexts might be used as well, for example, for some regions contexts may be neighborhood based and for some regions of the same video block it can be fixed or position based. An advantage of the position based approach is that it is not necessary to calculate context coefficient-wise, it can be done once for a region. Also, the division might be dependent on the coefficient positions or the position of the rectangular sub-block containing this coefficient.

[0156] For a coefficient with (x, y) coordinates, regions can be defined according to the coefficient position. For example, if the condition (x + y >= threshold) is true, then this coefficient may be assigned to region R2; otherwise, it may be assigned to region R1. In a similar manner this can be done based on a rectangular shaped sub-block, for the sub-block with (X, Y) coordinates, regions can be defined according to the rectangular sub-block position. For example, if the condition (X + Y >= Threshold) is true than this coefficient may be assigned to region R2, otherwise it may be assigned to R1. The threshold may be fixed to some predefined value, like integer number (e.g., equal to 0 or 1) or might be dependent on TU size.

[0157] Alternatively, a context model may be different for the different regions, but still use the same method for context calculation. For example, context may be calculated based on neighborhood, but for different regions it uses an offset. An offset can be fixed, video block size dependent, or be dependent on one or more of: coefficient position in the video block and/or rectangular sub-block, position of the rectangular sub-block containing the current coefficient in the video block, or any combination of these conditions.

[0158] With an offset, the context may be calculated according to equation (3).

$$Ctx = offset(region) + \sum_{p \in S} (coef_p != 0) \tag{3}$$

[0159] Alternatively, the context may be calculated according to a function using Ctx as an input, for instance, Ctx = (Ctx+1)>>1.

[0160] FIGS. 14A and 14B are conceptual diagrams that illustrate example assignment of context offsets for each region of a video block. In these examples, regions R1 and R2 are defined based on rectangular sub-blocks and scan direction, and offsets are

different for regions R1 and R2. Offset values offset1 and offset2 could be any integer numbers, for example offset1 = 0, offset2 = 3. Other divisions into regions are also possible. For example, a number of regions can be more than two. It should be noted that, 2x8 and 8x2 rectangular sub-blocks, depending on coefficient scanning directions, were used in this disclosure as an example. Similar methods can be used for other rectangular-shaped sub-blocks with size MxN without restriction.

[0161] In general, this disclosure describes diagonal based, square, e.g., 4x4, sub-block based, and rectangular, e.g., 2x8 and 8x2, sub-block based division of video blocks. In other examples, other types of division are possible, and division can be flexible based on various shapes, e.g., rectangular, square, triangular and the like, with different sizes. This disclosure also describes dividing video blocks into any number of regions. This disclosure further describes grouping coefficients into regions based on square sub-block, rectangular sub-blocks, or based on other groupings such as diagonal divisions of a video block. Thresholds and offsets described above are also provided as an example, other values or neighbor dependencies could be exploited.

[0162] Similar techniques as described in this disclosure can be used for non-square transform units or other shapes of units. The described techniques may be applied to significance map coding, and to other syntax and bin coding of transform coefficients without limitation. In addition, this disclosure typically refers to the video blocks as TU blocks, but the techniques may be applied to any of TUs, PUs, CUs, LCUs or other groups of blocks.

[0163] FIG. 15 is a flowchart illustrating an example method for encoding a current block. The current block may comprise a current CU or a portion of the current CU. Although described with respect to video encoder 20 (FIGS. 1 and 2), it should be understood that other devices may be configured to perform a method similar to that of FIG. 15.

[0164] In this example, video encoder 20 initially predicts the current block (150). For example, video encoder 20 may calculate one or more prediction units (PUs) for the current block. Video encoder 20 may then calculate a residual block for the current block, e.g., to produce a transform unit (TU) (152). To calculate the residual block, video encoder 20 may calculate a difference (that is, pixel-by-pixel differences) between the original, uncoded block and the predicted block for the current block. Video encoder 20 may then transform and quantize coefficients of the residual block (154).

Next, video encoder 20 may scan the quantized transform coefficients of the residual block (156).

[0165] During the scan, video encoder 20 may determine a region in which a current coefficient occurs, and in this manner, video encoder 20 may determine regions in which the various coefficients occur (158). In accordance with the techniques of this disclosure, video encoder 20 may determine regions in which coefficients occur based on, for example, positions of the coefficients or positions of sub-blocks in which the coefficients occur. Video encoder 20 may determine regions using any of the techniques described with respect to FIGS. 9–14, or other similar techniques. For example, as shown in FIG. 9A, video encoder 20 may be configured to determine whether a coefficient occurs in a first region including one or more sub-blocks, or a second region including sub-blocks outside the first region.

[0166] Video encoder 20 may further determine contexts for entropy encoding coefficients based on the regions (160). That is, video encoder 20 may determine, for each coefficient, a context for encoding the coefficient based on the region in which the coefficient occurs. For example, as discussed above, video encoder 20 may determine the context based on a position of the coefficient in the block, a position of a sub-block including the coefficient in the block, an offset to be applied to a calculated context, or the like based on the region in which the coefficient occurs.

[0167] Likewise, video encoder 20 may entropy encode the coefficients using the determined contexts (162). In particular, video encoder 20 may entropy encode one or more syntax elements representative of the coefficients using the context. For example, video encoder 20 may entropy encode one or more of significance information for the coefficients, level information for the significant coefficients, and/or sign information for the significant coefficients. Significance information may comprise significant_coeff_flag data. Level information may comprise coeff_abs_level_greater1_flag, coeff_abs_level_greater2_flag, and coeff_abs_level_remaining. Sign information may comprise coeff_sign_flag. Video encoder 20 may then output the entropy encoded data for the coefficients (164).

[0168] In this manner, the method of FIG. 15 represents an example of a method including determining a context for coding a transform coefficient of a video block based on a region of the video block in which the transform coefficient occurs, and entropy coding the transform coefficient using the determined context. Moreover, the region may comprise one of a first region comprising one or more upper-left 4x4 sub-

blocks of transform coefficients of the video block and a second region comprising transform coefficients of the video block outside the first region.

[0169] FIG. 16 is a flowchart illustrating an example method for decoding a current block of video data. The current block may comprise a current CU or a portion of the current CU. Although described with respect to video decoder 30 (FIGS. 1 and 3), it should be understood that other devices may be configured to perform a method similar to that of FIG. 16.

[0170] Video decoder 30 may predict the current block (200), e.g., using an intra- or inter-prediction mode to calculate a predicted block for the current block. Video decoder 30 may also receive entropy encoded data for the current block, such as entropy encoded data for coefficients of a residual block corresponding to the current block (202).

[0171] In accordance with the techniques of this disclosure, video decoder 30 may determine regions in which the coefficients will occur (204), e.g., during an inverse scan and entropy decoding process. That is, video decoder 30 may determine the position of the next transform coefficient based on the position of a previously decoded transform coefficient and a next significant transform coefficient in scan order. Video decoder 30 may further determine a region of the block in which this position occurs. Video decoder 30 may similarly determine regions for each of the coefficients in a similar manner.

[0172] Moreover, video decoder 30 may determine regions in which coefficients will occur based on, for example, positions of the coefficients or positions of sub-blocks in which the coefficients will occur. Video decoder 30 may determine regions using any of the techniques described with respect to FIGS. 9–14, or other similar techniques. For example, as shown in FIG. 9A, video decoder 30 may be configured to determine whether a coefficient occurs in a first region including one or more sub-blocks, or a second region including sub-blocks outside the first region.

[0173] Furthermore, video decoder 30 may determine contexts for decoding the coefficients based on the determined regions (206). That is, video decoder 30 may determine, for each coefficient, a context for decoding the coefficient based on the region in which the coefficient occurs. For example, as discussed above, video decoder 30 may determine the context based on a position of the coefficient in the block, a position of a sub-block including the coefficient in the block, an offset to be applied to a calculated context, or the like, based on the region in which the coefficient will occur.

[0174] Video decoder 30 may entropy decode the entropy coded data to reproduce coefficients of the block using the determined contexts (208). In particular, video decoder 30 may entropy decode one or more syntax elements representative of the coefficients using the context. For example, video decoder 30 may entropy decode one or more of significance information for the coefficients, level information for the significant coefficients, and/or sign information for the significant coefficients. Significance information may comprise significant_coeff_flag data. Level information may comprise coeff_abs_level_greater1_flag, coeff_abs_level_greater2_flag, and coeff_abs_level_remaining. Sign information may comprise coeff_sign_flag. Video decoder 30 may then regenerate the block (e.g., the TU) to include the decoded transform coefficients in their respective positions (210). That is, as discussed above, video decoder 30 may inverse scan the reproduced coefficients to create a block of quantized transform coefficients.

[0175] Video decoder 30 may then inverse quantize and inverse transform the coefficients to produce a residual block (212). Video decoder 30 may ultimately decode the current block by combining the predicted block and the residual block (214). That is, video decoder 30 may mathematically combine the pixel values of the predicted block with co-located pixel values of the residual block to decode and reproduce the original block.

[0176] In this manner, the method of FIG. 16 represents an example of a method including determining a context for coding a transform coefficient of a video block based on a region of the video block in which the transform coefficient occurs, and entropy coding the transform coefficient using the determined context. Moreover, the region may comprise one of a first region comprising one or more upper-left 4x4 sub-blocks of transform coefficients of the video block and a second region comprising transform coefficients of the video block outside the first region.

[0177] In one or more examples, the functions described may be implemented in hardware, software, firmware, or any combination thereof. If implemented in software, the functions may be stored on or transmitted over, as one or more instructions or code, a computer-readable medium and executed by a hardware-based processing unit. Computer-readable media may include computer-readable storage media, which corresponds to a tangible medium such as data storage media, or communication media including any medium that facilitates transfer of a computer program from one place to another, e.g., according to a communication protocol. In this manner, computer-

readable media generally may correspond to (1) tangible computer-readable storage media which is non-transitory or (2) a communication medium such as a signal or carrier wave. Data storage media may be any available media that can be accessed by one or more computers or one or more processors to retrieve instructions, code and/or data structures for implementation of the techniques described in this disclosure. A computer program product may include a computer-readable medium.

[0178] By way of example, and not limitation, such computer-readable storage media can comprise RAM, ROM, EEPROM, CD-ROM or other optical disk storage, magnetic disk storage, or other magnetic storage devices, flash memory, or any other medium that can be used to store desired program code in the form of instructions or data structures and that can be accessed by a computer. Also, any connection is properly termed a computer-readable medium. For example, if instructions are transmitted from a website, server, or other remote source using a coaxial cable, fiber optic cable, twisted pair, digital subscriber line (DSL), or wireless technologies such as infrared, radio, and microwave, then the coaxial cable, fiber optic cable, twisted pair, DSL, or wireless technologies such as infrared, radio, and microwave are included in the definition of medium. It should be understood, however, that computer-readable storage media and data storage media do not include connections, carrier waves, signals, or other transient media, but are instead directed to non-transient, tangible storage media. Disk and disc, as used herein, includes compact disc (CD), laser disc, optical disc, digital versatile disc (DVD), floppy disk and Blu-ray disc, where disks usually reproduce data magnetically, while discs reproduce data optically with lasers. Combinations of the above should also be included within the scope of computer-readable media.

[0179] Instructions may be executed by one or more processors, such as one or more digital signal processors (DSPs), general purpose microprocessors, application specific integrated circuits (ASICs), field programmable logic arrays (FPGAs), or other equivalent integrated or discrete logic circuitry. Accordingly, the term "processor," as used herein may refer to any of the foregoing structure or any other structure suitable for implementation of the techniques described herein. In addition, in some aspects, the functionality described herein may be provided within dedicated hardware and/or software modules configured for encoding and decoding, or incorporated in a combined codec. Also, the techniques could be fully implemented in one or more circuits or logic elements.

[0180] The techniques of this disclosure may be implemented in a wide variety of devices or apparatuses, including a wireless handset, an integrated circuit (IC) or a set of ICs (e.g., a chip set). Various components, modules, or units are described in this disclosure to emphasize functional aspects of devices configured to perform the disclosed techniques, but do not necessarily require realization by different hardware units. Rather, as described above, various units may be combined in a codec hardware unit or provided by a collection of interoperative hardware units, including one or more processors as described above, in conjunction with suitable software and/or firmware.

[0181] Various examples have been described. These and other examples are within the scope of the following claims.

WHAT IS CLAIMED IS:

1. A method of coding video data, the method comprising:

determining a context for coding a transform coefficient of a video block based on a region of the video block in which the transform coefficient occurs; and

entropy coding the transform coefficient using the determined context.

2. The method of claim 1, wherein the region comprises one of a first region comprising one or more upper-left 4x4 sub-blocks of transform coefficients of the video block and a second region comprising transform coefficients of the video block outside the first region.

3. The method of claim 1, wherein the region comprises one of a plurality of regions of the video block, each of the regions comprising respective sets of one or more sub-blocks of the video block.

4. The method of claim 1, wherein coding the transform coefficient comprises coding one or more of significance information associated with the transform coefficient, level information of the transform coefficient, and sign information associated with the transform coefficient.

5. The method of claim 1, wherein the video block comprises one of a transform unit (TU), a prediction unit (PU), a coding unit (CU), a largest coding unit (LCU), and a group of blocks.

6. The method of claim 1, wherein determining the context comprises determining the context using one of position-based context information and neighborhood-based context information based on the region.

7. The method of claim 1, wherein determining the context comprises determining an offset applied to a position-based context for the video block based on the region, wherein the offset for the region is one of a fixed offset and an offset that is dependent on one or more of a size of the video block, a position of the transform coefficient within the video block, and a position of a sub-block that includes the transform coefficient within the video block.

8. The method of claim 1, further comprising determining the region from a plurality of regions of the video block, wherein each of the regions corresponds to a respective one of a plurality of transform unit (TU) sizes, and wherein determining the context comprises selecting a context that is shared between the region and a TU having the same size as the region.

9. The method of claim 1, further comprising determining the region from a plurality of regions of the video block, wherein each of the regions corresponds to a respective one of a plurality of transform unit (TU) sizes, and wherein determining the context comprises selecting a shared context for dedicated positions of transform coefficients between two or more TUs of different sizes, wherein the region has the same size as one of the two or more TUs of different sizes.

10. The method of claim 9, wherein the shared context for the dedicated positions of transform coefficients comprises a context for one of DC coefficients and high frequency coefficients shared between the two or more TUs of different sizes.

11. The method of claim 9, wherein the shared context for the dedicated positions of transform coefficients comprises a shared context between a first TU having a size of 4x4 transform coefficients and a second TU having a size of 8x8 transform coefficients.

12. The method of claim 1, wherein the video block comprises a non-square video block.

13. The method of claim 1, wherein entropy coding the transform coefficient comprises entropy decoding the transform coefficient using the determined context according to context adaptive binary arithmetic coding (CABAC).

14. The method of claim 1, wherein entropy coding the transform coefficient comprises entropy encoding the transform coefficient using the determined context according to context adaptive binary arithmetic coding (CABAC).

15. A device for coding video data, the device comprising a video coder configured to determine a context for coding a transform coefficient of a video block based on a region of the video block in which the transform coefficient occurs, and entropy code the transform coefficient using the determined context.

16. The device of claim 15, wherein the region comprises one of a first region comprising one or more upper-left 4x4 sub-blocks of transform coefficients of the video block and a second region comprising transform coefficients of the video block outside the first region.

17. The device of claim 15, wherein the region comprises one of a plurality of regions of the video block, each of the regions comprising respective sets of one or more sub-blocks of the video block.

18. The device of claim 15, wherein to code the transform coefficient, the video coder is configured to code one or more of significance information associated with the transform coefficient, level information of the transform coefficient, and sign information associated with the transform coefficient.

19. The device of claim 15, wherein the video block comprises one of a transform unit (TU), a prediction unit (PU), a coding unit (CU), a largest coding unit (LCU), and a group of blocks.

20. The device of claim 15, wherein the video coder is configured to determine the context using one of position-based context information and neighborhood-based context information based on the region.

21. The device of claim 15, wherein the video coder is further configured to determine an offset applied to a position-based context for the video block based on the region, wherein the offset for the region is one of a fixed offset and an offset that is dependent on one or more of a size of the video block, a position of the transform coefficient within the video block, and a position of a sub-block that includes the transform coefficient within the video block.

22. The device of claim 15, wherein the video coder comprises a video decoder configured to entropy decode the transform coefficient.

23. The device of claim 15, wherein the video coder comprises a video encoder configured to entropy encode the transform coefficient.

24. A device for coding video data, the device comprising:

means for determining a context for coding a transform coefficient of a video block based on a region of the video block in which the transform coefficient occurs; and

means for entropy coding the transform coefficient using the determined context.

25. The device of claim 24, wherein the region comprises one of a first region comprising one or more upper-left 4x4 sub-blocks of transform coefficients of the video block and a second region comprising transform coefficients of the video block outside the first region.

26. The device of claim 24, wherein the region comprises one of a plurality of regions of the video block, each of the regions comprising respective sets of one or more sub-blocks of the video block.

27. The device of claim 24, wherein the means for coding the transform coefficient comprises means for coding one or more of significance information associated with the transform coefficient, level information of the transform coefficient, and sign information associated with the transform coefficient.

28. The device of claim 24, wherein the video block comprises one of a transform unit (TU), a prediction unit (PU), a coding unit (CU), a largest coding unit (LCU), and a group of blocks.

29. The device of claim 24, wherein the means for determining the context comprises means for determining the context using one of position-based context information and neighborhood-based context information based on the region.

30. The device of claim 24, wherein the means for determining the context comprises means for determining an offset applied to a position-based context for the video block based on the region, wherein the offset for the region is one of a fixed offset and an offset that is dependent on one or more of a size of the video block, a position of the transform coefficient within the video block, and a position of a sub-block that includes the transform coefficient within the video block.

31. A computer-readable storage medium having stored thereon instructions that, when executed, cause a processor to:

determine a context for coding a transform coefficient of a video block based on a region of the video block in which the transform coefficient occurs; and

entropy code the transform coefficient using the determined context.

32. The computer-readable storage medium of claim 31, wherein the region comprises one of a first region comprising one or more upper-left 4x4 sub-blocks of transform coefficients of the video block and a second region comprising transform coefficients of the video block outside the first region.

33. The computer-readable storage medium of claim 31, wherein the region comprises one of a plurality of regions of the video block, each of the regions comprising respective sets of one or more sub-blocks of the video block.

34. The computer-readable storage medium of claim 31, wherein the instructions that cause the processor to code the transform coefficient comprise instructions that cause the processor to code one or more of significance information associated with the transform coefficient, level information of the transform coefficient, and sign information associated with the transform coefficient.

35. The computer-readable storage medium of claim 31, wherein the video block comprises one of a transform unit (TU), a prediction unit (PU), a coding unit (CU), a largest coding unit (LCU), and a group of blocks.

36. The computer-readable storage medium of claim 31, wherein the instructions that cause the processor to determine the context comprise instructions that cause the processor to determine the context using one of position-based context information and neighborhood-based context information based on the region.

37. The computer-readable storage medium of claim 31, wherein the instructions that cause the processor to determine the context comprise instructions that cause the processor to determine an offset applied to a position-based context for the video block based on the region, wherein the offset for the region is one of a fixed offset and an offset that is dependent on one or more of a size of the video block, a position of the transform coefficient within the video block, and a position of a sub-block that includes the transform coefficient within the video block.

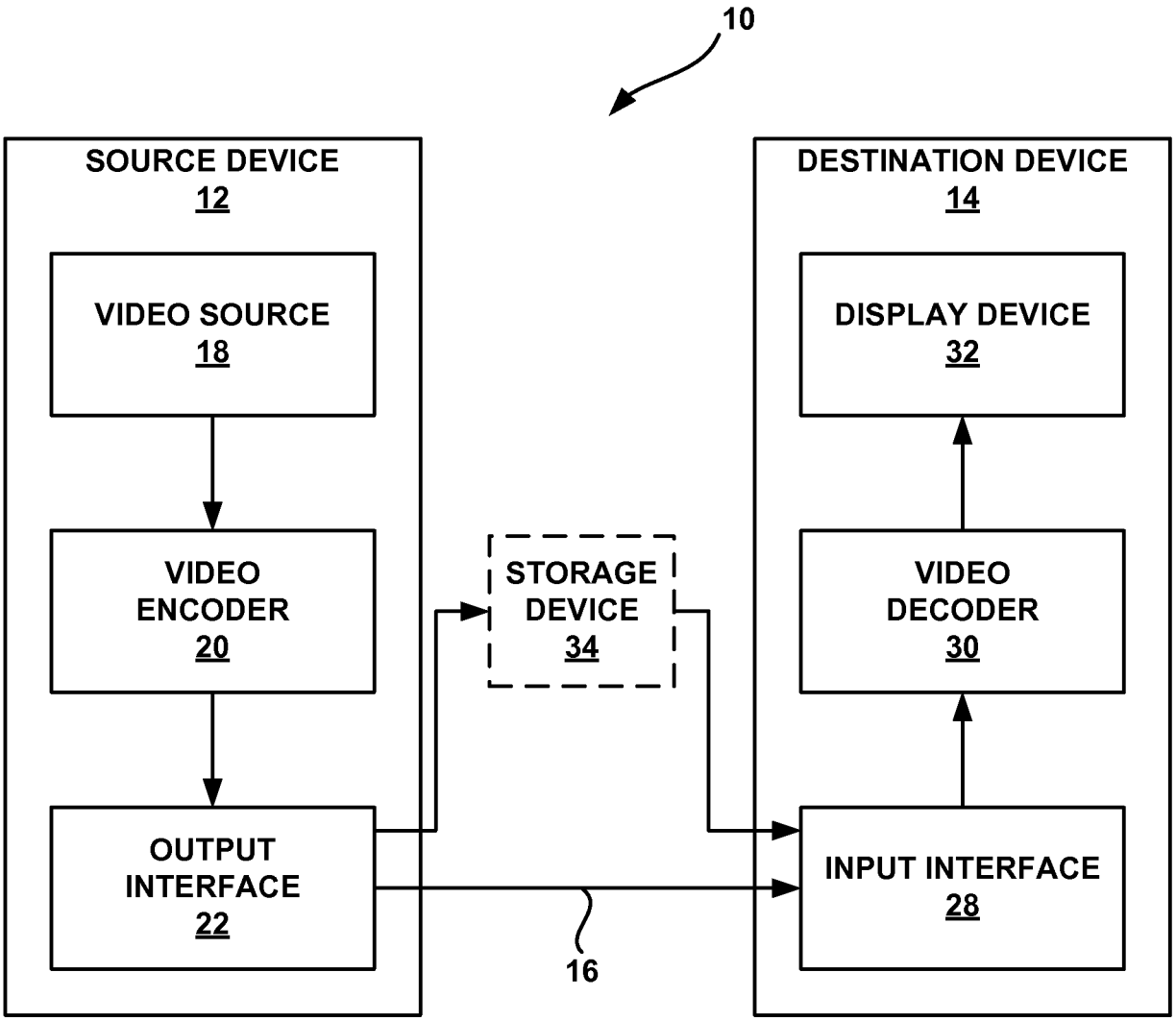

10
SOURCE DEVICE
12
VIDEO SOURCE
18
VIDEO
ENCODER
20
OUTPUT
INTERFACE
22
STORAGE
DEVICE
34
16
DESTINATION DEVICE
14
DISPLAY DEVICE
32
VIDEO
DECODER
30
INPUT INTERFACE
28

FIG. 1

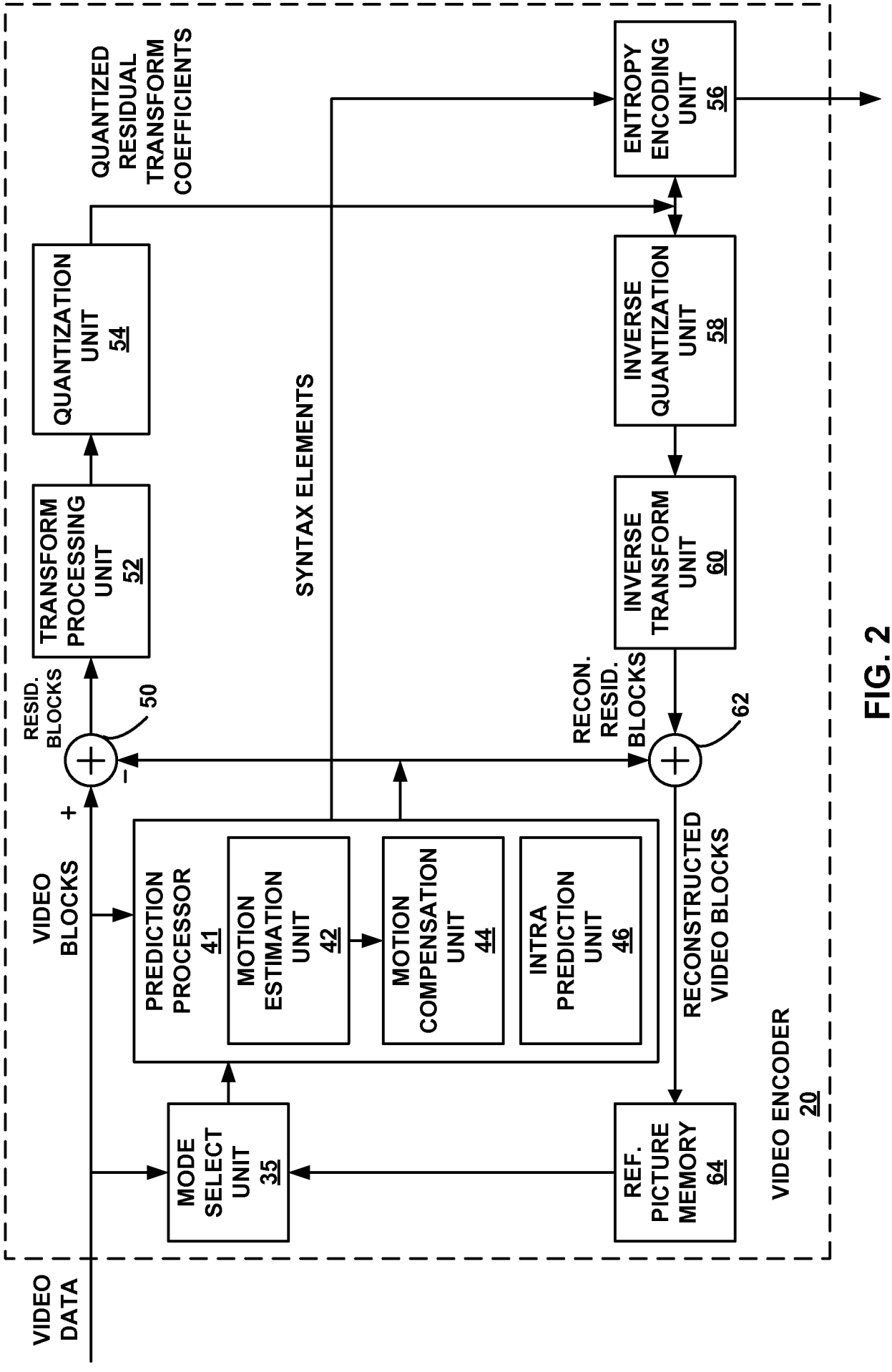

VIDEO DATA
VIDEO ENCODER 20
MODE SELECT UNIT 35
REF. PICTURE MEMORY 64
PREDICTION PROCESSOR 41
MOTION ESTIMATION UNIT 42
MOTION COMPENSATION UNIT 44
INTRA PREDICTION UNIT 46
VIDEO BLOCKS
RESID. BLOCKS
50
TRANSFORM PROCESSING UNIT 52
QUANTIZATION UNIT 54
QUANTIZED RESIDUAL TRANSFORM COEFFICIENTS
SYNTAX ELEMENTS
ENTROPY ENCODING UNIT 56
INVERSE QUANTIZATION UNIT 58
INVERSE TRANSFORM UNIT 60
RECON. RESID. BLOCKS
62
RECONSTRUCTED VIDEO BLOCKS

