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

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
4 August 2016 (04.08.2016)

WIPO | PCT

(10) International Publication Number
WO 2016/123488 A1

(51) International Patent Classification:
H04N 19/70 (2014.01) *H04N 19/463* (2014.01)
H04N 19/13 (2014.01) *H04N 19/42* (2014.01)
H04N 19/593 (2014.01)

(21) International Application Number:
PCT/US2016/015663

(22) International Filing Date:
29 January 2016 (29.01.2016)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
62/110,302 30 January 2015 (30.01.2015) US
15/009,477 28 January 2016 (28.01.2016) US

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

(72) Inventors: **KARCZEWICZ, Marta**; 5775 Morehouse Drive, San Diego, California 92121-1714 (US). **PU, Wei**; 5565 Wellesley Avenue, Apartment 5, Pittsburgh, Pennsylvania 15206 (US). **JOSHI, Rajan Laxman**; 5775 Morehouse Drive, San Diego, California 92121-1714 (US). **SEREGIN, Vadim**; 5775 Morehouse Drive, San Diego, California 92121-1714 (US).

(74) Agent: **MCADAMS, Paul M.**; 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, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IR, IS, JP, KE, KG, KN, KP, KR, KZ, LA, LC, LK, LR, LS, 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, SA, 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, ST, SZ,

[Continued on next page]

(54) Title: PALETTE INDEX GROUPING FOR HIGH THROUGHPUT CABAC CODING

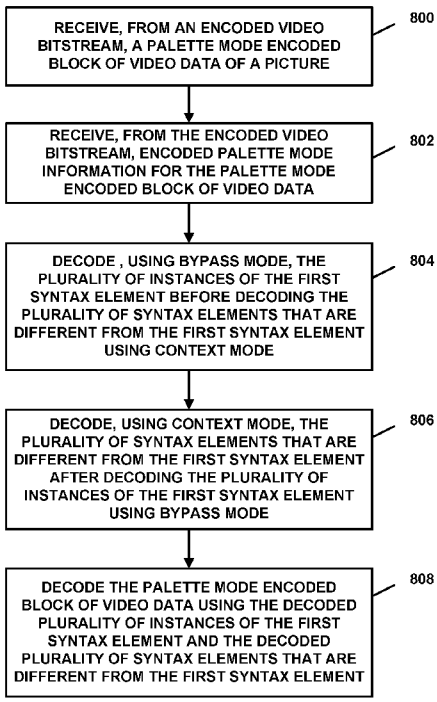


FIG. 8

(57) Abstract: In an example, a method of decoding video data may include receiving a palette mode encoded block of video data of a picture. The method may include receiving encoded palette mode information for the palette mode encoded block of video data. The encoded palette mode information may include a plurality of instances of a first syntax element and a plurality of syntax elements that are different from the first syntax element. The method may include decoding, using bypass mode, the plurality of instances of the first syntax element before decoding the plurality of syntax elements that are different from the first syntax element using context mode. The method may include decoding, using context mode, the plurality of syntax elements that are different from the first syntax element after decoding the plurality of instances of the first syntax element using bypass mode.

WO 2016/123488 A1

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, KM, ML, MR, NE, SN, TD, TG).

Published:

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

PALETTE INDEX GROUPING FOR HIGH THROUGHPUT CABAC CODING

[0001] This application claims the benefit of U.S. Provisional Patent Application No. 62/110,302 filed on January 30, 2015, which is hereby incorporated by reference herein in its entirety.

TECHNICAL FIELD

[0002] This disclosure relates to encoding and decoding content, and more specifically, encoding and decoding content according to a palette-based coding mode.

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), ITU-T H.265, High Efficiency Video Coding (HEVC), 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. 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 as 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 indicates 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 coefficients, which then may be quantized. The quantized coefficients, initially arranged in a two-dimensional array, may be scanned in order to produce a one-dimensional vector of coefficients, and entropy coding may be applied to achieve even more compression.

[0006] Content, such as an image, may be encoded and decoded using palette mode. Generally, palette mode is a technique involving use of a palette to represent content. Content may be encoded such that the content is represented by an index map that includes values corresponding to the palette. The index map may be decoded to reconstruct the content.

SUMMARY

[0007] Techniques of this disclosure relate to palette-based content coding. For example, in palette-based content coding, a content coder (e.g., a content coder such as a video encoder or a video decoder) may form a "palette" as a table of colors for representing the video data of the particular area (e.g., a given block). Palette-based content coding may, for example, be especially useful for coding areas of video data having a relatively small number of colors. Rather than coding actual pixel values (or their residuals), the content coder may code palette indices (e.g., index values) for one or more of the pixels that relate the pixels with entries in the palette representing the colors of the pixels. The techniques described in this disclosure may include techniques for various combinations of one or more of signaling palette-based coding modes, transmitting palettes, deriving palettes, deriving the value of non-transmitted syntax elements, transmitting palette-based coding maps and other syntax elements, predicting palette entries, coding runs of palette indices, entropy coding palette information, and various other palette coding techniques.

[0008] In one example, this disclosure describes a method of decoding video data comprising receiving, from an encoded video bitstream, a palette mode encoded block

of video data of a picture; receiving, from the encoded video bitstream, encoded palette mode information for the palette mode encoded block of video data, wherein the encoded palette mode information includes a plurality of instances of a first syntax element and a plurality of syntax elements that are different from the first syntax element; decoding, using bypass mode, the plurality of instances of the first syntax element before decoding the plurality of syntax elements that are different from the first syntax element using context mode; decoding, using context mode, the plurality of syntax elements that are different from the first syntax element after decoding the plurality of instances of the first syntax element using bypass mode; and decoding the palette mode encoded block of video data using the decoded plurality of instances of the first syntax element and the decoded plurality of syntax elements that are different from the first syntax element.

[0009] In another example, this disclosure describes a device for decoding video data comprising a memory configured to store the video data; and a video decoder in communication with the memory configured to: receive, from an encoded video bitstream, a palette mode encoded block of video data of a picture; receive, from the encoded video bitstream, encoded palette mode information for the palette mode encoded block of video data, wherein the encoded palette mode information includes a plurality of instances of a first syntax element and a plurality of syntax elements that are different from the first syntax element; decode, using bypass mode, the plurality of instances of the first syntax element before decoding the plurality of syntax elements that are different from the first syntax element using context mode; decode, using context mode, the plurality of syntax elements that are different from the first syntax element after decoding the plurality of instances of the first syntax element using bypass mode; and decode the palette mode encoded block of video data using the decoded plurality of instances of the first syntax element and the decoded plurality of syntax elements that are different from the first syntax element.

[0010] In another example, this disclosure describes a non-transitory computer-readable storage medium having instructions stored thereon that, when executed, cause one or more processors to receive, from an encoded video bitstream, a palette mode encoded block of video data of a picture; receive, from the encoded video bitstream, encoded palette mode information for the palette mode encoded block of video data, wherein the encoded palette mode information includes a plurality of instances of a first syntax element and a plurality of syntax elements that are different from the first syntax

element; decode, using bypass mode, the plurality of instances of the first syntax element before decoding the plurality of syntax elements that are different from the first syntax element using context mode; decode, using context mode, the plurality of syntax elements that are different from the first syntax element after decoding the plurality of instances of the first syntax element using bypass mode; and decode the palette mode encoded block of video data using the decoded plurality of instances of the first syntax element and the decoded plurality of syntax elements that are different from the first syntax element.

[0011] In another example, this disclosure describes a method of encoding video data comprising determining that a block of video data is to be coded in palette mode; encoding the block of video data using palette mode into an encoded bitstream, wherein encoding the block of video data using palette mode comprises: generating palette mode information for the block of video data, wherein the palette mode information includes a plurality of instances of a first syntax element and a plurality of syntax elements that are different from the first syntax element; encoding, using bypass mode, the plurality of instances of the first syntax element into the encoded bitstream before encoding the plurality of syntax elements that are different from the first syntax element into the encoded bitstream using context mode; and encoding, using context mode, the plurality of syntax elements that are different from the first syntax element into the encoded bitstream after encoding the plurality of instances of the first syntax element using bypass mode into the encoded bitstream.

[0012] In another example, this disclosure describes a device for encoding video data, the device comprising a memory configured to store the video data; and a video encoder in communication with the memory, the video encoder configured to: determine that a block of video data stored in the memory is to be encoded in palette mode; encode the block of video data using palette mode into an encoded bitstream, wherein the video encoder being configured to encode the block of video data using palette mode comprises the video encoder being configured to: generate palette mode information for the block of video data, wherein the palette mode information includes a plurality of instances of a first syntax element and a plurality of syntax elements that are different from the first syntax element; encode, using bypass mode, the plurality of instances of the first syntax element into the encoded bitstream before encoding the plurality of syntax elements that are different from the first syntax element into the encoded bitstream using context mode; and encode, using context mode, the plurality of syntax

elements that are different from the first syntax element into the encoded bitstream after encoding the plurality of instances of the first syntax element using bypass mode into the encoded bitstream.

[0013] The details of one or more examples of the disclosure are set forth in the accompanying drawings and the description below. Other features, objects, and advantages of the disclosure will be apparent from the description and drawings, and from the claims.

BRIEF DESCRIPTION OF DRAWINGS

[0014] FIG. 1 is a block diagram illustrating an example video coding system that may utilize the techniques described in this disclosure.

[0015] FIG. 2 is a block diagram illustrating an example video encoder that may perform the techniques described in this disclosure.

[0016] FIG. 3 is a block diagram illustrating an example video decoder that may perform the techniques described in this disclosure.

[0017] FIG. 4 is a conceptual diagram illustrating an example of determining palette entries for palette-based video coding, consistent with techniques of this disclosure.

[0018] FIG. 5 is a conceptual diagram illustrating an example of determining indices to a palette for a block of pixels, consistent with techniques of this disclosure.

[0019] FIG. 6 is a conceptual diagram illustrating an example of determining maximum copy above run-length, assuming raster scanning order, consistent with techniques of this disclosure.

[0020] FIG. 7 is a table illustrating changes to coding order of syntax elements for palette-mode.

[0021] FIG. 8 is a flowchart illustrating an example process for decoding video data consistent with techniques for palette-based video coding of this disclosure.

[0022] FIG. 9 is a flowchart illustrating an example process for encoding video data consistent with techniques for palette-based video coding of this disclosure.

DETAILED DESCRIPTION

[0023] Aspects of this disclosure are directed to techniques for content coding (e.g., video coding). In particular, this disclosure describes techniques for palette-based coding of content data (e.g., video data) and techniques for context-based adaptive binary arithmetic coding (CABAC) of palette coding information. In various examples

of this disclosure, techniques of this disclosure may be directed to processes of predicting or coding a block in palette mode to improve coding efficiency and/or reduce codec complexity, as described in greater detail below. For example, the disclosure describes techniques related to palette index grouping (such as advanced palette index grouping).

[0024] In a CABAC process, e.g., as described in D. Marpe, H. Schwarz, and T. Wiegand, "Context-based adaptive binary arithmetic coding in the H.264/AVC video compression standard," in *IEEE Trans. Cir. & Sys. Video Tech.*, Vol. 13, No. 7, July 2003, there are two modes: (1) bypass mode and (2) context mode. In bypass mode, there is no context update process. Therefore, bypass mode can achieve higher data throughput than context-based mode by exploiting hardware or ISA level parallelism. This benefit of bypass mode becomes larger as the number of bypass bins that can be processed together increases.

[0025] In a current palette mode coding design, as described in R. Joshi and J. Xu, "High efficient video coding (HEVC) screen content coding: Draft 2," JCTVC-S1005, in screen content coding, the syntax elements of **palette_index_idc** and **palette_escape_val** are CABAC bypass mode coded, and are interleaved with other syntax elements such as **palette_run_msb_id_plus1** which are CABAC context mode coded. This disclosure describes techniques of grouping the bypass mode coded syntax elements together. As used herein, "bypass mode coded" and "context mode coded" are respectively interchangeable with "bypass coded" and "context coded."

[0026] As used herein, instances of the term "content" may be changed to the term "video," and instances of the term "video" may be changed to the term "content." This is true regardless of whether the terms "content" or "video" are being used as an adjective, noun, or other part of speech. For example, reference to a "content coder" also includes reference to a "video coder," and reference to a "video coder" also includes reference to a "content coder." Similarly, reference to "content" also includes reference to "video," and reference to "video" also includes reference to "content."

[0027] As used herein, "content" refers to any type of content. For example, "content" may refer to video, screen content, image, any graphical content, any displayable content, or any data corresponding thereto (e.g., video data, screen content data, image data, graphical content data, displayable content data, and the like).

[0028] As used herein, the term "video" may refer to screen content, movable content, a plurality of images that may be presented in a sequence, or any data corresponding

thereto (e.g., screen content data, movable content data, video data, image data, and the like).

[0029] As used herein, the term "image" may refer to a single image, one or more images, one or more images amongst a plurality of images corresponding to a video, one or more images amongst a plurality of images not corresponding to a video, a plurality of images corresponding to a video (e.g., all of the images corresponding to the video or less than all of the images corresponding to the video), a sub-part of a single image, a plurality of sub-parts of a single image, a plurality of sub-parts corresponding to a plurality of images, one or more graphics primitives, image data, graphical data, and the like.

[0030] In traditional video coding, images are assumed to be continuous-tone and spatially smooth. Based on these assumptions, various tools have been developed such as block-based transforms, filtering, and other coding tools, and such tools have shown good performance for natural content videos. However, in applications like remote desktop, collaborative work and wireless display, computer-generated screen content may be the dominant content to be compressed. This type of screen content tends to have discrete-tone, sharp lines, and high contrast object boundaries. The assumption of continuous-tone and smoothness may no longer apply, and thus, traditional video coding techniques may be inefficient in compressing content (e.g., screen content).

[0031] In one example of palette-based video coding, a video encoder may encode a block of video data by determining a palette for the block (e.g., coding the palette explicitly, predicting the palette, or a combination thereof), locating an entry in the palette to represent the value(s) of one or more pixels, and encoding both the palette and the block with index values that indicate the entry in the palette used to represent the pixel values of the block. In some examples, the video encoder may signal the palette and/or the index values in an encoded bitstream. In turn, a video decoder may obtain, from an encoded bitstream, a palette for a block, as well as index values for the individual pixels of the block. The video decoder may relate the index values of the pixels to entries of the palette to reconstruct the various pixel values of the block.

[0032] For example, a particular area of video data may be assumed to have a relatively small number of colors. A video coder (e.g., a video encoder or video decoder) may code (e.g., encode or decode) a so-called "palette" to represent the video data of the particular area. The palette may be expressed as an index (e.g., table) of colors or pixel values representing the video data of the particular area (e.g., a given block). The video

coder may code the index, which relates one or more pixel values to the appropriate value in the palette. Each pixel may be associated with an entry in the palette that represents the color of the pixel. For example, the palette may include the most dominant pixel values in the given block. In some cases, the most dominant pixel values may include the one or more pixel values that occur most frequently within the block. Additionally, in some cases, a video coder may apply a threshold value to determine whether a pixel value is to be included as one of the most dominant pixel values in the block. According to various aspects of palette-based coding, the video coder may code index values indicative of one or more of the pixels values of the current block, instead of coding actual pixel values or their residuals for a current block of video data. In the context of palette-based coding, the index values indicate respective entries in the palette that are used to represent individual pixel values of the current block. The description above is intended to provide a general description of palette-based video coding.

[0033] Palette-based coding may be particularly suitable for screen generated content coding or other content where one or more traditional coding tools are inefficient. The techniques for palette-based coding of video data may be used with one or more other coding techniques, such as techniques for inter- or intra-predictive coding. For example, as described in greater detail below, an encoder or decoder, or combined encoder-decoder (codec), may be configured to perform inter- and intra-predictive coding, as well as palette-based coding.

[0034] In some examples, the palette-based coding techniques may be configured for use with one or more video coding standards. For example, High Efficiency Video Coding (HEVC) is a new video coding standard being developed by the Joint Collaboration Team on Video Coding (JCT-VC) of ITU-T Video Coding Experts Group (VCEG) and ISO/IEC Motion Picture Experts Group (MPEG). The finalized HEVC standard document is published as "ITU-T H.265, SERIES H: AUDIOVISUAL AND MULTIMEDIA SYSTEMS Infrastructure of audiovisual services – Coding of moving video - High efficiency video coding," Telecommunication Standardization Sector of International Telecommunication Union (ITU), April 2013.

[0035] To provide more efficient coding of screen generated content, the JCT-VC is developing an extension to the HEVC standard, referred to as the HEVC Screen Content Coding (SCC) standard. A recent working draft of the HEVC SCC standard, referred to as "HEVC SCC Draft 2" or "WD2," is described in document JCTVC-S1005, R. Joshi

and J. Xu, "HEVC screen content coding draft text 2," Joint Collaborative Team on Video Coding (JCT-VC) of ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11, 19th Meeting: Strasbourg, FR, 17-24 October 2014.

[0036] With respect to the HEVC framework, as an example, the palette-based coding techniques may be configured to be used as a coding unit (CU) mode. In other examples, the palette-based coding techniques may be configured to be used as a prediction unit (PU) mode in the framework of HEVC. Accordingly, all of the following disclosed processes described in the context of a CU mode may, additionally or alternatively, apply to PU. However, these HEVC-based examples should not be considered a restriction or limitation of the palette-based coding techniques described herein, as such techniques may be applied to work independently or as part of other existing or yet to be developed systems/standards. In these cases, the unit for palette coding can be square blocks, rectangular blocks or even regions of non-rectangular shape.

[0037] In some examples, a palette may be derived for one or more CUs, PUs, or any region of data (e.g., any block of data). For example, a palette may comprise (and may consist of) the most dominant pixel values in the current CU, where CU is the region of data for this particular example. The size and the elements of the palette are first transmitted from a video encoder to a video decoder. The size and/or the elements of the palette can be directly coded or predictively coded using the size and/or the elements of the palette in the neighboring CUs (e.g. above and/or left coded CU). After that, the pixel values in the CU are encoded based on the palette according to a certain scanning order. For each pixel location in the CU, a flag (e.g., palette_flag or escape_flag) may be first transmitted to indicate whether the pixel value is included in the palette. For those pixel values that map to an entry in the palette, the palette index associated with that entry is signaled for the given pixel location in the CU. Instead of sending the flag (e.g., palette_flag or escape_flag), for those pixel values that do not exist in the palette, a special index may be assigned to the pixel and the actual pixel value (possibly in quantized form) may be transmitted for the given pixel location in the CU. These pixels are referred to as "escape pixels." An escape pixel can be coded using any existing entropy coding method such as fixed length coding, unary coding, etc. In some examples, one or more techniques described herein may utilize a flag such as palette_flag or escape_flag. In other examples, one or more techniques described herein may not utilize a flag such as palette_flag or escape_flag.

[0038] Samples in a block of video data may be processed (e.g., scanned) using a horizontal raster scanning order or other scanning order. For example, the video encoder may convert a two-dimensional block of palette indices into a one-dimensional array by scanning the palette indices using a horizontal raster scanning order. Likewise, the video decoder may reconstruct a block of palette indices using the horizontal raster scanning order. Accordingly, this disclosure may refer to a previous sample as a sample that precedes the sample currently being coded in the block in the scanning order. It should be appreciated that scans other than a horizontal raster scan, such as vertical raster scanning order, may also be applicable. The example above, as well as other examples set forth in this disclosure, is intended to provide a general description of palette-based video coding.

[0039] FIG. 1 is a block diagram illustrating an example video coding system 10 that may utilize the techniques of this disclosure. As used herein, the term "video coder" refers generically to both video encoders and video decoders. In this disclosure, the terms "video coding" or "coding" may refer generically to video encoding or video decoding. Video encoder 20 and video decoder 30 of video coding system 10 represent examples of devices that may be configured to perform techniques for palette-based video coding and entropy coding (e.g., CABAC) in accordance with various examples described in this disclosure. For example, video encoder 20 and video decoder 30 may be configured to selectively code various blocks of video data, such as CUs or PUs in HEVC coding, using either palette-based coding or non-palette based coding. Non-palette based coding modes may refer to various inter-predictive temporal coding modes or intra-predictive spatial coding modes, such as the various coding modes specified by the HEVC standard.

[0040] As shown in FIG. 1, video coding system 10 includes a source device 12 and a destination device 14. Source device 12 generates encoded video data. Accordingly, source device 12 may be referred to as a video encoding device or a video encoding apparatus. Destination device 14 may decode the encoded video data generated by source device 12. Accordingly, destination device 14 may be referred to as a video decoding device or a video decoding apparatus. Source device 12 and destination device 14 may be examples of video coding devices or video coding apparatuses.

[0041] Source device 12 and destination device 14 may comprise a wide range of devices, including desktop computers, mobile computing devices, notebook (e.g., laptop) computers, tablet computers, set-top boxes, telephone handsets such as so-called

"smart" phones, televisions, cameras, display devices, digital media players, video gaming consoles, in-car computers, or the like.