FIG. 2

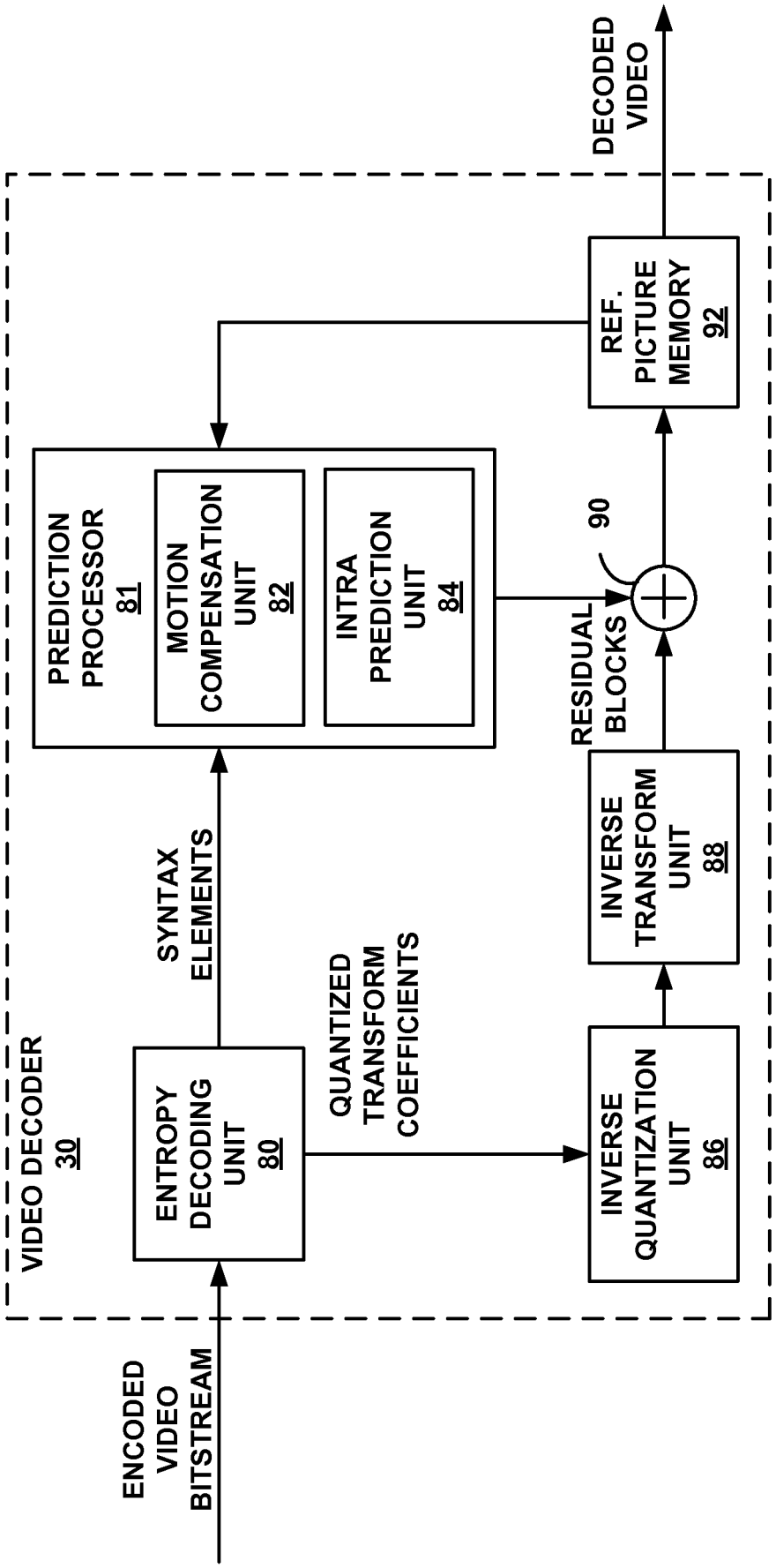

ENCODED VIDEO BITSTREAM
VIDEO DECODER 30
ENTROPY DECODING UNIT 80
SYNTAX ELEMENTS
QUANTIZED TRANSFORM COEFFICIENTS
PREDICTION PROCESSOR 81
MOTION COMPENSATION UNIT 82
INTRA PREDICTION UNIT 84
INVERSE QUANTIZATION UNIT 86
INVERSE TRANSFORM UNIT 88
RESIDUAL BLOCKS
90
REF. PICTURE MEMORY 92
DECODED VIDEO

FIG. 3

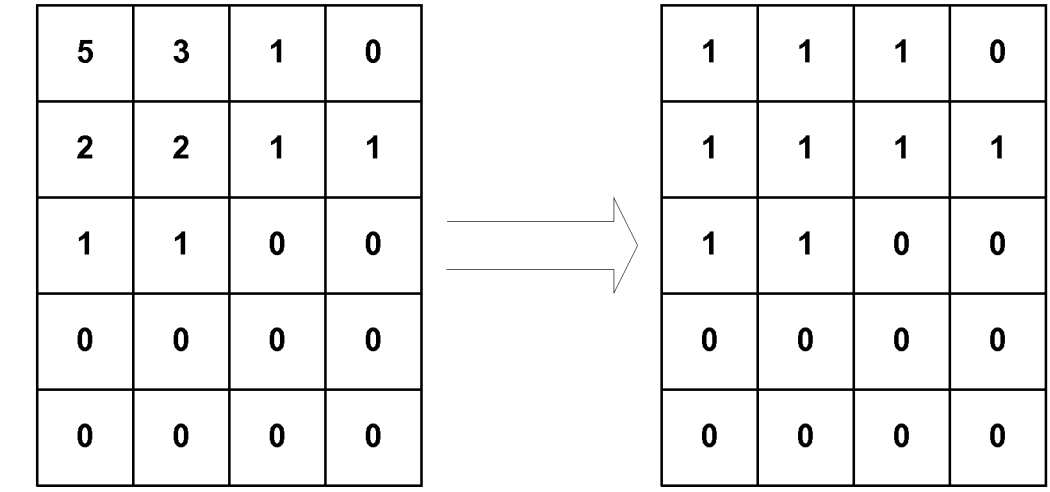

5 3 1 0
2 2 1 1
1 1 0 0
0 0 0 0
0 0 0 0
1 1 1 0
1 1 1 1
1 1 0 0
0 0 0 0
0 0 0 0
TRANSFORM COEFFICIENTS
SIGNIFICANCE MAP

FIG. 4

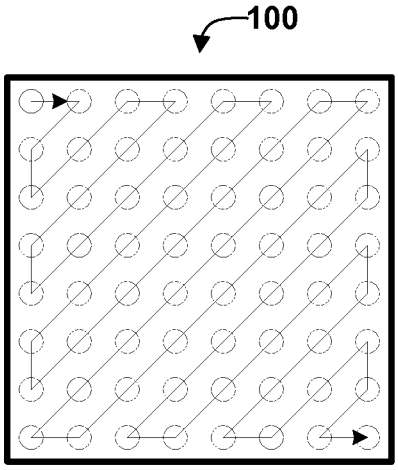


FIG. 5A

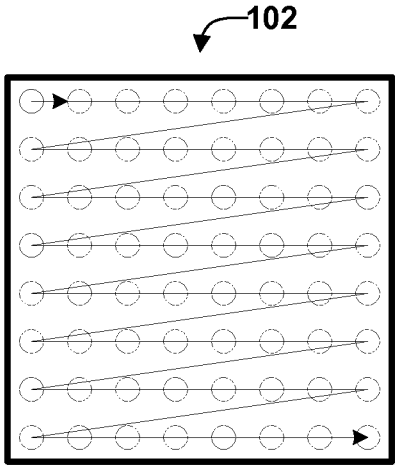


FIG. 5B

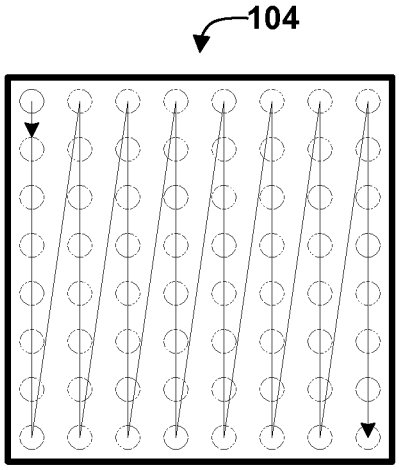


FIG. 5C

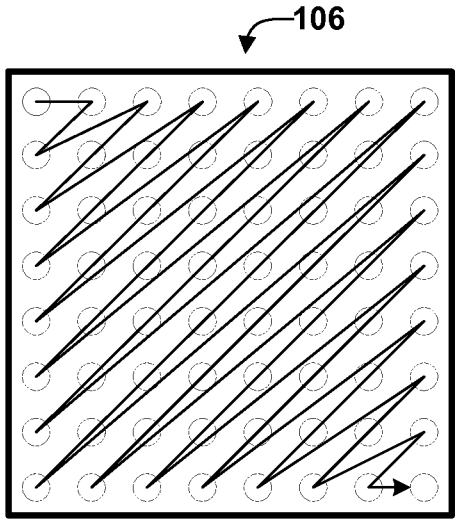


FIG. 5D

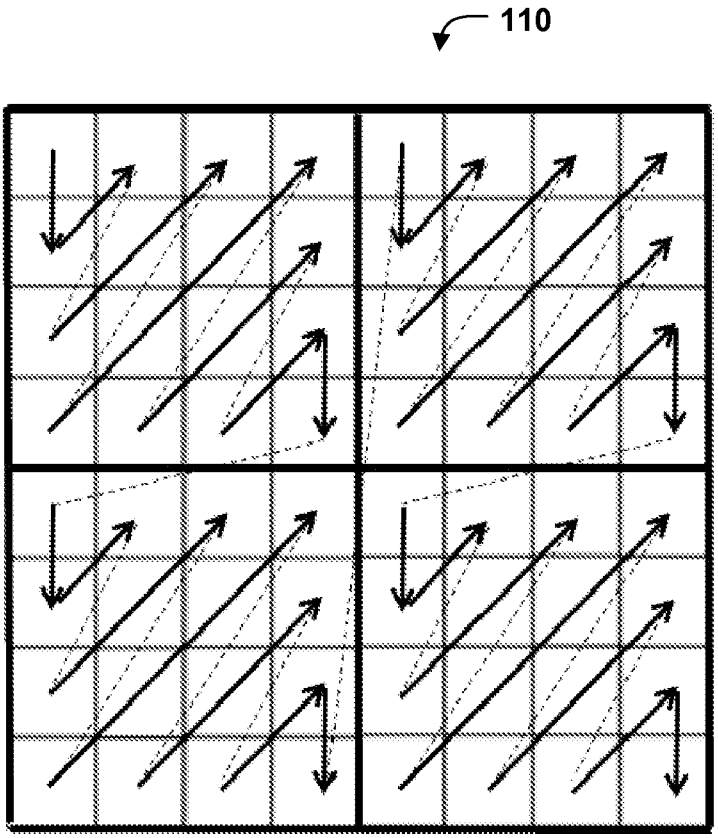


FIG. 6

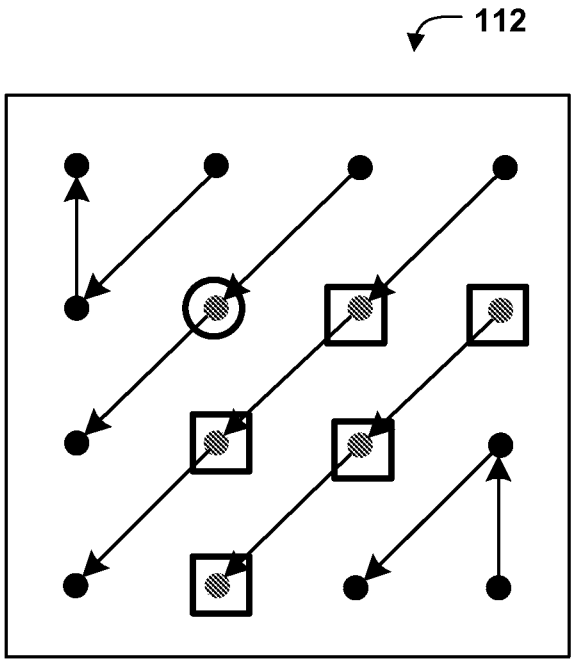


FIG. 7

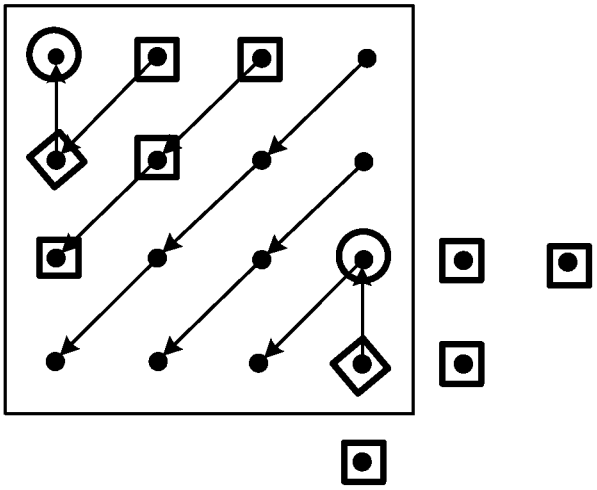


FIG. 8A

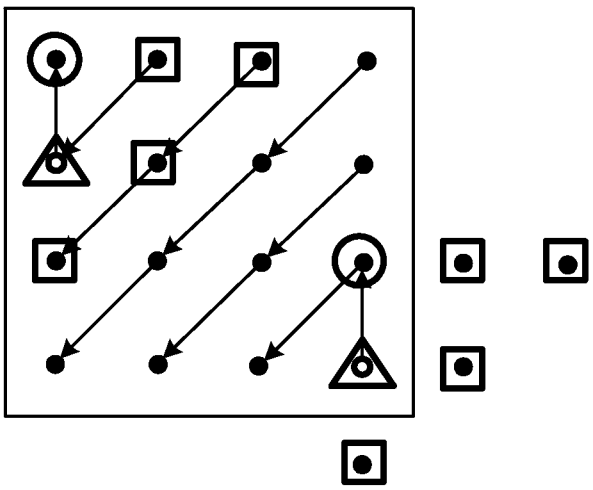


FIG. 8B

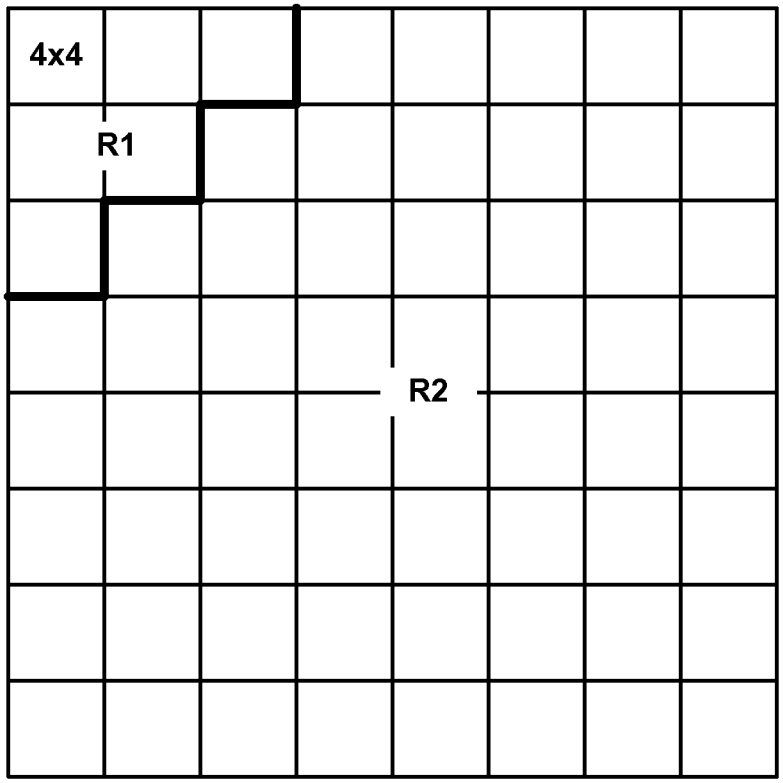


FIG. 9A

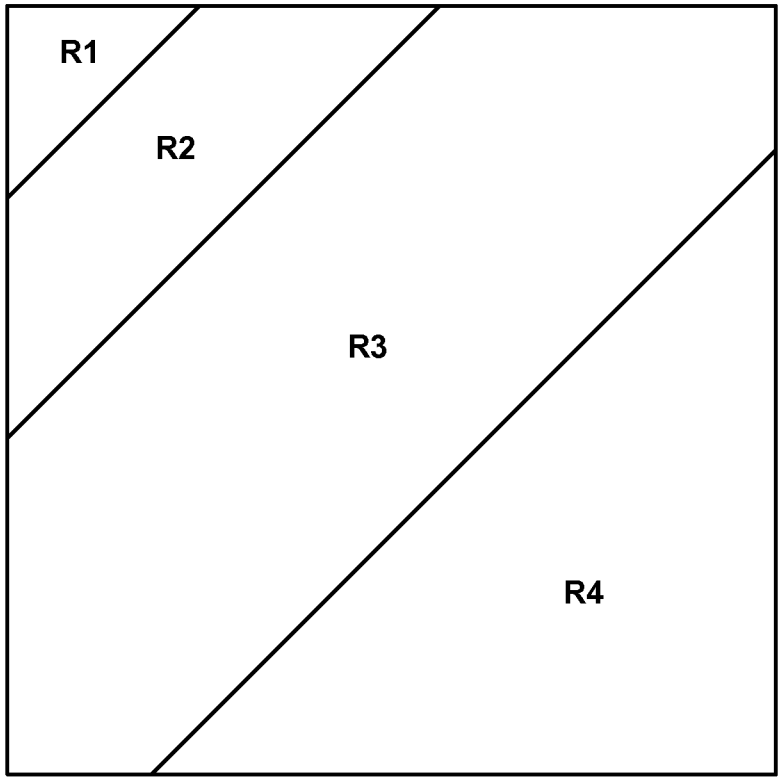


FIG. 9B

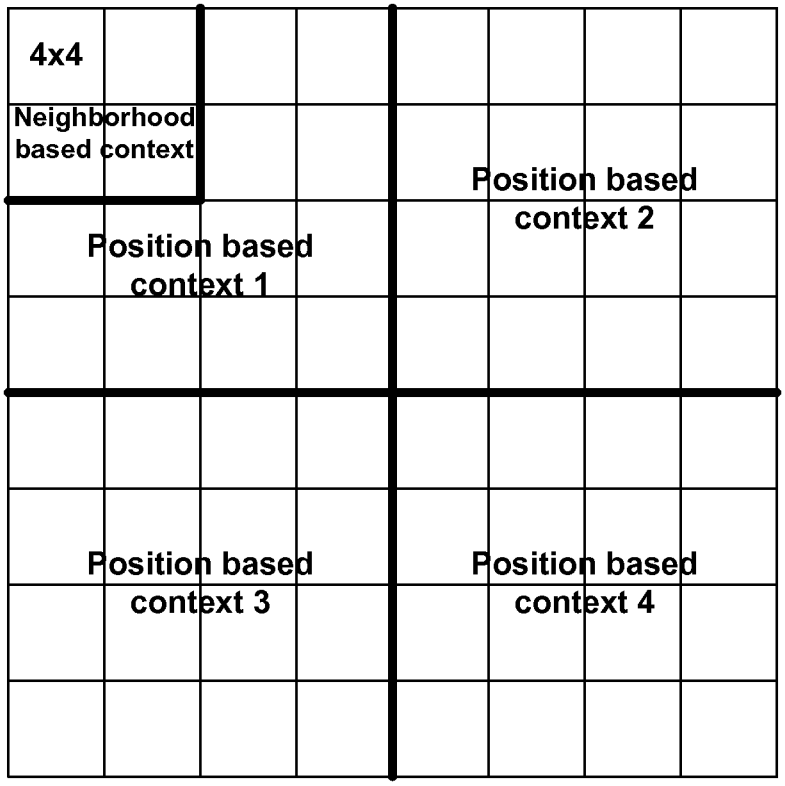

4x4
Neighborhood based context
Position based context 1
Position based context 2
Position based context 3
Position based context 4