[0042] Destination device 14 may receive encoded video data from source device 12 via a channel 16. Channel 16 may comprise one or more media or devices capable of moving the encoded video data from source device 12 to destination device 14. In one example, channel 16 may comprise one or more communication media that enable source device 12 to transmit encoded video data directly to destination device 14 in real-time. In this example, source device 12 may modulate the encoded video data according to a communication standard, such as a wireless communication protocol, and may transmit the modulated video data to destination device 14. The one or more communication media may include wireless and/or wired communication media, such as a radio frequency (RF) spectrum or one or more physical transmission lines. The one or more communication media may form part of a packet-based network, such as a local area network, a wide-area network, or a global network (e.g., the Internet). The one or more communication media may include routers, switches, base stations, or other equipment that facilitate communication from source device 12 to destination device 14.

[0043] In another example, channel 16 may include a storage medium that stores encoded video data generated by source device 12. In this example, destination device 14 may access the storage medium via, for example, disk access or card access. The storage medium may include a variety of locally-accessed data storage media such as Blu-ray discs, DVDs, CD-ROMs, flash memory, or other suitable digital storage media for storing encoded video data.

[0044] In a further example, channel 16 may include a file server or another intermediate storage device that stores encoded video data generated by source device 12. In this example, destination device 14 may access encoded video data stored at the file server or other intermediate storage device via streaming or download. The file server may be a type of server capable of storing encoded video data and transmitting the encoded video data to destination device 14. Example file servers include web servers (e.g., for a website), file transfer protocol (FTP) servers, network attached storage (NAS) devices, and local disk drives.

[0045] Destination device 14 may access the encoded video data through a standard data connection, such as an Internet connection. Example types of data connections may include wireless channels (e.g., Wi-Fi connections), wired connections (e.g., DSL, cable modem, etc.), or combinations of both that are suitable for accessing encoded

video data stored on a file server. The transmission of encoded video data from the file server may be a streaming transmission, a download transmission, or a combination of both.

[0046] Source device 12 and destination device 14 may be configured to perform palette-based coding and entropy coding (e.g., CABAC) consistent with this disclosure. The techniques of this disclosure for palette-based coding or CABAC, however, are not limited to wireless applications or settings. The techniques may be applied to video coding in support 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 video data for storage on a data storage medium, decoding of video data stored on a data storage medium, or other applications. In some examples, video coding 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.

[0047] Video coding system 10 illustrated in FIG. 1 is merely an example and the techniques of this disclosure may apply to video coding settings (e.g., video encoding or video decoding) that do not necessarily include any data communication between the encoding and decoding devices. In other examples, data is retrieved from a local memory, streamed over a network, or the like. A video encoding device may encode and store data to memory, and/or a video decoding device may retrieve and decode data from memory. In many examples, the encoding and decoding is performed by devices that do not communicate with one another, but simply encode data to memory and/or retrieve and decode data from memory.

[0048] In the example of FIG. 1, source device 12 includes a video source 18, a video encoder 20, and an output interface 22. In some examples, output interface 22 may include a modulator/demodulator (modem) and/or a transmitter. Video source 18 may include a video capture device, e.g., a video camera, a video archive containing previously-captured video data, a video feed interface to receive video data from a video content provider, and/or a computer graphics system for generating video data, or a combination of such sources of video data.

[0049] Video encoder 20 may encode video data from video source 18. In some examples, source device 12 directly transmits the encoded video data to destination device 14 via output interface 22. In other examples, the encoded video data may also

be stored onto a storage medium or a file server for later access by destination device 14 for decoding and/or playback.

[0050] In the example of FIG. 1, destination device 14 includes an input interface 28, a video decoder 30, and a display device 32. In some examples, input interface 28 includes a receiver and/or a modem. Input interface 28 may receive encoded video data over channel 16. Display device 32 may be integrated with or may be external to destination device 14. In general, display device 32 displays decoded video data. Display device 32 may comprise 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.

[0051] This disclosure may generally refer to video encoder 20 "signaling" or "transmitting" certain information to another device, such as video decoder 30. The term "signaling" or "transmitting" may generally refer to the communication of syntax elements and/or other data used to decode the compressed video data. Such communication may occur in real- or near-real-time. Alternately, such communication may occur over a span of time, such as might occur when storing syntax elements to a computer-readable storage medium in an encoded bitstream at the time of encoding, which then may be retrieved by a decoding device at any time after being stored to this medium. Thus, while video decoder 30 may be referred to as "receiving" certain information, the receiving of information does not necessarily occur in real- or near-real-time and may be retrieved from a medium at some time after storage.

[0052] Video encoder 20 and video decoder 30 each may be implemented as any of a variety of suitable circuitry, such as one or more microprocessors, digital signal processors (DSPs), application-specific integrated circuits (ASICs), field-programmable gate arrays (FPGAs), discrete logic, hardware, or any combinations thereof. If the techniques are implemented partially in software, a device may store instructions for the software in a suitable, non-transitory computer-readable storage medium and may execute the instructions in hardware using one or more processors to perform the techniques of this disclosure. Any of the foregoing (including hardware, software, a combination of hardware and software, etc.) may be considered to be one or more processors. 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.

[0053] In some examples, video encoder 20 and video decoder 30 operate according to a video compression standard, such as HEVC standard mentioned above, and described in the HEVC standard. In addition to the base HEVC standard, there are ongoing efforts to produce scalable video coding, multiview video coding, and 3D coding extensions for HEVC. In addition, palette-based coding modes, e.g., as described in this disclosure, may be provided for extension of the HEVC standard. In some examples, the techniques described in this disclosure for palette-based coding may be applied to encoders and decoders configured to operation according to other video coding standards. Accordingly, application of a palette-based coding mode for coding of coding units (CUs) or prediction units (PUs) in an HEVC codec is described for purposes of example.

[0054] In HEVC and other video coding standards, a video sequence typically includes a series of pictures. Pictures may also be referred to as "frames." A picture may include three sample arrays, denoted S_L, S_{Cb} and S_{Cr}. S_L is a two-dimensional array (i.e., a block) of luma samples. S_{Cb} is a two-dimensional array of Cb chrominance samples. S_{Cr} is a two-dimensional array of Cr chrominance samples. Chrominance samples may also be referred to herein as "chroma" samples. In other instances, a picture may be monochrome and may only include an array of luma samples.

[0055] To generate an encoded representation of a picture, video encoder 20 may generate a set of coding tree units (CTUs). Each of the CTUs may be a coding tree block of luma samples, two corresponding coding tree blocks of chroma samples, and syntax structures used to code the samples of the coding tree blocks. A coding tree block may be an NxN block of samples. A CTU may also be referred to as a "tree block" or a "largest coding unit" (LCU). The CTUs of HEVC may be broadly analogous to the macroblocks of other standards, such as H.264/AVC. However, a CTU is not necessarily limited to a particular size and may include one or more coding units (CUs). A slice may include an integer number of CTUs ordered consecutively in the raster scan. A coded slice may comprise a slice header and slice data. The slice header of a slice may be a syntax structure that includes syntax elements that provide information about the slice. The slice data may include coded CTUs of the slice.

[0056] This disclosure may use the term "video unit" or "video block" or "block" to refer to one or more sample blocks and syntax structures used to code samples of the one or more blocks of samples. Example types of video units or blocks may include CTUs, CUs, PUs, transform units (TUs), macroblocks, macroblock partitions, and so

on. In some contexts, discussion of PUs may be interchanged with discussion of macroblocks or macroblock partitions.

[0057] To generate a coded CTU, video encoder 20 may recursively perform quad-tree partitioning on the coding tree blocks of a CTU to divide the coding tree blocks into coding blocks, hence the name "coding tree units." A coding block is an NxN block of samples. A CU may be a coding block of luma samples and two corresponding coding blocks of chroma samples of a picture that has a luma sample array, a Cb sample array and a Cr sample array, and syntax structures used to code the samples of the coding blocks. Video encoder 20 may partition a coding block of a CU into one or more prediction blocks. A prediction block may be a rectangular (i.e., square or non-square) block of samples on which the same prediction is applied. A prediction unit (PU) of a CU may be a prediction block of luma samples, two corresponding prediction blocks of chroma samples of a picture, and syntax structures used to predict the prediction block samples. Video encoder 20 may generate predictive luma, Cb and Cr blocks for luma, Cb and Cr prediction blocks of each PU of the CU.

[0058] Video encoder 20 may use intra prediction or inter prediction to generate the predictive blocks for a PU. If video encoder 20 uses intra prediction to generate the predictive blocks of a PU, video encoder 20 may generate the predictive blocks of the PU based on decoded samples of the picture associated with the PU.

[0059] If video encoder 20 uses inter prediction to generate the predictive blocks of a PU, video encoder 20 may generate the predictive blocks of the PU based on decoded samples of one or more pictures other than the picture associated with the PU. Video encoder 20 may use uni-prediction or bi-prediction to generate the predictive blocks of a PU. When video encoder 20 uses uni-prediction to generate the predictive blocks for a PU, the PU may have a single motion vector (MV). When video encoder 20 uses bi-prediction to generate the predictive blocks for a PU, the PU may have two MVs.

[0060] After video encoder 20 generates predictive blocks (e.g., predictive luma, Cb and Cr blocks) for one or more PUs of a CU, video encoder 20 may generate residual blocks for the CU. Each sample in a residual block of the CU may indicate a difference between a sample in a predictive block of a PU of the CU and a corresponding sample in a coding block of the CU. For example, video encoder 20 may generate a luma residual block for the CU. Each sample in the CU's luma residual block indicates a difference between a luma sample in one of the CU's predictive luma blocks and a corresponding sample in the CU's original luma coding block. In addition, video

encoder 20 may generate a Cb residual block for the CU. Each sample in the CU's Cb residual block may indicate a difference between a Cb sample in one of the CU's predictive Cb blocks and a corresponding sample in the CU's original Cb coding block. Video encoder 20 may also generate a Cr residual block for the CU. Each sample in the CU's Cr residual block may indicate a difference between a Cr sample in one of the CU's predictive Cr blocks and a corresponding sample in the CU's original Cr coding block.

[0061] Furthermore, video encoder 20 may use quad-tree partitioning to decompose the residual blocks (e.g., luma, Cb and Cr residual blocks) of a CU into one or more transform blocks (e.g., luma, Cb and Cr transform blocks). A transform block may be a rectangular block of samples on which the same transform is applied. A transform unit (TU) of a CU may be a transform block of luma samples, two corresponding transform blocks of chroma samples, and syntax structures used to transform the transform block samples. Thus, each TU of a CU may be associated with a luma transform block, a Cb transform block, and a Cr transform block. The luma transform block associated with the TU may be a sub-block of the CU's luma residual block. The Cb transform block may be a sub-block of the CU's Cb residual block. The Cr transform block may be a sub-block of the CU's Cr residual block.

[0062] Video encoder 20 may apply one or more transforms to a transform block to generate a coefficient block for a TU. A coefficient block may be a two-dimensional array of transform coefficients. A transform coefficient may be a scalar quantity. For example, video encoder 20 may apply one or more transforms to a luma transform block of a TU to generate a luma coefficient block for the TU. Video encoder 20 may apply one or more transforms to a Cb transform block of a TU to generate a Cb coefficient block for the TU. Video encoder 20 may apply one or more transforms to a Cr transform block of a TU to generate a Cr coefficient block for the TU.

[0063] After generating a coefficient block (e.g., a luma coefficient block, a Cb coefficient block or a Cr coefficient block), video encoder 20 may quantize the coefficient block. Quantization generally refers to a process in which transform coefficients are quantized to possibly reduce the amount of data used to represent the transform coefficients, providing further compression. After video encoder 20 quantizes a coefficient block, video encoder 20 may entropy encode syntax elements indicating the quantized transform coefficients. For example, video encoder 20 may perform

Context-Adaptive Binary Arithmetic Coding (CABAC) on the syntax elements indicating the quantized transform coefficients.

[0064] With respect to CABAC, as an example, video encoder 20 and video decoder 30 may select a probability model (also referred to as a context model) to code symbols associated with a block of video data based on context. For example, a context model (Ctx) may be an index or offset that is applied to select one of a plurality of different contexts, each of which may correspond to a particular probability model. Accordingly, a different probability model is typically defined for each context. After encoding or decoding the bin, the probability model is further updated based on a value of the bin to reflect the most current probability estimates for the bin. For example, a probability model may be maintained as a state in a finite state machine. Each particular state may correspond to a specific probability value. The next state, which corresponds to an update of the probability model, may depend on the value of the current bin (e.g., the bin currently being coded). Accordingly, the selection of a probability model may be influenced by the values of the previously coded bins, because the values indicate, at least in part, the probability of the bin having a given value. The context coding process described above may generally be referred to as a context-adaptive coding mode.

[0065] Hence, video encoder 20 may encode a target symbol using a probability model. Likewise, video decoder 30 may parse a target symbol using the probability model. In some instances, video encoder 20 may code syntax elements using a combination of context adaptive and non-context adaptive coding. For example, video encoder 20 may context code bins by selecting a probability model or "context model" that operates on context to code some of the bins. In contrast, for other bins, video encoder 20 may bypass code bins by bypassing, or omitting the regular arithmetic coding process when coding the bins. In such examples, video encoder 20 may use a fixed probability model to bypass code the bins. That is, bypass coded bins do not include context or probability updates.

[0066] Video encoder 20 may output a bitstream that includes the entropy-encoded syntax elements. The bitstream may also include syntax elements that are not entropy encoded. The bitstream may include a sequence of bits that forms a representation of coded pictures and associated data. The bitstream may comprise a sequence of network abstraction layer (NAL) units. Each of the NAL units includes a NAL unit header and encapsulates a raw byte sequence payload (RBSP). The NAL unit header may include a syntax element that indicates a NAL unit type code. The NAL unit type code specified

by the NAL unit header of a NAL unit indicates the type of the NAL unit. A RBSP may be a syntax structure containing an integer number of bytes that is encapsulated within a NAL unit. In some instances, an RBSP includes zero bits.

[0067] Different types of NAL units may encapsulate different types of RBSPs. For example, a first type of NAL unit may encapsulate an RBSP for a picture parameter set (PPS), a second type of NAL unit may encapsulate an RBSP for a coded slice, a third type of NAL unit may encapsulate an RBSP for supplemental enhancement information (SEI), and so on. NAL units that encapsulate RBSPs for video coding data (as opposed to RBSPs for parameter sets and SEI messages) may be referred to as video coding layer (VCL) NAL units.

[0068] Video decoder 30 may receive a bitstream generated by video encoder 20. In addition, video decoder 30 may parse the bitstream to decode syntax elements from the bitstream. Video decoder 30 may reconstruct the pictures of the video data based at least in part on the syntax elements decoded from the bitstream. The process to reconstruct the video data may be generally reciprocal to the process performed by video encoder 20. For instance, video decoder 30 may use MVs of PUs to determine predictive blocks for the inter-predicted PUs of a current CU. Likewise, video decoder 30 may generate intra-predicted blocks for PU's of a current CU. In addition, video decoder 30 may inverse quantize transform coefficient blocks associated with TUs of the current CU. Video decoder 30 may perform inverse transforms on the transform coefficient blocks to reconstruct transform blocks associated with the TUs of the current CU. Video decoder 30 may reconstruct the coding blocks of the current CU by adding the samples of the predictive blocks for PUs of the current CU to corresponding residual values obtained from inverse quantization and inverse transformation of the transform blocks of the TUs of the current CU. By reconstructing the coding blocks for each CU of a picture, video decoder 30 may reconstruct the picture.

[0069] In some examples, video encoder 20 and video decoder 30 may be configured to perform palette-based coding. For example, in palette based coding, rather than performing the intra-predictive or inter-predictive coding techniques described above, video encoder 20 and video decoder 30 may code a so-called palette as a table of colors or pixel values representing the video data of a particular area (e.g., a given block). In this way, rather than coding actual pixel values or their residuals for a current block of video data, the video coder may code index values for one or more of the pixels values

of the current block, where the index values indicate entries in the palette that are used to represent the pixel values of the current block.

[0070] For example, video encoder 20 may encode a block of video data by determining a palette for the block, locating an entry in the palette to represent the value of each pixel, and encoding the palette and the index values for the pixels relating the pixel value to the palette. Video decoder 30 may obtain, from an encoded bitstream, a palette for a block, as well as index values for the pixels of the block. Video decoder 30 may match the index values of the individual pixels to entries of the palette to reconstruct the pixel values of the block. In instances where the index value associated with an individual pixel does not match any index value of the corresponding palette for the block, video decoder 30 may identify such a pixel as an escape pixel, for the purposes of palette-based coding.

[0071] As described in more detail below, the basic idea of palette-based coding is that, for a given block of video data to be coded, video encoder 20 may derive a palette that includes the most dominant pixel values in the current block. For instance, the palette may refer to a number of pixel values which are determined or assumed to be dominant and/or representative for the current CU. Video encoder 20 may first transmit the size and the elements of the palette to video decoder 30. Additionally, video encoder 20 may encode the pixel values in the given block according to a certain scanning order. For each pixel included in the given block, video encoder 20 may signal the index value that maps the pixel value to a corresponding entry in the palette. If the pixel value is not included in the palette (i.e., no palette entry exists that specifies a particular pixel value of the palette-coded block), then such a pixel is defined as an "escape pixel." In accordance with palette-based coding, video encoder 20 may encode and signal an index value that is reserved for an escape pixel. In some examples, video encoder 20 may also encode and signal the pixel value (or a quantized version thereof) for an escape pixel included in the given block. For example, video decoder 30 may be configured to determine whether a pixel value matches or is otherwise close to a palette entry based on a distortion metric (e.g., MSE, SAD, and the like

[0072] Upon receiving the encoded video bitstream signaled by video encoder 20, video decoder 30 may first determine the palette based on the information received from video encoder 20. Video decoder 30 may then map the received index values associated with the pixel locations in the given block to entries of the palette to reconstruct the pixel values of the given block. In some instances, video decoder 30 may determine that a

pixel of a palette-coded block is an escape pixel, such as by determining that the pixel is palette-coded with an index value reserved for escape pixels. In instances where video decoder 30 identifies an escape pixel in a palette-coded block, video decoder 30 may receive the pixel value (or a quantized version thereof) for an escape pixel included in the given block. Video decoder 30 may reconstruct the palette-coded block by mapping the individual pixel values to the corresponding palette entries, and by using the pixel value (or a quantized version thereof) to reconstruct any escape pixels included in the palette-coded block.

[0073] A stated above, in an example palette-coding mode, a palette may include entries numbered by an index. Each entry may represent color component values or intensities (for example, in color spaces such as YCbCr, RGB, YUV, CMYK, or other formats), which can be used as a predictor for a block or as final reconstructed block samples. As described in standard submission document JCTVC-Q0094 (Wei Pu et al., "AHG10: Suggested Software for Palette Coding based on RExt6.0," JCTVC-Q0094, Valencia, ES, 27 March – 4 April 2014) a palette may include entries that are copied from a predictor palette. A predictor palette may include palette entries from blocks previously coded using palette mode or other reconstructed samples. For each entry in the predictor palette, a binary flag is sent to indicate whether that entry is copied to the current palette (indicated by flag = 1). This is referred to as the binary palette prediction vector. Additionally the current palette may comprise (e.g., consist of) new entries signaled explicitly. The number of new entries may be signaled as well.

[0074] As another example, in palette mode, a palette may include entries numbered by an index representing color component values that may be used as predictors for block samples or as final reconstructed block samples. Each entry in the palette may contain, for example, one luma component (e.g., luma value), two chroma components (e.g., two chroma values), or three color components (e.g., RGB, YUV, etc.). Previously decoded palette entries may be stored in a list. This list may be used to predict palette entries in the current palette mode CU, for example. A binary prediction vector may be signaled in the bitstream to indicate which entries in the list are re-used in the current palette. In some examples, run-length coding may be used to compress the binary palate predictor. For example, a run-length value may be coded using 0th order Exp-Golomb code.

[0075] In this disclosure, it will be assumed that each palette entry specifies the values for all color components of a sample. However, the concepts of this disclosure are applicable to using a separate palette and/or a separate palette entry for each color

component. Also, it is assumed that samples in a block are processed using horizontal raster scanning order. However, other scans such as vertical raster scanning order are also applicable. As mentioned above, a palette may contain predicted palette entries, for example, predicted from the palette(s) used to code the previous block(s), and the new entries which may be specific for the current block and are signaled explicitly. The encoder and decoder may know the number of the predicted and new palette entries and a sum of them may indicate the total palette size in a block.

[0076] As proposed in the example of JCTVC-Q0094 cited above, each sample in a block coded with the palette may belong to one of the three modes, as set forth below:

- Escape mode. In this mode, the sample value is not included into a palette as a palette entry and the quantized sample value is signaled explicitly for all color components. It is similar to the signaling of the new palette entries, although for new palette entries, the color component values are not quantized.
- CopyAbove mode (also called CopyFromTop mode). In this mode, the palette entry index for the current sample is copied from the sample located directly above the current sample in a block of samples. In other examples, for copy above mode, a block of video data may be transposed so that the sample above the block is actually the sample to the left of the block.
- Value mode (also called index mode). In this mode, the value of the palette entry index is explicitly signaled.

[0077] As described herein, a palette entry index may be referred as a palette index or simply index. These terms can be used interchangeably to describe techniques of this disclosure. In addition, as described in greater detail below, a palette index may have one or more associated color or intensity values. For example, a palette index may have a single associated color or intensity value associated with a single color or intensity component of a pixel (e.g., an Red component of RGB data, a Y component of YUV data, or the like). In another example, a palette index may have multiple associated color or intensity values. In some instances, palette-based video coding may be applied to code monochrome video. Accordingly, "color value" may generally refer to any color or non-color component used to generate a pixel value.