FIG. 10

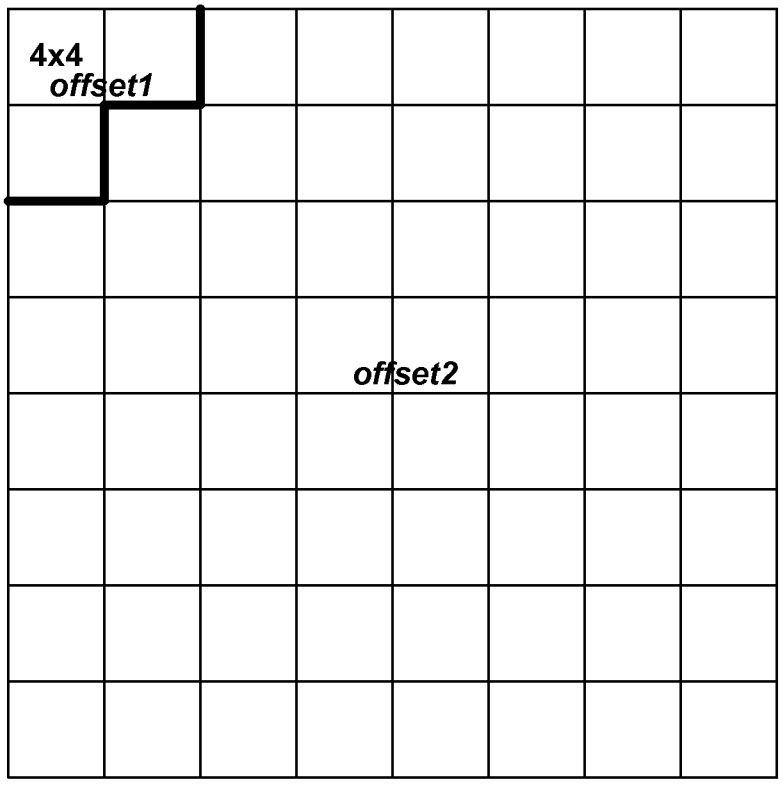

4x4
offset1
offset2

FIG. 11

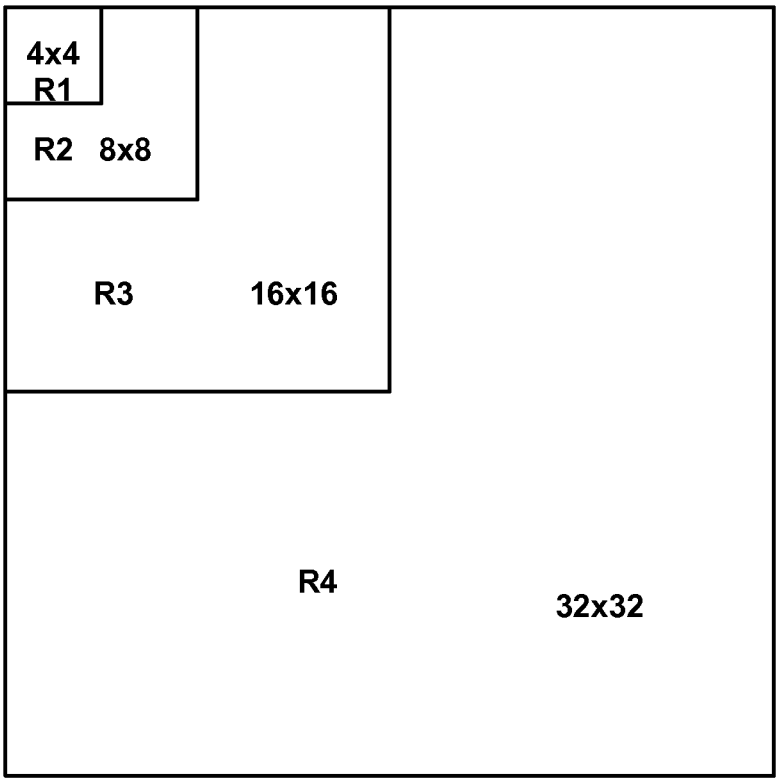


FIG. 12

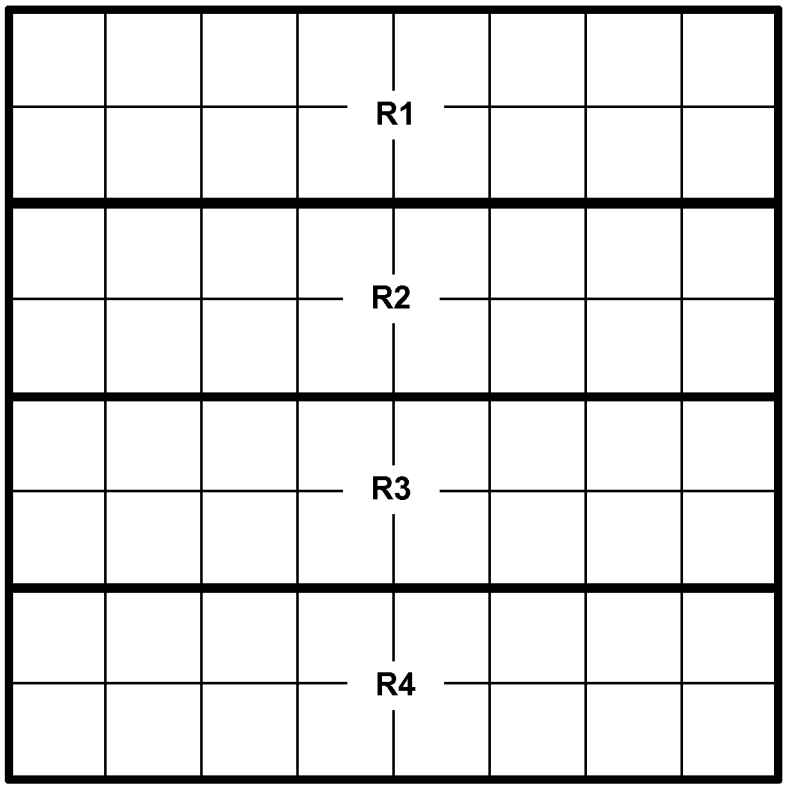


FIG. 13A

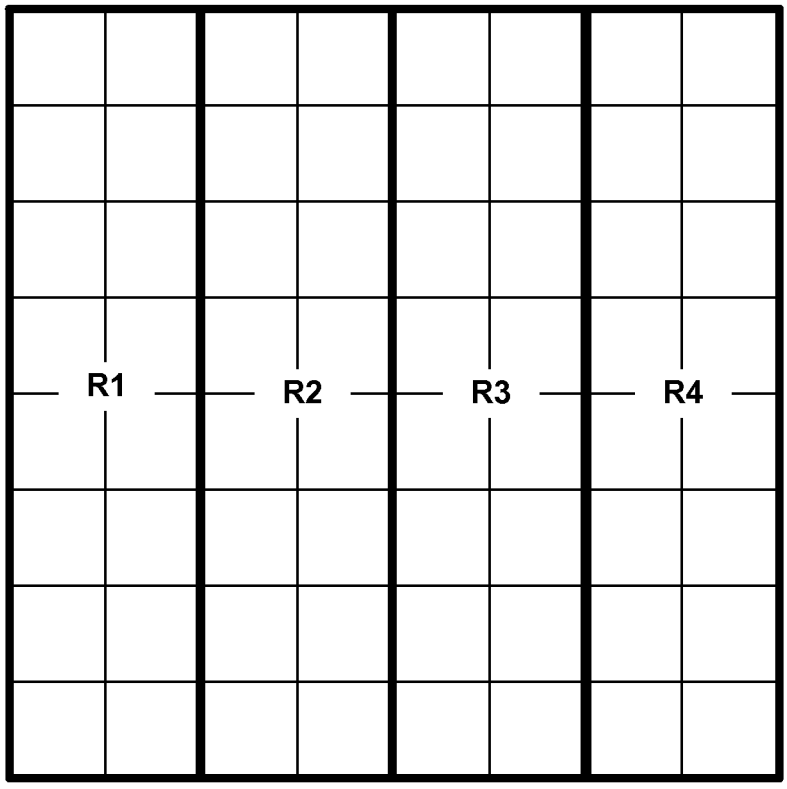


FIG. 13B

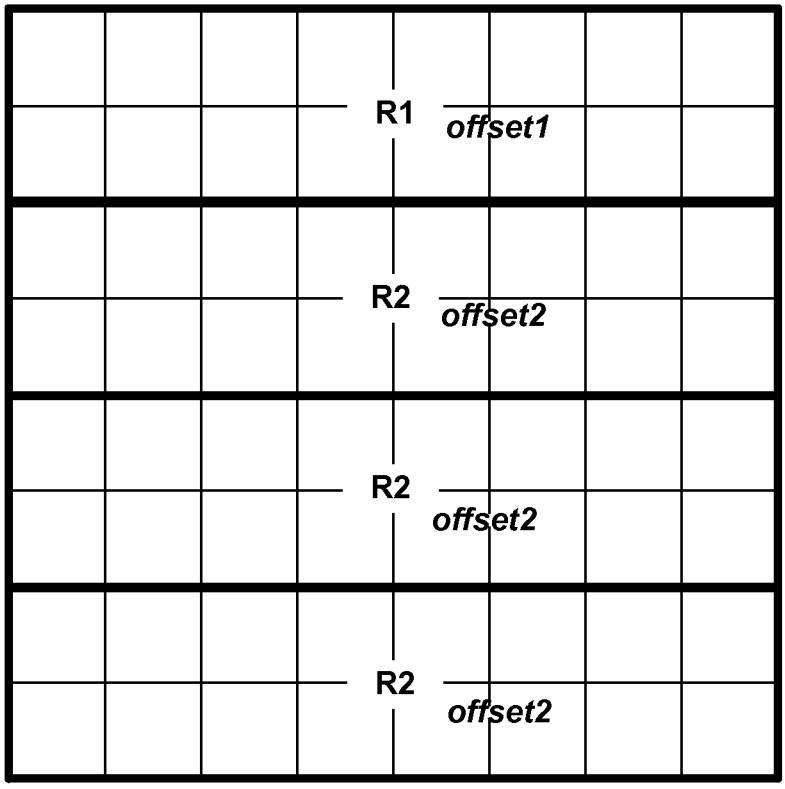


FIG. 14A

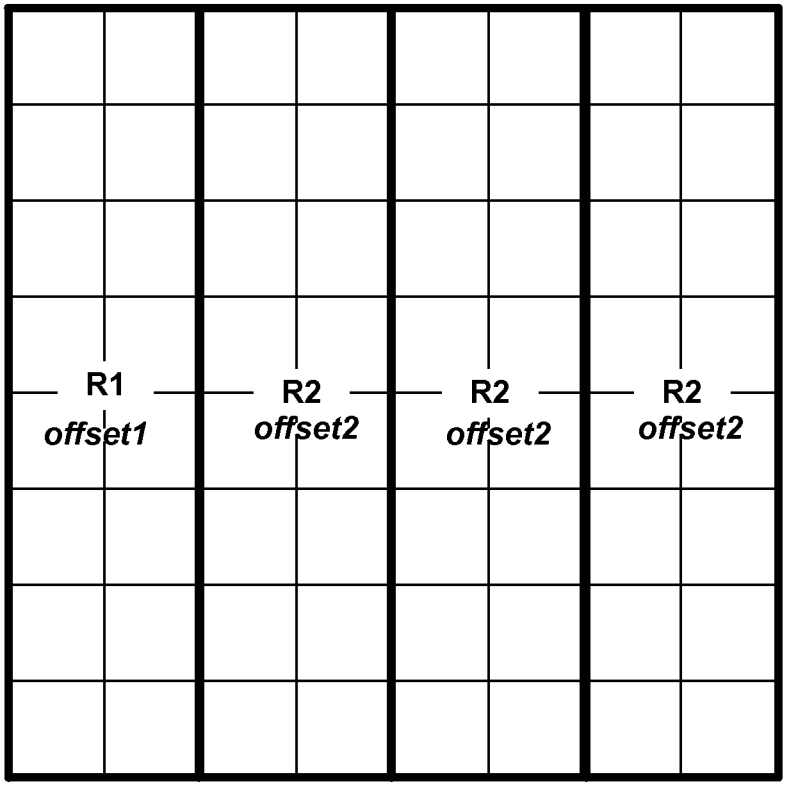


FIG. 14B

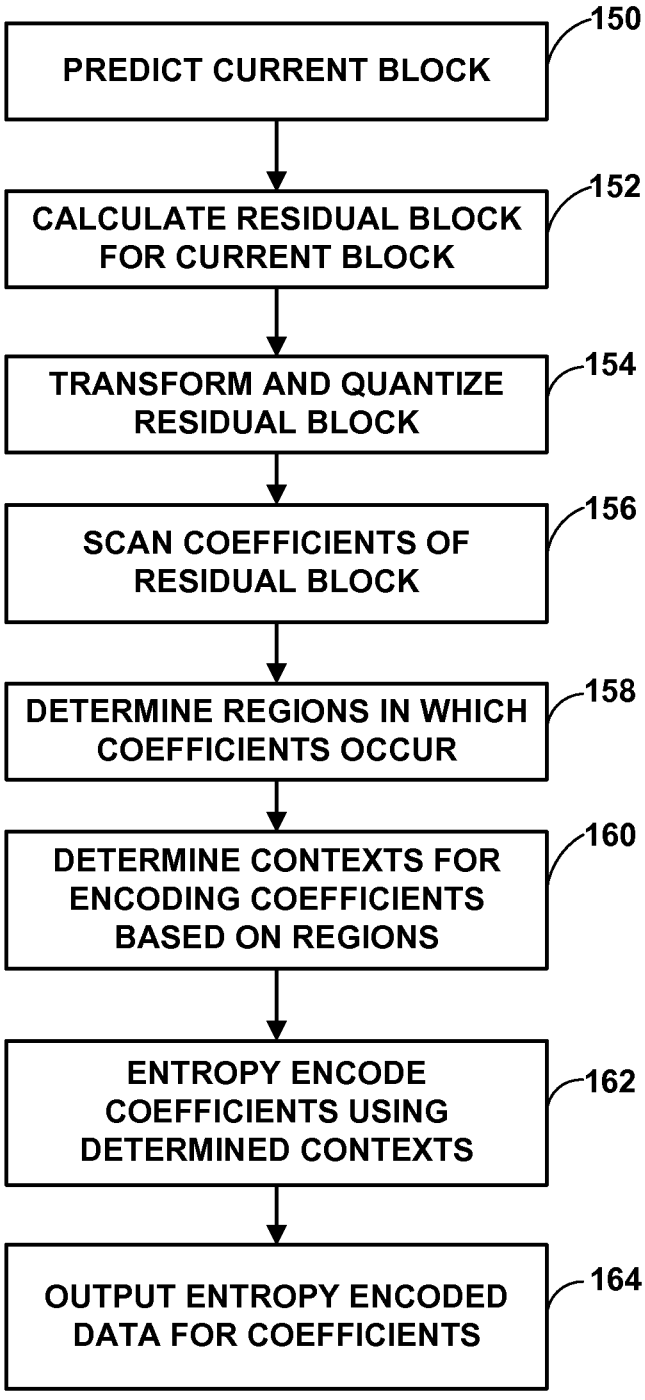

PREDICT CURRENT BLOCK
150
CALCULATE RESIDUAL BLOCK FOR CURRENT BLOCK
152
TRANSFORM AND QUANTIZE RESIDUAL BLOCK
154
SCAN COEFFICIENTS OF RESIDUAL BLOCK
156
DETERMINE REGIONS IN WHICH COEFFICIENTS OCCUR
158
DETERMINE CONTEXTS FOR ENCODING COEFFICIENTS BASED ON REGIONS
160
ENTROPY ENCODE COEFFICIENTS USING DETERMINED CONTEXTS
162
OUTPUT ENTROPY ENCODED DATA FOR COEFFICIENTS
164

FIG. 15

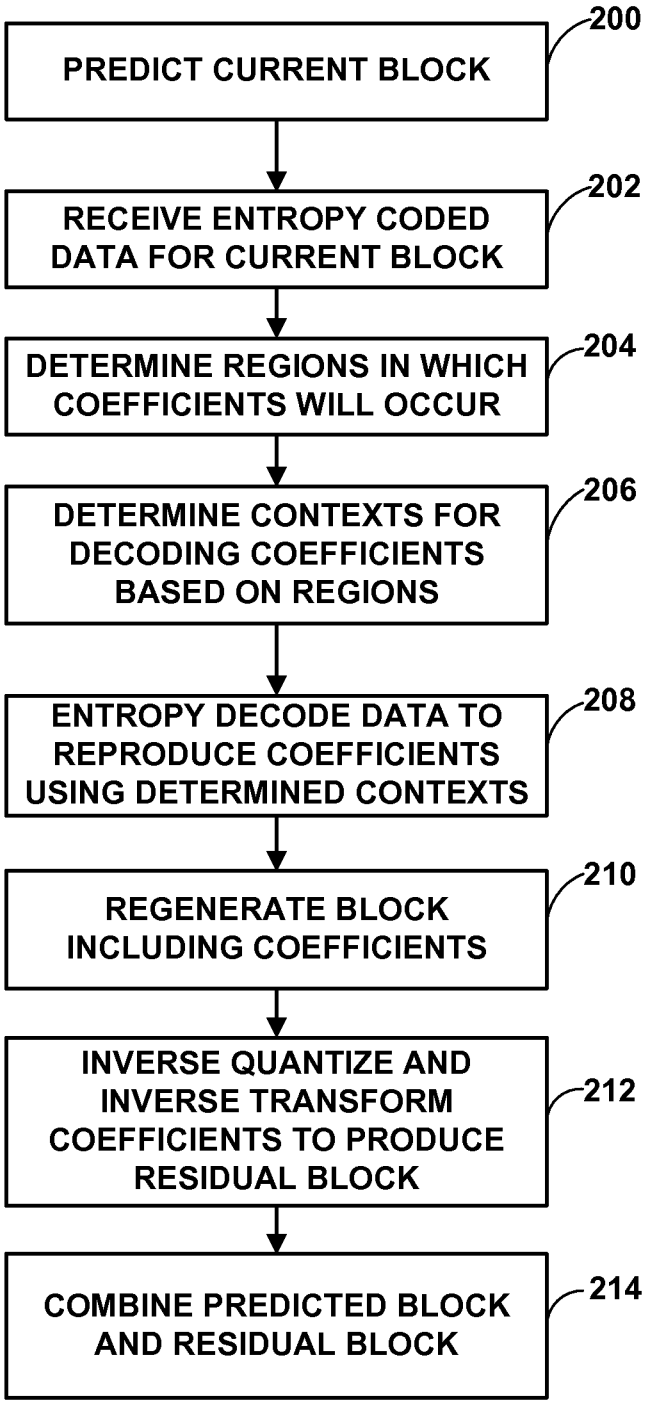

PREDICT CURRENT BLOCK
200
RECEIVE ENTROPY CODED DATA FOR CURRENT BLOCK
202
DETERMINE REGIONS IN WHICH COEFFICIENTS WILL OCCUR
204
DETERMINE CONTEXTS FOR DECODING COEFFICIENTS BASED ON REGIONS
206
ENTROPY DECODE DATA TO REPRODUCE COEFFICIENTS USING DETERMINED CONTEXTS
208
REGENERATE BLOCK INCLUDING COEFFICIENTS
210
INVERSE QUANTIZE AND INVERSE TRANSFORM COEFFICIENTS TO PRODUCE RESIDUAL BLOCK
212
COMBINE PREDICTED BLOCK AND RESIDUAL BLOCK
214

FIG. 16

(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
18 July 2013 (18.07.2013)

WIPO | PCT

(10) International Publication Number
WO 2013/106710 A3

(51) International Patent Classification:
H04N 7/26 (2006.01)

(21) International Application Number:
PCT/US2013/021234

(22) International Filing Date:
11 January 2013 (11.01.2013)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:

61/586,668	13 January 2012 (13.01.2012)	US
61/588,595	19 January 2012 (19.01.2012)	US
61/597,097	9 February 2012 (09.02.2012)	US
13/738,534	10 January 2013 (10.01.2013)	US

(71) Applicant: **QUALCOMM INCORPORATED** [US/US]; ATTN: International IP Administration, 5775 Morehouse Drive, San Diego, California 92121-1714 (US).

(72) Inventors: **SEREGIN, Vadim**; 5775 Morehouse Drive, San Diego, California 92121 (US). **SOLE ROJALS, Joel**; 5775 Morehouse Drive, San Diego, California 92121 (US). **KARCZEWICZ, Marta**; 5775 Morehouse Drive, San Diego, California 92121 (US).

(74) Agent: **DAWLEY, Brian R.**; Shumaker & Sieffert, P.A., 1625 Radio Drive, Suite 300, Woodbury, Minnesota 55125 (US).

(81) Designated States *(unless otherwise indicated, for every kind of national protection available)*: AE, AG, AL, AM, AO, AT, AU, AZ, BA, BB, BG, BH, BN, BR, BW, BY, BZ, CA, CH, CL, CN, CO, CR, CU, CZ, DE, DK, DM,

[Continued on next page]

(54) Title: DETERMINING CONTEXTS FOR CODING TRANSFORM COEFFICIENT DATA IN VIDEO CODING

(57) Abstract: In one example, a device for coding video data includes a video coder configured to determine a context for coding a transform coefficient of a video block based on a region of the video block in which the transform coefficient occurs, and entropy code the transform coefficient using the determined context. The region may comprise one of a first region comprising one or more upper-left 4x4 sub-blocks of transform coefficients of the video block and a second region comprising transform coefficients of the video block outside the first region.

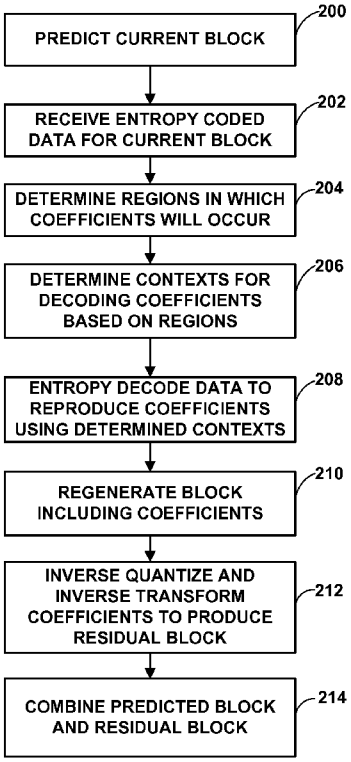


FIG. 16

WO 2013/106710 A3

DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IS, JP, KE, KG, KM, KN, KP, KR, KZ, LA, LC, LK, LR, LS, LT, LU, LY, MA, MD, ME, MG, MK, MN, MW, MX, MY, MZ, NA, NG, NI, NO, NZ, OM, PA, PE, PG, PH, PL, PT, QA, RO, RS, RU, RW, SC, SD, SE, SG, SK, SL, SM, ST, SV, SY, TH, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, ZA, ZM, ZW.

(84) Designated States *(unless otherwise indicated, for every kind of regional protection available)*: ARIPO (BW, GH, GM, KE, LR, LS, MW, MZ, NA, RW, SD, SL, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, RU, TJ, TM), European (AL, AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HR, HU, IE, IS, IT, LT, LU, LV, MC, MK, MT, NL, NO, PL, PT, RO, RS, SE, SI, SK, SM, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, ML, MR, NE, SN, TD, TG).

Published:

— *with international search report (Art. 21(3))*

— *before the expiration of the time limit for amending the claims and to be republished in the event of receipt of amendments (Rule 48.2(h))*

(88) Date of publication of the international search report:
20 February 2014

INTERNATIONAL SEARCH REPORT

International application No
PCT/US2013/021234

A. CLASSIFICATION OF SUBJECT MATTER
INV. H04N7/26
ADD.

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)
H04N

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)
EPO-Internal, INSPEC, WPI Data

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	WO 2011/128303 A2 (FRAUNHOFER GES FORSCHUNG [DE]; WIEGAND THOMAS [DE]; KIRCHHOFFER HEINER) 20 October 2011 (2011-10-20) abstract page 19	1-37
X	WO 03/027940 A1 (NOKIA CORP [FI]; NOKIA INC [US]) 3 April 2003 (2003-04-03) abstract page 24	1-37

☐ Further documents are listed in the continuation of Box C.

☒ See patent family annex.

* Special categories of cited documents :

"A" document defining the general state of the art which is not considered to be of particular relevance

"E" earlier application or patent but published on or after the international filing date

"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)

"O" document referring to an oral disclosure, use, exhibition or other means

"P" document published prior to the international filing date but later than the priority date claimed

"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention

"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone

"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art

"&" document member of the same patent family

Date of the actual completion of the international search	Date of mailing of the international search report
20 December 2013	08/01/2014

Name and mailing address of the ISA/	Authorized officer
European Patent Office, P.B. 5818 Patentlaan 2 NL - 2280 HV Rijswijk Tel. (+31-70) 340-2040, Fax: (+31-70) 340-3016	Kulak, Eray

INTERNATIONAL SEARCH REPORT

Information on patent family members

International application No
PCT/US2013/021234

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
WO 2011128303 A2	20-10-2011	CN 102939755 A	20-02-2013
		EP 2559244 A2	20-02-2013
		JP 2013524707 A	17-06-2013
		KR 20130006678 A	17-01-2013
		TW 201142759 A	01-12-2011
		US 2013051459 A1	28-02-2013
		WO 2011128303 A2	20-10-2011
WO 03027940 A1	03-04-2003	AU 2002334271 B2	07-08-2008
		CN 1585958 A	23-02-2005
		EP 1435063 A1	07-07-2004
		EP 2007147 A2	24-12-2008
		JP 5230890 B2	10-07-2013
		JP 2005504471 A	10-02-2005
		JP 2012080551 A	19-04-2012
		US 2003081850 A1	01-05-2003
		WO 03027940 A1	03-04-2003

(19)中华人民共和国国家知识产权局

(12)发明专利申请

(10)申请公布号 CN 104054340 A
(43)申请公布日 2014.09.17

(21)申请号 201380005233.4
(22)申请日 2013.01.11
(30)优先权数据
61/586,668 2012.01.13 US
61/588,595 2012.01.19 US
61/597,097 2012.02.09 US
13/738,534 2013.01.10 US
(85)PCT国际申请进入国家阶段日
2014.07.11
(86)PCT国际申请的申请数据
PCT/US2013/021234 2013.01.11
(87)PCT国际申请的公布数据
WO2013/106710 EN 2013.07.18
(71)申请人 高通股份有限公司
地址 美国加利福尼亚州
(72)发明人 瓦迪姆·谢廖金
霍埃尔·索赖·罗哈斯
马尔塔·卡切维奇
(74)专利代理机构 北京律盟知识产权代理有限责任公司 11287
代理人 宋献涛
(51)Int.Cl.
H04N 19/119(2014.01)
H04N 19/18(2014.01)
H04N 19/13(2014.01)
H04N 19/60(2014.01)
H04N 19/91(2014.01)

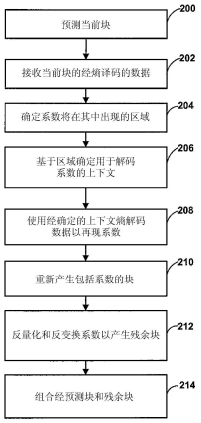
权利要求书3页 说明书25页 附图13页

(54)发明名称

确定在视频译码中用于译码变换系数数据的上下文

(57)摘要

在一个实例中,一种用于译码视频数据的装置包括视频译码器,所述视频译码器经配置以基于其中出现变换系数的视频块的区域来确定用于译码所述视频块的所述变换系数的上下文,且使用所述经确定上下文来熵译码所述变换系数。所述区域可包含第一区域和第二区域中的一者,所述第一区域包含所述视频块的变换系数的一或多个左上 4×4 子块,所述第二区域包含所述视频块的在所述第一区域外部的变换系数。



CN 104054340 A

1. 一种译码视频数据的方法，所述方法包含：

基于其中出现变换系数的视频块的区域来确定用于译码所述视频块的所述变换系数的上下文；及

使用所述经确定上下文来熵译码所述变换系数。

2. 根据权利要求 1 所述的方法，其中所述区域包含第一区域和第二区域中的一者，所述第一区域包含所述视频块的变换系数的一或多个左上 4×4 子块，所述第二区域包含所述视频块的在所述第一区域外部的变换系数。

3. 根据权利要求 1 所述的方法，其中所述区域包含所述视频块的多个区域中的一者，所述区域中的每一者包含所述视频块的一或多个子块的相应集合。

4. 根据权利要求 1 所述的方法，其中译码所述变换系数包含译码与所述变换系数相关联的有效性信息、所述变换系数的水平信息和与所述变换系数相关联的正负号信息中的一者或一者以上。

5. 根据权利要求 1 所述的方法，其中所述视频块包含变换单元 TU、预测单元 PU、译码单元 CU、最大译码单元 LCU 和块群组中的一者。

6. 根据权利要求 1 所述的方法，其中确定所述上下文包含基于所述区域使用基于位置的上下文信息和基于邻域的上下文信息中的一者来确定所述上下文。

7. 根据权利要求 1 所述的方法，其中确定所述上下文包含基于所述区域来确定应用于所述视频块的基于位置的上下文的偏移，其中所述用于所述区域的偏移为固定偏移及取决于所述视频块的大小、所述变换系数在所述视频块内的位置和包括所述变换系数的子块在所述视频块内的位置中的一者或一者以上的偏移中的一者。

8. 根据权利要求 1 所述的方法，其进一步包含从所述视频块的多个区域确定所述区域，其中所述区域中的每一者对应于多个变换单元 TU 大小中的相应一者，且其中确定所述上下文包含选择在所述区域与具有与所述区域相同的大小的 TU 之间共享的上下文。

9. 根据权利要求 1 所述的方法，其进一步包含从所述视频块的多个区域确定所述区域，其中所述区域中的每一者对应于多个变换单元 TU 大小中的相应一者，且其中确定所述上下文包含选择在不同大小的两个或两个以上 TU 之间共享的用于变换系数的专用位置的上下文，其中所述区域具有与不同大小的所述两个或两个以上 TU 中的一者相同的大小。

10. 根据权利要求 9 所述的方法，其中用于变换系数的所述专用位置的所述共享上下文包含在不同大小的所述两个或两个以上 TU 之间共享的用于 DC 系数和高频率系数中的一者的上下文。

11. 根据权利要求 9 所述的方法，其中用于变换系数的所述专用位置的所述共享上下文包含具有 4×4 变换系数的大小的第一 TU 与具有 8×8 变换系数的大小的第二 TU 之间的共享上下文。

12. 根据权利要求 1 所述的方法，其中所述视频块包含非正方形视频块。

13. 根据权利要求 1 所述的方法，其中熵译码所述变换系数包含根据上下文自适应二进制算术译码 CABAC 使用所述经确定上下文来熵解码所述变换系数。

14. 根据权利要求 1 所述的方法，其中熵译码所述变换系数包含根据上下文自适应二进制算术译码 CABAC 使用所述经确定上下文来熵编码所述变换系数。

15. 一种用于译码视频数据的装置，所述装置包含视频译码器，所述视频译码器经配置

以基于其中出现变换系数的视频块的区域来确定用于译码所述视频块的所述变换系数的上下文，且使用所述经确定上下文来熵译码所述变换系数。

16. 根据权利要求 15 所述的装置，其中所述区域包含第一区域和第二区域中的一者，所述第一区域包含所述视频块的变换系数的一或多个左上 4×4 子块，所述第二区域包含所述视频块的在所述第一区域外部的变换系数。

17. 根据权利要求 15 所述的装置，其中所述区域包含所述视频块的多个区域中的一者，所述区域中的每一者包含所述视频块的一或多个子块的相应集合。

18. 根据权利要求 15 所述的装置，其中为了译码所述变换系数，所述视频译码器经配置以译码与所述变换系数相关联的有效性信息、所述变换系数的水平信息和与所述变换系数相关联的正负号信息中的一者或一者以上。

19. 根据权利要求 15 所述的装置，其中所述视频块包含变换单元 TU、预测单元 PU、译码单元 CU、最大译码单元 LCU 和块群组中的一者。

20. 根据权利要求 15 所述的装置，其中所述视频译码器经配置以基于所述区域使用基于位置的上下文信息和基于邻域的上下文信息中的一者来确定所述上下文。

21. 根据权利要求 15 所述的装置，其中所述视频译码器进一步经配置以基于所述区域来确定应用于所述视频块的基于位置的上下文的偏移，其中所述用于所述区域的偏移为固定偏移及取决于所述视频块的大小、所述变换系数在所述视频块内的位置和包括所述变换系数的子块在所述视频块内的位置中的一者或一者以上的偏移中的一者。

22. 根据权利要求 15 所述的装置，其中所述视频译码器包含视频解码器，所述视频解码器经配置以熵解码所述变换系数。

23. 根据权利要求 15 所述的装置，其中所述视频译码器包含视频编码器，所述视频编码器经配置以熵编码所述变换系数。

24. 一种译码视频数据的装置，所述装置包含 ：

用于基于其中出现变换系数的视频块的区域来确定用于译码所述视频块的所述变换系数的上下文的装置 ；及

用于使用所述经确定上下文来熵译码所述变换系数的装置。

25. 根据权利要求 24 所述的装置，其中所述区域包含第一区域和第二区域中的一者，所述第一区域包含所述视频块的变换系数的一或多个左上 4×4 子块，所述第二区域包含所述视频块的在所述第一区域外部的变换系数。

26. 根据权利要求 24 所述的装置，其中所述区域包含所述视频块的多个区域中的一者，所述区域中的每一者包含所述视频块的一或多个子块的相应集合。

27. 根据权利要求 24 所述的装置，其中所述用于译码所述变换系数的装置包含用于译码与所述变换系数相关联的有效性信息、所述变换系数的水平信息和与所述变换系数相关联的正负号信息中的一者或一者以上的装置。

28. 根据权利要求 24 所述的装置，其中所述视频块包含变换单元 TU、预测单元 PU、译码单元 CU、最大译码单元 LCU 和块群组中的一者。

29. 根据权利要求 24 所述的装置，其中所述用于确定所述上下文的装置包含用于基于所述区域使用基于位置的上下文信息和基于邻域的上下文信息中的一者来确定所述上下文的装置。

30. 根据权利要求 24 所述的装置，其中所述用于确定所述上下文的装置包含用于基于所述区域来确定应用于所述视频块的基于位置的上下文的偏移的装置，其中所述用于所述区域的偏移为固定偏移及取决于所述视频块的大小、所述变换系数在所述视频块内的位置和包括所述变换系数的子块在所述视频块内的位置中的一者或一者以上的偏移中的一者。

31. 一种其上存储有指令的计算机可读存储媒体，所述指令在被执行时致使处理器：

基于其中出现变换系数的视频块的区域来确定用于译码所述视频块的所述变换系数的上下文；及

使用所述经确定上下文来熵译码所述变换系数。

32. 根据权利要求 31 所述的计算机可读存储媒体，其中所述区域包含第一区域和第二区域中的一者，所述第一区域包含所述视频块的变换系数的一或多个左上 4×4 子块，所述第二区域包含所述视频块的在所述第一区域外部的变换系数。

33. 根据权利要求 31 所述的计算机可读存储媒体，其中所述区域包含所述视频块的多个区域中的一者，所述区域中的每一者包含所述视频块的一或多个子块的相应集合。

34. 根据权利要求 31 所述的计算机可读存储媒体，其中所述致使所述处理器译码所述变换系数的指令包含致使所述处理器译码与所述变换系数相关联的有效性信息、所述变换系数的水平信息和与所述变换系数相关联的正负号信息中的一者或一者以上的指令。

35. 根据权利要求 31 所述的计算机可读存储媒体，其中所述视频块包含变换单元 TU、预测单元 PU、译码单元 CU、最大译码单元 LCU 和块群组中的一者。

36. 根据权利要求 31 所述的计算机可读存储媒体，其中所述致使所述处理器确定所述上下文的指令包含致使所述处理器基于所述区域使用基于位置的上下文信息和基于邻域的上下文信息中的一者来确定所述上下文的指令。

37. 根据权利要求 31 所述的计算机可读存储媒体，其中所述致使所述处理器确定所述上下文的指令包含致使所述处理器基于所述区域来确定应用于所述视频块的基于位置的上下文的偏移的指令，其中所述用于所述区域的偏移为固定偏移及取决于所述视频块的大小、所述变换系数在所述视频块内的位置和包括所述变换系数的子块在所述视频块内的位置中的一者或一者以上的偏移中的一者。

确定在视频译码中用于译码变换系数数据的上下文

[0001] 本申请案主张2012年1月13日申请的第61/586,668号美国临时申请案、2012年1月19日申请的第61/588,595号美国临时申请案和2012年2月9日申请的第61/597,097号美国临时申请案的权益，所述申请案中的每一者的全部内容特此以引用的方式并入。

技术领域

[0002] 本发明涉及视频译码。

背景技术

[0003] 数字视频能力可并入到广泛范围的装置中，所述装置包括数字电视、数字直播系统、无线广播系统、个人数字助理(PDA)、膝上型或桌上型计算机、平板型计算机、电子书阅读器、数字相机、数字记录装置、数字媒体播放器、视频游戏装置、视频游戏机、蜂窝式或卫星无线电电话、所谓的“智能型手机”、视频电话会议装置、视频串流装置等。数字视频装置实施视频压缩技术，例如以下各者中描述的技术：由MPEG-2、MPEG-4、ITU-T H.263、ITU-T H.264/MPEG-4第10部分(高级视频译码(AVC))定义的标准、目前在开发中的高效率视频译码(HEVC)标准，和这些标准的扩展。视频装置可通过实施所述视频压缩技术而较有效率地发射、接收、编码、解码和/或存储数字视频信息。

[0004] 视频压缩技术执行空间(图片内)预测和/或时间(图片间)预测，以减少或移除视频序列中所固有的冗余。对于基于块的视频译码而言，可将视频切片(即，视频帧或视频帧的一部分)分割成视频块(其还可被称作树块、译码单元(CU)和/或译码节点)。使用相对于同一图片中的相邻块中的参考样本的空间预测来编码图片的帧内译码(I)切片中的视频块。图片的帧间译码(P或B)切片中的视频块可使用相对于同一图片中的相邻块中的参考样本的空间预测或相对于其它参考图片中的参考样本的时间预测。图片可被称作帧，且参考图片可被称作参考帧。

[0005] 空间预测或时间预测导致用于待译码块的预测性块。残余数据表示待译码的原始块与预测性块之间的像素差。根据指向形成预测性块的参考样本的块的运动向量和指示经译码块与预测性块之间的差的残余数据来编码帧间译码块。根据帧内译码模式和残余数据来编码帧内译码块。为进行进一步压缩，可将残余数据从像素域变换到变换域，从而产生残余变换系数，可接着量化残余变换系数。可扫描最初布置成二维阵列的经量化的变换系数以便产生变换系数的一维向量，且可应用熵译码以实现更进一步压缩。

发明内容

[0006] 一般来说，本发明描述关于确定用于熵译码(例如，使用上下文自适应二进制算术译码(CABAC))视频数据的上下文的技术。CABAC译码大体上涉及确定当译码各种语法元素的二进制表示时的上下文。语法元素的实例包括用于变换系数的数据，例如，指示变换系数是否有效的数据、有效的变换系数的正负号，和有效的变换系数的水平值。变换系数大体上对应于例如变换单元(TU)的变换块的系数。本发明描述用于基于变换块的变换系数在

其中出现的区域来确定用于译码变换系数的上下文的技术。

[0007] 在一个实例中，一种译码视频数据的方法包括基于视频块的变换系数在其中出现的区域来确定用于译码所述视频块的所述变换系数的上下文，及使用所述经确定上下文来熵译码所述变换系数。

[0008] 在另一实例中，一种用于译码视频数据的装置包括视频译码器，所述视频译码器经配置以基于视频块的变换系数在其中出现的区域来确定用于译码所述视频块的所述变换系数的上下文，且使用所述经确定上下文来熵译码所述变换系数。

[0009] 在另一实例中，一种用于译码视频数据的装置包括用于基于视频块的变换系数在其中出现的区域来确定用于译码所述视频块的所述变换系数的上下文的装置，及用于使用所述经确定上下文来熵译码所述变换系数的装置。

[0010] 在另一实例中，一种计算机可读存储媒体具有存储于其上的指令，所述指令在被执行时致使处理器基于视频块的变换系数在其中出现的区域来确定用于译码所述视频块的所述变换系数的上下文，且使用所述经确定上下文来熵译码所述变换系数。

[0011] 在另一实例中，一种解码视频数据的方法包括：确定视频块的变换系数是否为 DC 变换系数；在所述变换系数经确定为所述视频块的所述 DC 变换系数时，在不考虑所述视频块的大小的情况下基于为所述 DC 变换系数的所述变换系数来确定用于解码所述变换系数的上下文；及使用所述经确定上下文熵解码所述变换系数。

[0012] 在另一实例中，一种用于解码视频数据的装置包括视频解码器，所述视频解码器经配置以：确定视频块的变换系数是否为 DC 变换系数；在所述变换系数经确定为所述视频块的所述 DC 变换系数时，在不考虑所述视频块的大小的情况下基于为所述 DC 变换系数的所述变换系数来确定用于解码所述变换系数的上下文；及使用所述经确定上下文熵解码所述变换系数。

[0013] 在另一实例中，一种用于解码视频数据的装置包括：用于确定视频块的变换系数是否为 DC 变换系数的装置；用于在所述变换系数经确定为所述视频块的所述 DC 变换系数时在不考虑所述视频块的大小的情况下基于为所述 DC 变换系数的所述变换系数来确定用于解码所述变换系数的上下文的装置；及用于使用所述经确定上下文熵解码所述变换系数的装置。

[0014] 在另一实例中，一种计算机可读存储媒体具有存储于其上的指令，所述指令在被执行时致使处理器进行如下操作：确定视频块的变换系数是否为 DC 变换系数；在所述变换系数经确定为所述视频块的所述 DC 变换系数时，在不考虑所述视频块的大小的情况下基于为所述 DC 变换系数的所述变换系数来确定用于解码所述变换系数的上下文；及使用所述经确定上下文熵解码所述变换系数。

[0015] 在另一实例中，一种编码视频数据的方法包括：确定视频块的变换系数是否为 DC 变换系数；在所述变换系数经确定为所述视频块的所述 DC 变换系数时，在不考虑所述视频块的大小的情况下基于为所述 DC 变换系数的所述变换系数来确定用于编码所述变换系数的上下文；及使用所述经确定上下文熵编码所述变换系数。

[0016] 在另一实例中，一种用于编码视频数据的装置包括视频编码器，所述视频编码器经配置以：确定视频块的变换系数是否为 DC 变换系数；在所述变换系数经确定为所述视频块的所述 DC 变换系数时，在不考虑所述视频块的大小的情况下基于为所述 DC 变换系数的

所述变换系数来确定用于编码所述变换系数的上下文；及使用所述经确定上下文熵编码所述变换系数。

[0017] 在另一实例中，一种用于编码视频数据的装置包括：用于确定视频块的变换系数是否为 DC 变换系数的装置；用于在所述变换系数经确定为所述视频块的所述 DC 变换系数时在不考虑所述视频块的大小的情况下基于为所述 DC 变换系数的所述变换系数来确定用于编码所述变换系数的上下文的装置；及用于使用所述经确定上下文熵编码所述变换系数的装置。

[0018] 在另一实例中，一种计算机可读存储媒体具有存储于其上的指令，所述指令在被执行时致使处理器进行如下操作：确定视频块的变换系数是否为 DC 变换系数；在所述变换系数经确定为所述视频块的所述 DC 变换系数时，在不考虑所述视频块的大小的情况下基于为所述 DC 变换系数的所述变换系数来确定用于编码所述变换系数的上下文；及使用所述经确定上下文熵编码所述变换系数。

[0019] 在另一实例中，一种解码视频数据的方法包括：确定当前子块的一或多个相邻子块的经译码的子块旗标的值；基于所述经译码的子块旗标的所述值来确定用于解码所述当前子块的变换系数的上下文；及使用所述经确定上下文熵解码所述变换系数。

[0020] 在另一实例中，一种用于解码视频数据的装置包括视频解码器，所述视频解码器经配置以：确定当前子块的一或多个相邻子块的经译码的子块旗标的值；基于所述经译码的子块旗标的所述值来确定用于解码所述当前子块的变换系数的上下文；及使用所述经确定上下文熵解码所述变换系数。

[0021] 在另一实例中，一种用于解码视频数据的装置包括：用于确定当前子块的一或多个相邻子块的经译码的子块旗标的值的装置；用于基于所述经译码的子块旗标的所述值来确定用于解码所述当前子块的变换系数的上下文的装置；及用于使用所述经确定上下文熵解码所述变换系数的装置。

[0022] 在另一实例中，一种计算机可读存储媒体具有存储于其上的指令，所述指令在被执行时致使处理器进行如下操作：确定当前子块的一或多个相邻子块的经译码的子块旗标的值；基于所述经译码的子块旗标的所述值来确定用于解码所述当前子块的变换系数的上下文；及使用所述经确定上下文熵解码所述变换系数。

[0023] 在另一实例中，一种编码视频数据的方法包括：确定当前子块的一或多个相邻子块的经译码的子块旗标的值；基于所述经译码的子块旗标的所述值来确定用于编码所述当前子块的变换系数的上下文；及使用所述经确定上下文熵编码所述变换系数。

[0024] 在另一实例中，一种用于编码视频数据的装置包括视频编码器，所述视频编码器经配置以：确定当前子块的一或多个相邻子块的经译码的子块旗标的值；基于所述经译码的子块旗标的所述值来确定用于编码所述当前子块的变换系数的上下文；及使用所述经确定上下文熵编码所述变换系数。

[0025] 在另一实例中，一种用于编码视频数据的装置包括：用于确定当前子块的一或多个相邻子块的经译码的子块旗标的值的装置；用于基于所述经译码的子块旗标的所述值来确定用于编码所述当前子块的变换系数的上下文的装置；及用于使用所述经确定上下文熵编码所述变换系数的装置。

[0026] 在另一实例中，一种计算机可读存储媒体具有存储于其上的指令，所述指令在被

执行时致使处理器进行如下操作：确定当前子块的一或多个相邻子块的经译码的子块旗标的值；基于所述经译码的子块旗标的所述值来确定用于编码所述当前子块的变换系数的上下文；及使用所述经确定上下文熵编码所述变换系数。

[0027] 在附图和以下描述中阐述一或多个实例的细节。其它特征、目标和优点将从所述描述和图式及从权利要求书显而易见。

附图说明

[0028] 图 1 为说明可利用本发明中所描述的帧间预测技术的实例视频编码和解码系统的方框图。

[0029] 图 2 为说明可实施本发明中所描述的帧间预测技术的实例视频编码器的方框图。

[0030] 图 3 为说明可实施本发明中所描述的帧间预测技术的实例视频解码器的方框图。

[0031] 图 4 为说明视频块中的变换系数与和所述视频块相关联的有效性映射之间的关系的概念图。

[0032] 图 5A 到图 5D 为说明使用 z 字形扫描次序、水平扫描次序、垂直扫描次序和对角线扫描次序扫描的视频数据块的实例的概念图。

[0033] 图 6 为说明划分为子块以用于变换系数译码的实例视频块的概念图。

[0034] 图 7 为说明实例五点支持的概念图，所述五点支持用以定义使用反向对角线扫描次序扫描的视频块中的系数的有效性映射的上下文模型。

[0035] 图 8A 和图 8B 为说明五点支持内的上下文相依性的概念图。

[0036] 图 9A 和图 9B 为说明视频块到两个或两个以上区域的实例划分的概念图。

[0037] 图 10 为说明针对视频块的每一区域的基于邻域或基于位置的上下文的实例指派的概念图。

[0038] 图 11 为说明针对视频块的每一区域的上下文偏移的实例指派的概念图。

[0039] 图 12 为说明视频块基于与现有上下文模型相关的 TU 大小的到两个或两个以上区域的实例内嵌式划分的概念图。

[0040] 图 13A 和图 13B 为说明视频块到两个或两个以上区域的实例划分的概念图。

[0041] 图 14A 和图 14B 为说明针对视频块的每一区域的上下文偏移的实例指派的概念图。

[0042] 图 15 为说明用于编码当前块的实例方法的流程图。

[0043] 图 16 为说明用于解码视频数据的当前块的实例方法的流程图。

具体实施方式

[0044] 一般来说，本发明描述关于确定用于视频数据的熵译码（例如，使用上下文自适应二进制算术译码 (CABAC)）的上下文的技术。CABAC 译码通常涉及确定当译码各种语法元素的二进制表示时的上下文。语法元素包括（例如）用于变换系数的数据，例如，指示变换系数是否有效的数据、有效的变换系数的正负号，和有效的变换系数的水平值。变换系数大体上对应于例如变换单元 (TU) 的变换块的系数。本发明描述用于基于变换块的变换系数在其中出现的区域来确定用于译码变换系数的上下文的技术。

[0045] 一般来说，根据本发明的技术，视频译码器可经配置以基于变换系数在其中出现

的区域来确定用于译码变换系数的上下文，且接着使用经确定上下文来熵译码变换系数。可以多种方式将视频块划分为区域。图 9A 和图 11 说明将视频块划分为包括一或多个左上子块（例如，4×4 个子块）的第一区域和包括在所述第一区域外部的子块的第二区域的实例。图 9B 说明沿对角线方向将视频块划分为区域的实例。图 10 说明将视频块划分为四等分且将左上四等分进一步划分为包括左上四等分的左上部分的子块的第一子区域和包括所述左上四等分的在所述第一子区域外部的子块的第二子区域的实例。图 12 说明将视频块划分为对应于视频块大小（例如，4×4、8×8、16×16 和 32×32）的区域的实例。图 13A 说明将视频块划分为水平矩形区域的实例。图 13B 说明将视频块划分为垂直矩形区域的实例。以下更详细地描述这些图。

[0046] 在各种实例中，视频译码器可经配置而以多种方式（例如，基于变换系数在其中出现的区域）来确定用于译码变换系数的上下文。举例来说，视频译码器可经配置而使用一些区域的基于位置的上下文信息或其它区域的基于邻域的上下文信息来确定上下文。在一些实例中，特定区域内的所有变换系数可使用基于所述区域所确定的同一上下文来译码。在其它实例中，可基于上下文邻域来确定一区域内的变换系数的上下文。在另外实例中，视频译码器可基于变换系数在其中出现的区域来确定待应用于上下文的偏移。即，所述区域中的每一者可与待应用于上下文的特定上下文偏移相关联。

[0047] 本发明的技术可减少带宽消耗，进而导致在译码变换系数的语法元素时位的节省。所述语法元素可包括以下各者中的任一者或其全部：有效系数旗标（其指示对应变换系数是否有效，即，非零）；有效系数的正负号；有效系数是否具有大于 1 的绝对值的指示；具有大于 1 的绝对值的有效系数是否具有大于 2 的绝对值的指示；及 / 或具有大于 2 的绝对值的系数的剩余水平值。

[0048] 图 1 为说明可利用本发明中所描述的技术的实例视频编码和解码系统 10 的方框图。如图 1 中所展示，系统 10 包括源装置 12，源装置 12 产生稍后待由目的地装置 14 解码的经编码视频数据。源装置 12 和目的地装置 14 可包含广泛范围的装置中的任一者，包括桌上型计算机、笔记型（即，膝上型）计算机、平板型计算机、机顶盒、电话手机（例如，所谓“智能型”手机）、所谓“智能型”板、电视、摄影机、显示装置、数字媒体播放器、视频游戏机、视频串流装置，或其类似者。在一些情形下，源装置 12 和目的地装置 14 可经装备以进行无线通信。

[0049] 目的地装置 14 可经由链路 16 接收待解码的经编码视频数据。链路 16 可包含能够将经编码的视频数据从源装置 12 移动到目的地装置 14 的任何类型的媒体或装置。在一实例中，链路 16 可包含用以使源装置 12 能够将经编码视频数据直接实时发射到目的地装置 14 的通信媒体。可根据通信标准（例如，无线通信协议）调制经编码视频数据，且将经编码视频数据发射到目的地装置 14。通信媒体可包含任何无线或有线通信媒体，例如，射频 (RF) 频谱或一或多个物理传输线。通信媒体可形成基于包的网络（例如，局域网、广域网或例如因特网的全球网络）的部分。通信媒体可包括路由器、交换器、基站，或可用以促进从源装置 12 到目的地装置 14 的通信的任何其它设备。

[0050] 或者，经编码数据可从输出接口 22 输出到存储装置 34。类似地，可通过输入接口从存储装置 34 存取经编码数据。存储装置 34 可包括多种分布式或本地存取的数据存储媒体中的任一者，例如，硬盘驱动器、蓝光光盘、DVD、CD-ROM、快闪存储器、易失性或非易失性

存储器，或用于存储经编码视频数据的任何其它合适数字存储媒体。在另一实例中，存储装置 34 可对应于可保持由源装置 12 产生的经编码视频的文件服务器或另一中间存储装置。目的地装置 14 可经由串流传输或下载从存储装置 34 存取所存储的视频数据。文件服务器可为能够存储经编码视频数据且将所述经编码视频数据发射到目的地装置 14 的任何类型的服务器。实例文件服务器包括网站服务器（例如，用于网站）、FTP 服务器、网络附加存储 (NAS) 装置或本地磁盘驱动器。目的地装置 14 可经由任何标准数据连接（包括因特网连接）而存取经编码视频数据。此数据连接可包括适合于存取存储于文件服务器上的经编码视频数据的无线信道（例如，Wi-Fi 连接）、有线连接（例如，DSL、缆线调制解调器，等等），或所述两者的组合。经编码视频数据从存储装置 34 的传输可为串流传输、下载传输或所述两者的组合。

[0051] 本发明的技术未必限于无线应用或环境。所述技术可应用于支持多种多媒体应用中的任一者的视频译码，所述应用例如：空中电视广播、有线电视发射、卫星电视发射、（例如）经由因特网的串流视频传输、供存储于数据存储媒体上的数字视频的编码、存储于数据存储媒体上的数字视频的解码，或其它应用。在一些实例中，系统 10 可经配置以支持单向或双向视频发射以支持例如视频串流、视频播放、视频广播和 / 或视频电话的应用。

[0052] 在图 1 的实例中，源装置 12 包括视频源 18、视频编码器 20 和输出接口 22。在一些情形下，输出接口 22 可包括调制器 / 解调器（调制解调器）和 / 或发射器。在源装置 12 中，视频源 18 可包括源，例如，视频俘获装置（例如，摄像机）、含有先前俘获的视频的视频存档、用以从视频内容提供者接收视频的视频馈送接口，和 / 或用于产生计算机图形数据作为源视频的计算机图形系统，或这些源的组合。作为一个实例，如果视频源 18 为摄像机，则源装置 12 和目的地装置 14 可形成所谓的相机电话或视频电话。然而，一股本发明中所描述的技术可适用于视频译码，且可应用于无线和 / 或有线应用。

[0053] 经俘获、经预先俘获或经计算机产生的视频可由视频编码器 20 来编码。可经由源装置 12 的输出接口 22 将经编码视频数据直接发射到目的地装置 14。还可（或替代性地）将经编码视频数据存储到存储装置 34 上以供目的地装置 14 或其它装置稍后存取以用于解码和 / 或播放。

[0054] 目的地装置 14 包括输入接口 28、视频解码器 30 和显示装置 32。在一些情形下，输入接口 28 可包括接收器和 / 或调制解调器。目的地装置 14 的输入接口 28 经由链路 16 接收经编码视频数据。经由链路 16 传达或在存储装置 34 上提供的经编码视频数据可包括由视频编码器 20 产生的多种语法元素，其供例如视频解码器 30 的视频解码器在解码视频数据时使用。所述语法元素可与在通信媒体上传输、存储于存储媒体上或存储于文件服务器上的经编码视频数据包括在一起。

[0055] 显示装置 32 可与目的地装置 14 集成在一起或在目的地装置 14 外部。在一些实例中，目的地装置 14 可包括集成式显示装置且还经配置以与外部显示装置介接。在其它实例中，目的地装置 14 可为显示装置。一股来说，显示装置 32 向用户显示经解码视频数据，且可包含多种显示装置中的任一者，例如，液晶显示器 (LCD)、等离子体显示器、有机发光二极管 (OLED) 显示器或另一类型的显示装置。

[0056] 视频编码器 20 和视频解码器 30 可根据视频压缩标准（例如，目前在开发中的高效率视频译码 (HEVC) 标准）而操作，且可遵照 HEVC 测试模型 (HM)。或者，视频编码器 20

和视频解码器 30 可根据例如 ITU-T H.264 标准或者被称作 MPEG-4 第 10 部分（高级视频译码 (AVC)）的其它专属或工业标准或这些标准的扩展而操作。举例来说，标准的扩展包括可缩放视频译码 (SVC)、多视图视频译码 (MVC)、例如译码深度信息的三维 (3D) 译码等。然而，本发明的技术不限于任何特定译码标准或标准扩展。视频压缩标准的其它实例包括 MPEG-2 和 ITU-T H.263。

[0057] 尽管图 1 中未展示，但在一些方面中，视频编码器 20 和视频解码器 30 可各自与音频编码器和解码器集成在一起，且可包括适当的 MUX-DEMUX 单元或其它硬件和软件以处置共同数据串流或单独数据串流中的音频和视频两者的编码。如果适用，则在一些实例中，MUX-DEMUX 单元可遵照 ITU H.223 多路复用器协议，或例如用户数据报协议 (UDP) 的其它协议。

[0058] 视频编码器 20 和视频解码器 30 各自可实施为多种合适编码器电路中的任一者，例如一或多个微处理器、数字信号处理器 (DSP)、专用集成电路 (ASIC)、现场可编程门阵列 (FPGA)、离散逻辑、软件、硬件、固件或其任何组合。当所述技术部分地以软件实施时，装置可将用于软件的指令存储于合适的非暂时性计算机可读媒体中，且在硬件中使用一或多个处理器来执行所述指令以执行本发明的技术。视频编码器 20 和视频解码器 30 中的每一者可包括于一或多个编码器或解码器中，其中任一者可集成为相应装置中的组合式编码器 / 解码器（编解码器）的部分。

[0059] JCT-VC 正致力于 HEVC 标准的开发。HEVC 标准化努力是基于视频译码装置的演进模型，其被称作 HEVC 测试模型 (HM)。HM 假设视频译码装置相对于根据（例如）ITU-TH.264/AVC 的现有装置的若干额外能力。举例来说，尽管 H.264 提供九个帧内预测编码模式，但 HM 可提供多达三十三个帧内预测编码模式。

[0060] 一股来说，HM 的工作模型描述视频帧或图片可被划分成包括明度样本和色度样本两者的树块或最大译码单元 (LCU) 的序列。树块具有与 H.264 标准的宏块的用途类似的用途。切片包括按译码次序的众多连续树块。可将视频帧或图片分割成一或多个切片。每一树块可根据四叉树而分裂成译码单元 (CU)。举例来说，树块（作为四叉树的根节点）可分裂成四个子节点，且每一子节点又可为上代节点，且分裂成另外四个子节点。最后未分裂的子节点（作为四叉树的叶节点）包含译码节点，即，经译码视频块。与经译码的位流相关联的语法数据可定义树块可分裂的最大次数，且还可定义译码节点的最小大小。

[0061] CU 包括译码节点和与所述译码节点相关联的预测单元 (PU) 和变换单元 (TU)。CU 的大小对应于译码节点的大小，且形状必须为正方形。CU 的大小的范围可从 8×8 个像素直到具有最大 64×64 个像素或大于 64×64 个像素的树块的大小。每一 CU 可含有一或多个 PU 和一或多个 TU。与 CU 相关联的语法数据可描述（例如）CU 到一或多个 PU 的分割。分割模式可在 CU 是被跳过还是经直接模式编码、经帧内预测模式编码还是经帧间预测模式编码之间不同。PU 可分割成非正方形。与 CU 相关联的语法数据还可描述（例如）根据四叉树的 CU 到一或多个 TU 的分割。TU 的形状可为正方形或非正方形。

[0062] HEVC 标准允许实现根据 TU 的变换，变换可针对不同 CU 而不同。通常基于针对经分割 LCU 所定义的给定 CU 内的 PU 的大小来设定 TU 大小，尽管可能并非总是如此情形。TU 通常具有与 PU 相同的大小，或小于 PU。在一些实例中，可使用称为“残余四叉树”(RQT) 的四叉树结构而将对应于 CU 的残余样本再分为更小的单元。RQT 的叶节点可称作变换单元

(TU)。可变换与 TU 相关联的像素差值以产生可量化的变换系数。

[0063] 一股来说，PU 包括与预测过程有关的数据。举例来说，当将 PU 以帧内模式编码时，所述 PU 可包括描述所述 PU 的帧内预测模式的数据。作为另一实例，当将 PU 以帧间模式编码时，所述 PU 可包括定义所述 PU 的运动向量的数据。定义 PU 的运动向量的数据可描述（例如）运动向量的水平分量、运动向量的垂直分量、运动向量的分辨率（例如，四分之一像素精度或八分之一像素精度）、运动向量所指向的参考图片，和 / 或运动向量的参考图片列表。