[0078] A run value may indicate a run of palette index values that are coded using the same palette-coding mode. For example, with respect to Value mode, a video coder (e.g., video encoder 20 or video decoder 30) may code an index value and a run value that indicates a number of consecutive subsequent samples in a scan order that have the

same index value and that are being coded with the palette index. With respect to CopyAbove mode, the video coder may code an indication that an index value for the current sample value is the same as an index value of an above-neighboring sample (e.g., a sample that is positioned above the sample currently being coded in a block) and a run value that indicates a number of consecutive subsequent samples in a scan order that also copy an index value from an above-neighboring sample. Accordingly, in the examples above, a run of palette index values refers to a run of palette values having the same value or a run of index values that are copied from above-neighboring samples.

[0079] Hence, the run may specify, for a given mode, the number of subsequent samples that belong to the same mode. In some instances, signaling an index value and a run value may be similar to run-length coding. In an example for purposes of illustration, a string of consecutive palette index values of an index block corresponding to a block of video data may be 0, 2, 2, 2, 2, 5. Each index value corresponds to a sample in the block of video data. In this example, a video coder may code the second sample (e.g., the first palette index value of "2") using Value mode. After coding an index value of 2, the video coder may code a run of 3, which indicates that the three subsequent samples also have the same palette index value of 2. In a similar manner, coding a run of four palette indices after coding an index using CopyAbove mode may indicate that a total of five palette indices are copied from the corresponding palette index values in the row above the sample position currently being coded.

[0080] Using the palette, video encoder 20 and/or video decoder 30 may be configured to code a block of samples (e.g., a block of video data) into an index block, where the index block is a block including index values that map to one or more palette entries, and, in some examples, one or more escape pixel values. Video encoder 20 may be configured to entropy encode the index block to compress the index block. Similarly, video decoder 30 may be configured to entropy decode an encoded index block to generate the index block from which video decoder 30 may generate a block of samples (e.g., the block of video data encoded by encoder 20). For example, run-length based entropy coding may be used to compress and decompress the index block. In some examples, video encoder 20 and video decoder 30 may be configured to respectively entropy encode and decode the index block using CABAC.

[0081] To apply CABAC coding to information (e.g., a syntax element, an index block such as the index values of the index block, or other information), a video coder (e.g., video encoder 20 and video decoder 30) may perform binarization on the information.

Binarization refers to the process of converting information into a series of one or more bits. Each series of one or more bits may be referred to as "bins." Binarization is a lossless process and may include one or a combination of the following coding techniques: fixed length coding, unary coding, truncated unary coding, truncated Rice coding, Golomb coding, exponential Golomb coding, Golomb-Rice coding, any form of Golomb coding, any form of Rice coding, and any form of entropy coding. For example, binarization may include representing the integer value of 5 as 00000101 using an 8-bit fixed length technique or as 11110 using a unary coding technique.

[0082] After binarization, a video coder may identify a coding context. The coding context may identify probabilities of coding bins having particular values. For instance, a coding context may indicate a 0.7 probability of coding a 0-valued bin and a 0.3 probability of coding a 1-valued bin. After identifying the coding context, the video coder may arithmetically code that bin based on the context, which is known as context mode coding. Bins coded using a CABAC context mode coding may be referred to as "context bins."

[0083] Further, rather than performing context mode coding on all bins, a video coder (e.g., video encoder 20 and video decoder 30) may code some bins using bypass CABAC coding (e.g., bypass mode coding). Bypass mode coding refers to the process of arithmetically coding a bin without using an adaptive context (e.g., a coding context). That is, the bypass coding engine does not select contexts and may assume a probability of 0.5 for both symbols (0 and 1). Although bypass mode coding may not be as bandwidth-efficient as context mode coding, it may be computationally less expensive to perform bypass mode coding on a bin rather than to perform context mode coding on the bin. Further, performing bypass mode coding may allow for a higher degree of parallelization and throughput. Bins coded using bypass mode coding may be referred to as "bypass bins."

[0084] Video encoder 20 and video decoder 30 may be configured with a CABAC coder (e.g., a CABAC encoder and a CABAC decoder, respectively). A CABAC coder may include a context mode coding engine to perform CABAC context mode coding and a bypass mode coding engine to perform bypass mode coding. If a bin is context mode coded, the context mode coding engine is used to code this bin. The context mode coding engine may need more than two processing cycles to code a single bin. However, with proper pipeline design, a context mode coding engine may only need

n+M cycles to encode n bins, where M is the overhead to start the pipeline. M is usually greater than 0.

[0085] At the start of the CABAC coding process (i.e., every switch from bypass mode to context mode and vice versa), pipeline overhead is introduced. If a bin is bypass mode coded, the bypass mode coding engine is used to code this bin. The bypass mode coding engine may be expected to need only one cycle to code n-bit information, where n may be greater than one. Thus, the total number of cycles to code a set of bypass bins and context bins may be reduced if all of the bypass bins within the set are coded together and all of the context bins within the set are coded together. In particular, coding the bypass bins together before or after transitioning to context mode coding can save the overhead required to restart the context mode coding engine. For example, video encoder 20 and video decoder 30 may be configured to switch between bypass mode to context mode while respectively encoding or decoding a block of video data using palette mode. In another example, video encoder 20 and video decoder 30 may be configured to reduce the number of times the encoding or decoding process switches between bypass mode to context mode when encoding or decoding a block of video data using palette mode.

[0086] The techniques described in this disclosure may include techniques for various combinations of one or more of signaling palette-based video coding modes, transmitting palettes, deriving palettes, signaling scanning order, deriving scanning order, and transmitting palette-based video coding maps and other syntax elements. For example, techniques of this disclosure may be directed to entropy coding palette information. In some examples, the techniques of this disclosure may, among other things, be used to increase coding efficiency and reduce coding inefficiencies associated with palette-based video coding. Accordingly, as described in greater detail below, the techniques of this disclosure may, in some instances, improve efficiency and improve bitrate when coding video data using a palette mode.

[0087] As described above, in the current palette mode design in screen content coding, the syntax elements of **palette_index_idc** and **palette_escape_val** are CABAC bypass coded, and are interleaved with other syntax elements (e.g., **palette_run_msb_id_plus1**) that are CABAC context coded. However, it may be beneficial to group the bypass coded information (e.g., syntax elements) together, which may improve coding efficiency and/or reduce codec complexity.

[0088] The syntax element of **palette_index_idc** may be an indication of an index to the array represented by currentPaletteEntries, as defined in, for example, JCTVC-S1005. The value of palette_index_idc may be in the range of 0 to (adjustedIndexMax-1), inclusive. The syntax element of **palette_escape_val** may specify the quantized escape coded sample value for a component, as defined in, for example, JCTVC-S1005. **palette_run_msb_id_plus1** minus 1 may specify the index of the most significant bit in the binary representation of paletteRun, as defined in, for example, JCTVC-S1005. The variable paletteRun may specify the number of consecutive locations minus 1 with the same palette index as the position in the above row when palette_run_type_flag is equal to COPY_ABOVE_MODE or specifies the number of consecutive locations minus 1 with the same palette index when palette_run_type_flag is equal to COPY_INDEX_MODE, as defined in, for example, JCTVC-S1005. Additional details regarding **palette_index_idc**, **palette_escape_val**, **palette_run_msb_id_plus1**, currentPaletteEntries, adjustedIndexMax, and paletteRun may be found in JCTVC-S1005.

[0089] In some examples, this disclosure describes a method of grouping all of the syntax elements **palette_index_idc** at the front of the palette index block coding section to improve CABAC throughput. For instance, video encoder 20 may be configured to encode all of the syntax elements **palette_index_idc** at the front of the palette index block coding section. For example, video encoder 20 may be configured to encode all of the syntax elements **palette_index_idc** before encoding syntax elements to be context mode encoded. Similarly, video decoder 30 may be configured to decode all of the syntax elements **palette_index_idc** at the front of the palette index block coding section. For example, video decoder 30 may be configured to decode all of the syntax elements **palette_index_idc** before decoding context mode encoded syntax elements.

[0090] As another example, video encoder 20 may be configured to bypass mode encode all of the syntax elements **palette_index_idc** at the front of the palette index block coding section such that all of the syntax elements **palette_index_idc** are encoded before encoding syntax element(s) related to palette run type (e.g., CopyAbove mode or index mode) and/or run length (e.g., palette_run_msb_id_plus1). Similarly, video decoder 30 may be configured to decode all of the syntax elements **palette_index_idc** for a block at the front of the palette index block coding section of the block such that all of the syntax elements **palette_index_idc** are decoded before decoding syntax

element(s) related to palette run type (e.g., CopyAbove mode or index mode) and/or run length (e.g., palette_run_msb_id_plus1).

[0091] syntax element(s) related to palette run type (e.g., CopyAbove mode or index mode) and/or run length (e.g., palette_run_msb_id_plus1)

[0092] As another example, example, video encoder 20 may be configured to encode all of the syntax elements **palette_index_idc** before context encoding syntax element(s) related to palette run type (e.g., CopyAbove mode or index mode) and/or run length (e.g., palette_run_msb_id_plus1). Similarly, video decoder 30 may be configured to decode all of the syntax elements **palette_index_idc** before context decoding syntax element(s) related to palette run type (e.g., CopyAbove mode or index mode) and/or run length (e.g., palette_run_msb_id_plus1).

[0093] As another example, video encoder 20 may be configured to encode all of the syntax elements **palette_index_idc** within the palette block coding section before encoding syntax elements to be context mode encoded. Similarly, video decoder 30 may be configured to decode all of the syntax elements **palette_index_idc** within the palette block coding section before decoding context mode encoded syntax elements. As another example, video encoder 20 may be configured to encode all of the syntax elements **palette_index_idc** within the palette block coding section before context encoding syntax element(s) related to palette run type (e.g., CopyAbove mode or index mode) and/or run length (e.g., palette_run_msb_id_plus1). Similarly, video decoder 30 may be configured to decode all of the syntax elements **palette_index_idc** within the palette block coding section before context decoding syntax element(s) related to palette run type (e.g., CopyAbove mode or index mode) and/or run length (e.g., palette_run_msb_id_plus1).

[0094] In general, video encoder 20 and video decoder 30 may be configured to not interleave the encoding or decoding of **palette_index_idc** in bypass mode with syntax elements that are to be encoded or decoded using context mode, respectively. For example, video encoder 20 and video decoder 30 may be configured to not interleave the encoding or decoding of **palette_index_idc** in bypass mode with syntax element(s) related to palette run type (e.g., CopyAbove mode or index mode) and/or run length (e.g., palette_run_msb_id_plus1) that are to be encoded or decoded using context mode, respectively. As another example, video encoder 20 may be configured to bypass encode all instances of the **palette_index_idc** syntax element before context encoding a syntax element that requires context mode. Similarly, video decoder 30 may be

configured to bypass decode all instances of the **palette_index_idc** syntax element before context decoding a syntax element that requires context mode. As another example, video encoder 20 may be configured to bypass encode all instances of the **palette_index_idc** syntax element before context encoding syntax element(s) related to palette run type (e.g., CopyAbove mode or index mode) and/or run length (e.g., palette_run_msb_id_plus1). Similarly, video decoder 30 may be configured to bypass decode all instances of the **palette_index_idc** syntax element before context decoding syntax element(s) related to palette run type (e.g., CopyAbove mode or index mode) and/or run length (e.g., palette_run_msb_id_plus1).

[0095] Video encoder 20 and video decoder 30 may also respectively encode and decode a value representing the number of occurrences of **palette_index_idc**. Video encoder 20 and video decoder 30 may use the value representing the number of occurrences of **palette_index_idc** to respectively encode or decode each of the syntax elements **palette_index_idc**. The techniques described in this disclosure may also remove the redundancy of palette run length related syntax elements, and remove the redundancy of **palette_run_type_flag** and **palette_index_idc**.

[0096] In some examples, this disclosure describes a method of grouping all of the syntax elements **palette_escape_val** at the front of the palette index block coding section of a block (e.g., a PU or a CU) to improve CABAC throughput. For instance, video encoder 20 may be configured to encode all of the syntax elements **palette_escape_val** at the front of the palette index block coding section of a block. For example, video encoder 20 may be configured to bypass mode encode all of the syntax elements **palette_escape_val** at the front of the palette index block coding section such that all of the syntax elements **palette_escape_val** are encoded before encoding syntax element(s) related to palette run type (e.g., CopyAbove mode or index mode) and/or run length (e.g., palette_run_msb_id_plus1). Similarly, video decoder 30 may be configured to decode all of the syntax elements **palette_escape_val** for a block at the front of the palette index block coding section of the block such that all of the syntax elements **palette_escape_val** are decoded before decoding syntax element(s) related to palette run type (e.g., CopyAbove mode or index mode) and/or run length (e.g., palette_run_msb_id_plus1). As another example, video encoder 20 may be configured to encode all of the syntax elements **palette_escape_val** before encoding syntax elements to be context mode encoded. For example, video encoder 20 may be configured to encode all of the syntax elements **palette_escape_val** before context

encoding syntax element(s) related to palette run type (e.g., CopyAbove mode or index mode) and/or run length (e.g., palette_run_msb_id_plus1). Similarly, video decoder 30 may be configured to decode all of the syntax elements **palette_escape_val** for a block at the front of the palette index block coding section of the block. For example, video decoder 30 may be configured to decode all of the syntax elements **palette_escape_val** before decoding context mode encoded syntax elements in a block.

[0097] As another example, video encoder 20 may be configured to encode all of the syntax elements **palette_escape_val** within the palette block coding section of a block before encoding syntax elements to be context mode encoded. Similarly, video decoder 30 may be configured to decode all of the syntax elements **palette_escape_val** within the palette block coding section of a block before decoding context mode encoded syntax elements of the block.

[0098] In general, video encoder 20 and video decoder 30 may be configured to not interleave the encoding or decoding of **palette_escape_val** for a block (e.g., a PU or a CU) in bypass mode with syntax elements that are to be encoded or decoded using context mode for the block, respectively. For example, video encoder 20 and video decoder 30 may be configured to not interleave the encoding or decoding of **palette_escape_val** in bypass mode with syntax element(s) related to palette run type (e.g., CopyAbove mode or index mode) and/or run length (e.g., palette_run_msb_id_plus1) that are to be encoded or decoded using context mode, respectively. As another example, video encoder 20 may be configured to bypass encode all instances of the **palette_escape_val** syntax element for a block before context encoding a syntax element that requires context mode. Similarly, video decoder 30 may be configured to bypass decode all instances of the **palette_escape_val** syntax element of a block (e.g., a PU or a CU) before context decoding a syntax element that requires context mode of the block.

[0099] Video encoder 20 and video decoder 30 may also respectively encode and decode a value representing the number of occurrences of **palette_escape_val** for a block. Video encoder 20 and video decoder 30 may use the value representing the number of occurrences of **palette_escape_val** to respectively encode or decode each of the syntax elements **palette_escape_val** for the block. The techniques described in this disclosure may reduce the dynamic range of **palette_index_idc** for a block, which may result in improved coding efficiency.

[0100] The techniques, aspects, and/or examples described herein may be utilized in conjunction with one another in any combination or separately from one another. For instance, video encoder 20 and video decoder 30 may be configured to perform any one or any suitable combination of one or more of the techniques, aspects, and/or examples described herein.

[0101] In some examples, to improve CABAC throughput, a video coder (e.g., video encoder 20) may be configured to group all of the occurrences of the syntax element **palette_index_idc** as described above. For example, the video coder (e.g., video encoder 20) may be configured to group all of the occurrences of the syntax element **palette_index_idc** in the current block (e.g., a PU or a CU) at the front of the index coding section for the current block. Similarly, a video decoder (e.g., video decoder 30) may be configured to decode all of the syntax elements **palette_index_idc** as described above. FIG. 7 illustrates one example where video encoder 20 may be configured to group all of the occurrences of the syntax element **palette_index_idc** in the current block (e.g., a CU) at, for example, the front of the index coding block relative to R. Joshi and J. Xu, "High efficient video coding (HEVC) screen content coding: Draft 2," JCTVC-S1005, Section 7.3.3.8. This aspect of the disclosure is referred to as Aspect 1. Specifically, FIG. 7 illustrates an example of video encoder 20 relocating an instance of the syntax element **palette_index_idc** to the front of the index coding block (which may also be referred to as the palette block coding section or the front of the index coding block). By relocating the illustrated instance of the syntax element **palette_index_idc**, video encoder 20 may be configured to improve CABAC throughput by coding all instances of the syntax element **palette_index_idc** using bypass mode and switching over to context mode to code palette information occurring after all instances of the syntax element **palette_index_idc** in the index coding block are bypass mode encoded.

[0102] According to the disclosure of JCTVC-S1005, one instance of **palette_index_idc** would be coded in bypass mode, then one instance of a syntax element related to palette run type and one instance of **palette_run_msb_id_plus1** would be coded in context mode, and the process would repeat while (scanPos < nCbS * nCbS), meaning that the video encoder would switch back and forth between bypass mode coding and context mode coding because the syntax elements to be coded using bypass mode are not grouped together. This is depicted in FIG. 7 with the ellipse immediately below the while loop of "while(scanPos<nCbS * nCbS)" (i.e., the ellipse excludes the information showing that a syntax element related to palette run type is

encoded using context mode), the box surrounding the if-statement with the consequent of the **palette_index_idc** syntax element being under the while loop of "while(scanPos<nCbS * nCbS)," and the subsequent pseudo-code. However, as described above, FIG. 7 also depicts Aspect 1 of this disclosure, which is the grouping (which may also be referred to as the re-location) of one or more instances of the syntax element **palette_index_idc** to, for example, the front of the index coding block. By re-locating one or more syntax elements (e.g., or other palette information) to be encoded using bypass mode, a video encoder (e.g., video encoder 20) may increase the throughput of entropy coding by reducing the number of times the video encoder or video decoder must switch between bypass mode encoding and context mode encoding. Similarly, by re-locating one or more syntax elements in such a manner, the throughput of a video decoder (e.g., video decoder 30) may increase because the number of times the video decoder must switch between bypass mode decoding and context mode decoding is reduced. In some examples of the techniques described in this disclosure, all instances of the **palette_index_idc** syntax element would be coded in bypass mode before an instance of **palette_run_msb_id_plus1** would be coded in context mode.

[0103] In some examples, video encoder 20 may be configured to signal the number of occurrences (e.g., instances) of the syntax element **palette_index_idc** using a syntax element named, for example, **num_palette_index**. For example, video encoder 20 may signal a value for **num_palette_index** in a bitstream, where the value is representative of the number of occurrences of the syntax element **palette_index_idc**. In some examples, video encoder 20 may be configured to not signal an index value as **palette_index_idc.** In such examples, video decoder 30 may be configured to infer the index value. For example, an occurrence of **palette_index_idc** may be counted in **num_palette_index**, which may be equal to the number of times a run type (e.g., COPY_INDEX_MODE) occurs in a particular block. Even when a run type (e.g., COPY_INDEX_MODE) is inferred or **palette_index_idc** is inferred, it still counts towards **num_palette_index**. As used herein, reference to a number of indices parsed, decoded, or remaining to be decoded may, in some examples, refer to the number of COPY_INDEX_MODE irrespective of whether the mode or the index is inferred. Video decoder 30 may be configured to determine the number of occurrences (e.g., instances) of syntax element **palette_index_idc** by, for example, decoding an encoded value corresponding to the **num_palette_index** syntax element from a bitstream. This aspect of the disclosure is referred to as Aspect 2. Video encoder 20 and video decoder

30 may be configured to implement Aspect 1 with Aspect 2 or without Aspect 2. Syntax wise, Aspect 2 may, according to some examples, be defined as:

indices_idc_coding() {	
num_palette_index	ae(v)
for (i = 0; i < num_palette_index; i++)	
palette_index_idc	ae(v)
}	

[0104] In some examples, video encoder 20 and video decoder 30 may be configured to implement (e.g., by enabling) Aspects 1 and 2 only when the variable indexMax is greater than 1. This aspect of the disclosure is referred to as Aspect 3. The variable indexMax may specify the number of distinct values that a palette index has for the current coding unit. In some examples, indexMax may refer to the quantity of (palette size + palette_escape_val_present_flag).

[0105] In some examples, Aspects 1 and 2 may be disabled when : (a) there is no escape pixel (i.e. palette_escape_val_present_flag = = 0) in the current block and the palette size is less than 2; or (b) there may be at least one escape pixel (i.e. palette_escape_val_present_flag = = 1) in the current block and the palette size is equal to 0. In other examples, video encoder 20 and video decoder 30 may be configured to implement (e.g., by enabling) Aspects 1 and 2 only when the variable indexMax is greater than 2. Similarly, in examples where indexMax is equal to (palette size + palette_escape_val_present_flag), Aspects 1 and 2 may be enabled (e.g., implemented) when indexMax is greater than 1. For example, if palette size is 0 and palette_escape_val_present_flag is 1, all the pixels in the block are escape pixels; and, as such, the indices are already known. As another example, if palette_escape_val_present_flag is 0 and palette size is 1, again, each pixel has an index 0; and, as such, no signaling of indices may be necessary.