[0064] 一股来说，TU 用于变换过程和量化过程。具有一或多个 PU 的给定 CU 还可包括一或多个 TU。在预测之后，视频编码器 20 可计算对应于 PU 的残余值。残余值包含像素差值，可使用 TU 将所述像素差值变换成变换系数，进行量化和扫描以产生串行化变换系数以用于熵译码。本发明通常使用术语“视频块”来指代 CU 的译码节点。在一些特定情形下，本发明还可使用术语“视频块”来指代树块（即，LCU），或 CU（其包括译码节点和 PU 和 TU）。

[0065] 视频序列通常包括一系列视频帧或图片。图片群组 (GOP) 通常包含视频图片中的一系列的一或多个视频图片。GOP 可在 GOP 的标头、图片中的一者或一者以上的标头中或在别处包括语法数据，所述语法数据描述包括于 GOP 中的数个图片。图片的每一切片可包括描述相应切片的编码模式的切片语法数据。视频编码器 20 通常对个别视频切片内的视频块进行操作，以便编码视频数据。视频块可对应于 CU 内的译码节点。视频块可具有固定或变化的大小，且可根据指定译码标准而在大小方面不同。

[0066] 作为一实例，HM 支持以各种 PU 大小进行预测。假定特定 CU 的大小为 2N×2N，HM 支持以 2N×2N 或 N×N 的 PU 大小进行帧内预测，且以 2N×2N、2N×N、N×2N 或 N×N 的对称 PU 大小进行帧间预测。HM 还支持以 2N×nU、2N×nD、nL×2N 和 nR×2N 的 PU 大小的不对称分割以进行帧间预测。在不对称分割中，CU 的一个方向未经分割，而另一方向分割成 25％和 75％。CU 的对应于 25％分割区的部分由“n”继之以“上”、“下”、“左”或“右”的指示来指示。因此，例如，“2N×nU”指代在水平方向上经分割而具有顶部的 2N×0.5N PU 和底部的 2N×1.5N PU 的 2N×2N CU。

[0067] 在本发明中，“N×N”与“N 乘 N”可互换地使用以指代视频块在垂直尺寸与水平尺寸方面的像素尺寸，例如，16×16 个像素或 16 乘 16 个像素。一股来说，16×16 块在垂直方向中将具有 16 个像素 (y = 16) 且在水平方向中将具有 16 个像素 (x = 16)。同样地，N×N 块通常在垂直方向上具有 N 个像素，且在水平方向上具有 N 个像素，其中 N 表示非负整数值。可按行和列来布置块中的像素。此外，块未必需要在水平方向中与在垂直方向中具有相同数目个像素。举例来说，块可包含 N×M 个像素，其中 M 未必等于 N。

[0068] 在使用 CU 的 PU 进行帧内预测性或帧间预测性译码之后，视频编码器 20 可计算 CU 的 TU 的残余数据。PU 可包含空间域（还称作像素域）中的像素数据，且 TU 可包含在将（例如）离散余弦变换 (DCT)、整数变换、小波变换或概念上类似的变换等变换应用于残余视频数据之后的变换域中的系数。残余数据可对应于未经编码图片的像素与对应于 PU 的预测值之间的像素差。视频编码器 20 可形成包括 CU 的残余数据的 TU，且接着变换所述 TU 以产生 CU 的变换系数。

[0069] 在应用任何变换以产生变换系数之后，视频编码器 20 可执行变换系数的量化。量化大体上指代如下过程：将变换系数量化以可能地减少用以表示所述系数的数据的量，从

而提供进一步压缩。所述量化过程可减少与所述系数中的一些或全部相关联的位深度。举例来说，可在量化期间将 n 位值下舍入到 m 位值，其中 n 大于 m。

[0070] 在一些实例中，视频编码器 20 和视频解码器 30 可利用预定义扫描次序来扫描经量化的变换系数以产生可经熵编码的串行化向量。在其它实例中，视频编码器 20 和视频解码器 30 可执行自适应扫描。在扫描所述经量化变换系数以形成一维向量之后，或在扫描期间，视频编码器 20 可熵编码所述一维向量，例如，根据上下文自适应可变长度译码 (CAVLC)、上下文自适应二进制算术译码 (CABAC)、基于语法的上下文自适应二进制算术译码 (SBAC)、概率区间分割熵 (PIPE) 译码或另一熵编码方法。视频解码器 30 可熵解码所述系数，执行反量化过程和反变换过程以再现残余数据，且组合所述残余数据与预测性数据以产生经解码视频数据。视频编码器 20 还可熵编码与经编码视频数据相关联的语法元素以供视频解码器 30 用于解码视频数据。

[0071] 为了执行 CABAC，视频编码器 20 和视频解码器 30 可将上下文模型内的上下文指派给待译码的符号。所述上下文可关于（例如）符号的相邻值是否非零。根据本发明的技术，视频编码器 20 和 / 或视频解码器 30 可经配置以基于视频块的变换系数在其中出现的区域来确定用于熵译码（例如，熵编码或熵解码）变换系数的上下文。

[0072] 视频编码器 20 和视频解码器 30 可经配置而具有视频块（例如，变换单元）的各区域的定义。举例来说，视频编码器 20 和视频解码器 30 可经配置而具有各种大小的视频块的区域的定义。在一些实例中，视频编码器 20 可确定借以将视频块划分为区域的方法和表示将如何划分所述块的码数据。所述区域中的每一者可与用于确定出现于相应区域内的变换系数的上下文的相应值和 / 或技术相关联。

[0073] 举例来说，视频块的特定区域可与基于邻域的上下文确定方案相关联，而视频块的另一区域可与基于位置的上下文确定方案相关联。作为另一实例，视频块的一区域可与偏移相关联，所述偏移将被应用于经确定用于位于所述区域中的变换系数的上下文。同一视频块的不同区域可与用于计算上下文的不同偏移值和 / 或不同技术相关联。

[0074] 作为一个实例，视频块可包括两个不同区域 :第一区域，其包括视频块的左上隅角中的一或多个子块（例如，4×4 个变换系数子块）;及第二区域，其包括视频块的未包括于第一区域中的其它子块。更具体来说，视频编码器 20 和视频解码器 30 可确定子块的 x 坐标和 y 坐标，且通过比较 x 与 y 的总和与阈值来确定所述子块是在第一区域中还是在第二区域中。如果 x 与 y 的总和小于所述阈值，则视频编码器 20 和视频解码器 30 可确定所述子块在第一区域中，且否则视频编码器 20 和视频解码器 30 可确定所述子块在第二区域中。视频编码器 20 和视频解码器 30 可基于视频块的系数在第一区域的子块中还是在第二区域的子块中来确定用于所述系数的上下文。

[0075] 举例来说，在一些区域中，所述上下文可为固定上下文，其中视频编码器 20 和视频解码器 30 使用所述固定上下文来译码所述区域中的变换系数。即，视频编码器 20 和视频解码器 30 可将同一上下文应用于所述区域中的所有变换系数。或者，所述区域中的子块中的每一者可与确定上下文的同一方法（例如，固定上下文方法）相关联，但区域中的不同子块可具有不同固定上下文。视频编码器 20 和视频解码器 30 可基于子块在区域中的位置来确定所述子块的固定上下文。作为又一实例，可将固定上下文指派给区域内的个别变换系数位置。即，视频编码器 20 和视频解码器 30 可基于变换系数在视频块、子块和 / 或区域

中的位置来确定用于译码区域内的变换系数的上下文。

[0076] 作为另一实例，在一些区域中，可根据相邻子块来定义上下文模型。举例来说，视频编码器 20 和视频解码器 30 可经配置而具有用于特定区域内的每一子块的上下文的集合。即，区域中的每一子块可与上下文的一相应集合相关联。视频编码器 20 和视频解码器 30 可针对相应子块中的每一变换系数从上下文的集合选择适当上下文。用于一个子块的上下文的集合可不同于用于另一子块的上下文的集合。

[0077] 作为又一实例，用于区域中的每一子块的个别旗标可经译码而表示在对应子块中是否存在任何有效（即，非零）系数。这些旗标可称作经译码的子块旗标。所述旗标可用于选择用于译码子块中的变换系数的上下文。举例来说，视频编码器 20 和视频解码器 30 可基于一或多个相邻子块的旗标的值来确定用于译码子块中的变换系数的上下文。举例来说，所述旗标可具有为 0 或 1 的二进制值，且视频编码器 20 和视频解码器 30 可基于右边相邻的子块和下方相邻的子块（还称作底部相邻的子块）的旗标值的总和来确定用于译码当前子块中的变换系数的上下文。还可将其它公式用于计算子块的上下文。

[0078] 视频编码器 20 和视频解码器 30 可经配置以单独地或按任何组合实施本发明的技术中的任一者或全部。这些技术的一个实例组合为视频编码器 20 和视频解码器 30 可经配置以将变换单元划分为子块（例如，4×4 个像素子块），且接着基于特定变换系数在子块中的位置且基于用于一或多个相邻子块（例如，左边相邻的子块和底部相邻的子块）的经译码块旗标来确定用于译码所述子块的所述变换系数的数据的上下文。

[0079] 视频编码器 20 和视频解码器 30 可经配置以使用在这些各种实例中所确定的上下文来译码表示变换系数的一或多个语法元素。变换系数可包括各种类型的语法元素。举例来说，变换系数可包括 significant_coeff_flag，其指示变换系数是否具有非零值（即，为有效的）。如果变换系数有效，则变换系数可包括指示变换系数的值是大于还是小于 0 的正负号值（例如，coeff_sign_flag），和指示变换系数的绝对值是否大于 1 的值（例如，coeff_abs_level_greater1_flag）。如果变换系数具有大于 1 的绝对值，则变换系数可包括指示变换系数是否具有大于 2 的绝对值的值（例如，coeff_abs_level_greater2_flag）。如果变换系数具有大于 2 的绝对值，则变换系数可包括指示变换系数的绝对值减去二的值（例如，coeff_abs_level_remaining）。

[0080] 视频编码器 20 和视频解码器 30 的 CABAC 译码器可使用根据本发明的技术所确定的上下文来译码这些值中的任一者或全部。额外或替代性地，视频编码器 20 和视频解码器 30 可使用根据本发明的技术所确定的上下文来译码指示最后有效系数的位置的数据（例如，last_significant_coeff_x_prefix、last_significant_coeff_x_suffix、last_significant_coeff_y_prefix 和 last_significant_coeff_y_suffix）。

[0081] 视频编码器 20 和视频解码器 30 可经配置以单独地或按任何组合执行本发明中所描述的技术中的任何一者或一者以上。以下描述用于基于视频块的变换系数在其中出现的区域来确定用于译码视频块的变换系数的上下文且使用所述经确定上下文来熵译码变换系数的各种技术。以下参看图 9 到图 14 来描述所述技术的实例。一般来说，使用经确定上下文来译码变换系数包括使用经确定上下文来译码变换系数的一或多个语法元素。确定上下文大体上包括确定变换系数在其中出现的区域且基于所述区域来确定上下文。举例来说，所述区域可与特定上下文或上下文的集合相关联，和 / 或与用于确定上下文的一或多

种技术相关联。

[0082] 图 2 为说明可实施本发明中所描述的帧间预测技术的实例视频编码器 20 的方框图。视频编码器 20 可执行视频切片内的视频块的帧内译码和帧间译码。帧内译码依赖于空间预测以减少或移除给定视频帧或图片内的视频的空间冗余。帧间译码依赖于时间预测以减少或移除视频序列的邻近帧或图片内的视频的时间冗余。帧内模式（I 模式）可指代若干基于空间的压缩模式中的任一者。例如单向预测（P 模式）或双向预测（B 模式）的帧间模式可指代若干基于时间的压缩模式中的任一者。

[0083] 在图 2 的实例中，视频编码器 20 包括模式选择单元 35、预测处理器 41、参考图片存储器 64、求和器 50、变换处理单元 52、量化单元 54 和熵编码单元 56。预测处理器 41 包括运动估计单元 42、运动补偿单元 44 和帧内预测单元 46。对于视频块重构而言，视频编码器 20 还包括反量化单元 58、反变换单元 60 和求和器 62。还可包括解块滤波器（图 2 中未展示）以对块边界滤波从而移除来自经重构视频的成块效应假影。视需要，所述解块滤波器将通常对求和器 62 的输出滤波。除解块滤波器以外，还可使用额外的环路滤波器（环路内或环路后）。

[0084] 如图 2 中所展示，视频编码器 20 接收视频数据，且模式选择单元 35 将所述数据分割为视频块。此分割还可包括分割为切片、瓦片（tile）或其它较大单元，以及（例如）根据 LCU 和 CU 的四叉树结构的视频块分割。视频编码器 20 大体上说明了编码待编码的视频切片内的视频块的组件。可将所述切片划分为多个视频块（且有可能划分为视频块的集合，其称为瓦片）。预测处理器 41 可基于误差结果（例如，译码速率和失真的水平）针对当前视频块选择多个可能译码模式中的一者，例如，多个帧内译码模式中的一者或多个帧间译码模式中的一者。预测处理器 41 可将所得帧内或帧间译码块提供到求和器 50 以产生残余块数据且提供到求和器 62 以重构经编码块以用作参考图片。

[0085] 预测处理器 41 内的帧内预测单元 46 可执行当前视频块相对于处于与待译码的当前块相同的帧或切片中的一或多个相邻块的帧内预测性译码，以提供空间压缩。预测处理器 41 内的运动估计单元 42 和运动补偿单元 44 执行相对于一或多个参考图片中的一或多个预测性块的当前视频块的帧间预测性译码，以提供时间压缩。

[0086] 运动估计单元 42 可经配置以根据视频序列的预定模式确定视频切片的帧间预测模式。所述预定模式可将序列中的视频切片指定为 P 切片、B 切片或 GPB 切片。运动估计单元 42 和运动补偿单元 44 可高度集成，但出于概念性目的而单独加以说明。由运动估计单元 42 执行的运动估计为产生运动向量的过程，所述运动向量估计视频块的运动。运动向量（例如）可指示当前视频帧或图片内的视频块的 PU 相对于参考图片内的预测性块的位移。

[0087] 预测性块为被发现与待译码的视频块的 PU 在像素差方面压缩紧密匹配的块，可通过绝对差总和 (SAD)、平方差总和 (SSD) 或其它差量度来确定所述像素差。在一些实例中，视频编码器 20 可计算存储于参考图片存储器 64 中的参考图片的子整数像素位置的值。举例来说，视频编码器 20 可内插参考图片的四分之一像素位置、八分之一像素位置或其它分率像素位置的值。因此，运动估计单元 42 可执行相对于全像素位置和分数像素位置的运动搜索，且以分数像素精度输出运动向量。

[0088] 运动估计单元 42 通过比较经帧间译码切片中的视频块的 PU 的位置与参考图片的

预测性块的位置来计算所述 PU 的运动向量。参考图片可选自第一参考图片列表（列表 0）或第二参考图片列表（列表 1），所述列表中的每一者识别存储于参考图片存储器 64 中的一或多个参考图片。运动估计单元 42 将所计算的运动向量发送到熵编码单元 56 和运动补偿单元 44。

[0089] 由运动补偿单元 44 执行的运动补偿可涉及基于由运动估计确定的运动向量来提取或产生预测性块，其有可能执行子像素精度的内插。一旦接收到当前视频块的 PU 的运动向量，运动补偿单元 44 便可在参考图片列表中的一者中找到运动向量所指向的预测性块。运动补偿单元 44 还可产生与视频块和视频切片相关联的语法元素以供视频解码器 30 在解码视频切片的视频块时使用。

[0090] 如以上所描述，作为由运动估计单元 42 和运动补偿单元 44 执行的帧间预测的替代方案，帧内预测单元 46 可对当前块进行帧内预测。具体来说，帧内预测单元 46 可确定用以编码当前块的帧内预测模式。在一些实例中，帧内预测单元 46 可（例如）在单独编码回合期间使用各种帧内预测模式来编码当前块，且帧内预测单元 46（或在一些实例中，为模式选择单元 35）可从经测试模式中选择要使用的适当帧内预测模式。举例来说，帧内预测单元 46 可使用对各种经测试的帧内预测模式的速率 - 失真分析而计算速率 - 失真值，且在经测试模式当中选择具有最佳速率 - 失真特性的帧内预测模式。速率 - 失真分析通常确定经编码块与经编码以产生所述经编码块的原始的未经编码的块之间的失真（或误差）的量，以及用以产生经编码块的位速率（即，位数目）。帧内预测单元 46 可从失真和速率计算各种经编码块的比率以确定哪一帧内预测模式展现块的最佳速率 - 失真值。

[0091] 在任一情形下，在选择用于块的帧内预测模式之后，帧内预测单元 46 可将指示用于块的选定帧内预测模式的信息提供到熵编码单元 56。熵编码单元 56 可根据本发明的技术来编码指示选定帧内预测模式的信息。视频编码器 20 可在经发射的位流配置数据中包括各种块的编码上下文的定义及将用于所述上下文中的每一者的最有可能的帧内预测模式、帧内预测模式索引表和经修改的帧内预测模式索引表的指示，所述位流配置数据可包括多个帧内预测模式索引表和多个经修改的帧内预测模式索引表（还称作码字映射表）。

[0092] 在预测处理器 41 经由帧间预测或帧内预测产生用于当前视频块的预测性块之后，视频编码器 20 通过从当前视频块减去预测性块而形成残余视频块。求和器 50 表示执行此计算的单元。残余块中的残余视频数据可包括于一或多个 TU 中且应用于变换处理单元 52。变换处理单元 52 通常将残余视频数据从像素域转换到变换域（例如，频域）。变换处理单元 52 可使用例如离散余弦变换 (DCT) 或概念上类似的变换的变换将残余视频数据变换为残余变换系数。或者，变换处理单元 52 可将二维 (2D) 变换（在水平方向和垂直方向两者上）应用于 TU 中的残余数据。

[0093] 变换处理单元 52 可将所得变换系数发送到量化单元 54。量化单元 54 量化所述变换系数以进一步减少位速率。所述量化过程可减少与所述系数中的一些或全部相关联的位深度。可通过调整量化参数而修改量化的程度。

[0094] 在量化之后，熵编码单元 56 熵编码经量化的变换系数。举例来说，熵编码单元 56 可执行上下文自适应可变长度译码 (CAVLC)、上下文自适应二进制算术译码 (CABAC)、基于语法的上下文自适应二进制算术译码 (SBAC)、概率区间分割熵 (PIPE) 译码或另一熵编码方法或技术。所述熵编码大体上包括扫描经量化的变化系数（本文中为了简要起见大体上

简称为“变换系数”）一或多次，及在每一扫描期间熵译码用于变换系数的语法元素，例如，指示对应变换系数是否有效、是否具有大于 1 或 2 的绝对值、有效系数的绝对值（或其一部分，例如，大于 2 的部分）和正负号的语法元素。

[0095] 根据本发明的技术，熵编码单元 56 可基于视频块（例如，变换单元）的变换系数在其中出现的区域来确定用于译码（即，熵编码）所述视频块的变换系数的上下文。举例来说，在扫描期间，熵编码单元 56 可确定变换系数在视频块中的位置，且确定所述位置出现于哪一区域中。另外，熵编码单元 56 可包括定义视频块的区域的配置数据。

[0096] 举例来说，熵编码单元 56 可经配置有一阈值。在此实例中，熵编码单元 56 可确定定义变换系数的位置的 x 坐标和 y 坐标是否具有大于所述阈值的总和（即，x+y）。在此实例中，第一区域对应于 x 坐标值和 y 坐标值的总和小于所述阈值的变换系数，且第二区域对应于 x 坐标值和 y 坐标值的总和大于或等于所述阈值的变换系数。或者，可使用多个阈值来定义多个区域。在图 9B 中展示以此方式定义的区域的实例，以下更详细地描述图 9B。

[0097] 作为另一实例，熵编码单元 56 可经配置以确定子块（包括变换系数）在视频块中的位置。子块可对应于 4×4 变换系数子块。即，视频块可包括多个非重叠子块，每一非重叠子块具有相同的大小，例如，4×4 个变换系数。为了确定子块的区域，熵编码单元 56 可比较子块（例如，子块的特定变换系数，例如，子块的左上变换系数）的 x 坐标和 y 坐标的总和与阈值。x 坐标和 y 坐标的总和是否小于阈值可指示子块的变换系数是包括于第一区域还是第二区域中。

[0098] 举例来说，令 Cij 表示具有在位置 (i,j) 处的左上变换系数的子块的位置，其中 x ＝ i 和 y ＝ j。另外，令 T 定义所述阈值。熵编码单元 56 可使用以下伪码来确定子块的变换系数在其中出现的区域：

[0099] (i+j ＜ T) ? 区域 1 :区域 2。

[0100] 在此实例中，当 i+j 小于 T 时（即，子块的 x 坐标和 y 坐标的总和小于阈值），熵编码单元 56 确定子块的所有变换系数出现于区域 1 中，而当 i+j 大于或等于 T 时（即，子块的 x 坐标和 y 坐标的总和大于或等于阈值），熵编码单元 56 确定子块的所有变换系数出现于区域 2 中。以下参看图 9 到图 14 更详细地描述区域的这些和其它实例。

[0101] 熵编码单元 56 可经配置而以各种方式基于区域来确定上下文。举例来说，熵编码单元 56 可使用变换系数在视频块中的位置或变换系数在其中出现的 4×4 子块的位置、基于变换系数在其中出现的区域来确定用于译码变换系数的上下文。

[0102] 或者，可根据相邻的 4×4 子块来定义上下文模型。举例来说，熵编码单元 56 可将可用上下文的相应集合指派给每一 4×4 子块，且（例如）基于变换系数在子块中的位置来为子块中的待译码的当前变换系数选择上下文中的一者。可将上下文的集合指派给相应子块，以使得每一子块可具有可用上下文的不同集合。作为又一实例，熵编码单元 56 可将上下文计算为 ctx ＝ Right4x4SubBlockFlag+Bottom4x4SubBlockFlag。在此情形下，Right4x4SubBlockFlag 表示用于右边相邻的子块的经译码的子块旗标，而 Bottom4x4SubBlockFlag 表示用于底部相邻的经译码的子块旗标的经译码的子块旗标。

[0103] 在一些实例中，熵编码单元 56 可将偏移应用于用于熵编码变换系数的经确定上下文，且可进一步基于变换系数在其中出现的区域来确定待应用的偏移。即，熵编码单元 56 可以相同的通用方式计算用于两个或两个以上区域的系数的基本上下文，但不同区域可具

有不同的对应偏移值。因此，熵编码单元 56 可基于区域所映射到的偏移（即，与所述区域相关联的偏移）将偏移应用于经计算的上下文值。

[0104] 熵编码单元 56 可确定变换系数是否为 DC（直流）变换系数（通常存在于变换块的左上隅角中），且基于变换系数在其中出现的区域以及所述变换系数是否为 DC 变换系数来选择用于译码变换系数的上下文。举例来说，熵编码单元 56 可使用用于专用位置的共享上下文来确定用于变换系数的上下文。即，所述共享上下文可包含应用于出现在特定位置处（例如，子块的左上隅角）的所有变换系数的同一上下文。因此，所述共享上下文可进一步包括待在译码 DC 变换系数而非出现于其它子块的左上位置处的非 DC 变换系数时应用的特定上下文的指示。

[0105] 额外或替代性地，共享上下文可包含在不同大小的块之间的用于出现于所述块的特定位置处的变换系数的共享上下文。举例来说，熵编码单元 56 可经配置以在译码任何大小（例如，4×4、8×8、16×16 或其类似者）的视频块（例如，TU）的 DC 变换系数时应用同一上下文。即，熵编码单元 56 可包括将 DC 变换系数（对于任何大小的块）映射到用于译码 DC 变换系数的同一上下文数据的数据。换句话说，熵编码单元 56 可经配置以使用经确定用于 DC 变换系数的上下文来译码 DC 变换系数，而不考虑正被译码的当前视频块的大小。通常，DC 变换系数为视频块的左上系数。

[0106] 在通过熵编码单元 56 进行的熵编码之后，可将经编码的位流传输到视频解码器 30，或经存档以用于稍后传输或由视频解码器 30 检索。熵编码单元 56 还可熵编码运动向量、帧内模式指示和正被译码的当前视频切片的其它语法元素。

[0107] 反量化单元 58 和反变换单元 60 分别应用反量化和反变换，以在像素域中重构残余块以供稍后用作参考图片的参考块。运动补偿单元 44 可通过将残余块加到参考图片列表中的一者内的参考图片中的一者的预测性块来计算参考块。运动补偿单元 44 还可将一或多个内插滤波器应用于经重构的残余块以计算子整数像素值以用于运动估计中。求和器 62 将经重构的残余块添加到由运动补偿单元 44 产生的运动补偿预测块，以产生参考块以供存储于参考图片存储器 64 中。参考块可由运动估计单元 42 和运动补偿单元 44 用作参考块以对后续视频帧或图片中的块进行帧间预测。

[0108] 以此方式，视频编码器 20 表示视频译码器的一实例，其经配置以基于视频块的变换系数在其中出现的区域来确定用于译码视频块的变换系数的上下文，且使用经确定上下文来熵译码所述变换系数。所述区域可包含第一区域和第二区域中的一者，所述第一区域包含视频块的变换系数的一或多个左上 4×4 子块，所述第二区域包含视频块的在所述第一区域外部的变换系数。

[0109] 图 3 为说明可实施本发明中所描述的帧间预测技术的实例视频解码器 30 的方框图。在图 3 的实例中，视频解码器 30 包括熵解码单元 80、预测处理器 81、反量化单元 86、反变换单元 88、求和器 90 和参考图片存储器 92。预测处理器 81 包括运动补偿单元 82 和帧内预测单元 84。在一些实例中，视频解码器 30 可执行与参考来自图 2 的视频编码器 20 所描述的编码回合大体上互逆的解码回合。

[0110] 在解码过程期间，视频解码器 30 从视频编码器 20 接收表示经编码的视频切片的视频块和相关联语法元素的经编码的视频位流。视频解码器 30 的熵解码单元 80 熵解码所述位流以产生经量化的系数、运动向量和其它语法元素。熵解码单元 80 将运动向量、帧内

模式指示和其它预测相关语法元素转发到预测处理器 81。熵解码单元 80 将经量化的系数以块（例如，TU）的形式转发到反量化单元 86。视频解码器 30 可接收视频切片层级和 / 或视频块层级的语法元素。

[0111] 具体来说，根据本发明的技术，熵解码单元 80 可基于块的变换系数在其中出现的区域来确定用于熵解码变换系数的上下文。具体来说，一旦变换系数位于块内，熵解码单元 80 便可基于块的变换系数将在其中出现的区域来确定所述上下文。熵解码单元 80 可经配置以如以下参看图 9 到图 14 所解释来确定所述区域或其它所述区域。举例来说，如图 9A 中所展示，熵解码单元 80 可经配置以确定一变换系数将出现于包括块的左上隅角中的一或多个子块的第一区域中还是出现于包括在所述第一区域外部的子块的第二区域中，且基于所述变换系数将出现于第一区域还是第二区域中来确定上下文。