[0106] In some examples, video encoder 20 may be configured to implement Aspects 1 and 2 such that the last occurrence (e.g., instance) of the syntax element **palette_run_type_flag**[xC][yC] is signaled by video encoder 20 at the front of the palette index block coding section. This aspect of the disclosure is referred to as Aspect 4. Specifically, the syntax table may be updated by, according to some example, adding a new syntax element **palette_last_run_type_flag** as follows:

indices_idc_coding() {	
num_palette_index	ae(v)
for (i = 0; i < num_palette_index; i++)	
palette_index_idc	ae(v)
palette_last_run_type_flag	ae(v)
}	

[0107] Video decoder 30 may be configured to determine the last occurrence (e.g., instance) of the syntax element **palette_run_type_flag**[xC][yC] by, for example, decoding an encoded **palette_last_run_type_flag** syntax element from a bitstream. The syntax element of **palette_last_run_type_flag** may be bypass mode coded or context mode coded in, for example, CABAC. In examples where the **palette_last_run_type_flag** syntax element is context mode coded, the **palette_last_run_type_flag** syntax element may share the same context(s) with **palette_run_type_flag**[xC][yC], or the **palette_last_run_type_flag** syntax element may have its own context(s) that are independent from the context(s) of **palette_run_type_flag**[xC][yC].

[0108] In some examples, video decoder 30 may be configured to decode the syntax element **palette_index_idc** such that the dynamic range adjustment process is disabled for the first occurrence (e.g., instance) of the **palette_index_idc** syntax element. This aspect of the disclosure is referred to as Aspect 5. Specifically, a process very similar to the adjustedIndexMax variable's derivation procedure specified in JCTVC-S1005 Section 7.4.9.6 is used. For comparison purposes, JCTVC-S1005 describes that the variable adjustedIndexMax may be derived as follows:

adjustedIndexMax = indexMax
if(scanPos > 0)
adjustedIndexMax − = 1

[0109] However, according to Aspect 5 of this disclosure, the variable adjustIndexMax may be derived as set forth below. For example, for each block, a variable isFirstIndex is initialized to 1 before parsing. In some examples, the variable adjustedIndexMax may be derived as follows:

adjustedIndexMax = indexMax
palette_index_idc

```
if( isFirstIndex ) {
  adjustedIndexMax − = isFirstIndex
  isFirstIndex = 0
}
```

[0110] In some examples, video decoder 30 may be configured to check one or more conditions before parsing and decoding the paletteRun. This aspect of the disclosure is referred to as Aspect 6. The variable paletteRun may specify the number of consecutive locations minus 1 with the same palette index as the position in the above row when palette_run_type_flag is equal to COPY_ABOVE_MODE or specify the number of consecutive locations minus 1 with the same palette index when palette_run_type_flag is equal to COPY_INDEX_MODE, as disclosed by JCTVC-S1005, for example.

[0111] Referring to the one or more conditions that video decoder 30 may be configured to check, if video decoder 30 determines that one or more of the conditions are satisfied, video decoder 30 may be configured to bypass the parsing and decoding process for the syntax elements related to the current paletteRun (i.e. **palette_run_msb_id_plus1** and **palette_run_refinement_bits**). In such an example, video decoder 30 may be configured to implicitly derive the current paletteRun as running to the end of the current block, i.e., equal to maxPaletteRun. The list of one or more conditions relating to Aspect 6 include: (i) the number of parsed/decoded **palette_index_idc** syntax elements equal to **num_palette_index**; or, alternatively, a variable paletteIndicesLeft may be defined that equals **num_palette_index** minus the number of indices received, and with such a definition, this condition may be stated as paletteIndicesLeft is equal to zero; and/or (ii) the current palette run type **palette_run_type_flag**[xC][yC] equals to the last palette run type **palette_last_run_type_flag**.

[0112] In some examples, if conditions (i) and (ii) set forth above for Aspect 6 are not satisfied simultaneously, video encoder 20 may be configured to code the palette run length into the bitstream. This aspect of the disclosure is referred to as Aspect 7. In other examples, if conditions (i) and (ii) set forth above for Aspect 6 are not satisfied simultaneously, video encoder 20 may be configured to code the palette run length into the bitstream. According to the current draft specification JCTVC-S1005, a parameter specifying the maximum achievable run length is required as input, where the parameter is equal to maxPaletteRun = nCbS * nCbS − scanPos – 1. According to this disclosure, however, video encoder 20 may be configured to reduce the parameter specifying the maximum achievable run length to maxPaletteRun = nCbS * nCbS – scanPos – 1 –

paletteIndicesLeft to improve coding efficiency. As used herein, nCbS specifies the size of the current block.

[0113] In some examples, a normative constraint may be imposed on video encoder 20 requiring that it never signals a palette with unused entries if a block is not in palette share mode (i.e., **palette_share_flag**[x0][y0] = = 0). This aspect of the disclosure is referred to as Aspect 8.

[0114] In some examples, for palette mode not using palette-share, video decoder 30 may be configured to bypass the decoding of the current occurrence (e.g., instance) of the syntax element **palette_index_idc** when one or more of the following conditions are satisfied: condition 1 where **num_palette_index** equals indexMax, and condition 2 where paletteIndicesLeft = = 1. In such examples, video decoder 30 may be configured to implicitly derive the value for the current occurrence of the syntax element **palette_index_idc** as an index that is in the palette, but has yet to appear in the index map during the decoding process (e.g., has not appeared in the index map up to this point in the decoding process). This aspect of the disclosure is referred to as Aspect 9.

[0115] Video decoder 30 may be configured to derive the value for the current occurrence of the syntax element **palette_index_idc** as set forth above for Aspect 9 because condition 1 requires that every index between 0 and (indexMax – 1), inclusively, be signaled and only be signaled once. Therefore, after the first (indexMax – 1) index values are signaled, video decoder 30 may be configured to derive the last index value as the number between 0 and (indexMax − 1), which has yet to appear during the decoding process for the current index map.

[0116] In some examples, video decoder 30 may be configured to bypass the decoding of the current occurrence (e.g., instance) of the syntax element **palette_run_type_flag**[xC][yC] when one or both of the following conditions are satisfied: condition 1 where paletteIndicesLeft equals 0, and condition 2 where the current pixel is at the last position of the block in scanning order. In such examples, video decoder 30 may be configured to implicitly derive the value for the current occurrence of the syntax element **palette_run_type_flag**[xC][yC]. For example, when condition 1 is satisfied, **palette_run_type_flag**[xC][yC] video decoder 30 may be configured to derive the value for the current occurrence of the syntax element **palette_run_type_flag**[xC][yC] as COPY_ABOVE_MODE. As another example, when condition 1 is satisfied, **palette_run_type_flag**[xC][yC] video decoder 30 may be configured to derive the value for the current occurrence of the syntax element

palette_run_type_flag[xC][yC] as COPY_INDEX_MODE if paletteIndicesLeft > 0, and as COPY_ABOVE_MODE if paletteIndicesLeft = 0. This aspect of the disclosure is referred to as Aspect 10.

[0117] As described herein, video encoder 20 and video decoder 30 may be configured to determine when a condition is satisfied. For example, with respect to Aspect 10, video decoder 30 may be configured to determine whether condition 1 is satisfied. Similarly, video decoder 30 may be configured to determine whether condition 2 is satisfied. In response to determining that condition 1 or condition 2 is satisfied, video decoder 30 may be configured to derive the value for the current occurrence of the syntax element **palette_run_type_flag**[xC][yC] as set forth above.

[0118] In some examples, video encoder 20 and video decoder 30 may be configured to respectively encode or decode the **num_palette_index** syntax element using any golomb code family. For example, video encoder 20 and video decoder 30 may be configured to respectively encode or decode the **num_palette_index** syntax element using, for example, Golomb Rice code, exponential Golomb code, Truncated Rice code, Unary code, or a concatenation of Golomb Rice and exponential Golomb code. This aspect of the disclosure is referred to as Aspect 11.

[0119] In other examples, video encoder 20 and video decoder 30 may be configured to respectively encode or decode the **num_palette_index** syntax element using any truncated version of any golomb code family. For example, video encoder 20 and video decoder 30 may be configured to respectively encode or decode the **num_palette_index** syntax element using, for example, truncated Golomb Rice code, truncated Exponential Golomb code, truncated Truncated Rice code, truncated Unary code, or a concatenation of truncated Rice code and exponential Golomb code such as the code used to code coeff_abs_level_remaining syntax elements. This aspect of the disclosure is referred to as Aspect 12.

[0120] In some examples, any golomb parameters relating to Aspects 11 or 12 depend upon the CU size, indexMax, palette size, and/or **palette_escape_val_present_flag**. Such dependency may be expressed as equations or a lookup table. In some examples, video encoder 20 may be configured to signal the lookup table or the parameters in the equations such that they are received by video decoder 30 in, for example, the SPS/PPS/Slice header. Alternatively or additionally, the parameters may be adaptively updated on a block-by-block basis. This aspect of the disclosure is referred to as Aspect 13. In some examples, the golomb parameter cRiceParam may depend on

indexMax, palette size, and/or palette_escape_val_present_flag. The golomb parameter cRiceParam may change from block to block.

[0121] In some examples, video encoder 20 may be configured to predictively encode **num_palette_index** by signaling the difference between the value of **num_palette_index** and an offset value, which may be expressed by a syntax element named, for example, numPaletteIndexCoded. This aspect of the disclosure is referred to as Aspect 14. For example, video encoder 20 may be configured to predictively encode **num_palette_index** by signaling a value for numPaletteIndexCoded, where numPaletteIndexCoded = **num_palette_index** – IndexOffsetValue. Similarly, video decoder 30 may be configured to predictively decode **num_palette_index** by, for example, determining a value for numPaletteIndexCoded from a bitstream. Since numPaletteIndexCoded = **num_palette_index** – IndexOffsetValue, video decoder 30 may be configured to determine the value of **num_palette_index** based on the determined value of numPaletteIndexCoded and the value of IndexOffsetValue.

[0122] In some examples, the variable IndexOffsetValue may be a constant. For example, IndexOffsetValue may equal a constant value of X for palette share mode or may equal a constant value of Y for non-palette share mode, where X and Y are integers. In some examples, X and Y may be the same (e.g., X equals Y such as equaling 1). In other examples, X and Y may be different (e.g., X does not equal Y). For example, IndexOffsetValue may equal 9 when palette share mode is used, and IndexOffsetValue may equal 33 when non-share mode is used. In some examples, the variable IndexOffsetValue may depend on the syntax element **palette_share_flag**[x0][y0]. In other examples, the variable IndexOffsetValue may depend on the variable indexMax. For example, IndexOffsetValue may equal indexMax. In some examples, video encoder 20 may be configured to signal IndexOffsetValue in the SPS/PPS/Slice header. Alternatively or additionally, the variable IndexOffsetValue may be adaptively updated block-by-block, meaning that the value corresponding to the variable IndexOffsetValue may be adaptively updated block-by-block.

[0123] In some examples, video encoder 20 and video decoder 30 may be configured to respectively encode or decode numPaletteIndexCoded may be coded using any golomb code family or any truncated golomb family, such as a concatenation of Golomb Rice and exponential Golomb code. For example, when IndexOffsetValue equals 1, numPaletteIndexCoded equals **num_palette_index** – 1.

[0124] In some examples, video encoder 20 and video decoder 30 may be configured to respectively encode or decode numPaletteIndexCoded using any golomb code family. For example, video encoder 20 and video decoder 30 may be configured to respectively encode or decode numPaletteIndexCoded using, for example, Golomb Rice code, exponential Golomb code, Truncated Rice code, Unary code, or a concatenation of Golomb Rice and exponential Golomb code.

[0125] In other examples, video encoder 20 and video decoder 30 may be configured to respectively encode or decode numPaletteIndexCoded using any truncated version of any golomb code family. For example, video encoder 20 and video decoder 30 may be configured to respectively encode or decode numPaletteIndexCoded using, for example, truncated Golomb Rice code, truncated Exponential Golomb code, truncated Truncated Rice code, truncated Unary code, or a concatenation of truncated Rice code and exponential Golomb code such as the code used to code coeff_abs_level_remaining syntax elements.

[0126] To code numPaletteIndexCoded, video encoder 20 may be configured to determine the sign of numPaletteIndexCoded. Video encoder 20 may be configured to signal a flag indicating whether the value of numPaletteIndexCoded is negative or not (e.g., whether the determined sign is positive or negative). This aspect of the disclosure is referred to as Aspect 15. In some examples, video encoder 20 may be configured to signal the flag, and then signal the value of numPaletteIndexCoded. In other examples, video encoder 20 may be configured to signal the value of numPaletteIndexCoded, and then signal the flag. Video encoder 20 may be configured to encode the flag using bypass mode or context mode. If context coded, the contexts may depend on CU size, indexMax, palette size, and/or **palette_escape_val_present_flag**.

[0127] As described above, video encoder 20 may be configured to determine the sign of numPaletteIndexCoded according to some examples. If the determined sign of numPaletteIndexCoded is negative, video encoder 20 may be configured to encode the value of (1 – numPaletteIndexCoded) into the bitstream. If the determined sign of numPaletteIndexCoded is positive, video encoder 20 may be configured to encode the value of numPaletteIndexCoded into the bitsteam. Video encoder 20 may be configured to encode the value of (1 – numPaletteIndexCoded) or the value numPaletteIndexCoded) using differnet golomb code parameters depending on, for example, the sign of numPaletteIndexCoded, CU size, indexMax, palette size, and/or **palette_escape_val_present_flag**.

[0128] In some examples, video encoder 20 may be configured to represent the negative parts of numPaletteIndexCoded using a mapping operation, which may be in addition to or may be an alternative to Aspect 15. This aspect of the disclosure is referred to as Aspect 16. For example, a mapping interval may be introduced and defined as a variable mapInterval. Video encoder 20 may be configured to, using variable mapInterval, map negative values of numPaletteIndexCoded to equally spaced positive values equal to: mapInterval × (−numPaletteIndexCoded) – 1. The corresponding positive value of numPaletteIndexCoded may be shifted accordingly to accommodate the positions taken by the mapped negative values.

[0129] For example, if mapInterval = 2, and numPaletteIndexCoded is chosen from {-3, -2, -1, 0, 1, 2, 3}, then the mapping can be illustrated as in Table I below. In this example, video encoder 20 may be configured to encode the values of numPaletteIndexCode using the mapped values in Table I. For example, video encoder 20 may be configured to entropy encode the mapped values into binary form.

Table I. Codeword Mapping Example

numPaletteIndexCoded	mapped value
-3	5
-2	3
-1	1
0	0
1	2
2	4
3	6

[0130] In some examples, video encoder 20 may be configured to represent the negative parts of numPaletteIndexCoded using a mapping operation as described with respect to Aspect 16. Video encoder 20 may also be configured to remove one or more redundancies that may be present when implementing Aspect 16. This aspect of the disclosure is referred to as Aspect 17. For example, the number of negative values of numPaletteIndexCoded may range from A={-1, -2, . . . , -IndexOffsetValue + 1}. As another example, the number of negative values of numPaletteIndexCode may range from A={-1, -2, . . . , -IndexOffsetValue + 1, IndexOffsetValue}. In either of these examples, the mapped value only needs to reserve (IndexOffsetValue – 1) or IndexOffsetValue positions for the negative numPaletteIndexCoded values. For example, if mapInterval = 2, and numPaletteIndexCoded is chosen form {-3, -2, -1, 0, 1,

2, 3, 4, 5, 6, 7, 8}, the mapping is illustrated in Table II below. In this example, video encoder 20 may be configured to encode the values of numPaletteIndexCode using the mapped values in Table II. For example, video encoder 20 may be configured to entropy encode the mapped values into binary form.

Table II. Codeword Mapping Example

numPaletteIndexCoded	mapped value
-3	5
-2	3
-1	1
0	0
1	2
2	4
3	*6*
4	*7*
5	*8*
6	*9*
7	*10*
8	*11*

[0131] As shown in Table II above, video encoder 20 may be configured to encode the mapped values corresponding to the values of numPaletteIndexCode such that negative and positive values of numPaletteIndexCode are not interleaved after a certain value. For example, in the example of Table II above, there is no interleaving of positive and negative values of numPaletteIndexCoded via the mapped values beginning with value 3 of numPaletteIndexCoded (i.e., positive values 3-8 of numPaletteIndexCoded map to mapped values 6-11).

[0132] As described above, video encoder 20 may also be configured to remove one or more redundancies that may be present when implementing Aspect 16. Another redundancy example different from the redundancy example described above includes: As **num_palette_index** is upper bounded by the total number of pixels in the current block, numPaletteIndexCoded is also upper bounded. Therefore, after allocating the positions for all of the possibilities of the positive codeword, the negative values can be mapped to the following positions without interleaving. For example, if mapInterval = 2, and numPaletteIndexCoded is chosen form {-5, -4, -3, -2, -1, 0, 1, 2, 3}, the mapping is illustrated in Table III below. In this example, video encoder 20 may be configured to encode the values of numPaletteIndexCode using the mapped values in Table III. For

example, video encoder 20 may be configured to entropy encode the mapped values into binary form.

Table III. Codeword Mapping Example

numPaletteIndexCoded	mapped value
-5	*8*
-4	*7*
-3	5
-2	3
-1	1
0	0
1	2
2	4
3	6

[0133] As shown in Table III above, video encoder 20 may be configured to encode the mapped values corresponding to the values of numPaletteIndexCode such that negative and positive values of numPaletteIndexCode are not interleaved after a certain value. For example, in the example of Table III above, there is no interleaving of positive and negative values of numPaletteIndexCoded via the mapped values beginning with value 4 of numPaletteIndexCoded (i.e., negative values -4 and -5 of numPaletteIndexCoded map to mapped values 7 and 8).

[0134] In some examples, video encoder 20 may be configured to further decouple the relationship between palette index and palette run. This aspect of the disclosure is referred to as Aspect 18. For example, instead of allowing the palette run coding's contexts depend on parsed or decoded indices, video encoder 20 may be configured to make the palette run coding's contexts depend on the previous palette run length or depend on the previous run's **palette_run_msb_id_plus1**, indexMax, and/or CU size.

[0135] In some examples, to further group bypass bins, video encoder 20 may be configured to signal the number of escape indices in a palette block as well as escape values before signaling the palette run type (i.e. **palette_run_type_flag**[xC][yC]) as follows. This aspect of the disclosure is referred to as Aspect 19. Italicized portions illustrate changes relative to previous version(s) of JCT-VC S1005, and bolded portions as well as the "ae(v)" in the right column indicate the signaling of a syntax element.

...	
if(currentPaletteSize != 0)	
palette_escape_val_present_flag	ae(v)
if(palette_escape_val_present_flag || (indexMax > 0))	
escape_idc_coding()	
if(palette_escape_val_present_flag) {	
if(cu_qp_delta_enabled_flag && !IsCuQpDeltaCoded) {	
cu_qp_delta_palette_abs	ae(v)
if(cu_qp_delta_palette_abs)	
cu_qp_delta_palette_sign_flag	ae(v)
}	
if(cu_chroma_qp_offset_enabled_flag && !IsCuChromaQpOffsetCoded) {	
cu_chroma_qp_palette_offset_flag	ae(v)
if(cu_chroma_qp_offset_flag && chroma_qp_offset_list_len_minus1 > 0)	
cu_chroma_qp_palette_offset_idx	ae(v)
}	
}	
if(indexMax > 0)	
palette_transpose_flag	ae(v)
scanPos = 0	
while(scanPos < nCbS * nCbS) {	
…	

[0136] In the example above, escape_idc_coding() consists of signaling the number of escape indices and escape values corresponding to each escape index. The number of escape indices in a palette block may not be signaled if **palette_escape_val_present_flag** is 0 or if indexMax is equal to 0. In the former case, the number of escape indices is inferred to be 0 and no escape values are signaled. In the latter case of indexMax equal to 0, the number of escape indices is inferred to be equal to the block size when **palette_escape_val_present_flag** equals 1 and escape values are signaled, or the number of escape indices is inferred to be zero when **palette_escape_val_present_flag** equals 0.

[0137] In some examples, video encoder 20 may be configured to signal the number of escape indices using golomb code family. This aspect of the disclosure is referred to as

Aspect 20. For example, video encoder 20 may be configured to signal the number of escape indices using, for example, Golomb Rice code, exponential Golomb code, Truncated Rice code, Unary code, or a concatenation of Golomb Rice and exponential Golomb code. Truncated versions of the above codes may be used with maximum set equal to the block size.

[0138] In some examples, it is proposed to enforce a normative restriction on **palette_escape_val_present_flag** that when **palette_escape_val_present_flag** equals to 0, there is no escape pixel in the current block. This aspect of the disclosure is referred to as Aspect 21. When **palette_escape_val_present_flag** equals to 1, there is *at least one* escape pixel in the current block. With this restriction, in escape_idc_coding(), the number of escape indices minus 1 can be coded instead of number of escape indices to improve coding efficiency. In that case, the maximum value for truncated golomb code family may be adjusted to (blockSize-1), accordingly.