[0112] 同样，熵解码单元 80 可基于区域来确定上下文，这是因为熵解码单元 80 可通过用于计算或确定与每一区域中的系数相关联的上下文的一或多个各种技术进行配置。即，每一区域可与用于计算或确定上下文的一或多种技术相关联。举例来说，一区域可与在一或多个变换系数之间共享的上下文相关联。作为另一实例，一区域可与在所述区域的子块之间共享的上下文相关联。作为又一实例，一区域可与一偏移值相关联，所述偏移值将应用于经计算用于所述区域中的变换系数的上下文值。熵解码单元 80 可经配置以基于变换系数在其中出现的区域使用如本文中所描述的这些或其它技术来确定用于解码变换系数的上下文。熵解码单元 80 可接着使用经确定上下文来熵解码变换系数。

[0113] 额外或替代性地，共享上下文可包含在不同大小的块之间的用于出现于所述块的特定位置处的变换系数的共享上下文。举例来说，熵解码单元 80 可经配置以在译码任何大小（例如，4×4、8×8、16×16 或其类似者）的视频块（例如，TU）的 DC 变换系数时应用同一上下文。即，熵解码单元 80 可包括将 DC 变换系数（对于任何大小的块）映射到用于译码 DC 变换系数的同一上下文数据的数据。换句话说，熵解码单元 80 可经配置以使用经确定用于 DC 变换系数的上下文来译码 DC 变换系数，而不考虑正被译码的当前视频块的大小。通常，DC 变换系数为视频块的左上系数。

[0114] 当视频切片经译码为帧内译码（I）切片时，预测处理器 81 的帧内预测单元 84 可基于用信号通知的帧内预测模式和来自当前帧或图片的先前解码块的数据而产生用于当前视频切片的视频块的预测数据。在视频帧经译码为帧间译码（即，B、P 或 GPB）切片时，预测处理器 81 的运动补偿单元 82 基于运动向量和从熵解码单元 80 接收的其它语法元素而产生针对当前视频切片的视频块的预测性块。预测性块可从参考图片列表中的一者内的参考图片中的一者产生。视频解码器 30 可基于存储于参考图片存储器 92 中的参考图片使用默认建构技术来建构参考帧列表（列表 0 和列表 1）。

[0115] 运动补偿单元 82 通过解析运动向量和其它语法元素而确定当前视频切片的视频块的预测信息，且使用所述预测信息以产生针对正被解码的当前视频块的预测性块。举例来说，运动补偿单元 82 使用所接收的语法元素中的一些来确定用以译码视频切片的视频块的预测模式（例如，帧内预测或帧间预测）、帧间预测切片类型（例如，B 切片、P 切片或 GPB 切片）、切片的参考图片列表中的一者或一者以上的建构信息、切片的每一经帧间编码视频块的运动向量、切片的每一经帧间译码视频块的帧间预测状态，及用以解码当前视频切片中的视频块的其它信息。

[0116] 运动补偿单元 82 还可基于内插滤波器执行内插。运动补偿单元 82 可使用如由视频编码器 20 在视频块的编码期间使用的内插滤波器来计算参考块的子整数像素的内插值。在此情形下，运动补偿单元 82 可从所接收的语法元素确定由视频编码器 20 使用的内插滤波器且使用所述内插滤波器来产生预测性块。

[0117] 反量化单元 86 将位流中所提供且由熵解码单元 80 解码的经量化的变换系数反量化（即，解量化）。反量化过程可包括使用由视频编码器 20 针对视频切片中的每一视频块计算的量化参数以确定量化程度及（同样）应被应用的反量化的程度。反变换单元 88 将反变换（例如，反 DCT、反整数变换或概念上类似的反变换过程）应用于变换系数，以便在像素域中产生残余块。

[0118] 在一些情形下，反变换单元 88 可将二维 (2D) 反变换（在水平方向和垂直方向两者上）应用于系数。根据本发明的技术，反变换单元 88 可替代地应用水平一维 (1D) 反变换、垂直 1D 反变换，或不将变换应用于 TU 中的每一者中的残余数据。可将在视频编码器 20 处应用于残余数据的变换的类型用信号通知给视频解码器 30 以将适当类型的反变换应用于变换系数。

[0119] 在运动补偿单元 82 基于运动向量和其它语法元素产生当前视频块的预测性块之后，视频解码器 30 通过对来自反变换单元 88 的残余块与由运动补偿单元 82 产生的对应预测性块求和而形成经解码的视频块。求和器 90 表示执行此求和运算的（若干）组件。视需要，还可应用解块滤波器来对经解码块滤波以便移除成块效应假影。其它环路滤波器（在译码环路中或在译码环路后）也可用以使像素转变平滑，或以其它方式改进视频质量。接着将给定帧或图片中的经解码的视频块存储于参考图片存储器 92 中，参考图片存储器 92 存储用于后续运动补偿的参考图片。参考图片存储器 92 还存储经解码视频以用于稍后在显示装置（例如，图 1 的显示装置 32）上呈现。

[0120] 以此方式，视频解码器 30 表示视频译码器的一实例，其经配置以基于视频块的变换系数在其中出现的区域来确定用于译码视频块的变换系数的上下文，且使用经确定上下文来熵译码所述变换系数。所述区域可包含第一区域和第二区域中的一者，所述第一区域包含视频块的变换系数的一或多个左上 4×4 子块，所述第二区域包含视频块的在所述第一区域外部的变换系数。

[0121] 图 4 为说明视频块中的变换系数与和所述视频块相关联的有效性映射之间的关系的概念图。如图 4 中所说明，有效性映射包括“1”以指示视频块中的有效系数值（即，大于零的值）的每一实例。可在位流中用信号通知有效性映射，其可由视频解码器（例如，视频解码器 30）解码以确定待解码的视频块中的有效（即，大于零）系数的位置。更具体来说，可在位流中用信号通知视频块内的最后非零系数的位置。视频块中的最后非零系数的位置取决于用于视频块的扫描次序。可用信号通知额外语法元素以根据已知或可知晓的扫描次序相对于最后非零系数来指示其它有效系数。

[0122] 图 5A 到图 5D 为说明使用 z 字形扫描次序、水平扫描次序、垂直扫描次序和对角线扫描次序所扫描的视频数据的块的实例的概念图。如图 5A 到图 5D 中所展示，视频数据的 8×8 块（例如，CU 的 TU）可包括在对应块位置中的六十四个变换系数（以圆圈表示）。在此实例中，块 100、102、104 和 106 各自具有 8×8 的大小，且因此包括使用先前所描述的预测技术所产生的六十四个变换系数。

[0123] 根据本发明中所描述的技术，块 100、102、104 和 106 中的每一者中的六十四个变换系数可能已使用 2D 变换、水平 1D 变换和垂直 1D 变换中的一者进行了变换或可被反变换，或变换系数可根本不经变换。无论是否经变换，视频块 100、102、104 和 106 中的每一者中的系数均使用 z 字形扫描次序、水平扫描次序、垂直扫描次序和对角线扫描次序中的一者进行扫描以准备进行熵译码。

[0124] 如图 5A 中所展示，与块 100 相关联的扫描次序为 z 字形扫描次序。所述 z 字形扫描次序使得视频译码器（例如，视频编码器 20 或视频解码器 30）以如由图 5A 中的箭头所指示的对角线方式扫描块 100 的经量化的变换系数。在图 5D 中，类似地，对角线扫描次序使得视频译码器以如由图 5D 中的箭头所指示的对角线方式来扫描块 106 的经量化的变换系数。如图 5B 和图 5C 中所展示，与块 102 和 104 相关联的扫描次序分别为水平扫描次序和垂直扫描次序。水平扫描次序使得视频译码器以水平逐行或“光栅”方式扫描块 102 的经量化的变换系数，而垂直扫描次序使得视频译码器以垂直逐行或“旋转光栅”方式扫描块 104 的经量化的变换系数（还如由图 5B 和图 5C 中的箭头所指示）。

[0125] 在其它实例中，如以上所描述，块可具有小于或大于块 100、102、104 和 106 的大小的大小，且可包括更多或更少的经量化的变换系数和对应的块位置。在这些实例中，与特定块相关联的扫描次序可使得视频译码器以实质上与图 5A 到图 5D 的 8×8 块的实例中所展示的方式类似的方式来扫描块的经量化的变换系数，例如，可遵循先前所描述的扫描次序中的任一者来扫描 4×4 块或 16×16 块。

[0126] 尽管图 5A 到图 5D 中的扫描方向大体上被展示为从低频率系数进行到高频率系数，但在其它实例中，视频编码器 20 和视频解码器 30 可经配置以执行反向扫描次序，其中扫描可从高频率系数进行到低频率系数。即，视频编码器 20 和视频解码器 30 可以与图 5A 到图 5D 中所展示的次序相反的次序来扫描系数。

[0127] 图 6 为说明划分为子块以用于变换系数译码的实例视频块 110 的概念图。在当前 HM 中，将子块概念用于变换系数译码。视频译码器可将大于经确定的子块大小的任何变换单元 (TU) 再分为子块。举例来说，将视频块 110 划分为四个 4×4 子块。

[0128] 在图 6 的经说明实例中，视频译码器将视频块 110 划分为 4×4 子块。在其它实例中，视频译码器可将视频块划分为其它大小的子块，例如，8×8、16×16 等。如果视频译码器将同一子块大小用于帧或切片的所有 TU，则归因于通过子块大小实现的均一性，可在硬件实施中实现增益。举例来说，所有处理可在所述子块中分裂，而与 TU 大小无关。然而，对于执行本发明的技术而言，均一的子块大小并非必要的。

[0129] 对于系数译码而言，视频译码器可使用对角线扫描次序来扫描视频块 110 的每一 4×4 子块，如图 6 中所展示。在一些实例中，视频译码器可使用统一扫描来扫描每一子块的变换系数。在此情形下，将同一扫描次序用于有效性信息，即，有效性映射、系数水平、正负号等。在第一实例中，如图 6 中所展示，视频译码器可使用对角线扫描来扫描变换系数。在另一实例中，视频译码器可以与图 6 中所展示的次序相反的次序来扫描变换系数，例如，从右下隅角开始且进行到左上隅角的反向对角线扫描。在其它实例中，视频译码器可使用 z 字形、水平或垂直扫描来扫描变换系数。其它扫描方向 / 定向也是可能的。

[0130] 为了解释简单起见，本发明将视频块的子块描述为 4×4 子块。然而，本发明的技术还可应用于不同大小（例如，8×8、16×16 等）的子块。针对每一 4×4 块，译码

significant_coeffgroup_flag，且如果在子块中存在至少一个非零系数，则此旗标被设定为一，否则所述旗标等于零。如果对于给定子块而言 significant_coeffgroup_flag 非零，则以反向对角线次序来扫描 4×4 子块，且针对所述子块的每一系数来译码 significant_coeff_flag 以指示系数的有效性。这些旗标的群组可称作视频块的有效性映射。在某一实例中，替代于明确地用信号通知所述有效性映射，可使用相邻的 4×4 子块旗标或在 4×4 子块含有最后系数或 DC 系数时隐含地导出 significant_coeffgroup_flag。系数的绝对值也经译码，即，系数水平。

[0131] 尽管将图 6 的扫描的方向大体上展示为从低频率系数进行到高频率系数，但在其它实例中，视频编码器 20 和视频解码器 30 可经配置以执行反向扫描次序，其中所述扫描可从高频率系数进行到低频率系数。即，视频编码器 20 和视频解码器 30 可以与图 6 中所展示的次序相反的次序来扫描系数。

[0132] 图 7 为说明实例五点支持邻域的概念图，所述五点支持邻域用以定义用于选择使用反向对角线扫描次序扫描的视频块 112 中的系数的有效性映射的上下文的上下文模型。如以上所提及，对于上下文自适应译码而言，可基于描述变换系数具有为 0 的值或为 1 的值的概率的上下文模型来译码变换系数。关于有效性映射译码，上下文模型描述特定变换系数是否有效（即，非零）的概率。

[0133] 对于有效性映射译码而言，五点支持 S 可用以定义用以译码视频块 112 的变换系数的有效性映射的上下文模型。五点支持可称作“上下文支持邻域”或简单地称作“支持邻域”。即，视频译码器可依据所述支持来确定当前位置的有效性为一或零的概率。所述上下文支持邻域定义可用作用于译码当前系数的上下文的相邻系数（例如，其可包括有效性信息）。根据本发明的一些实例，所述上下文支持邻域可针对块或子块内的不同系数位置而不同。

[0134] 在图 7 中所展示的实例中，相对于由被圆圈环绕的圆点表示的当前或“目标”位置，五点支持 S 由被正方形环绕的圆点表示。可将上下文模型 Ctx（以下等式 (1)）定义为所述支持的每一点中的有效旗标的总和，其中如果对应变换系数非零，则可将有效性旗标设定为“1”，且否则便将有效性旗标设定为“0”。

[0135]

$$Ctx = \sum_{p \in S} (coef_p != 0) \qquad (1)$$

[0136] 因此，有效性旗标计数可少于或等于支持基数。ctx 的值未必为原始上下文值，但可以偏移的形式应用于基本上下文值，以导出待用以译码针对特定系数的数据的上下文。

[0137] 然而，当并行地计算用于一个以上变换系数（例如，与变换系数相关联的有效性信息）的上下文时（称作“并行有效性上下文计算”或简单地称作“并行上下文计算”），图 7 中所展示的支持 S 可并不合适。举例来说，使用图 7 中所展示的支持 S 可妨碍视频译码器并行地计算用于有效性信息的上下文的能力，因为支持 S 中的所有数据必须可用（例如，已经译码）以使得能够进行上下文的并行计算。在一些情况下，如以下关于图 8A 所描述，译码器可能被迫等候支持 S 中的一支持元素完成译码，之后才确定用于支持 S 中的另一支持元素的上下文。此延迟减少了视频译码器有效地处理有效性信息的能力。

[0138] 图 8A 和图 8B 为说明五点支持内的上下文相依性的概念图。举例来说，为了计算

带圆圈位置的有效性上下文，可能有必要解析由棱形（展示于图 8A 中）描绘的支持 S 内的位置的有效性旗标。如果要求并行地计算两个系数的有效性上下文，则所述解析可引入延迟，这是因为在扫描次序中所述棱形被定位成紧接在带圆圈元素之前。即，无法与由棱形标记的位置同时地计算带圆圈位置的上下文，这是因为带圆圈位置取决于由棱形标记的位置，且因此，必须在确定用于带圆圈位置的上下文之前译码由棱形标记的位置。

[0139] 为了解决此相依性，可从支持 S 移除特定元素，使得所述支持具有所谓的“洞”（由三角形环绕的未填充的圆点，展示于图 8B 中）。举例来说，洞中的有效性旗标被跳过且在上下文计算时不加以考虑（即，假定为零）。因此，不需要解析洞位置中的有效性旗标。5 点支持形状取决于位置以允许实现较佳的并行处理。

[0140] 图 9A 和图 9B 为说明一视频块到两个或两个以上区域的实例划分的概念图。在当前 HM 中，通过 5 点支持将邻域上下文模型化用于大于 8×8（即，16×16、32×32 和非正方形变换大小 16×4、4×16、32×8 和 8×32）的 TU 大小。然而，具有 5 点支持的上下文模型化可增大较大块大小的上下文计算的复杂性。图 9A 的区域 R1 表示包括视频块的变换系数的一或多个左上 4×4 子块的区域的一实例，而图 9A 的区域 R2 表示包括视频块的在区域 R1 外部的变换系数的区域的一实例。图 9A 还表示多个区域包含一或多个子块的相应集合的实例。

[0141] 根据本发明中所描述的技术，视频译码器（例如，视频编码器 20 或视频解码器 30）可将视频块划分为区域 R（例如，如图 9A 和图 9B 中所展示）且针对不同区域中的每一者使用不同上下文指派程序。举例来说，一些区域可使用固定的或基于位置的上下文且一些区域可使用基于邻域的上下文。如图 9A 中所说明，所述区域可基于 4×4 子块以使得在一个区域或另一区域中包括整个子块。而且，在一些实例中，划分为区域可为灵活的。如图 9B 中所说明，视频块可在对角线方向上划分为区域以使得子块的部分可包括于两个不同区域中。在其它实例中，所述划分可能视系数位置或含有此系数的 4×4 子块的位置而定。

[0142] 在一些实例中，可根据视频块中的系数位置或根据含有此系数的 4×4 子块的位置来定义上下文。或者，可能根据相邻的 4×4 子块来定义上下文模型。举例来说，同一 4×4 子块内的每一系数可使用一或若干个上下文，下一 4×4 子块的系数也可使用一或若干个上下文。然而，一个 4×4 子块的上下文可能不同于先前基于 4×4 子块的上下文。或者，可能将上下文计算为 Ctx = Right4x4SubBlockFlag+Bottom4x4SubBlockFlag，或视邻域而定的类似公式。再次，Right4x4SubBlockFlag 可表示右边相邻的子块的经译码的子块旗标（例如，指示右边相邻的 4×4 子块是否包括至少一个非零系数），且 Bottom4x4SubBlockFlag 可表示右边相邻的子块的经译码的子块旗标（例如，指示底部相邻的 4×4 子块是否包括至少一个非零系数）。

[0143] 图 10 为说明针对视频块的每一区域的基于邻域或基于位置的上下文的实例指派的概念图。如图 10 中所说明，还可能使用混合类型的上下文，例如，对于一些区域而言上下文可为基于邻域的，且对于同一视频块的一些区域而言上下文可为固定的或基于位置的。基于位置的方法的可能优点在于不必按逐系数的方式来计算上下文。替代地，视频译码器可针对区域中的所有系数计算上下文一次，使得区域中的所有系数具有相同上下文。图 10 表示多个区域包含一或多个子块的相应集合的实例。

[0144] 对于具有坐标 (x,y) 的系数而言，可根据系数位置来定义区域。举例来说，如果条

件（x+y ＞＝阈值）为真，则视频译码器可确定对应系数出现于区域 R2 内；否则，如果所述条件并不为真，则视频译码器确定对应系数出现于区域 R1 内。类似地，可基于 4×4 子块将坐标指派给区域。对于具有 (X,Y) 坐标的子块而言，可根据 4×4 子块位置来定义区域。举例来说，如果条件 (X+Y ＞＝阈值）为真，则视频译码器可确定对应系数出现于区域 R2 内；否则，视频译码器可确定对应系数出现于区域 R1 内。阈值可固定到某一预定义值（例如，等于 4、5、6、7 或 8 的整数），或可视视频块（例如，TU）大小而定。

[0145] 以此方式，图 10 表示如下实例：视频译码器可经配置以基于变换系数在其中出现的区域、使用基于所述区域的基于位置的上下文信息和基于邻域的上下文信息中的一者来确定用于译码变换系数的上下文。具体来说，如果变换系数在第一区域中，则视频译码器可使用第一上下文确定方法来确定用于译码变换系数的上下文。如果变换系数在第二区域中，则视频译码器可使用第二上下文确定方法来确定用于译码变换系数的上下文，其中第二上下文确定方法不同于第一上下文确定方法且第一区域不同于第二区域。在一实例中，第一区域和第二区域不重叠。再一次，第一和第二上下文确定方法的实例包括使用基于位置的上下文信息和基于邻域的上下文信息。

[0146] 图 11 为说明针对视频块的每一区域的上下文偏移的实例指派的概念图。对于不同区域而言上下文模型可为单独的，但仍将同一方法用于上下文计算。换句话说，可通过用于计算用于译码变换系数的上下文的一种方法配置视频译码器，但其可包括基于变换系数在其中出现的区域所确定的不同上下文模型。

[0147] 举例来说，可基于邻域来计算上下文，但对于不同区域，所述上下文使用偏移。针对每一区域的偏移可为固定的或取决于视频块大小、视频块或子块中的系数位置和视频块中的子块位置中的一者或一者以上。图 11 的区域 R1 表示包括视频块的变换系数的一或多个左上 4×4 子块的区域的另一实例，而图 11 的区域 R2 表示包括视频块的在区域 R1 外部的变换系数的区域的另一实例。图 11 也是多个区域包含一或多个子块的相应集合的实例。

[0148] 通过偏移，可根据等式 (2) 来计算上下文。

[0149]

$$Ctx = offset\,(region) + \sum_{p \in S} (coef_p != 0) \qquad (2)$$

[0150] 或者，视频译码器可根据将 Ctx 用作输入的函数（例如，Ctx ＝ (Ctx+1) ＞＞ 1）来计算上下文。

[0151] 在图 11 中展示基于区域的偏移的一个实例，其中区域 R1 和 R2 是基于 4×4 子块进行定义且对于区域 R1 和 R2 而言偏移不同。偏移值 offset1 和 offset2 可为任何整数，例如，offset1 ＝ 0，offset2 ＝ 3。在其它实例中，到区域的其它划分也是有可能的，且到两个以上区域的划分也是有可能的。

[0152] 图 12 为说明基于与现有上下文模型相关的 TU 大小的视频块到两个或两个以上区域的实例内嵌式划分的概念图。因为在当前 HM 中存在若干大小的 TU(4×4、8×8、16×16 和 32×32)，所以可使用内嵌式样的划分根据较小 TU 大小来进行较大块的划分，如图 12 中所说明。对于内嵌式划分而言，上下文计算的方法可共享且上下文模型自身可共享。

[0153] 举例来说，对于 TU 大小 32×32 而言，在表示 4×4TU 的区域 R1 中，上下文计算可使用与用于大小为 4×4 的实际 TU 的方法相同的用于上下文计算的方法。另外，可在大小

为 4×4 的 TU 与大小为 32×32 的 TU 的 R1 之间共享上下文模型，或可将偏移应用于大小为 4×4 的 TU 的上下文模型。到于 R2，可在大小为 8×8 的 TU 与大小为 32×32 的 TU 的 R2 之间共享上下文计算方法。R3 表示 16×16TU 区域，而 R4 表示 32×32TU 区域。此方法的可能优点在于：可将相同单元用于上下文计算，且可考虑到内嵌区域与 TU 之间的额外相关性。

[0154] 或者，使用内嵌式划分，可在所有 TU 或 TU 的某一群组之间针对专用位置共享一些有效性映射上下文模型。举例来说，可在具有从 4×4 到 32×32 的大小的所有 TU 之间共享对应于 DC 系数的上下文模型。作为另一实例，可在所有 TU 之间共享关于高频率系数的上下文模型。在这些情形下，大小为 32×32 的 TU 中的区域 R1（表示 4×4TU）可与具有大小 4×4、8×8、16×16、32×32 等中的任一者的 TU 使用相同的用于 DC 系数和 / 或高频率系数的上下文模型。

[0155] 作为另一实例，替代在所有 TU 之间进行共享，可仅在所有 TU 的子集或群组之间共享以上所描述的系数（例如，DC 和 / 或高频率系数）的上下文模型。举例来说，可仅在两种大小的 TU（例如，4×4TU 和 8×8TU）之间共享系数的上下文模型。在此情形下，大小为 32×32 的 TU 中的区域 R1（表示 4×4TU）可与具有大小 4×4 和 8×8 的 TU 使用相同的用于 DC 系数和 / 或高频率系数的上下文模型。

[0156] 以此方式，图 12 的实例表示视频译码器（例如，视频编码器 20 或视频解码器 30）可经配置以从视频块的多个区域中确定变换系数在其中出现的区域的实例，其中所述区域中的每一者对应于多个变换单元 (TU) 大小中的相应一者，且其中视频译码器通过选择在所述区域和与所述区域具有相同大小的 TU 之间共享的上下文而确定上下文。

[0157] 图 12 还表示视频译码器（例如，视频编码器 20 或视频解码器 30）可经配置以从视频块的多个区域中确定变换系数在其中出现的区域的实例，其中所述区域中的每一者对应于多个变换单元 (TU) 大小中的相应一者，且其中为了确定上下文，视频译码器选择在不同大小的两个或两个以上 TU 之间共享的用于变换系数的专用位置的上下文，其中所述区域具有与不同大小的两个或两个以上 TU 中的一者相同的大小。用于变换系数的专用位置的所述共享上下文可包含在不同大小的两个或两个以上 TU 之间共享的用于 DC 系数和高频率系数中的一者的上下文。额外或替代性地，用于变换系数的专用位置的所述共享上下文可包含在具有 4×4 变换系数的大小的第一 TU 与具有 8×8 变换系数的大小的第二 TU 之间共享的上下文。

[0158] 图 13A 和图 13B 为说明视频块到两个或两个以上区域的实例划分的概念图。以与以上关于区域是基于正方形（例如，4×4）子块的实例所描述的方式类似的方式，本发明的技术还描述用以基于矩形形状的子块将视频块（例如，TU）划分为两个或两个以上区域的分类方法。举例来说，可依据如图 13A 和图 13B 中所展示的系数扫描而将 2×8 和 8×2 子块用于 8×8 视频块。在此实例中，视频译码器将水平扫描应用于在图 13A 中所展示的块中的系数，且将垂直扫描应用于图 13B 中所展示的块。在图 13A 和图 13B 中所说明的实例中，一个正方形块表示单一系数，且整个视频块的大小为 8×8。

[0159] 根据本发明的技术，视频块可划分为不同矩形区域，例如，R1、R2、R3 和 R4。不同矩形区域中的每一者可具有不同上下文指派。举例来说，对于一些区域而言，可使用固定上下文。这些区域可基于以上所描述且在图 13A 和图 13B 中所展示的矩形（例如，2×8 或 8×2）子块而形成。举例来说，可根据视频块中的系数位置或根据含有这是数的矩形子块的位置

来定义上下文。

[0160] 或者，可能根据相邻的矩形形状的子块来定义上下文模型。举例来说，同一矩形子块内的每一系数可使用一或若干个上下文。另外，相邻矩形子块的系数还可使用一或若干个上下文。然而，一个矩形子块的上下文可不同于先前基于矩形子块的上下文。还可能使用混合类型的上下文，例如，对于一些区域而言上下文可为基于邻域的，且对于同一视频块的一些区域而言上下文可为固定的或基于位置的。基于位置的方法的优点在于，不必逐系数地计算上下文，可针对一区域一次性地进行。而且，所述划分可能视系数位置或含有这是数的矩形子块的位置而定。

[0161] 对于具有 (x,y) 坐标的系数而言，可根据系数位置来定义区域。举例来说，如果条件 (x+y ＞＝阈值) 为真，则可将此系数指派给区域 R2 ；否则，可将其指派给区域 R1。以类似方式，可基于矩形形状的子块进行此操作，对于具有 (X, Y) 坐标的子块而言，可根据矩形子块位置来定义区域。举例来说，如果条件 (X+Y ＞＝阈值) 为真，则可将此系数指派给区域 R2，否则，可将其指派给 R1。所述阈值可固定到某一预定义值 (如整数 (例如，等于 0 或 1)) 或可能视 TU 大小而定。

[0162] 或者，对于不同区域而言上下文模型可不同，但仍将同一方法用于上下文计算。举例来说，可基于邻域来计算上下文，但对于不同区域，所述上下文使用偏移。偏移可为固定的、视视频块大小而定的或视以下各者中的一者或一者以上而定 ：在视频块和 / 或矩形子块中的系数位置、含有当前系数的矩形子块在视频块中的位置，或这些条件的任何组合。

[0163] 通过偏移，可根据等式 (3) 来计算上下文。

[0164]

$$Ctx = offset\,(region) + \sum_{p \in S} (coef_p != 0) \qquad (3)$$

[0165] 或者，可根据将 Ctx 用作输入的函数 (例如，Ctx ＝ (Ctx+1) ＞＞ 1) 来计算上下文。

[0166] 图 14A 和图 14B 为说明针对视频块的每一区域的上下文偏移的实例指派的概念图。在这些实例中，区域 R1 和 R2 是基于矩形子块和扫描方向来定义，且对于区域 R1 和 R2 而言偏移不同。偏移值 offset1 和 offset2 可为任何整数，例如，offset1 ＝ 0，offset2 ＝ 3。到区域的其它划分也是有可能的。举例来说，区域的数目可大于二。应注意，视系数扫描方向而定，在本发明中使用 2×8 和 8×2 矩形子块作为一实例。在无限制的情况下，可将类似方法用于具有大小 M×N 的其它矩形形状子块。