[0139] In some examples, when the number of escape indices is signaled before coding the indices map block and when all of the escape indices have already been coded, then indexMax may be reduced by 1. Furthermore, if indexMax becomes 1, the index, run and mode coding is terminated since the indices for all the remaining samples may be inferred. This aspect of the disclosure is referred to as Aspect 22. As one example of Aspect 22, assume palette size equals 1 and **palette_escape_val_present_flag** equals 1. Ordinarily, the possible index values are 0 and 1, where 1 is used for escape sample(s). Under Aspect 22, video encoder 20 may be configured to signal the number of escape values/samples. Then, when the indices are being signaled and the last escape value/sample is encountered, both video encoder 20 and/or video decoder 30 may be configured to infer (e.g., determine) that there are no more escape values/samples. As such, video encoder 20 and/or video decoder 30 may be configured to determine that the only index value that can occur from the last escape value/sample to the end of the block is 0, meaning that video encoder 20 may be configured to not signal the mode, index value, and/or run value from the last escape value/sample to the end of the block.

[0140] In some examples, escape_idc_coding() is used in combination with indices_idc_coding(). This aspect of the disclosure is referred to as Aspect 23. In one example, the number of escape indices may be signaled before signaling the number of indices. In this case, only the number of non-escape indices need to be signaled in indices_idc_coding(). In one example, the number of escape indices may be signaled

after signaling the number of indices. In this case, the maximum value for truncated golomb code family may be adjusted to **num_palette_index**, accordingly.

[0141] Video encoder 20 and/or video decoder 30 may be configured to operate according to the techniques described in this disclosure. In general, video encoder 20 and/or video decoder 30 may be configured to determine that a current block is coded in palette mode, bypass mode code a plurality of instances of a first syntax element for reconstructing the current block, and after bypass mode code a plurality of instance of the first syntax element, context mode decoding a plurality of instances of a second syntax element for reconstructing the current block.

[0142] FIG. 2 is a block diagram illustrating an example video encoder 20 that may implement the techniques of this disclosure. FIG. 2 is provided for purposes of explanation and should not be considered limiting of the techniques as broadly exemplified and described in this disclosure. For purposes of explanation, this disclosure describes video encoder 20 in the context of HEVC coding and, for example, the SCC extension of HEVC. However, the techniques of this disclosure may be applicable to other coding standards or methods.

[0143] Video encoder 20 represents an example of a device that may be configured to perform techniques for palette-based coding and entropy coding (e.g., CABAC) in accordance with various examples described in this disclosure.

[0144] In the example of FIG. 2, video encoder 20 includes a block encoding unit 100, video data memory 101, a residual generation unit 102, a transform processing unit 104, a quantization unit 106, an inverse quantization unit 108, an inverse transform processing unit 110, a reconstruction unit 112, a filter unit 114, a decoded picture buffer 116, and an entropy encoding unit 118. Block encoding unit 100 includes an inter-prediction processing unit 120 and an intra-prediction processing unit 126. Inter-prediction processing unit 120 includes a motion estimation unit and a motion compensation unit (not shown). Video encoder 20 also includes a palette-based encoding unit 122 configured to perform various aspects of the palette-based coding techniques described in this disclosure. In other examples, video encoder 20 may include more, fewer, or different functional components.

[0145] Video data memory 101 may store video data to be encoded by the components of video encoder 20. The video data stored in video data memory 101 may be obtained, for example, from video source 18. Decoded picture buffer 116 may be a reference picture memory that stores reference video data for use in encoding video data by video

encoder 20, e.g., in intra- or inter-coding modes. Video data memory 101 and decoded picture buffer 116 may be formed by any of a variety of memory devices, such as dynamic random access memory (DRAM), including synchronous DRAM (SDRAM), magnetoresistive RAM (MRAM), resistive RAM (RRAM), or other types of memory devices. Video data memory 101 and decoded picture buffer 116 may be provided by the same memory device or separate memory devices. In various examples, video data memory 101 may be on-chip with other components of video encoder 20, or off-chip relative to those components.

[0146] Video encoder 20 may receive video data. Video encoder 20 may encode each CTU in a slice of a picture of the video data. Each of the CTUs may be associated with equally-sized luma coding tree blocks (CTBs) and corresponding CTBs of the picture. As part of encoding a CTU, block encoding unit 100 may perform quad-tree partitioning to divide the CTBs of the CTU into progressively-smaller blocks. The smaller block may be coding blocks of CUs. For example, block encoding unit 100 may partition a CTB associated with a CTU into four equally-sized sub-blocks, partition one or more of the sub-blocks into four equally-sized sub-sub-blocks, and so on.

[0147] Video encoder 20 may encode CUs of a CTU to generate encoded representations of the CUs (i.e., coded CUs). As part of encoding a CU, block encoding unit 100 may partition the coding blocks associated with the CU among one or more PUs of the CU. Thus, each PU may be associated with a luma prediction block and corresponding chroma prediction blocks. Video encoder 20 and video decoder 30 may support PUs having various sizes. As indicated above, the size of a CU may refer to the size of the luma coding block of the CU and the size of a PU may refer to the size of a luma prediction block of the PU. Assuming that the size of a particular CU is 2Nx2N, video encoder 20 and video decoder 30 may support PU sizes of 2Nx2N or NxN for intra prediction, and symmetric PU sizes of 2Nx2N, 2NxN, Nx2N, NxN, or similar for inter prediction. Video encoder 20 and video decoder 30 may also support asymmetric partitioning for PU sizes of 2NxnU, 2NxnD, nLx2N, and nRx2N for inter prediction.

[0148] Inter-prediction processing unit 120 may generate predictive data for a PU by performing inter prediction on each PU of a CU. The predictive data for the PU may include predictive blocks of the PU and motion information for the PU. Inter-prediction unit 121 may perform different operations for a PU of a CU depending on whether the PU is in an I slice, a P slice, or a B slice. In an I slice, all PUs are intra predicted. Hence, if the PU is in an I slice, inter-prediction unit 121 does not perform inter

prediction on the PU. Thus, for blocks encoded in I-mode, the predicted block is formed using spatial prediction from previously-encoded neighboring blocks within the same frame.

[0149] If a PU is in a P slice, the motion estimation unit of inter-prediction processing unit 120 may search the reference pictures in a list of reference pictures (e.g., "RefPicList0") for a reference region for the PU. The reference region for the PU may be a region, within a reference picture, that contains sample blocks that most closely corresponds to the sample blocks of the PU. The motion estimation unit of inter-prediction processing unit 120 may generate a reference index that indicates a position in RefPicList0 of the reference picture containing the reference region for the PU. In addition, the motion estimation unit may generate an MV that indicates a spatial displacement between a coding block of the PU and a reference location associated with the reference region. For instance, the MV may be a two-dimensional vector that provides an offset from the coordinates in the current decoded picture to coordinates in a reference picture. The motion estimation unit may output the reference index and the MV as the motion information of the PU. The motion compensation unit of inter-prediction processing unit 120 may generate the predictive blocks of the PU based on actual or interpolated samples at the reference location indicated by the motion vector of the PU.

[0150] If a PU is in a B slice, the motion estimation unit may perform uni-prediction or bi-prediction for the PU. To perform uni-prediction for the PU, the motion estimation unit may search the reference pictures of RefPicList0 or a second reference picture list ("RefPicList1") for a reference region for the PU. The motion estimation unit may output, as the motion information of the PU, a reference index that indicates a position in RefPicList0 or RefPicList1 of the reference picture that contains the reference region, an MV that indicates a spatial displacement between a prediction block of the PU and a reference location associated with the reference region, and one or more prediction direction indicators that indicate whether the reference picture is in RefPicList0 or RefPicList1. The motion compensation unit of inter-prediction processing unit 120 may generate the predictive blocks of the PU based at least in part on actual or interpolated samples at the reference region indicated by the motion vector of the PU.

[0151] To perform bi-directional inter prediction for a PU, the motion estimation unit may search the reference pictures in RefPicList0 for a reference region for the PU and may also search the reference pictures in RefPicList1 for another reference region for

the PU. The motion estimation unit may generate reference picture indexes that indicate positions in RefPicList0 and RefPicList1 of the reference pictures that contain the reference regions. In addition, the motion estimation unit may generate MVs that indicate spatial displacements between the reference location associated with the reference regions and a sample block of the PU. The motion information of the PU may include the reference indexes and the MVs of the PU. The motion compensation unit may generate the predictive blocks of the PU based at least in part on actual or interpolated samples at the reference regions indicated by the motion vectors of the PU.

[0152] In accordance with various examples of this disclosure, video encoder 20 may be configured to perform palette-based coding. With respect to the HEVC framework, as an example, the palette-based coding techniques may be configured to be used at the CU level. In other examples, the palette-based video coding techniques may be configured to be used at the PU level. In other examples, the palette-based coding techniques may be configured to be used at the sub-prediction unit (sub-PU) level (e.g., a sub-block of a prediction unit). Accordingly, all of the disclosed processes described herein (throughout this disclosure) in the context of a CU level may, additionally or alternatively, apply to a PU level or a sub-PU level. However, these HEVC-based examples should not be considered a restriction or limitation of the palette-based video coding techniques described herein, as such techniques may be applied to work independently or as part of other existing or yet to be developed systems/standards. In these cases, the unit for palette coding can be square blocks, rectangular blocks or even regions of non-rectangular shape.

[0153] Palette-based encoding unit 122, for example, may perform palette-based decoding when a palette-based encoding mode is selected, e.g., for a CU or PU. For example, palette-based encoding unit 122 may be configured to generate a palette having entries indicating pixel values, select pixel values in a palette to represent pixel values of at least some positions of a block of video data, and signal information associating at least some of the positions of the block of video data with entries in the palette corresponding, respectively, to the selected pixel values. Although various functions are described as being performed by palette-based encoding unit 122, some or all of such functions may be performed by other processing units, or a combination of different processing units.

[0154] According to aspects of this disclosure, palette-based encoding unit 122 may be configured to perform any combination of the techniques for palette coding described herein.

[0155] Intra-prediction processing unit 126 may generate predictive data for a PU by performing intra prediction on the PU. The predictive data for the PU may include predictive blocks for the PU and various syntax elements. Intra-prediction processing unit 126 may perform intra prediction on PUs in I slices, P slices, and B slices.

[0156] To perform intra prediction on a PU, intra-prediction processing unit 126 may use multiple intra prediction modes to generate multiple sets of predictive data for the PU. Intra-prediction processing unit 126 may use samples from sample blocks of neighboring PUs to generate a predictive block for a PU. The neighboring PUs may be above, above and to the right, above and to the left, or to the left of the PU, assuming a left-to-right, top-to-bottom encoding order for PUs, CUs, and CTUs. Intra-prediction processing unit 126 may use various numbers of intra prediction modes, e.g., 33 directional intra prediction modes. In some examples, the number of intra prediction modes may depend on the size of the region associated with the PU.

[0157] Block encoding unit 100 may select the predictive data for PUs of a CU from among the predictive data generated by inter-prediction processing unit 120 for the PUs or the predictive data generated by intra-prediction processing unit 126 for the PUs. In some examples, block encoding unit 100 selects the predictive data for the PUs of the CU based on rate/distortion metrics of the sets of predictive data. The predictive blocks of the selected predictive data may be referred to herein as the selected predictive blocks.

[0158] Residual generation unit 102 may generate, based on the luma, Cb and Cr coding block of a CU and the selected predictive luma, Cb and Cr blocks of the PUs of the CU, a luma, Cb and Cr residual blocks of the CU. For instance, residual generation unit 102 may generate the residual blocks of the CU such that each sample in the residual blocks has a value equal to a difference between a sample in a coding block of the CU and a corresponding sample in a corresponding selected predictive block of a PU of the CU.

[0159] Transform processing unit 104 may perform quad-tree partitioning to partition the residual blocks associated with a CU into transform blocks associated with TUs of the CU. Thus, in some examples, a TU may be associated with a luma transform block and two chroma transform blocks. The sizes and positions of the luma and chroma

transform blocks of TUs of a CU may or may not be based on the sizes and positions of prediction blocks of the PUs of the CU. A quad-tree structure known as a "residual quad-tree" (RQT) may include nodes associated with each of the regions. The TUs of a CU may correspond to leaf nodes of the RQT.

[0160] Transform processing unit 104 may generate transform coefficient blocks for each TU of a CU by applying one or more transforms to the transform blocks of the TU. Transform processing unit 104 may apply various transforms to a transform block associated with a TU. For example, transform processing unit 104 may apply a discrete cosine transform (DCT), a directional transform, or a conceptually similar transform to a transform block. In some examples, transform processing unit 104 does not apply transforms to a transform block. In such examples, the transform block may be treated as a transform coefficient block.

[0161] Quantization unit 106 may quantize the transform coefficients in a coefficient block. The quantization process may reduce the bit depth associated with some or all of the transform coefficients. For example, an n-bit transform coefficient may be rounded down to an m-bit transform coefficient during quantization, where n is greater than m. Quantization unit 106 may quantize a coefficient block associated with a TU of a CU based on a quantization parameter (QP) value associated with the CU. Video encoder 20 may adjust the degree of quantization applied to the coefficient blocks associated with a CU by adjusting the QP value associated with the CU. Quantization may introduce loss of information, thus quantized transform coefficients may have lower precision than the original ones.

[0162] Inverse quantization unit 108 and inverse transform processing unit 110 may apply inverse quantization and inverse transforms to a coefficient block, respectively, to reconstruct a residual block from the coefficient block. Reconstruction unit 112 may add the reconstructed residual block to corresponding samples from one or more predictive blocks generated by block encoding unit 100 to produce a reconstructed transform block associated with a TU. By reconstructing transform blocks for each TU of a CU in this way, video encoder 20 may reconstruct the coding blocks of the CU.

[0163] Filter unit 114 may perform one or more deblocking operations to reduce blocking artifacts in the coding blocks associated with a CU. Filter unit 114 may perform other filtering operations, including sample adaptive offset (SAO) filtering and/or adaptive loop filtering (ALF). Decoded picture buffer 116 may store the reconstructed coding blocks after filter unit 114 performs the one or more deblocking

operations on the reconstructed coding blocks. Inter-prediction processing unit 120 may use a reference picture that contains the reconstructed coding blocks to perform inter prediction on PUs of other pictures. In addition, intra-prediction processing unit 126 may use reconstructed coding blocks in decoded picture buffer 116 to perform intra prediction on other PUs in the same picture as the CU.

[0164] Entropy encoding unit 118 may receive data from other functional components of video encoder 20. For example, entropy encoding unit 118 may receive coefficient blocks from quantization unit 106 and may receive syntax elements from block encoding unit 100. Entropy encoding unit 118 may perform one or more entropy encoding operations on the data to generate entropy-encoded data. For example, entropy encoding unit 118 may perform a context-adaptive coding operation, such as a CABAC operation, context-adaptive variable length coding (CAVLC) operation, a variable-to-variable (V2V) length coding operation, a syntax-based context-adaptive binary arithmetic coding (SBAC) operation, a Probability Interval Partitioning Entropy (PIPE) coding operation, an Exponential-Golomb encoding operation, or another type of entropy encoding operation on the data. Video encoder 20 may output a bitstream that includes entropy-encoded data generated by entropy encoding unit 118. For instance, the bitstream may include data that represents a RQT for a CU.

[0165] In some examples, residual coding is not performed with palette coding. Accordingly, video encoder 20 may not perform transformation or quantization when coding using a palette coding mode. In addition, video encoder 20 may entropy encode data generated using a palette coding mode separately from residual data.

[0166] According to one or more of the techniques of this disclosure, video encoder 20, and specifically palette-based encoding unit 122, may perform palette-based video coding of predicted video blocks. As described above, a palette generated by video encoder 20 may be explicitly encoded and sent to video decoder 30, predicted from previous palette entries, predicted from previous pixel values, or a combination thereof.

[0167] In accordance with one or more techniques of this disclosure, video encoder 20 may be configured to determine that a current block is coded in palette mode, bypass mode encode a plurality of instances of a first syntax element for reconstructing the current block, and after bypass mode encode a plurality of instance of the first syntax element, context mode encode a plurality of instances of a second syntax element for reconstructing the current block, e.g., using a CABAC coding process. Video encoder 20 may be configured to bypass mode encode any two instances of the plurality of

instances of the first syntax element, e.g., using a bypass mode of a CABAC coding process, without interleaving with the context mode encoding of any one instance of the plurality of instances of the second syntax element. In one example, the first syntax element comprises one of a palette_index_idc syntax element or palette_escape_val syntax element, and the second syntax element comprises a palette_run_msb_id_plus1 syntax element. Video encoder 20 may be configured to bypass encode the plurality of instances of the first syntax element at a front of an index block coding section for the current block.

[0168] Video encoder 20 may be configured to encode a third syntax element indicating a number of instances of the first syntax element, wherein bypass mode encoding the plurality of instances of the first syntax element comprises bypass mode encoding the plurality of instances of the first syntax element based on the third syntax element. Video encoder 20 may encode the third syntax element using one of a Golomb Rice code, exponential Golomb code, Truncated Rice code, Unary code, a concatenation of Golomb Rice and exponential Golomb code, or a truncated version of any of the previous codes.

[0169] FIG. 3 is a block diagram illustrating an example video decoder 30 that is configured to perform the techniques of this disclosure. FIG. 3 is provided for purposes of explanation and is not limiting on the techniques as broadly exemplified and described in this disclosure. For purposes of explanation, this disclosure describes video decoder 30 in the context of HEVC coding. However, the techniques of this disclosure may be applicable to other coding standards or methods.

[0170] The details of palette coding described above with respect to encoder 20 are not repeated here with respect to decoder 30, but it is understood that decoder 30 may perform the reciprocal decoding process relative to any encoding process described herein with respect to encoder 20.

[0171] Video decoder 30 represents an example of a device that may be configured to perform techniques for palette-based coding and entropy coding (e.g., CABAC) in accordance with various examples described in this disclosure.

[0172] In the example of FIG. 3, video decoder 30 includes an entropy decoding unit 150, video data memory 151, a block decoding unit 152, an inverse quantization unit 154, an inverse transform processing unit 156, a reconstruction unit 158, a filter unit 160, and a decoded picture buffer 162. Block decoding unit 152 includes a motion compensation unit 164 and an intra-prediction processing unit 166. Video decoder 30

also includes a palette-based decoding unit 165 configured to perform various aspects of the palette-based coding techniques described in this disclosure. In other examples, video decoder 30 may include more, fewer, or different functional components.

[0173] Video data memory 151 may store video data, such as an encoded video bitstream, to be decoded by the components of video decoder 30. The video data stored in video data memory 151 may be obtained, for example, from computer-readable medium 16, e.g., from a local video source, such as a camera, via wired or wireless network communication of video data, or by accessing physical data storage media. Video data memory 151 may form a coded picture buffer (CPB) that stores encoded video data from an encoded video bitstream. Decoded picture buffer 162 may be a reference picture memory that stores reference video data for use in decoding video data by video decoder 30, e.g., in intra- or inter-coding modes. Video data memory 151 and decoded picture buffer 162 may be formed by any of a variety of memory devices, such as dynamic random access memory (DRAM), including synchronous DRAM (SDRAM), magnetoresistive RAM (MRAM), resistive RAM (RRAM), or other types of memory devices. Video data memory 151 and decoded picture buffer 162 may be provided by the same memory device or separate memory devices. In various examples, video data memory 151 may be on-chip with other components of video decoder 30, or off-chip relative to those components.

[0174] A coded picture buffer (CPB), which may be provided by video data memory 151, may receive and store encoded video data (e.g., NAL units) of a bitstream. Entropy decoding unit 150 may receive encoded video data (e.g., NAL units) from the CPB and parse the NAL units to decode syntax elements. Entropy decoding unit 150 may entropy decode entropy-encoded syntax elements in the NAL units. Block decoding unit 152, inverse quantization unit 154, inverse transform processing unit 156, reconstruction unit 158, and filter unit 160 may generate decoded video data based on the syntax elements extracted from the bitstream.

[0175] Video decoder 30 may be configured to perform a process generally reciprocal to that of video encoder 20 described herein. Similarly, video encoder 20 may be configured to perform a process generally reciprocal to that of video decoder 30 described herein. For example, disclosure that video decoder 30 may be configured to decode an encoded syntax element in a bitstream likewise necessarily discloses that video encoder 20 may be configured to encode the syntax element into the bitstream.

[0176] As another example, entropy decoding unit 150 may be configured to perform a process generally reciprocal to that of entropy encoding unit 118 described herein. According to aspects of this disclosure, entropy decoding unit 150 may be configured to entropy decode any code words generated by entropy encoding unit 118. For example, entropy decoding unit 150 may be configured to entropy decode uniform and non-uniform kth order truncated Exp-Golomb (TEGk)-encoded values, such as a binary palette prediction vector and/or a palette map for a CU. As another example, entropy decoding unit 150 may be configured to entropy decode a kth order Exp-Golomb (EGk) code word, a kth order truncated Exp-Golomb (TEGk) code word, a kth order non-uniform truncated Exp-Golomb (TEGk) code word, or any combination thereof.

[0177] The NAL units of the bitstream may include coded slice NAL units. As part of decoding the bitstream, entropy decoding unit 150 may extract and entropy decode syntax elements from the coded slice NAL units. Each of the coded slices may include a slice header and slice data. The slice header may contain syntax elements pertaining to a slice. The syntax elements in the slice header may include a syntax element that identifies a PPS associated with a picture that contains the slice.