[0167] 一股来说，本发明描述视频块的基于对角线、基于正方形 (例如，4×4) 子块和基于矩形 (例如，2×8 和 8×2) 子块的划分。在其它实例中，其它类型的划分为可能的，且划分可基于具有不同大小的各种形状 (例如，矩形、正方形、三角形等) 而为灵活的。本发明还描述将视频块划分为任何数目个区域。本发明进一步描述基于正方形子块、矩形子块或基于例如视频块的对角线划分的其它分组而将系数分组为区域。以上所描述的阈值和偏移还被提供作为一实例，可利用其它值或相邻相依性。

[0168] 如本发明中所描述的类似技术可用于非正方形的变换单元或其它形状的单元。所描述的技术可应用于有效性映射译码，且在无限制的情况下应用于变换系数的其它语法和二进位 (bin) 译码。另外，本发明通常将视频块称作 TU 块，但所述技术可应用于 TU、PU、CU、

LCU 或块的其它群组中的任一者。

[0169] 图 15 为说明用于编码当前块的实例方法的流程图。当前块可包含当前 CU 或所述当前 CU 的一部分。尽管关于视频编码器 20(图 1 和图 2) 进行了描述，但应理解，其它装置可经配置以执行类似于图 15 的方法的方法。

[0170] 在此实例中，视频编码器 20 最初预测当前块 (150)。举例来说，视频编码器 20 可计算当前块的一或多个预测单元 (PU)。视频编码器 20 可接着计算当前块的残余块以 (例如) 产生变换单元 (TU)(152)。为了计算所述残余块，视频编码器 20 可计算原始的未经译码块与当前块的预测块之间的差 (即，逐个像素的差)。视频编码器 20 可接着变换和量化残余块的系数 (154)。接下来，视频编码器 20 可扫描残余块的经量化的变换系数 (156)。

[0171] 在扫描期间，视频编码器 20 可确定当前系数在其中出现的区域，且以此方式，视频编码器 20 可确定各种系数在其中出现的区域 (158)。根据本发明的技术，视频编码器 20 可基于 (例如) 系数的位置或系数在其中出现的子块的位置来确定系数在其中出现的区域。视频编码器 20 可使用关于图 9 到图 14 所描述的技术中的任一者或其它类似技术来确定区域。举例来说，如图 9A 中所展示，视频编码器 20 可经配置以确定系数出现于包括一或多个子块的第一区域中还是包括在第一区域外部的子块的第二区域中。

[0172] 视频编码器 20 可进一步基于区域来确定用于熵编码系数的上下文 (160)。即，视频编码器 20 可针对每一系数基于系数在其中出现的区域来确定用于编码系数的上下文。举例来说，如以上所论述，视频编码器 20 可基于系数在其中出现的区域而基于系数在块中的位置、包括系数的子块在块中的位置、待应用于经计算的上下文的偏移或其类似者来确定上下文。

[0173] 同样，视频编码器 20 可使用经确定上下文来熵编码系数 (162)。具体来说，视频编码器 20 可使用上下文来熵编码表示系数的一或多个语法元素。举例来说，视频编码器 20 可熵编码系数的有效性信息、有效系数的水平信息和 / 或有效系数的正负号信息中的一者或一者以上。有效性信息可包含 significant_coeff_flag 数据。水平信息可包含 coeff_abs_level_greater1_flag、coeff_abs_level_greater2_flag 和 coeff_abs_level_remaining。正负号信息可包含 coeff_sign_flag。视频编码器 20 可接着输出系数的经熵编码的数据 (164)。

[0174] 以此方式，图 15 的方法表示一方法的一实例，所述方法包括基于视频块的变换系数在其中出现的区域来确定用于译码视频块的变换系数的上下文，及使用经确定上下文来熵译码所述变换系数。此外，所述区域可包含第一区域和第二区域中的一者，所述第一区域包含视频块的变换系数的一或多个左上 4×4 子块，所述第二区域包含视频块的在所述第一区域外部的变换系数。

[0175] 图 16 为说明用于解码视频数据的当前块的实例方法的流程图。当前块可包含当前 CU 或所述当前 CU 的一部分。尽管关于视频解码器 30(图 1 和图 3) 进行了描述，但应理解，其它装置可经配置以执行类似于图 16 的方法的方法。

[0176] 视频解码器 30 可预测当前块 (200)，例如，使用帧内或帧间预测模式以计算当前块的经预测块。视频解码器 30 还可接收当前块的经熵编码的数据，例如，对应于当前块的残余块的系数的经熵编码数据 (202)。

[0177] 根据本发明的技术，视频解码器 30 可 (例如) 在反向扫描和熵解码过程期间确定

系数将在其中出现的区域 (204)。即，视频解码器 30 可基于先前所解码的变换系数的位置和扫描次序中的下一有效变换系数来确定下一变换系数的位置。视频解码器 30 可进一步确定此位置在其中出现的块的区域。视频解码器 30 可以类似方式类似地确定系数中的每一者的区域。

[0178] 此外，视频解码器 30 可基于（例如）系数的位置或系数将在其中出现的子块的位置来确定系数将在其中出现的区域。视频解码器 30 可使用关于图 9 到图 14 所描述的技术中的任一者或其它类似技术来确定区域。举例来说，如图 9A 中所展示，视频解码器 30 可经配置以确定系数出现于包括一或多个子块的第一区域中还是包括在第一区域外部的子块的第二区域中。

[0179] 另外，视频解码器 30 可基于经确定区域来确定用于解码系数的上下文 (206)。即，视频解码器 30 可针对每一系数基于系数在其中出现的区域来确定用于解码系数的上下文。举例来说，如以上所论述，视频解码器 30 可基于系数将在其中出现的区域而基于系数在块中的位置、包括系数的子块在块中的位置、待应用于经计算的上下文的偏移或其类似者来确定上下文。

[0180] 视频解码器 30 可使用经确定上下文熵解码经熵译码的数据以再现块的系数 (208)。具体来说，视频解码器 30 可使用上下文来熵解码表示系数的一或多个语法元素。举例来说，视频解码器 30 可熵解码系数的有效性信息、有效系数的水平信息和 / 或有效系数的正负号信息中的一者或一者以上。有效性信息可包含 significant_coeff_flag 数据。水平信息可包含 coeff_abs_level_greater1_flag、coeff_abs_level_greater2_flag 和 coeff_abs_level_remaining。正负号信息可包含 coeff_sign_flag。视频解码器 30 可接着重新产生块（例如，TU）以包括在经解码的变换系数的相应位置中的经解码的变换系数 (210)。即，如以上所论述，视频解码器 30 可反向扫描经再现的系数以产生经量化的变换系数的块。

[0181] 视频解码器 30 可接着反量化和反变换所述系数以产生残余块 (212)。视频解码器 30 可最终通过组合所预测块与残余块来解码当前块 (214)。即，视频解码器 30 可以数学方式组合经预测块的像素值与残余块的位于相同位置的像素值以解码且再现原始块。

[0182] 以此方式，图 16 的方法表示一方法的一实例，所述方法包括基于视频块的变换系数在其中出现的区域来确定用于译码视频块的变换系数的上下文，及使用经确定上下文来熵译码所述变换系数。此外，所述区域可包含第一区域和第二区域中的一者，所述第一区域包含视频块的变换系数的一或多个左上 4×4 子块，所述第二区域包含视频块的在所述第一区域外部的变换系数。

[0183] 在一或多个实例中，所描述功能可以硬件、软件、固件或其任何组合来实施。如果以软件实施，则所述功能可作为一或多个指令或代码而存储于计算机可读媒体上或经由计算机可读媒体进行传输，且通过基于硬件的处理单元执行。计算机可读媒体可包括计算机可读存储媒体（其对应于例如数据存储媒体的有形媒体）或通信媒体，通信媒体包括（例如）根据通信协议促进计算机程式从一处传送到另一处的任何媒体。以此方式，计算机可读媒体大体上可对应于 (1) 非暂时性有形计算机可读存储媒体，或 (2) 例如信号或载波的通信媒体。数据存储媒体可为可由一或多个计算机或一或多个处理器存取以检索指令、代码和 / 或数据结构以用于实施本发明中所描述的技术的任何可用媒体。计算机程式产品可

包括计算机可读媒体。

[0184] 举例来说而非限制，所述计算机可读存储媒体可包含 RAM、ROM、EEPROM、CD-ROM 或其它光盘存储装置、磁盘存储装置或其它磁性存储装置，快闪存储器，或可用以存储呈指令或数据结构的形式的所要程序代码且可由计算机存取的任何其它媒体。而且，将任何连接适当地称为计算机可读媒体。举例来说，如果使用同轴电缆、光缆、双绞线、数字订户线 (DSL) 或无线技术（例如，红外线、无线电和微波）而从网站、服务器或其它远端源传输指令，则同轴电缆、光缆、双绞线、DSL 或无线技术（例如，红外线、无线电和微波）包括于媒体的定义中。然而，应理解，计算机可读存储媒体和数据存储媒体不包括连接、载波、信号或其它暂时媒体，而可替代地针对非暂时、有形存储媒体。如本文中所使用，磁盘和光盘包括压缩光盘 (CD)、激光光盘、光盘、数字多功能光盘 (DVD)、软性磁盘和蓝光光盘，其中磁盘通常以磁性方式再现数据，而光盘通过激光以光学方式再现数据。以上各者的组合也应包括于计算机可读媒体的范围内。

[0185] 可通过一或多个处理器来执行指令，例如，一或多个数字信号处理器 (DSP)、通用微处理器、专用集成电路 (ASIC)、现场可编程逻辑阵列 (FPGA) 或其它等效集成或离散逻辑电路。因此，如本文中所使用，术语“处理器”可指代前述结构或适合于实施本文中所描述的技术的任何其它结构中的任一者。另外，在一些方面中，可将本文所描述的功能性提供于经配置以用于编码和解码的专用硬件和 / 或软件模块内，或并入于组合式编解码器中。而且，所述技术可完全实施于一或多个电路或逻辑元件中。

[0186] 本发明的技术可实施于多种装置或设备中，所述装置或设备包括无线手持机、集成电路 (IC) 或一组 IC（例如，芯片组）。在本发明中描述各种组件、模块或单元以强调经配置以执行所揭示技术的装置的功能方面，但未必要求通过不同硬件单元来实现。而是，如以上所描述，可将各种单元组合于编解码器硬件单元中，或通过互操作性硬件单元（包括如以上所描述的一或多个处理器）的集合结合合适软件和 / 或固件来提供所述单元。

[0187] 已描述各种实例。这些和其它实例在所附权利要求书的范围内。

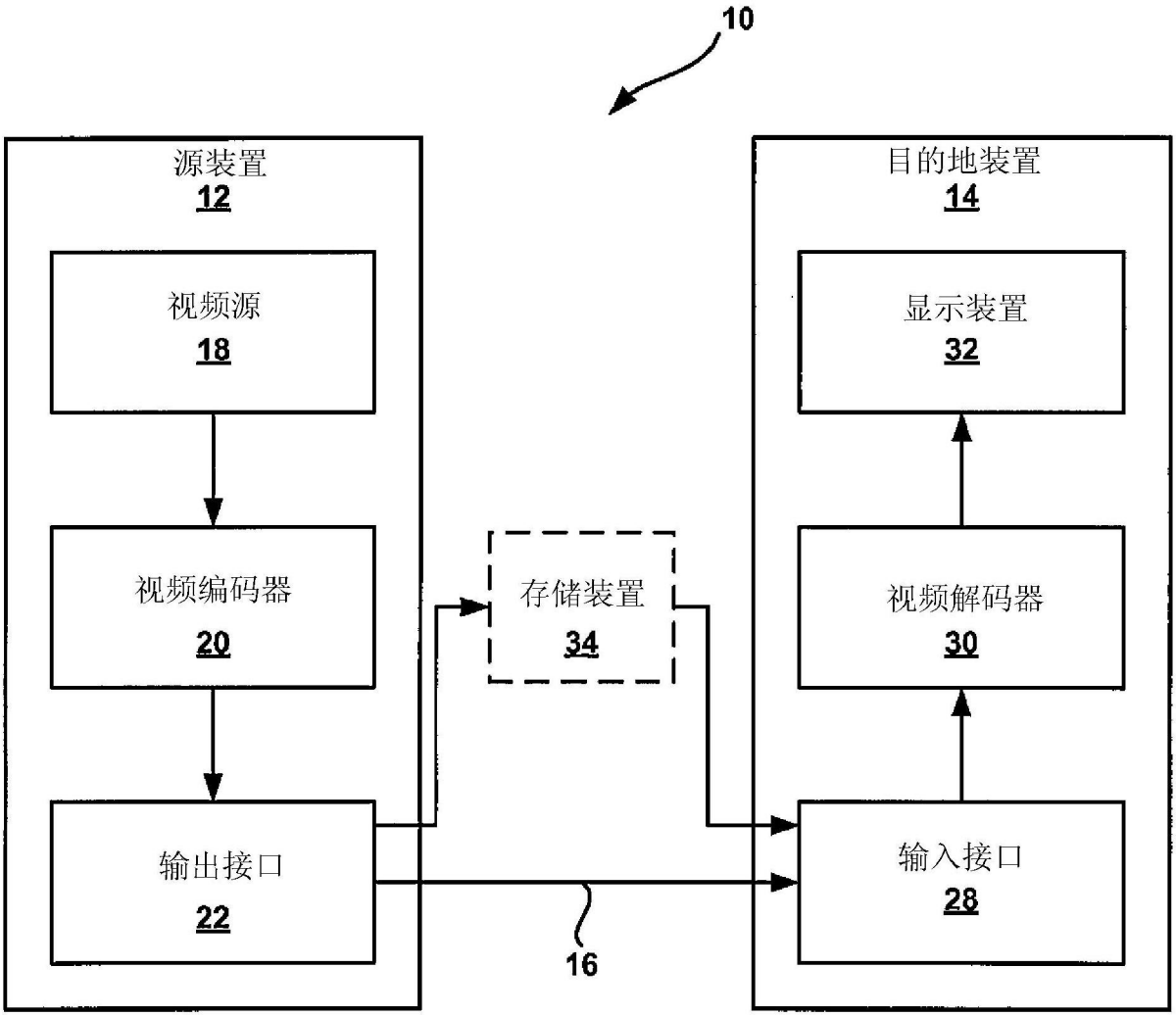

10
源装置
12
视频源
18
视频编码器
20
输出接口
22
存储装置
34
16
目的地装置
14
显示装置
32
视频解码器
30
输入接口
28

图 1

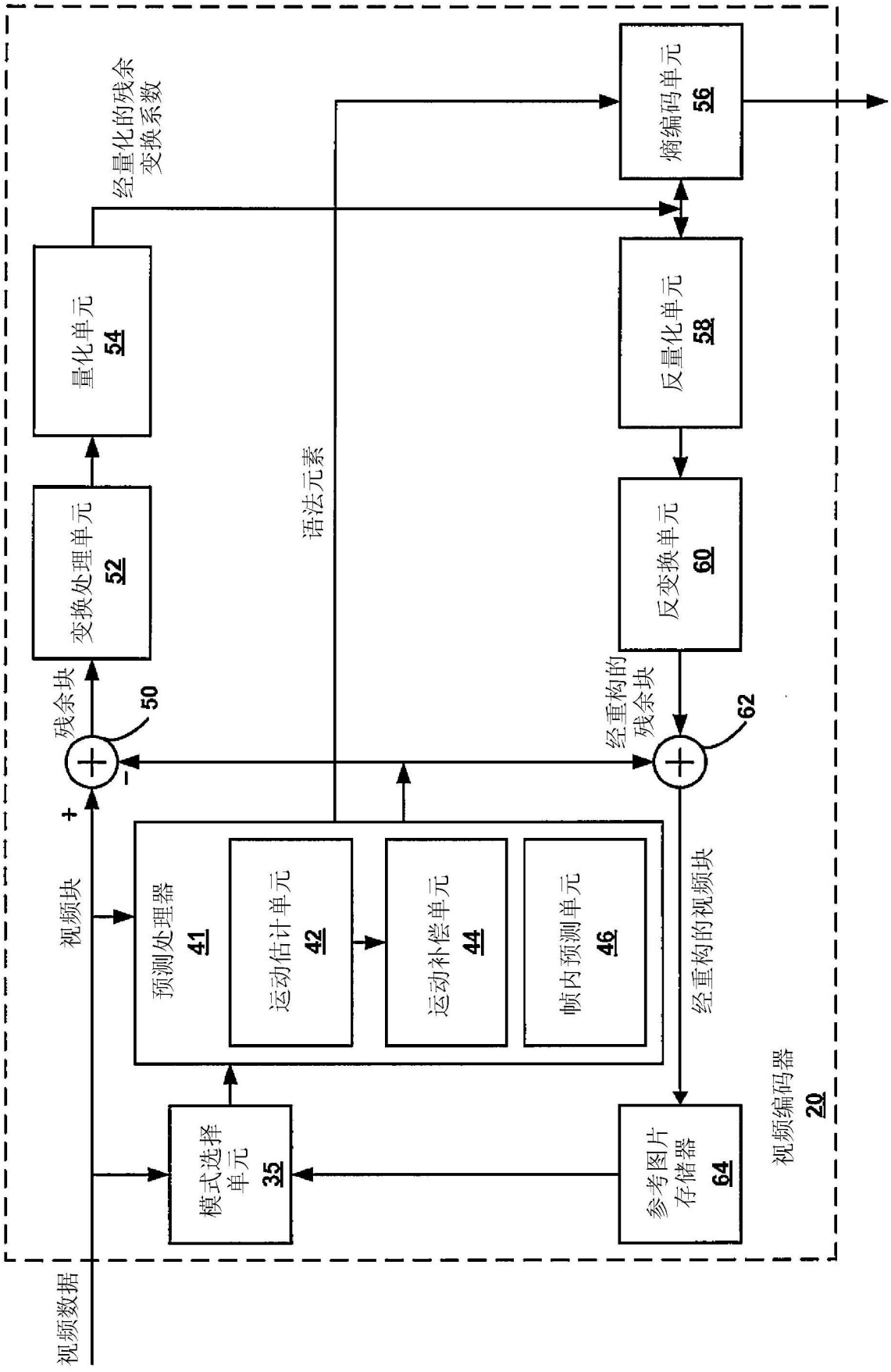

视频数据
视频块
+
-
残余块
50
变换处理单元
52
量化单元
54
经量化的残余
变换系数
熵编码单元
56
反量化单元
58
反变换单元
60
经重构的
残余块
62
语法元素
预测处理器
41
运动估计单元
42
运动补偿单元
44
帧内预测单元
46
模式选择
单元
35
参考图片
存储器
64
经重构的视频块
视频编码器
20

图 2

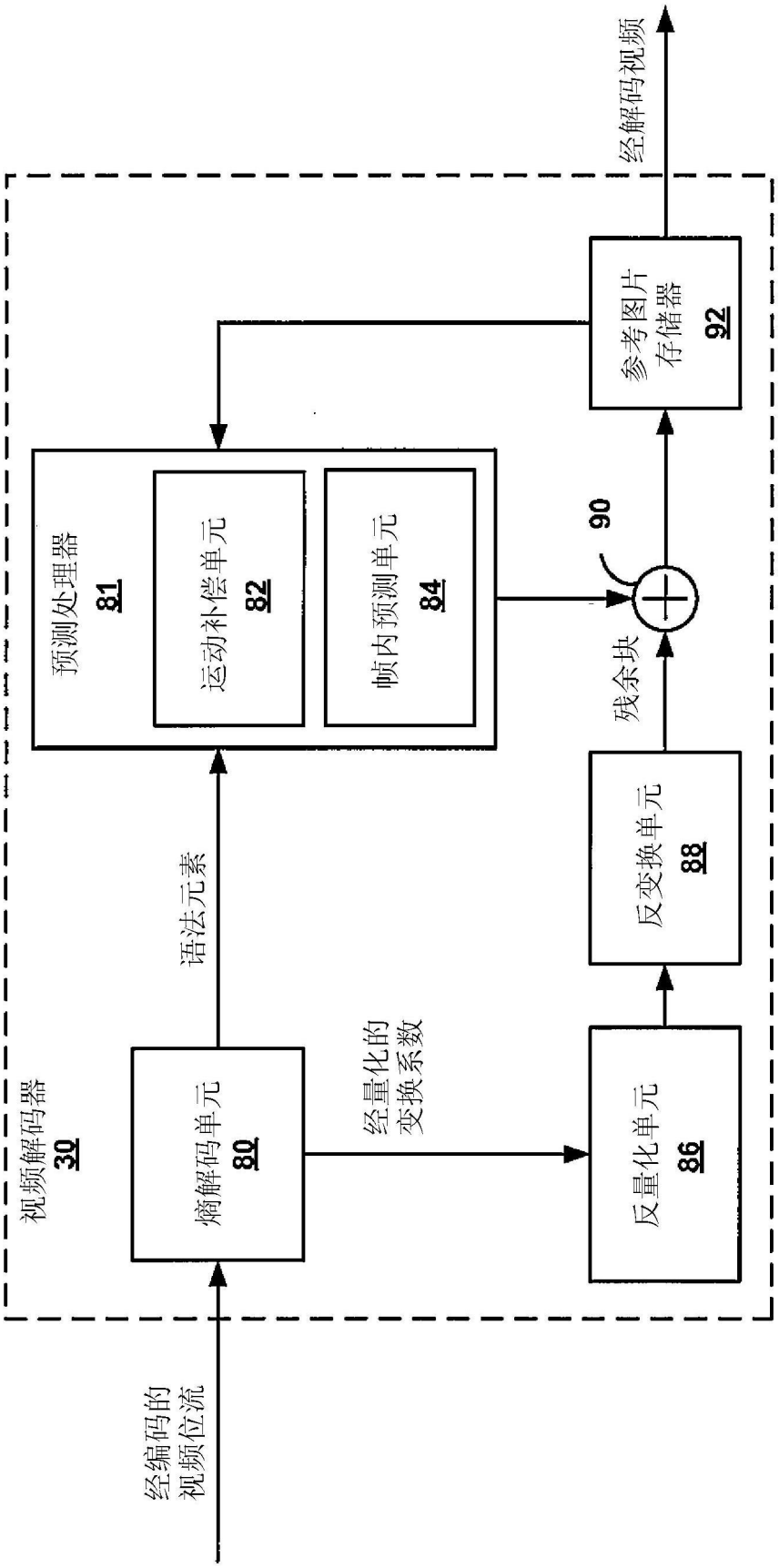

经解码视频
参考图片存储器 92
预测处理器 81
运动补偿单元 82
帧内预测单元 84
90
残余块
反变换单元 88
反量化单元 86
经量化的变换系数
语法元素
熵解码单元 80
视频解码器 30
经编码的视频位流

图 3

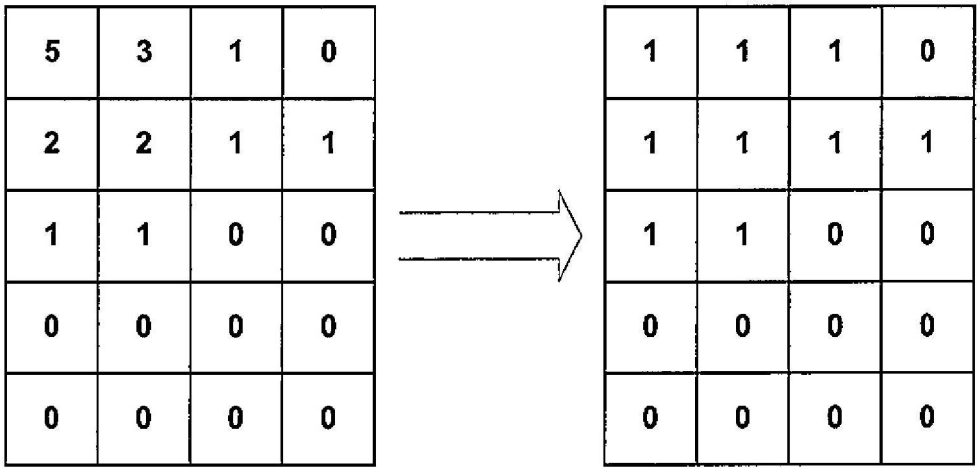

5 3 1 0
2 2 1 1
1 1 0 0
0 0 0 0
0 0 0 0
1 1 1 0
1 1 1 1
1 1 0 0
0 0 0 0
0 0 0 0
变换系数
有效性映射