[0178] In addition to decoding syntax elements from the bitstream, video decoder 30 may perform a reconstruction operation on a non-partitioned CU. To perform the reconstruction operation on a non-partitioned CU, video decoder 30 may perform a reconstruction operation on each TU of the CU. By performing the reconstruction operation for each TU of the CU, video decoder 30 may reconstruct residual blocks of the CU.

[0179] As part of performing a reconstruction operation on a TU of a CU, inverse quantization unit 154 may inverse quantize, i.e., de-quantize, coefficient blocks associated with the TU. Inverse quantization unit 154 may use a QP value associated with the CU of the TU to determine a degree of quantization and, likewise, a degree of inverse quantization for inverse quantization unit 154 to apply. That is, the compression ratio, i.e., the ratio of the number of bits used to represent original sequence and the compressed one, may be controlled by adjusting the value of the QP used when quantizing transform coefficients. The compression ratio may also depend on the method of entropy coding employed.

[0180] After inverse quantization unit 154 inverse quantizes a coefficient block, inverse transform processing unit 156 may apply one or more inverse transforms to the coefficient block in order to generate a residual block associated with the TU. For

example, inverse transform processing unit 156 may apply an inverse DCT, an inverse integer transform, an inverse Karhunen-Loeve transform (KLT), an inverse rotational transform, an inverse directional transform, or another inverse transform to the coefficient block.

[0181] If a PU is encoded using intra prediction, intra-prediction processing unit 166 may perform intra prediction to generate predictive blocks for the PU. Intra-prediction processing unit 166 may use an intra-prediction mode to generate the predictive luma, Cb and Cr blocks for the PU based on the prediction blocks of spatially-neighboring PUs. Intra-prediction processing unit 166 may determine the intra prediction mode for the PU based on one or more syntax elements decoded from the bitstream.

[0182] Block decoding unit 152 may construct a first reference picture list (RefPicList0) and a second reference picture list (RefPicList1) based on syntax elements extracted from the bitstream. Furthermore, if a PU is encoded using inter prediction, entropy decoding unit 150 may extract motion information for the PU. Motion compensation unit 164 may determine, based on the motion information of the PU, one or more reference regions for the PU. Motion compensation unit 164 may generate, based on samples blocks at the one or more reference blocks for the PU, predictive luma, Cb and Cr blocks for the PU.

[0183] Reconstruction unit 158 may use the luma, Cb and Cr transform blocks associated with TUs of a CU and the predictive luma, Cb and Cr blocks of the PUs of the CU, i.e., either intra-prediction data or inter-prediction data, as applicable, to reconstruct the luma, Cb and Cr coding blocks of the CU. For example, reconstruction unit 158 may add samples of the luma, Cb and Cr transform blocks to corresponding samples of the predictive luma, Cb and Cr blocks to reconstruct the luma, Cb and Cr coding blocks of the CU.

[0184] Filter unit 160 may perform a deblocking operation to reduce blocking artifacts associated with the luma, Cb and Cr coding blocks of the CU. Video decoder 30 may store the luma, Cb and Cr coding blocks of the CU in decoded picture buffer 162. Decoded picture buffer 162 may provide reference pictures for subsequent motion compensation, intra prediction, and presentation on a display device, such as display device 32 of FIG. 1. For instance, video decoder 30 may perform, based on the luma, Cb, and Cr blocks in decoded picture buffer 162, intra prediction or inter prediction operations on PUs of other CUs.

[0185] In accordance with various examples of this disclosure, video decoder 30 may be configured to perform palette-based coding. Palette-based decoding unit 165, for example, may perform palette-based decoding when a palette-based decoding mode is selected, e.g., for a CU or PU. For example, palette-based decoding unit 165 may be configured to generate a palette having entries indicating pixel values, receive information associating at least some pixel locations in a block of video data with entries in the palette, select pixel values in the palette based on the information, and reconstruct pixel values of the block based on the selected pixel values in the palette. Although various functions are described as being performed by palette-based decoding unit 165, some or all of such functions may be performed by other processing units, or a combination of different processing units.

[0186] Palette-based decoding unit 165 may receive palette coding mode information, and perform the above operations when the palette coding mode information indicates that the palette coding mode applies to the block. When the palette coding mode information indicates that the palette coding mode does not apply to the block, or when other mode information indicates the use of a different mode, palette-based decoding unit 165 decodes the block of video data using a non-palette based coding mode, e.g., such as an HEVC inter-predictive or intra-predictive coding mode. The block of video data may be, for example, a CU or PU generated according to an HEVC coding process. The palette-based coding mode may comprise one of a plurality of different palette-based coding modes, or there may be a single palette-based coding mode.

[0187] According to aspects of this disclosure, palette-based decoding unit 165 may be configured to perform any combination of the techniques for palette coding described herein. The details of palette coding described above with respect to encoder 20 are not repeated here with respect to decoder 30, but it is understood that decoder 30 may perform the reciprocal palette-based decoding process relative to any palette-based encoding process described herein with respect to encoder 20.

[0188] Video decoder 30 may be configured to determine that a current block is coded in palette mode, bypass mode decode a plurality of instances of a first syntax element for reconstructing the current block, e.g., using a bypass mode of a CABAC coding process, and after bypass mode decoding a plurality of instance of the first syntax element, context mode decode a plurality of instances of a second syntax element for reconstructing the current block, e.g., using a CABAC coding process. Video decoder 30 may bypass mode decode any two instances of the plurality of instances of the first

syntax element without interleaving with the context mode decoding of any one instance of the plurality of instances of the second syntax element. In some examples, the first syntax element comprises one of a palette_index_idc syntax element or palette_escape_val syntax element, and the second syntax element comprises a palette_run_msb_id_plus1 syntax element. Video decoder 30 may bypass decode the plurality of instances of the first syntax element at a front of an index block coding section for the current block.

[0189] Video decoder 30 may decode a third syntax element indicating a number of instances of the first syntax element, wherein bypass mode decoding the plurality of instances of the first syntax element comprises bypass mode decoding the plurality of instances of the first syntax element based on the third syntax element. Video decoder 30 may decode the third syntax element using one of a Golomb Rice code, exponential Golomb code, Truncated Rice code, Unary code, a concatenation of Golomb Rice and exponential Golomb code, or a truncated version of any of the previous codes.

[0190] FIG. 4 is a conceptual diagram illustrating an example of determining a palette for coding video data, consistent with techniques of this disclosure. The example of FIG. 4 includes a picture 178 having a first PAL (palette) coding unit (CU) 180 that is associated with first palettes 184 and a second PAL CU 188 that is associated with second palettes 192. As described in greater detail below and in accordance with the techniques of this disclosure, second palettes 192 are based on first palettes 184. Picture 178 also includes block 196 coded with an intra-prediction coding mode and block 200 that is coded with an inter-prediction coding mode.

[0191] The techniques of FIG. 4 are described in the context of video encoder 20 (FIG. 1 and FIG. 2) and video decoder 30 (FIG. 1 and FIG. 3) and with respect to the HEVC video coding standard for purposes of explanation. However, it should be understood that the techniques of this disclosure are not limited in this way, and may be applied by other video coding processors and/or devices in other video coding processes and/or standards.

[0192] In general, a palette refers to a number of pixel values that are dominant and/or representative for a CU currently being coded, CU 188 in the example of FIG. 4. First palettes 184 (which may also be referred to as indexes/indices 184) and second palettes 192 (which may also be referred to as indexes/indices 192) are shown as including multiple palettes (which may also be referred to as multiple indexes). In some examples, according to aspects of this disclosure, a video coder (such as video encoder

20 or video decoder 30) may code palettes (e.g., indexes) separately for each color component of a CU. For example, video encoder 20 may encode a palette for a luma (Y) component of a CU, another palette for a chroma (U) component of the CU, and yet another palette for the chroma (V) component of the CU. In this example, entries of the Y palette may represent Y values of pixels of the CU, entries of the U palette may represent U values of pixels of the CU, and entries of the V palette may represent V values of pixels of the CU.

[0193] In other examples, video encoder 20 may encode a single palette for all color components of a CU. In this example, video encoder 20 may encode a palette having an i-th entry that is a triple value, including Yi, Ui, and Vi. In this case, the palette includes values for each of the components of the pixels. Accordingly, the representation of palettes 184 and 192 as a set of palettes having multiple individual palettes is merely one example and not intended to be limiting.

[0194] In the example of FIG. 4, first palettes 184 includes three entries 202-206 having entry index value 1, entry index value 2, and entry index value 3, respectively. First palettes 184 relate the index values (e.g., the values shown in the left column of first palettes 184) to pixel values. For example, as shown in FIG. 4, one of first palettes 184 relates index values 1, 2, and 3 to pixel values A, B, and C, respectively. As described herein, rather than coding the actual pixel values of first CU 180, a video coder (such as video encoder 20 or video decoder 30) may use palette-based coding to code the pixels of the block using the indices 1-3 (which may also be expressed as index values 1-3). That is, for each pixel position of first CU 180, video encoder 20 may encode an index value for the pixel, where the index value is associated with a pixel value in one or more of first palettes 184. Video decoder 30 may obtain the index values from a bitstream and reconstruct the pixel values using the index values and one or more of first palettes 184. Thus, first palettes 184 are transmitted by video encoder 20 in an encoded video data bitstream for use by video decoder 30 in palette-based decoding.

[0195] In some examples, video encoder 20 and video decoder 30 may determine second palettes 192 based on first palettes 184. For example, video encoder 20 and/or video decoder 30 may locate one or more blocks from which the predictive palettes, in this example, first palettes 184, are determined. In some examples, such as the example illustrated in FIG. 4, video encoder 20 and/or video decoder 30 may locate the previously coded CU such as a left neighboring CU (first CU 180) when determining a predictive palette for second CU 188.

[0196] In the example of FIG. 4, second palettes 192 include three entries 208-212 having entry index value 1, entry index value 2, and entry index value 3, respectively. Second palettes 192 relate the index values (e.g., the values shown in the left column of first palettes 192) to pixel values. For example, as shown in FIG. 4, one of the second palettes 192 relates index values 1, 2, and 3 to pixel values A, B, and D, respectively. In this example, video encoder 20 may code one or more syntax elements indicating which entries of first palettes 184 are included in second palettes 192. In the example of FIG. 4, the one or more syntax elements are illustrated as a vector 216. Vector 216 has a number of associated bins (or bits), with each bin indicating whether the palette predictor associated with that bin is used to predict an entry of the current palette. For example, vector 216 indicates that the first two entries of first palettes 184 (202 and 204) are included in second palettes 192 (a value of "1" in vector 216), while the third entry of first palettes 184 is not included in second palettes 192 (a value of "0" in vector 216). In the example of FIG. 4, the vector is a Boolean vector.

[0197] In some examples, video encoder 20 and video decoder 30 may determine a palette predictor list (which may also be referred to as a palette predictor table) when performing palette prediction. The palette predictor list may include entries from palettes of one or more neighboring blocks that are used to predict one or more entries of a palette for coding a current block. Video encoder 20 and video decoder 30 may construct the list in the same manner. Video encoder 20 and video decoder 30 may code data (such as vector 216) to indicate which entries of the palette predictor list are to be included in a palette for coding a current block.

[0198] FIG. 5 is a conceptual diagram illustrating an example of determining indices to a palette for a block of pixels, consistent with techniques of this disclosure. For example, FIG. 5 includes an index block 240 (which may also be referred to as map 240 or index map 240) including index values (e.g., index values 1, 2, and 3) that relate respective positions of pixels associated with the index values to an entry of palettes 244.

[0199] While index block 240 is illustrated in the example of FIG. 5 as including an index value for each pixel position, it should be understood that in other examples, not all pixel positions may be associated with an index value relating the pixel value to an entry of palettes 244. That is, as noted above, in some examples, video encoder 20 may encode (and video decoder 30 may obtain, from an encoded bitstream) an indication of

an actual pixel value (or its quantized version) for a position in index block 240 if the pixel value is not included in palettes 244.

[0200] In some examples, video encoder 20 and video decoder 30 may be configured to code an additional map indicating which pixel positions are associated with which index values. For example, assume that the (i, j) entry in the index block 240 corresponds to the (i, j) position of a CU. Video encoder 20 may encode one or more syntax elements for each entry of the index block (i.e., each pixel position) indicating whether the entry has an associated index value. For example, video encoder 20 may encode a flag having a value of one to indicate that the pixel value at the (i, j) location in the CU is one of the values in palettes 244.

[0201] Video encoder 20 may, in such an example, also encode a palette (shown in the example of FIG. 5 as 244). In instances in which palettes 244 include a single entry and associated pixel value, video encoder 20 may skip the signaling of the index value. Video encoder 20 may encode the flag to have a value of zero to indicate that the pixel value at the (i, j) location in the CU is not one of the values in palettes 244. In this example, video encoder 20 may also encode an indication of the pixel value for use by video decoder 30 in reconstructing the pixel value. In some instances, the pixel value may be coded in a lossy manner.

[0202] The value of a pixel in one position of a CU may provide an indication of values of one or more other pixels in other positions of the CU. For example, there may be a relatively high probability that neighboring pixel positions of a CU will have the same pixel value or may be mapped to the same index value (in the case of lossy coding, in which more than one pixel value may be mapped to a single index value).

[0203] Accordingly, video encoder 20 may encode one or more syntax elements indicating a number of consecutive pixels or index values in a given scan order that have the same pixel value or index value. As noted above, the string of like-valued pixel or index values may be referred to herein as a run. In an example for purposes of illustration, if two consecutive pixels or indices in a given scan order have different values, the run is equal to zero. If two consecutive pixels or indices in a given scan order have the same value but the third pixel or index in the scan order has a different value, the run is equal to one. For three consecutive indices or pixels with the same value, the run is two, and so forth. Video decoder 30 may obtain the syntax elements indicating a run from an encoded bitstream and use the data to determine the number of consecutive locations that have the same pixel or index value.

[0204] In some examples in accordance with the techniques of this disclosure, entropy encoding unit 118 and entropy decoding unit 150 may be configured to entropy code index block 240. For example, entropy encoding unit 118 and entropy decoding unit 150 may be configured to entropy code run-lengths (e.g., run-length values or codes) and/or a binary palette prediction vector relating to an index block in palette mode.

[0205] FIG. 6 is a conceptual diagram illustrating an example of determining maximum copy above run-length, assuming an example of a raster scanning order, consistent with techniques of this disclosure. In the example of FIG. 6, if none of the pixels encompassed by dashed lines 280 is coded as an escape sample, the maximum possible run-length is 35 (i.e. the number of unshaded pixel positions). If one or more of the pixels within dashed lines 280 is coded as an escape sample, assuming that the pixel marked as the escape pixel (the pixel position with the "X") is the first escape pixel within dashed lines 280 in scanning order, then the maximum possible coded copy above run-length is five.

[0206] In some examples, video decoder 30 may only determine the run mode (e.g., the palette mode in which the pixels are coded) for the pixels within dashed lines 280. Hence, in the worst case, video decoder 30 makes the determination for BlockWidth-1 pixels. In some examples, video decoder 30 may be configured to implement certain restrictions regarding the maximum of number of pixels for which the run mode is checked. For example, video decoder 30 may only check the pixels within dashed lines 280 if the pixels are in the same row as the current pixel. Video decoder 30 may infer that all other pixels within dashed lines 280 are not coded as escape samples. The example in FIG. 6 assumes a raster scanning order. The techniques however, may be applied to other scanning orders, such as vertical, horizontal traverse, and vertical traverse.

[0207] In accordance with an example of this disclosure, if the current run mode is 'copy above,' the run-length's contexts for a current pixel may depend on the index value of the above-neighboring pixel's index relative to the current pixel. In this example, if the above-neighboring pixel relative to the current pixel is outside of the current CU, the video decoder assumes that the corresponding index equals to a predefined constant k. In some examples, $k = 0$.

[0208] During entropy coding, an entropy encoder or decoder may place bits of a symbol to be encoded or decoded into one or more bins. The bins may indicate whether a value of a symbol is equal to zero. The entropy coder or entropy decoder may use the

values of the bins to adjust entropy coding process. In some examples, an entropy encoder or decoder may also use bins to indicate whether a values is greater than a specific value, e.g., greater than zero, greater than one, etc.

[0209] In some examples, if the current mode is 'copy above,' the first bin of the run-length codeword selects one of the two candidate CABAC contexts based on whether the above-neighboring sample (e.g., pixel) relative to the current sample (e.g., pixel) equals to 0 or not.

[0210] As another example, if the current mode is 'copy previous,' the first bin of the run-length codeword selects one of the four candidate CABAC contexts based on whether the index value equals to 0, equals 1, equals to 2, or larger than 2.

[0211] FIG. 8 is a flowchart illustrating an example process for decoding video data consistent with techniques of this disclosure. The process of FIG. 8 is generally described as being performed by video decoder 30 for purposes of illustration, although a variety of other processors may also carry out the process shown in FIG. 8. In some examples, block decoding unit 152, palette-based decoding unit 165, and/or entropy decoding unit 150 may perform one or more processes shown in FIG. 8.

[0212] In the example of FIG. 8, video decoder 30 may be configured to receive, from an encoded video bitstream, a palette mode encoded block of video data of a picture (800). Video decoder 30 may be configured to receive, from the encoded video bitstream, encoded palette mode information for the palette mode encoded block of video data (802). In some examples, the encoded palette mode information may include a plurality of instances of a first syntax element and a plurality of syntax elements that are different from the first syntax element. For example, the first syntax element may include palette_index_idc or palette_escape_val, and the plurality of syntax elements that are different from the first syntax element may include a palette_run_msb_id_plus1 syntax element. As another example, the first syntax element may be an indication of an index to an array of palette entries or the first syntax element may specify a quantized escape coded sample value for a color component corresponding to an escape sample. The plurality of syntax elements that are different from the first syntax element may include a syntax element that specifies an index of a most significant bit in a binary representation of a variable representing run length and a syntax element that specifies a run type mode.

[0213] As another example, the plurality of syntax elements that are different from the first syntax element may be any and all syntax elements that are different from the first

syntax element. As described herein with respect to some examples, the plurality of syntax elements that are different from the first syntax element may also be different from second, third, and/or fourth syntax elements. In such examples, the plurality of syntax elements that are different from the first, second, third, and fourth syntax elements may be any and all syntax elements that are different from the first, second, third, and/or fourth syntax elements. In some examples, the plurality of syntax elements that are different from the first syntax element may be any and all syntax elements that are not bypass mode decoded and/or that are not to be bypass mode decoded.

[0214] Video decoder 30 may be configured to decode, using bypass mode, e.g., the bypass mode of a CABAC coding process, the plurality of instances of the first syntax element before decoding the plurality of syntax elements that are different from the first syntax element using context mode (804). Video decoder 30 may be configured to decode, using context mode, e.g., the regular CABAC mode (rather than the bypass mode), the plurality of syntax elements that are different from the first syntax element after decoding the plurality of instances of the first syntax element using bypass mode (806). In some examples, the plurality of instances of the first syntax element includes all instances of the first syntax element for the palette mode encoded block of video data. In such examples, all instances of the first syntax element are decoded using bypass mode before decoding any subsequent data, such as the plurality of syntax elements that are different from the first syntax element. Otherwise stated, video decoder 30 may be configured to decode, using context mode, the plurality of syntax elements that are different from the first syntax element after decoding all instances of the first syntax element for the palette mode encoded block of video data using bypass mode.

[0215] Video decoder 30 may be configured to decode the palette mode encoded block of video data using the decoded plurality of instances of the first syntax element and the decoded plurality of syntax elements that are different from the first syntax element (808). In some examples, the plurality of instances of the first syntax element are grouped together such that switching between bypass mode and context mode while decoding the palette mode encoded block of video data is reduced.

[0216] In some examples, the encoded palette mode information may include a second syntax element indicating a number of instances of the first syntax element (e.g., indicating how many instances of the first syntax element there are for the palette mode encoded block of video data). The plurality of syntax elements that are different from

the first syntax element may also be different from the second syntax element. In such examples, video decoder 30 may be configured to decode, using bypass mode, the second syntax element before decoding the plurality of syntax elements that are different from the first syntax element and the second syntax element. In some examples, no instance of the second syntax element is interleaved between any two instances of the first syntax element for the palette mode encoded block of video data. In some examples, video decoder 30 may be configured to determine, after decoding a number of instances of the first syntax element equal to the number indicated by the second syntax element, that subsequent data in the encoded video bitstream following the number of instances of the first syntax element correspond to the plurality of syntax elements that are different from the first syntax element and the second syntax element. In some examples, video decoder 30 may be configured to decode the second syntax element using a concatenation of truncated Rice code and exponential Golomb code.

[0217] In some examples, the encoded palette mode information may include a third syntax element and a fourth syntax element. In such examples, video decoder 30 may be configured to decode the third syntax element to determine a value corresponding to the third syntax element indicative of whether the palette mode encoded block of video data includes an escape pixel. Video decoder 30 may be configured to decode the fourth syntax element to determine a value corresponding to the fourth syntax element indicative of palette size. Video decoder 30 may be configured to decode, based on the determined values respectively corresponding to the third and fourth syntax elements, the plurality of syntax elements that are different from the first syntax element and the second syntax element using context mode after decoding the plurality of instances of the first syntax element and the second syntax element using bypass mode.

[0218] In some examples, the encoded palette mode information may include another syntax element, and video decoder 30 may be configured to decode this other syntax element to determine a value corresponding to this other syntax element that specifies a number of distinct values that a palette index has for the palette mode encoded block of video data. Video decoder 30 may be configured to decode, based on the determined value corresponding to this other syntax element, the plurality of syntax elements that are different from the first syntax element and the second syntax element using context mode after decoding the plurality of instances of the first syntax element and the second syntax element using bypass mode.

[0219] In some examples, the encoded palette mode information may include another syntax element, and video decoder 30 may be configured to decode this other syntax element to determine a value corresponding to this other syntax element indicative of a last instance of a syntax element of palette_run_type_flag[xC][yC] for the palette mode encoded block of video data.

[0220] In some examples, video decoder 30 may be configured to determine the encoded block of video data has one or more escape samples. In such examples, video decoder 30 may be configured to decode a last escape sample in the encoded block of video data among the one or more escape samples. Video decoder 30 may be configured to infer an index value that applies to samples of the encoded block of video data following the last escape sample. Video decoder 30 may be configured to decode the samples of the encoded block of video data following the last escape sample using the inferred index value for each sample of the samples following the last escape sample.

[0221] In some examples, video decoder 30 may be configured to determine a number of palette indices received. In such examples, video decoder 30 may be configured to determine a number of palette indices left based on the number of palette indices received and the number of instances of the first syntax element. Video decoder 30 may be configured to determine a maximum possible run value for the encoded block of video data based on the number of palette indices received and the number of instances of the first syntax element. In some examples, video decoder 30 may be configured to determine the maximum possible run value for the encoded block of video data according to: nCbS * nCbS – scanPos – 1 – paletteIndicesLeft, where nCbS specifies a size of the encoded block of video data, scanPos specifies scan position, and paletteIndicesLeft specifies the number of palette indices left.

[0222] FIG. 9 is a flowchart illustrating an example process for encoding video data consistent with techniques of this disclosure. The process of FIG. 9 is generally described as being performed by video encoder 20 for purposes of illustration, although a variety of other processors may also carry out the process shown in FIG. 9. In some examples, block encoding unit 100, palette-based encoding unit 122, and/or entropy encoding unit 118 may perform one or more processes shown in FIG. 9.

[0223] In the example of FIG. 9, video encoder 20 may be configured to determine that a block of video data is to be encoded in palette mode (900). Video encoder 20 may be configured to encode the block of video data using palette mode into an encoded

bitstream (902). In some examples, video encoder 20 may be configured to generate palette mode information for the block of video data (904). The palette mode information may include a plurality of instances of a first syntax element and a plurality of syntax elements that are different from the first syntax element. For example, the first syntax element may include palette_index_idc or palette_escape_val, and the plurality of syntax elements that are different from the first syntax element may include a palette_run_msb_id_plus1 syntax element. As another example, the first syntax element may be an indication of an index to an array of palette entries or the first syntax element may specify a quantized escape coded sample value for a color component corresponding to an escape sample. The plurality of syntax elements that are different from the first syntax element may include a syntax element that specifies an index of a most significant bit in a binary representation of a variable representing run length and a syntax element that specifies a run type mode.

[0224] As another example, the plurality of syntax elements that are different from the first syntax element may be any and all syntax elements that are different from the first syntax element. As described herein with respect to some examples, the plurality of syntax elements that are different from the first syntax element may also be different from second, third, and/or fourth syntax elements. In such examples, the plurality of syntax elements that are different from the first, second, third, and fourth syntax elements may be any and all syntax elements that are different from the first, second, third, and/or fourth syntax elements. In some examples, the plurality of syntax elements that are different from the first syntax element may be any and all syntax elements that are not bypass mode encoded and/or that are not to be bypass mode encoded.

[0225] Video encoder 20 may be configured to encode, using bypass mode, e.g., the bypass mode of a CABAC coding process, the plurality of instances of the first syntax element into the encoded bitstream before encoding the plurality of syntax elements that are different from the first syntax element into the encoded bitstream using context mode (906). Video encoder 20 may be configured to encode, using context mode, e.g., the regular CABAC context-based mode, the plurality of syntax elements that are different from the first syntax element into the encoded bitstream after encoding the plurality of instances of the first syntax element using bypass mode into the encoded bitstream (908). In some examples, the plurality of instances of the first syntax element are grouped together such that switching between bypass mode and context mode while encoding the palette mode encoded block of video data is reduced.

[0226] In some examples, the plurality of instances of the first syntax element includes all instances of the first syntax element for the block of video data. In such examples, all instances of the first syntax element are encoded using bypass mode before encoding any subsequent data, such as the plurality of syntax elements that are different from the first syntax element. Otherwise stated, video encoder 20 may be configured to encode, using context mode, the plurality of syntax elements that are different from the first syntax element after encoding all instances of the first syntax element for the block of video data using bypass mode.

[0227] In some examples, the palette mode information may include a second syntax element indicating a number of instances of the first syntax element (e.g., indicating how many instances of the first syntax element there are for the block of video data). The plurality of syntax elements that are different from the first syntax element may also be different from the second syntax element. In such examples, video encoder 20 may be configured to encode, using bypass mode, the second syntax element into the encoded bitstream before the encoding of the plurality of syntax elements that are different from the first syntax element and the second syntax element. In some examples, video encoder 20 may be configured to encode the plurality of instances of the first syntax element such that no instance of the second syntax element is interleaved between any two instances of the first syntax element for the palette mode encoded block of video data in the encoded bitstream. In some examples, video encoder 20 may be configured to encode the second syntax element into the encoded bitstream after the encoded plurality of instances of the first syntax element in the encoded bitstream. For example, video encoder 20 may be configured to first encode all instances of the first syntax element, and then encode the second syntax element into the encoded bitstream. In some examples, video encoder 20 may be configured to encode the second syntax element using a concatenation of truncated Rice code and exponential Golomb code.

[0228] In some examples, the palette mode information may include a third syntax element and a fourth syntax element. In such examples, video encoder 20 may be configured to encode a value corresponding to the third syntax element indicative of whether the block of video data includes an escape pixel into the encoded bitstream. Video encoder 20 may be configured to a value corresponding to the fourth syntax element indicative of palette size into the encoded bitstream. In some examples, the palette mode information may include another syntax element, and video encoder 20 may be configured to encode a value corresponding to this other syntax element that

specifies a number of distinct values that a palette index has for the block of video data into the encoded bitstream.

[0229] In some examples, the encoded palette mode information may include another syntax element, and video encoder 20 may be configured to encode a value corresponding to this other syntax element indicative of a last instance of a syntax element of palette_run_type_flag[xC][yC] for the block of video data.

[0230] In some examples, video encoder 20 may be configured to encode a last escape sample in the block of video data among the one or more escape samples. In such examples, video encoder 20 may be configured to infer an index value that applies to samples of the block of video data following the last escape sample. Video encoder 20 may be configured to encode the samples of the block of video data following the last escape sample using the inferred index value for each sample of the samples following the last escape sample.

[0231] It should be understood that all of the techniques described herein may be used individually or in combination. For example, video encoder 20 and/or one or more components thereof and video decoder 30 and/or one or more components thereof may perform the techniques described in this disclosure in any combination.

[0232] It is to be recognized that depending on the example, certain acts or events of any of the techniques described herein can be performed in a different sequence, may be added, merged, or left out altogether (e.g., not all described acts or events are necessary for the practice of the techniques). Moreover, in certain examples, acts or events may be performed concurrently, e.g., through multi-threaded processing, interrupt processing, or multiple processors, rather than sequentially. In addition, while certain aspects of this disclosure are described as being performed by a single module or unit for purposes of clarity, it should be understood that the techniques of this disclosure may be performed by a combination of units or modules associated with a video coder.

[0233] Certain aspects of this disclosure have been described with respect to the developing HEVC standard for purposes of illustration. However, the techniques described in this disclosure may be useful for other video coding processes, including other standard or proprietary video coding processes not yet developed.

[0234] The techniques described above may be performed by video encoder 20 (FIGS. 1 and 2) and/or video decoder 30 (FIGS. 1 and 3), both of which may be generally referred to as a video coder. Likewise, video coding may refer to video encoding or video decoding, as applicable.

[0235] In accordance with this disclosure, the term "or" may be interrupted as "and/or" where context does not dictate otherwise. Additionally, while phrases such as "one or more" or "at least one" or the like may have been used for some features disclosed herein but not others; the features for which such language was not used may be interpreted to have such a meaning implied where context does not dictate otherwise.

[0236] While particular combinations of various aspects of the techniques are described above, these combinations are provided merely to illustrate examples of the techniques described in this disclosure. Accordingly, the techniques of this disclosure should not be limited to these example combinations and may encompass any conceivable combination of the various aspects of the techniques described in this disclosure.

[0237] 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.

[0238] 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.

[0239] 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 gate 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.

[0240] 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.

[0241] Various examples have been described herein. Any combination of the described systems, operations, functions, or examples is contemplated. These and other examples are within the scope of the following claims.

WHAT IS CLAIMED IS:

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

receiving, from an encoded video bitstream, a palette mode encoded block of video data of a picture;

receiving, from the encoded video bitstream, encoded palette mode information for the palette mode encoded block of video data, wherein the encoded palette mode information includes a plurality of instances of a first syntax element and a plurality of syntax elements that are different from the first syntax element;

decoding, using bypass mode, the plurality of instances of the first syntax element before decoding the plurality of syntax elements that are different from the first syntax element using context mode;

decoding, using context mode, the plurality of syntax elements that are different from the first syntax element after decoding the plurality of instances of the first syntax element using bypass mode; and

decoding the palette mode encoded block of video data using the decoded plurality of instances of the first syntax element and the decoded plurality of syntax elements that are different from the first syntax element.

2. The method of claim 1, wherein the plurality of instances of the first syntax element includes all instances of the first syntax element for the palette mode encoded block of video data.

3. The method of claim 1, wherein the first syntax element is an indication of an index to an array of palette entries or specifies a quantized escape coded sample value for a color component corresponding to an escape sample, and wherein the plurality of syntax elements that are different from the first syntax element includes a syntax element that specifies an index of a most significant bit in a binary representation of a variable representing run length and a syntax element that specifies a run type mode.

4. The method of claim 1, wherein the first syntax element is palette_index_idc or palette_escape_val, and wherein the plurality of syntax elements that are different from the first syntax element includes a palette_run_msb_id_plus1 syntax element.

5. The method of claim 1, wherein the plurality of instances of the first syntax element are grouped together such that switching between bypass mode and context mode while decoding the palette mode encoded block of video data is reduced.

6. The method of claim 1, wherein the encoded palette mode information includes a second syntax element indicating a number of instances of the first syntax element, wherein the plurality of syntax elements that are different from the first syntax element are different from the second syntax element, and wherein the method further comprises:

decoding, using bypass mode, the second syntax element before the decoding of the plurality of syntax elements that are different from the first syntax element and the second syntax element.

7. The method of claim 6, wherein no instance of the second syntax element is interleaved between any two instances of the first syntax element for the palette mode encoded block of video data.

8. The method of claim 6, further comprising:

determining, after decoding a number of instances of the first syntax element equal to the number indicated by the second syntax element, that subsequent data in the encoded video bitstream following the number of instances of the first syntax element correspond to the plurality of syntax elements that are different from the first syntax element and the second syntax element.

9. The method of claim 6, wherein the encoded palette mode information includes a third syntax element and a fourth syntax element, wherein the method further comprises:

decoding the third syntax element to determine a value corresponding to the third syntax element indicative of whether the palette mode encoded block of video data includes an escape sample;

decoding the fourth syntax element to determine a value corresponding to the fourth syntax element indicative of palette size; and

decoding, based on the determined values respectively corresponding to the third and fourth syntax elements, the plurality of syntax elements that are different from the first syntax element and the second syntax element using context mode after decoding the plurality of instances of the first syntax element and the second syntax element using bypass mode.

10. The method of claim 6, wherein the encoded palette mode information includes a third syntax element, wherein the method further comprises:

decoding the third syntax element to determine a value corresponding to the third syntax element that specifies a number of distinct values that a palette index has for the palette mode encoded block of video data; and

decoding, based on the determined value corresponding to the third syntax element, the plurality of syntax elements that are different from the first syntax element and the second syntax element using context mode after decoding the plurality of instances of the first syntax element and the second syntax element using bypass mode.

11. The method of claim 6, wherein the encoded palette mode information includes a third syntax element, wherein the method further comprises:

decoding the third syntax element to determine a value corresponding to the third syntax element indicative of a last instance of a syntax element of palette_run_type_flag[xC][yC] for the palette mode encoded block of video data.

12. The method of claim 6, further comprising:

decoding the second syntax element using a concatenation of truncated Rice code and exponential Golomb code.

13. The method of claim 1, further comprising:

determining the encoded block of video data has one or more escape samples;

decoding a last escape sample in the encoded block of video data among the one or more escape samples;

inferring an index value that applies to samples of the encoded block of video data following the last escape sample; and

decoding the samples of the encoded block of video data following the last escape sample using the inferred index value for each sample of the samples following the last escape sample.

14. The method of claim 6, further comprising:

determining a number of palette indices received;

determining a number of palette indices left based on the number of palette indices received and the number of instances of the first syntax element; and

determining a maximum possible run value for the encoded block of video data based on the number of palette indices received and the number of instances of the first syntax element.

15. The method of claim 14, further comprising:

determining the maximum possible run value for the encoded block of video data according to: nCbS * nCbS – scanPos – 1 – paletteIndicesLeft, wherein nCbS specifies a size of the encoded block of video data, scanPos specifies scan position, and paletteIndicesLeft specifies the number of palette indices left.

16. A device for decoding video data, the device comprising:

a memory configured to store the video data; and

a video decoder in communication with the memory, the video decoder configured to:

receive a palette mode encoded block of video data of a picture from the memory;

receive encoded palette mode information for the palette mode encoded block of video data, wherein the encoded palette mode information includes a plurality of instances of a first syntax element and a plurality of syntax elements that are different from the first syntax element;

decode, using bypass mode, the plurality of instances of the first syntax element before decoding the plurality of syntax elements that are different from the first syntax element using context mode;

decode, using context mode, the plurality of syntax elements that are different from the first syntax element after decoding the plurality of instances of the first syntax element using bypass mode; and

decode the palette mode encoded block of video data using the decoded plurality of instances of the first syntax element and the decoded plurality of syntax elements that are different from the first syntax element.

17. The device of claim 16, wherein the plurality of instances of the first syntax element includes all instances of the first syntax element for the palette mode encoded block of video data.

18. The device of claim 16, wherein the first syntax element is an indication of an index to an array of palette entries or specifies a quantized escape coded sample value for a color component corresponding to an escape sample, and wherein the plurality of syntax elements that are different from the first syntax element includes a syntax element that specifies an index of a most significant bit in a binary representation of a variable representing run length and a syntax element that specifies a run type mode.

19. The device of claim 16, wherein the first syntax element is palette_index_idc or palette_escape_val, and wherein the plurality of syntax elements that are different from the first syntax element includes a palette_run_msb_id_plus1 syntax element.

20. The device of claim 16, wherein the plurality of instances of the first syntax element are grouped together such that switching between bypass mode and context mode while decoding the palette mode encoded block of video data is reduced.

21. The device of claim 16, wherein the encoded palette mode information includes a second syntax element indicating a number of instances of the first syntax element, wherein the plurality of syntax elements that are different from the first syntax element are different from the second syntax element, and wherein the video decoder is further configured to:

decode, using bypass mode, the second syntax element before the decoding of the plurality of syntax elements that are different from the first syntax element and the second syntax element.

22. The device of claim 21, wherein no instance of the second syntax element is interleaved between any two instances of the first syntax element for the palette mode encoded block of video data.

23. The device of claim 21, wherein the video decoder is further configured to determine, after decoding a number of instances of the first syntax element equal to the number indicated by the second syntax element, that subsequent data in the encoded video bitstream following the number of instances of the first syntax element correspond to the plurality of syntax elements that are different from the first syntax element and the second syntax element.

24. The device of claim 21, wherein the encoded palette mode information includes a third syntax element and a fourth syntax element, wherein the video decoder is further configured to:

decode the third syntax element to determine a value corresponding to the third syntax element indicative of whether the palette mode encoded block of video data includes an escape sample;

decode the fourth syntax element to determine a value corresponding to the fourth syntax element indicative of palette size; and

decode, based on the determined values respectively corresponding to the third and fourth syntax elements, the plurality of syntax elements that are different from the first syntax element and the second syntax element using context mode after decoding the plurality of instances of the first syntax element and the second syntax element using bypass mode.

25. The device of claim 21, wherein the encoded palette mode information includes a third syntax element, wherein the video decoder is further configured to:

decode the third syntax element to determine a value corresponding to the third syntax element that specifies a number of distinct values that a palette index has for the palette mode encoded block of video data; and

decode, based on the determined value corresponding to the third syntax element, the plurality of syntax elements that are different from the first syntax element and the second syntax element using context mode after decoding the plurality of instances of the first syntax element and the second syntax element using bypass mode.

26. The device of claim 21, wherein the encoded palette mode information includes a third syntax element, wherein the video decoder is further configured to:

decode the third syntax element to determine a value corresponding to the third syntax element indicative of a last instance of a syntax element of palette_run_type_flag[xC][yC] for the palette mode encoded block of video data.

27. The device of claim 21, wherein the video decoder is further configured to:

decode the second syntax element using a concatenation of truncated Rice code and exponential Golomb code.

28. The device of claim 16, wherein the video decoder is further configured to:

determine the encoded block of video data has one or more escape samples;

decode a last escape sample in the encoded block of video data among the one or more escape samples;

infer an index value that applies to samples of the encoded block of video data following the last escape sample; and

decode the samples of the encoded block of video data following the last escape sample using the inferred index value for each sample of the samples following the last escape sample.

29. The device of claim 21, wherein the video decoder is further configured to:

determine a number of palette indices received;

determine a number of palette indices left based on the number of palette indices received and the number of instances of the first syntax element; and

determine a maximum possible run value for the encoded block of video data based on the number of palette indices received and the number of instances of the first syntax element.

30. The device of claim 29, wherein the video decoder is further configured to:

determine the maximum possible run value for the encoded block of video data according to: nCbS * nCbS – scanPos – 1 – paletteIndicesLeft, wherein nCbS specifies a size of the encoded block of video data, scanPos specifies scan position, and paletteIndicesLeft specifies the number of palette indices left.

31. A non-transitory computer-readable storage medium having instructions stored thereon that, when executed, cause one or more processors to:

receive a palette mode encoded block of video data of a picture from a memory;

receive encoded palette mode information for the palette mode encoded block of video data, wherein the encoded palette mode information includes a plurality of instances of a first syntax element and a plurality of syntax elements that are different from the first syntax element;

decode, using bypass mode, the plurality of instances of the first syntax element before decoding the plurality of syntax elements that are different from the first syntax element using context mode;

decode, using context mode, the plurality of syntax elements that are different from the first syntax element after decoding the plurality of instances of the first syntax element using bypass mode; and

decode the palette mode encoded block of video data using the decoded plurality of instances of the first syntax element and the decoded plurality of syntax elements that are different from the first syntax element.

32. A method of encoding video data, the method comprising:

determining that a block of video data is to be coded in palette mode;

encoding the block of video data using palette mode into an encoded bitstream, wherein encoding the block of video data using palette mode comprises:

generating palette mode information for the block of video data, wherein the palette mode information includes a plurality of instances of a first syntax element and a plurality of syntax elements that are different from the first syntax element;

encoding, using bypass mode, the plurality of instances of the first syntax element into the encoded bitstream before encoding the plurality of syntax elements that are different from the first syntax element into the encoded bitstream using context mode; and

encoding, using context mode, the plurality of syntax elements that are different from the first syntax element into the encoded bitstream after encoding the plurality of instances of the first syntax element using bypass mode into the encoded bitstream.

33. The method of claim 32, wherein the plurality of instances of the first syntax element includes all instances of the first syntax element for the block of video data.

34. The method of claim 32, wherein the first syntax element is an indication of an index to an array of palette entries or specifies a quantized escape coded sample value for a color component corresponding to an escape sample, and wherein the plurality of syntax elements that are different from the first syntax element includes a syntax element that specifies an index of a most significant bit in a binary representation of a variable representing run length and a syntax element that specifies a run type mode.

35. The method of claim 32, wherein the first syntax element is palette_index_idc or palette_escape_val, and wherein the plurality of syntax elements that are different from the first syntax element includes a palette_run_msb_id_plus1 syntax element.

36. The method of claim 32, wherein the plurality of instances of the first syntax element are grouped together such that switching between bypass mode and context mode while encoding the palette mode encoded block of video data is reduced.

37. The method of claim 32, wherein the palette mode information includes a second syntax element indicating a number of instances of the first syntax element, wherein the plurality of syntax elements that are different from the first syntax element are different from the second syntax element, and wherein the method further comprises:

encoding, using bypass mode, the second syntax element into the encoded bitstream before the encoding of the plurality of syntax elements that are different from the first syntax element and the second syntax element.

38. The method of claim 37, wherein no instance of the second syntax element is interleaved between any two instances of the first syntax element for the block of video data.

39. The method of claim 37, further comprising:

encoding the second syntax element into the encoded bitstream after the encoded plurality of instances of the first syntax element in the encoded bitstream.