图 4

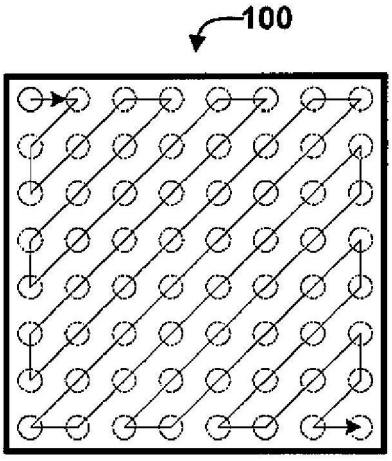

100

图 5A

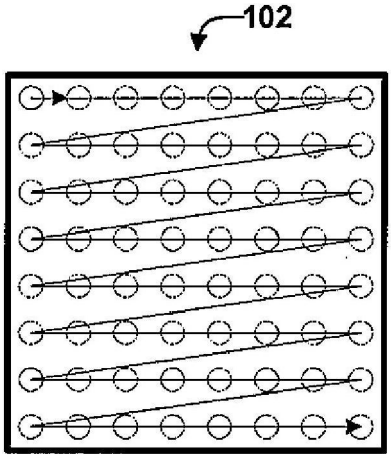

102

图 5B

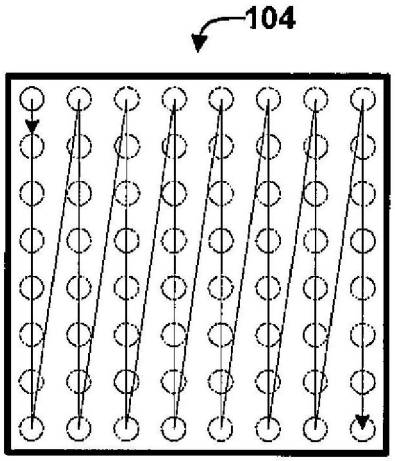


图 5C

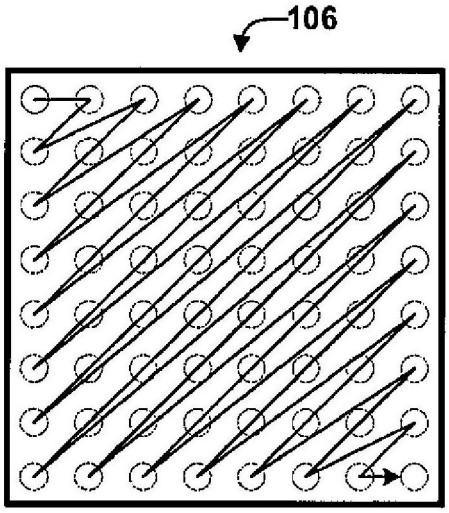


图 5D

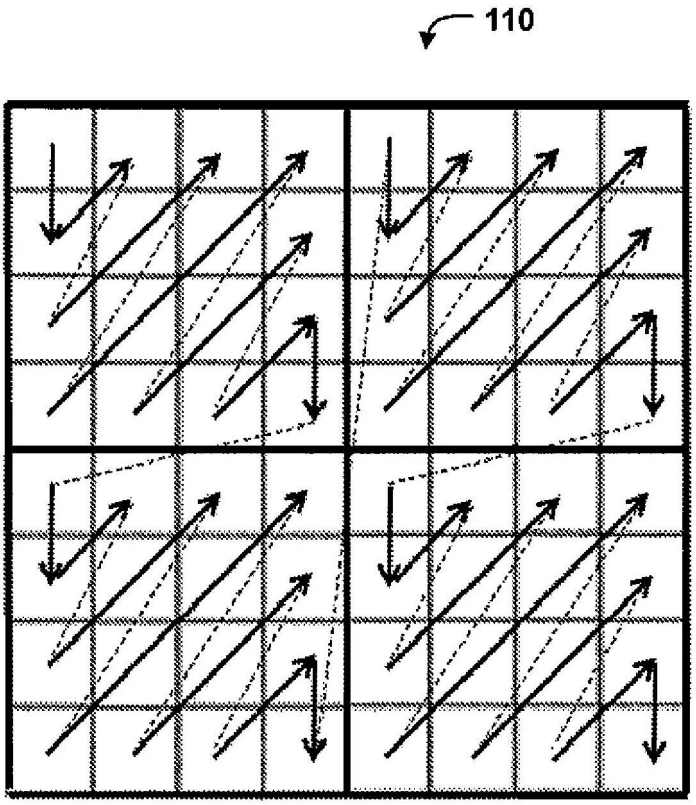


图 6

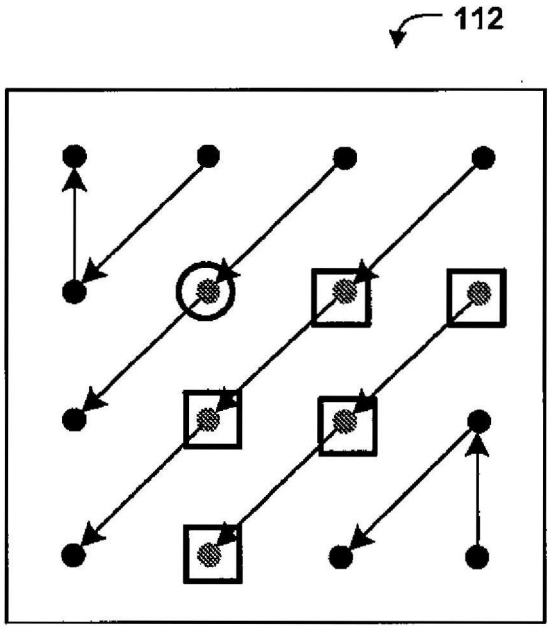


图 7

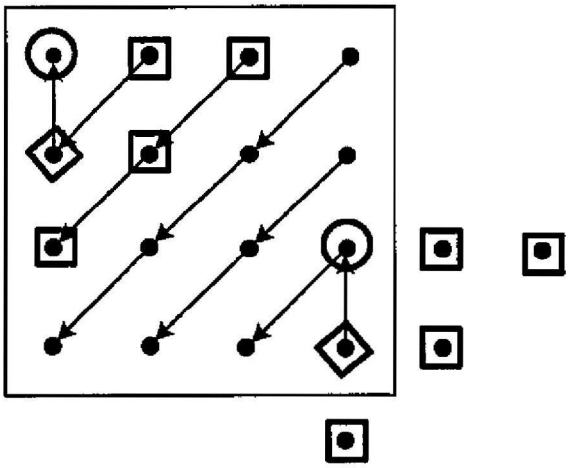


图 8A

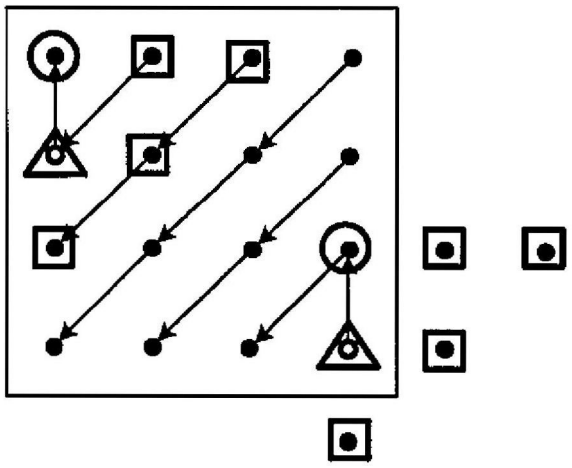


图 8B

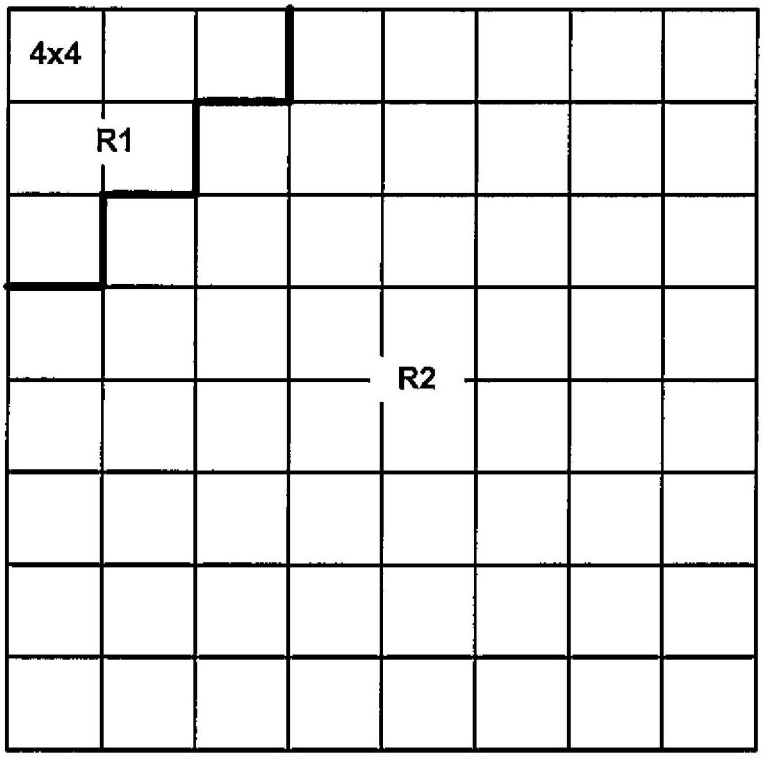


图 9A

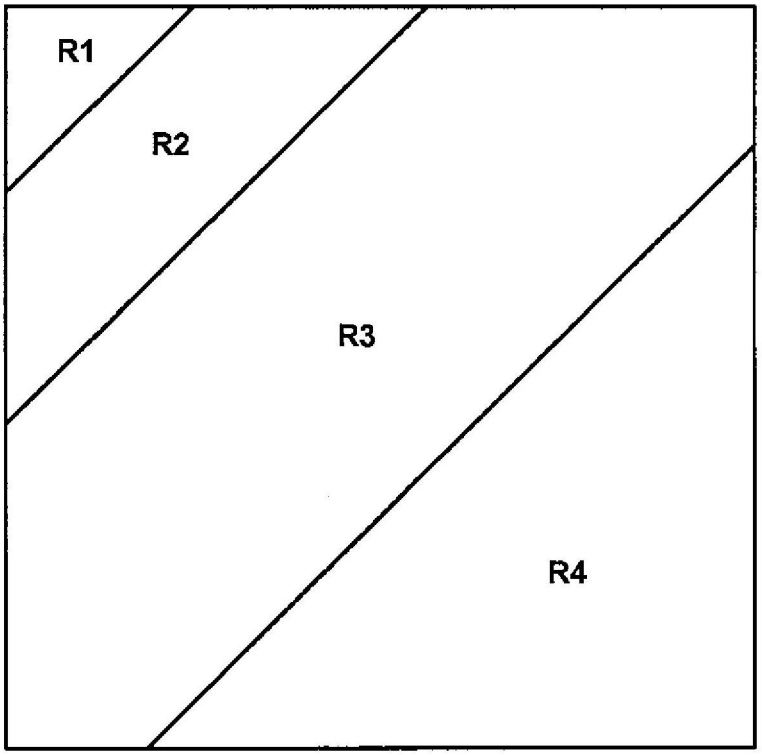


图 9B

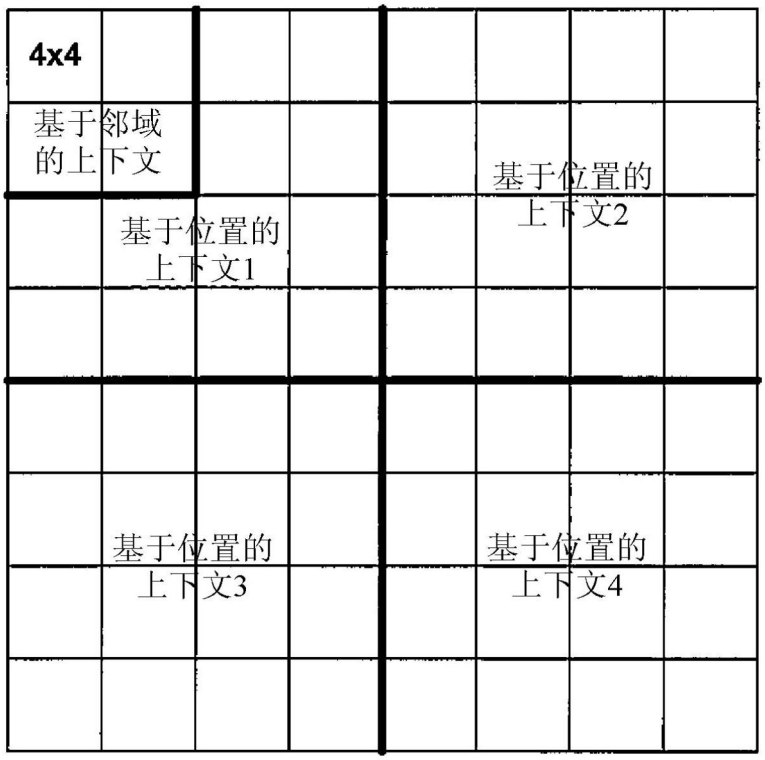

4x4
基于邻域的上下文
基于位置的上下文1
基于位置的上下文2
基于位置的上下文3
基于位置的上下文4

图 10

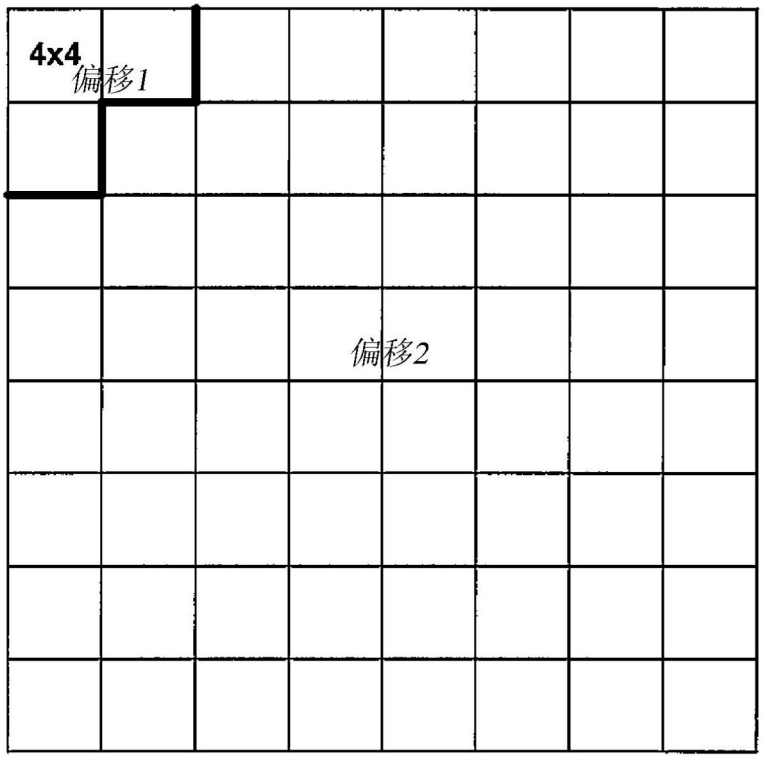

4x4
偏移1
偏移2

图 11

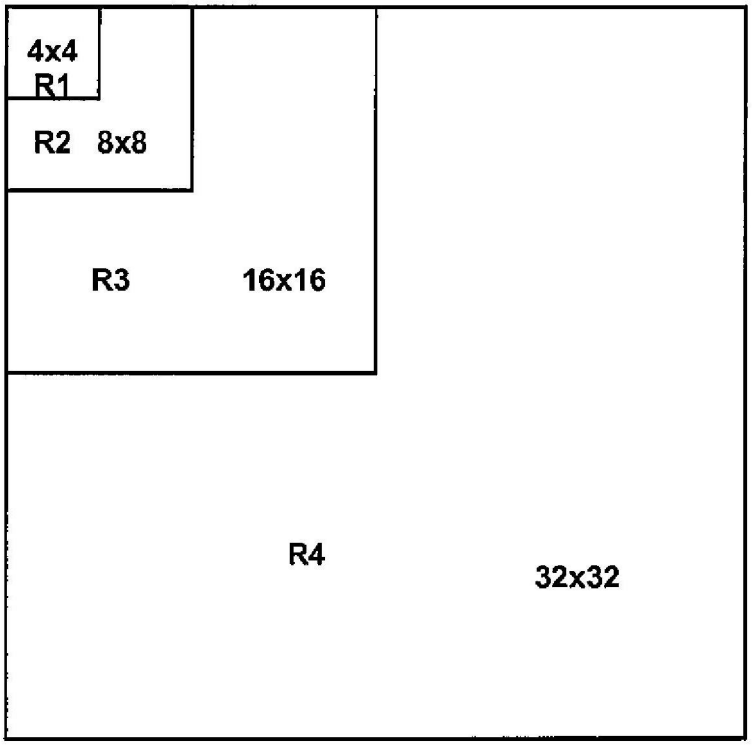


图 12

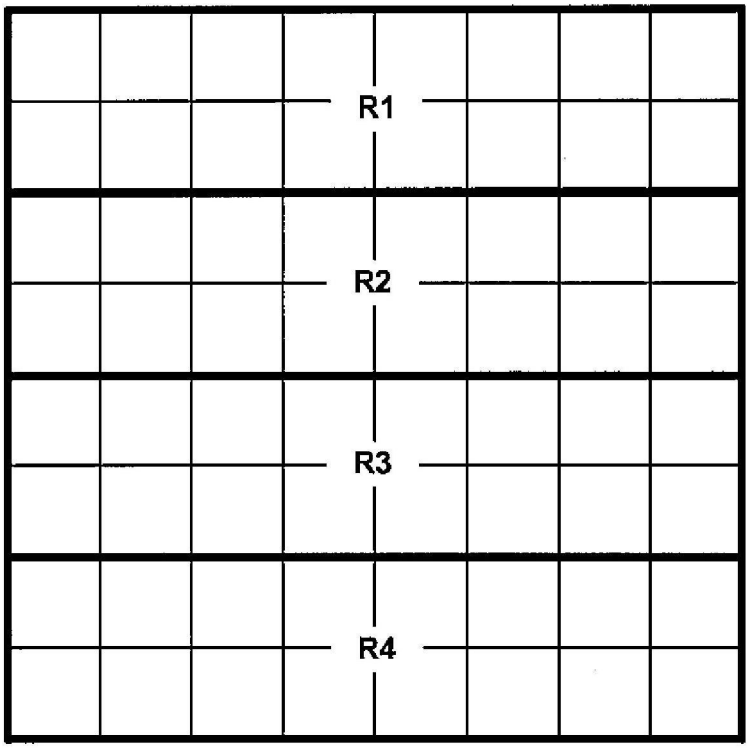


图 13A

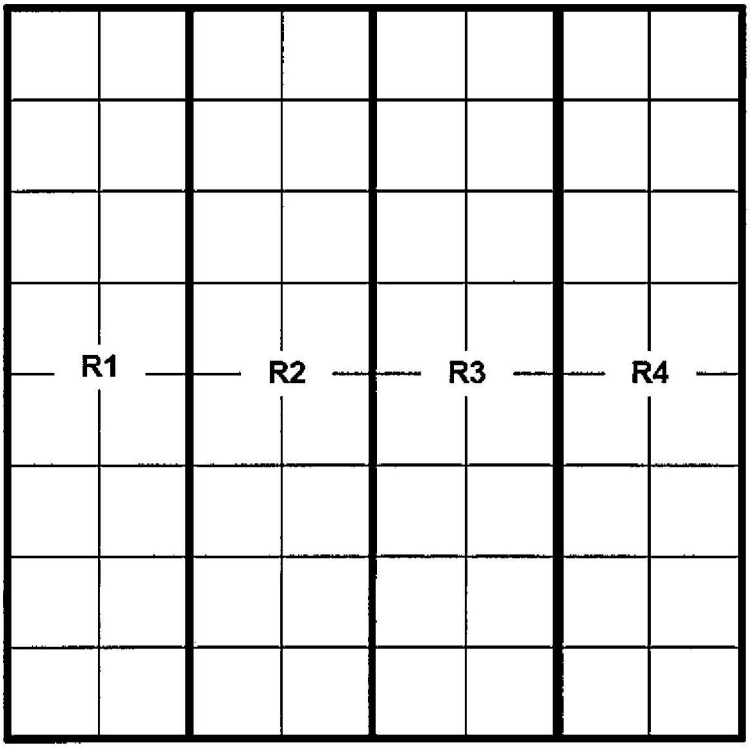


图 13B

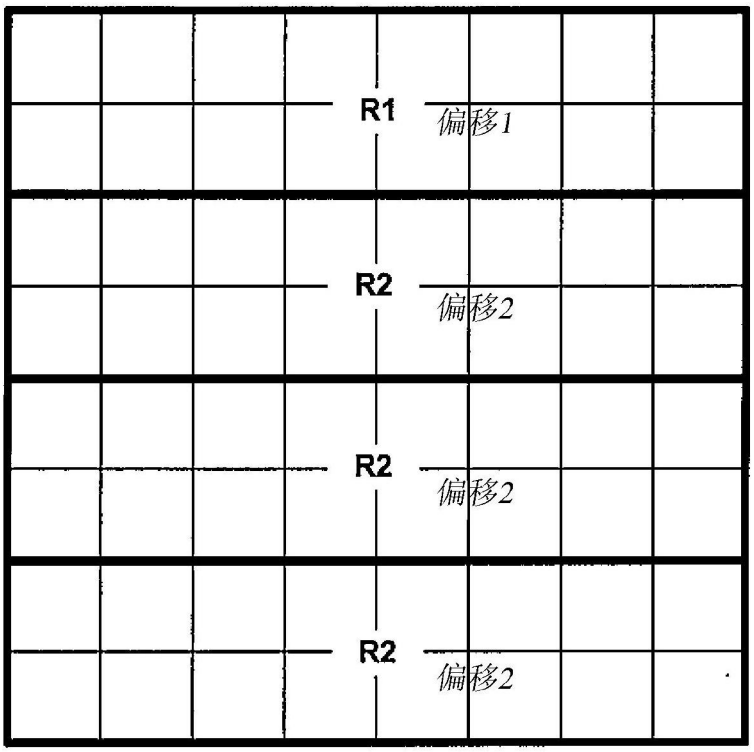


图 14A

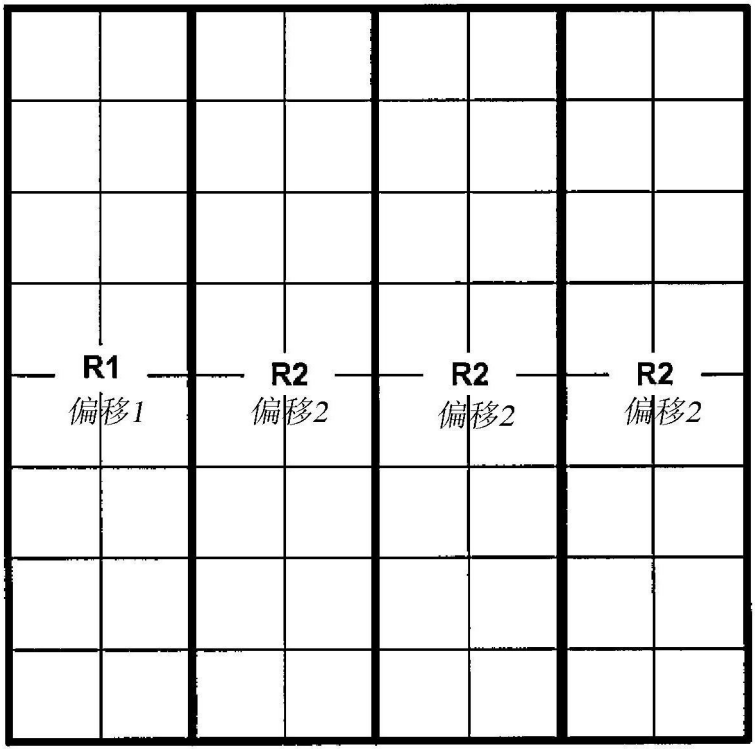


图 14B

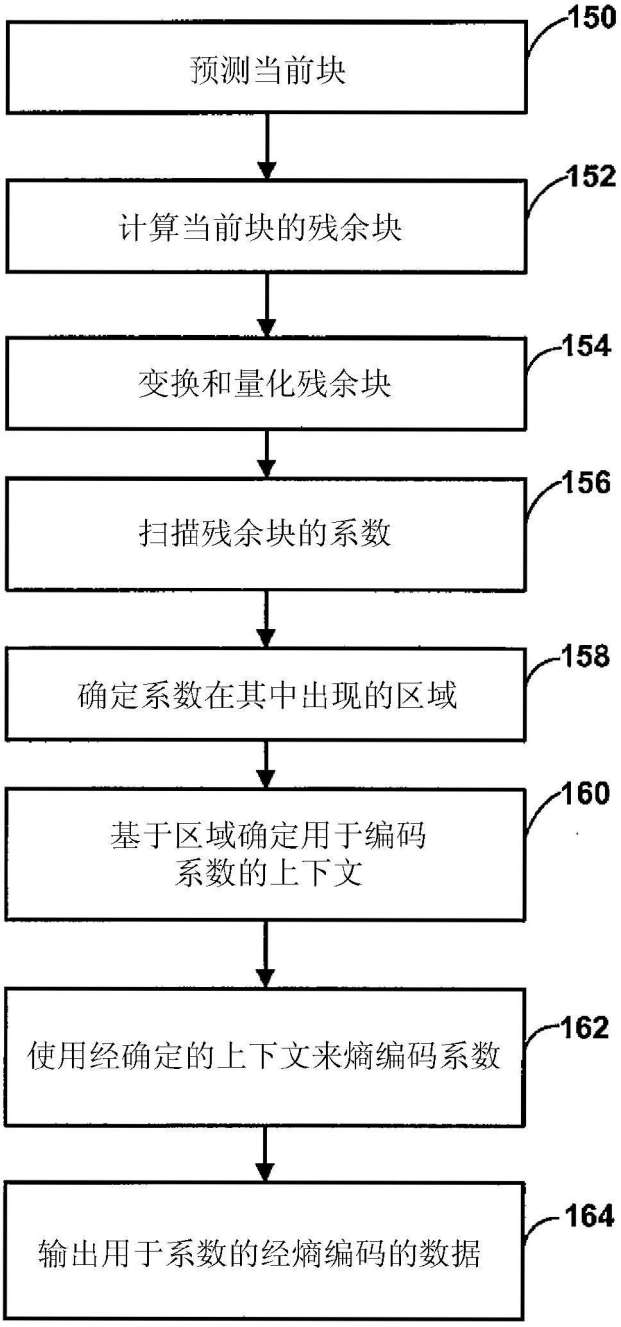

150
预测当前块
152
计算当前块的残余块
154
变换和量化残余块
156
扫描残余块的系数
158
确定系数在其中出现的区域
160
基于区域确定用于编码系数的上下文
162
使用经确定的上下文来熵编码系数
164
输出用于系数的经熵编码的数据

图 15

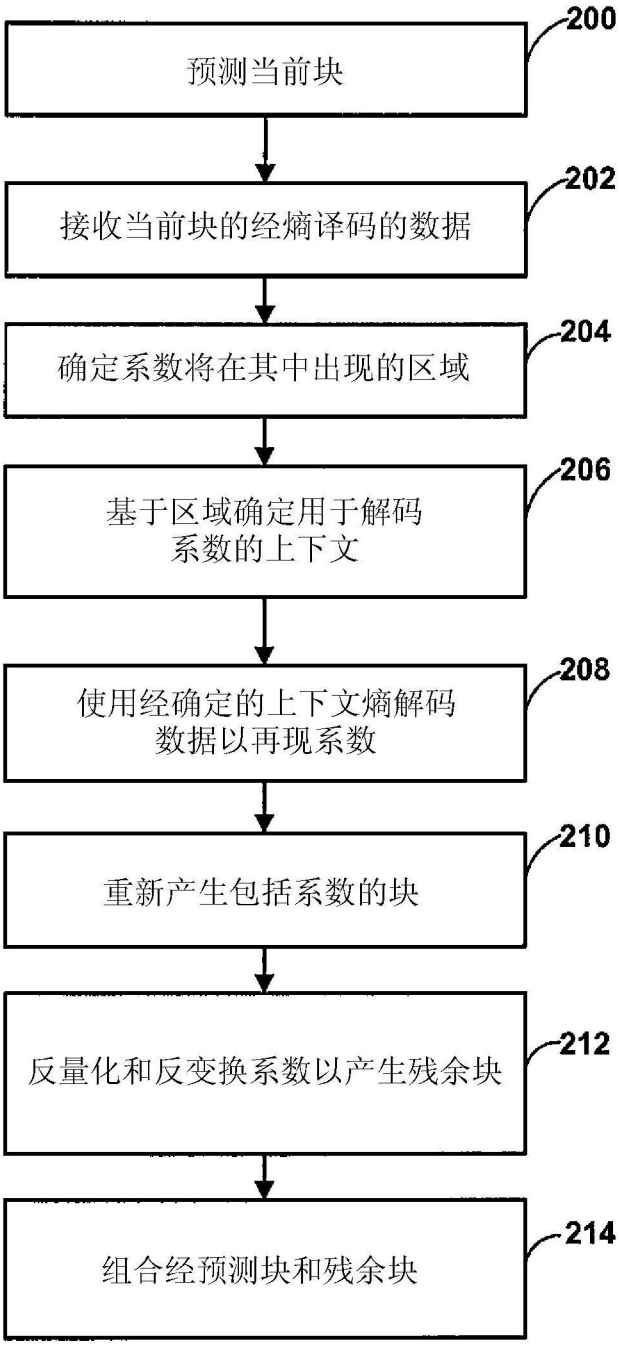

预测当前块
200
接收当前块的经熵译码的数据
202
确定系数将在其中出现的区域
204
基于区域确定用于解码
系数的上下文
206
使用经确定的上下文熵解码
数据以再现系数
208
重新产生包括系数的块
210
反量化和反变换系数以产生残余块
212
组合经预测块和残余块
214

图 16