40. The method of claim 37, wherein the palette mode information includes a third syntax element and a fourth syntax element, wherein the method further comprises:

encoding a value corresponding to the third syntax element indicative of whether the block of video data includes an escape sample into the encoded bitstream; and

encoding a value corresponding to the fourth syntax element indicative of palette size into the encoded bitstream.

41. The method of claim 37, wherein the palette mode information includes a third syntax element, wherein the method further comprises:

encoding a value corresponding to the third syntax element that specifies a number of distinct values that a palette index has for the block of video data into the encoded bitstream.

42. The method of claim 37, wherein the palette mode information includes a third syntax element, wherein the method further comprises:

encoding a value corresponding to the third syntax element indicative of a last instance of a syntax element of palette_run_type_flag[xC][yC] for the block of video data.

43. The method of claim 37, further comprising:

encoding the second syntax element using a concatenation of truncated Rice code and exponential Golomb code.

44. The method of claim 32, further comprising:

encoding a last escape sample in the block of video data among the one or more escape samples;

inferring an index value that applies to samples of the block of video data following the last escape sample; and

encoding the samples of the block of video data following the last escape sample using the inferred index value for each sample of the samples following the last escape sample.

45. A device for encoding video data, the device comprising:

a memory configured to store the video data; and

a video encoder in communication with the memory, the video encoder configured to:

determine that a block of video data stored in the memory is to be encoded in palette mode;

encode the block of video data using palette mode into an encoded bitstream, wherein the video encoder being configured to encode the block of video data using palette mode comprises the video encoder being configured to:

generate palette mode information for the block of video data, wherein the palette mode information includes a plurality of instances of a first syntax element and a plurality of syntax elements that are different from the first syntax element;

encode, using bypass mode, the plurality of instances of the first syntax element into the encoded bitstream before encoding the plurality of syntax elements that are different from the first syntax element into the encoded bitstream using context mode; and

encode, using context mode, the plurality of syntax elements that are different from the first syntax element into the encoded bitstream after encoding the plurality of instances of the first syntax element using bypass mode into the encoded bitstream.

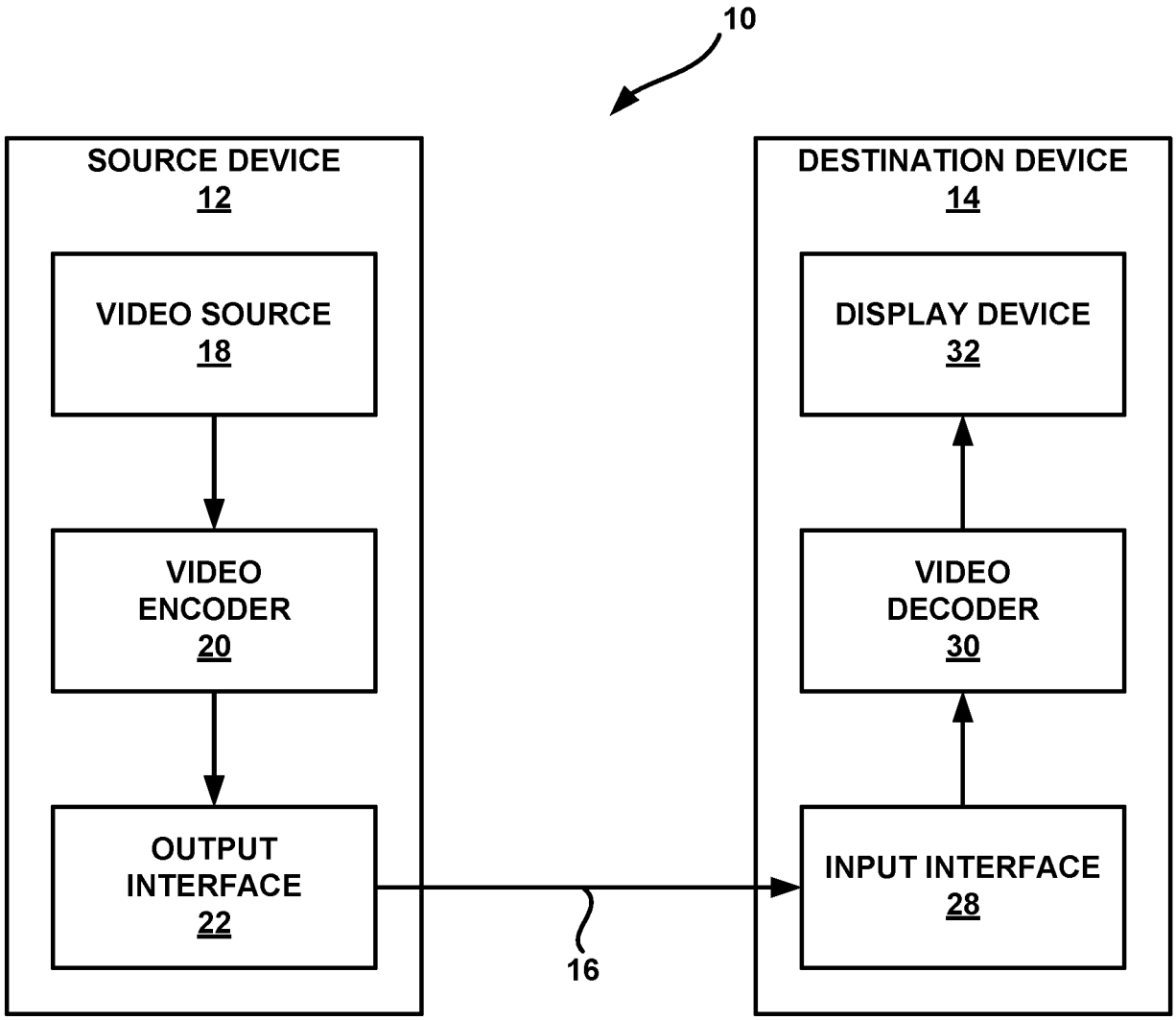

10
SOURCE DEVICE
12
VIDEO SOURCE
18
VIDEO
ENCODER
20
OUTPUT
INTERFACE
22
16
DESTINATION DEVICE
14
DISPLAY DEVICE
32
VIDEO
DECODER
30
INPUT INTERFACE
28

FIG. 1

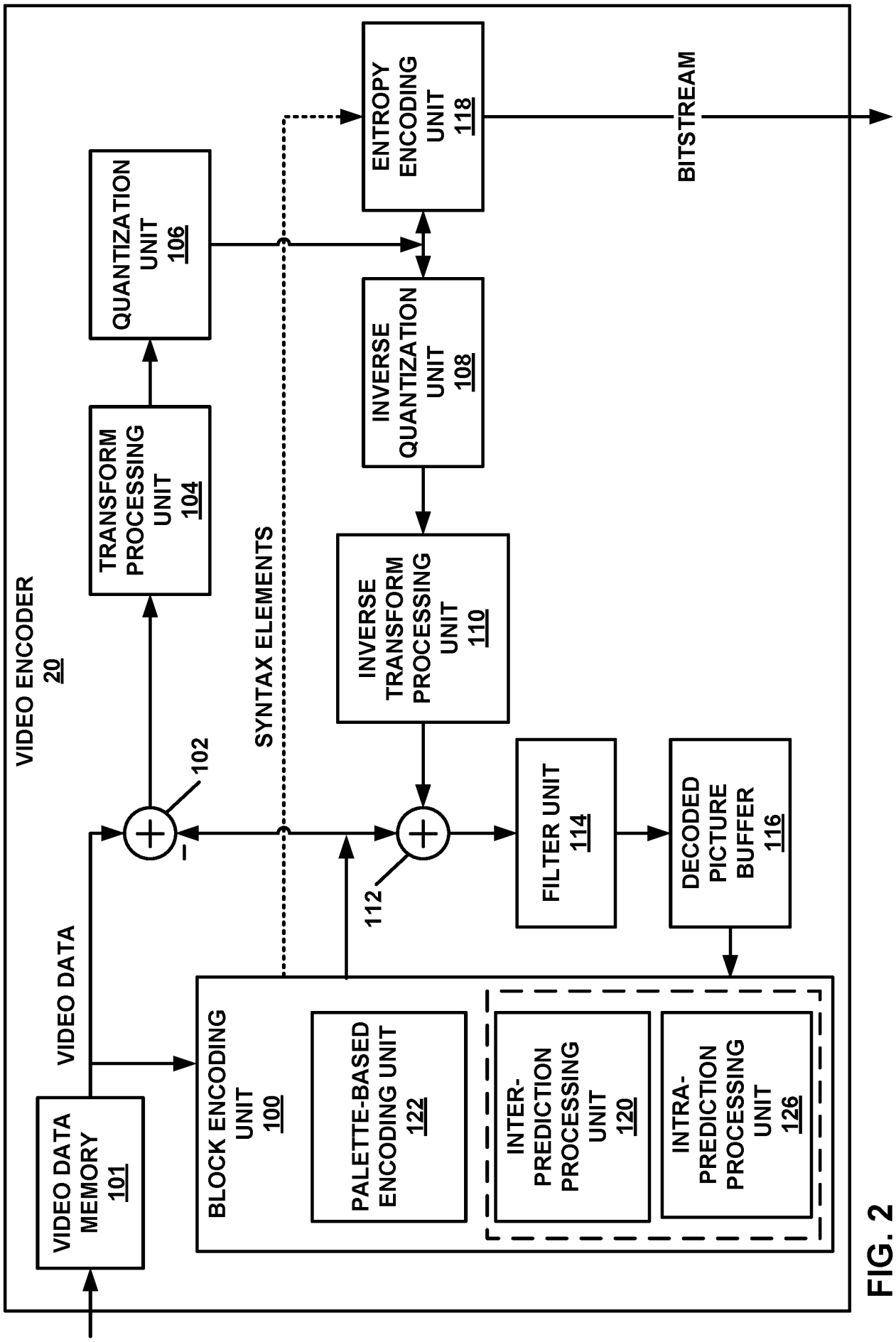

VIDEO ENCODER
20
VIDEO DATA MEMORY
101
VIDEO DATA
BLOCK ENCODING UNIT
100
PALETTE-BASED ENCODING UNIT
122
INTER-PREDICTION PROCESSING UNIT
120
INTRA-PREDICTION PROCESSING UNIT
126
102
TRANSFORM PROCESSING UNIT
104
QUANTIZATION UNIT
106
INVERSE QUANTIZATION UNIT
108
INVERSE TRANSFORM PROCESSING UNIT
110
112
FILTER UNIT
114
DECODED PICTURE BUFFER
116
ENTROPY ENCODING UNIT
118
SYNTAX ELEMENTS
BITSTREAM

FIG. 2

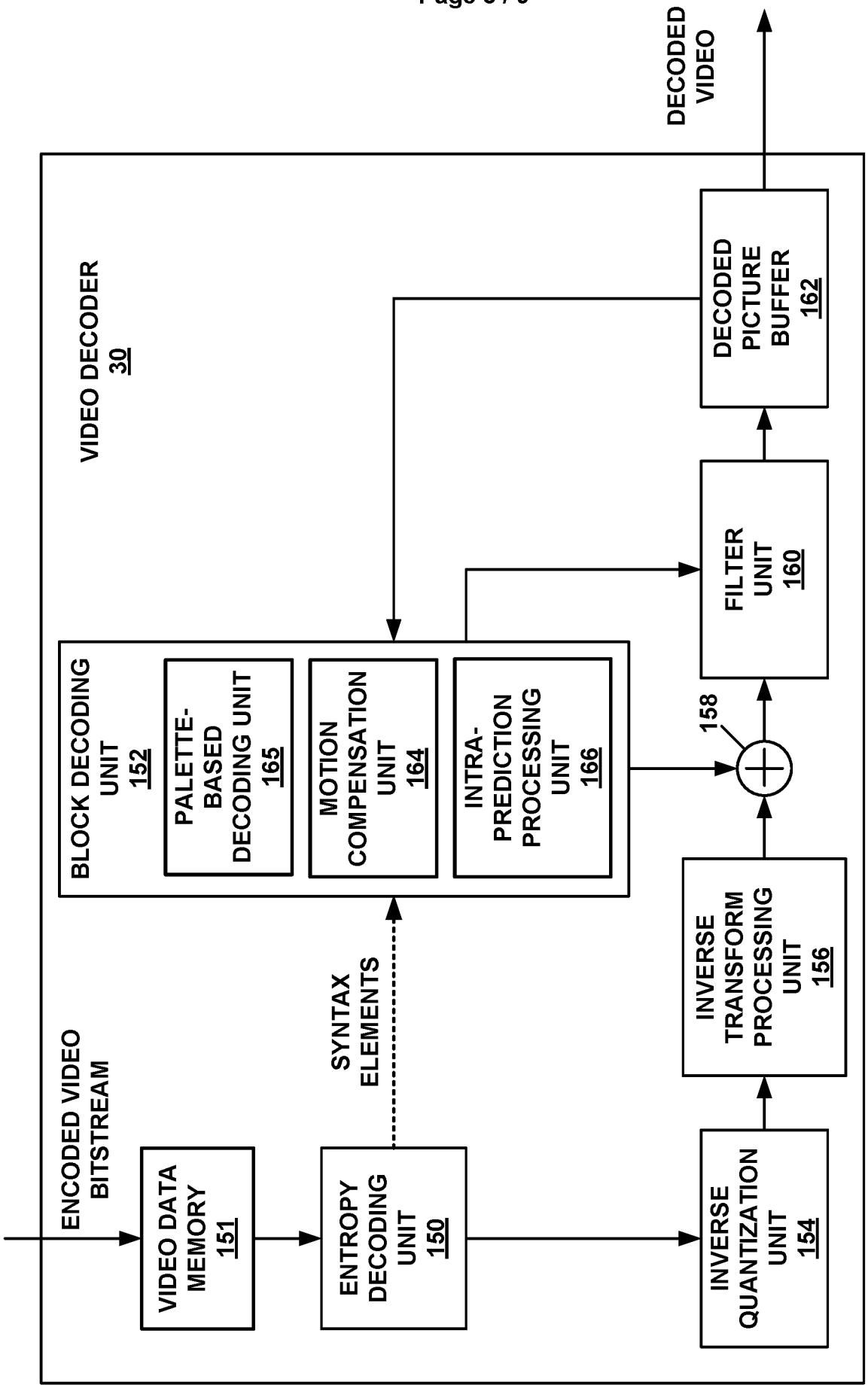

ENCODED VIDEO BITSTREAM
VIDEO DATA MEMORY 151
ENTROPY DECODING UNIT 150
SYNTAX ELEMENTS
INVERSE QUANTIZATION UNIT 154
INVERSE TRANSFORM PROCESSING UNIT 156
158
BLOCK DECODING UNIT 152
PALETTE-BASED DECODING UNIT 165
MOTION COMPENSATION UNIT 164
INTRA-PREDICTION PROCESSING UNIT 166
FILTER UNIT 160
DECODED PICTURE BUFFER 162
VIDEO DECODER 30
DECODED VIDEO

FIG. 3

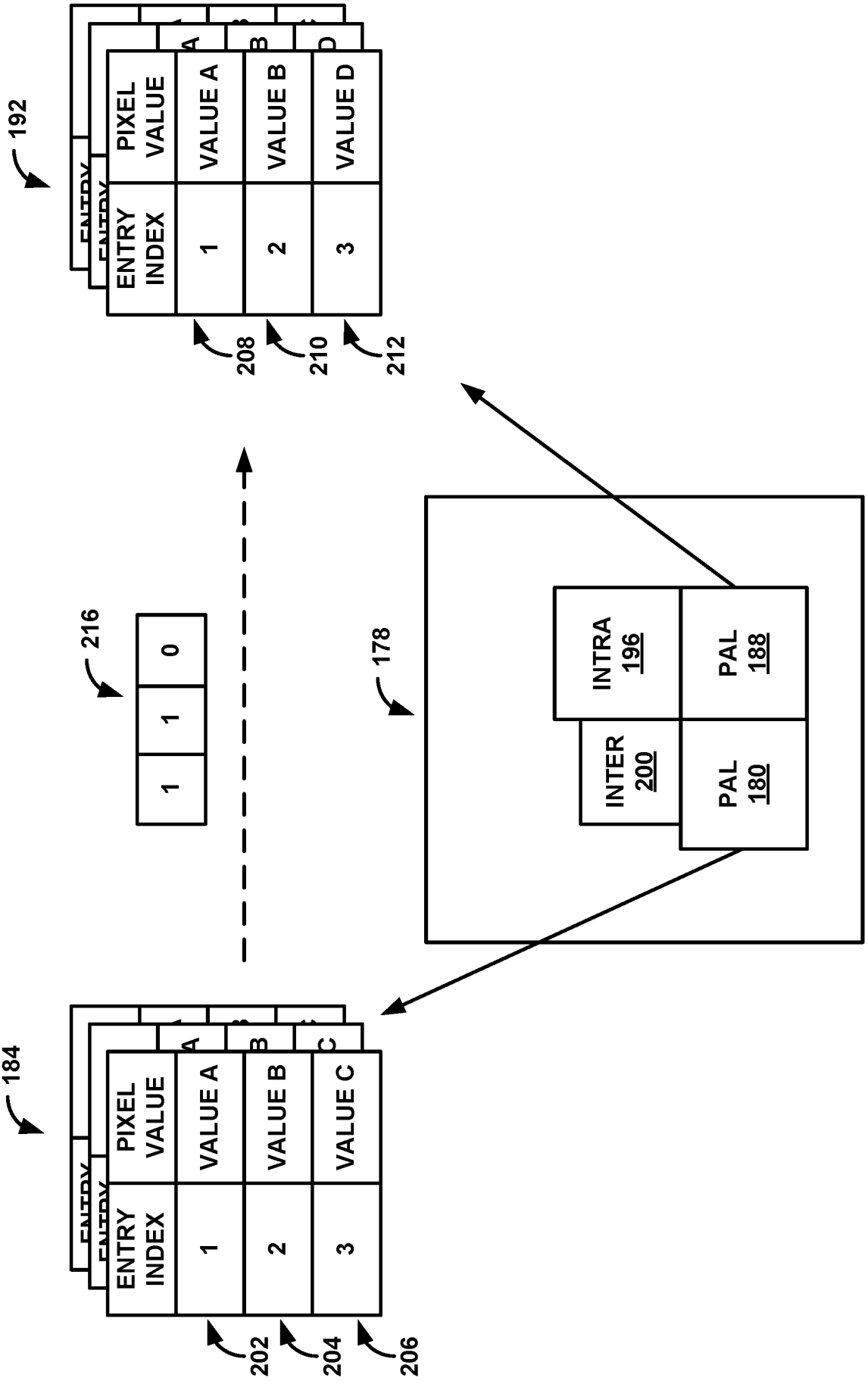

192
ENTRY
ENTRY
PIXEL VALUE
A
B
D
ENTRY INDEX
1
2
3
VALUE A
VALUE B
VALUE D
208
210
212
216
1
1
0
178
INTRA 196
PAL 188
INTER 200
PAL 180
184
ENTRY
ENTRY
PIXEL VALUE
A
B
C
ENTRY INDEX
1
2
3
VALUE A
VALUE B
VALUE C
202
204
206

FIG. 4

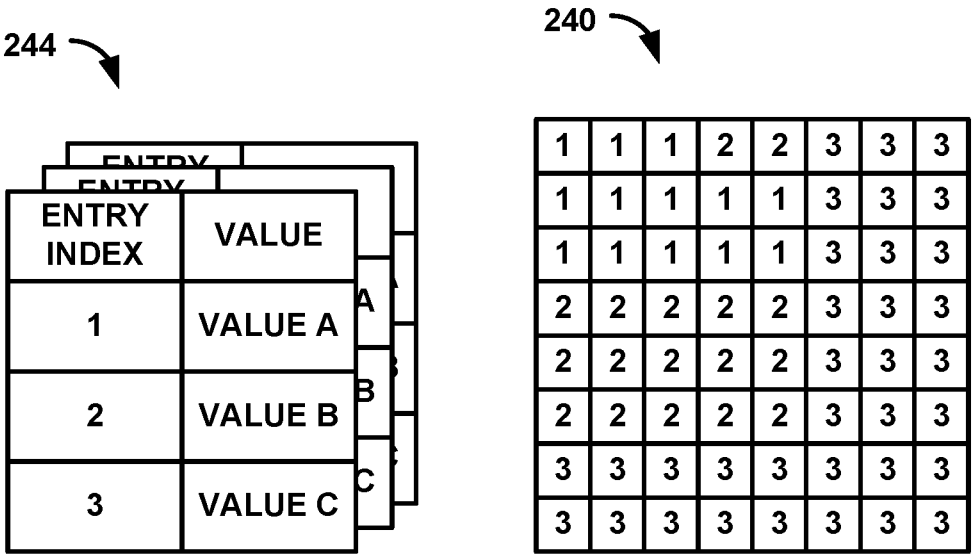

244
240
ENTRY INDEX
VALUE
1
VALUE A
2
VALUE B
3
VALUE C
1 1 1 2 2 3 3 3
1 1 1 1 1 3 3 3
1 1 1 1 1 3 3 3
2 2 2 2 2 3 3 3
2 2 2 2 2 3 3 3
2 2 2 2 2 3 3 3
3 3 3 3 3 3 3 3
3 3 3 3 3 3 3 3

FIG. 5

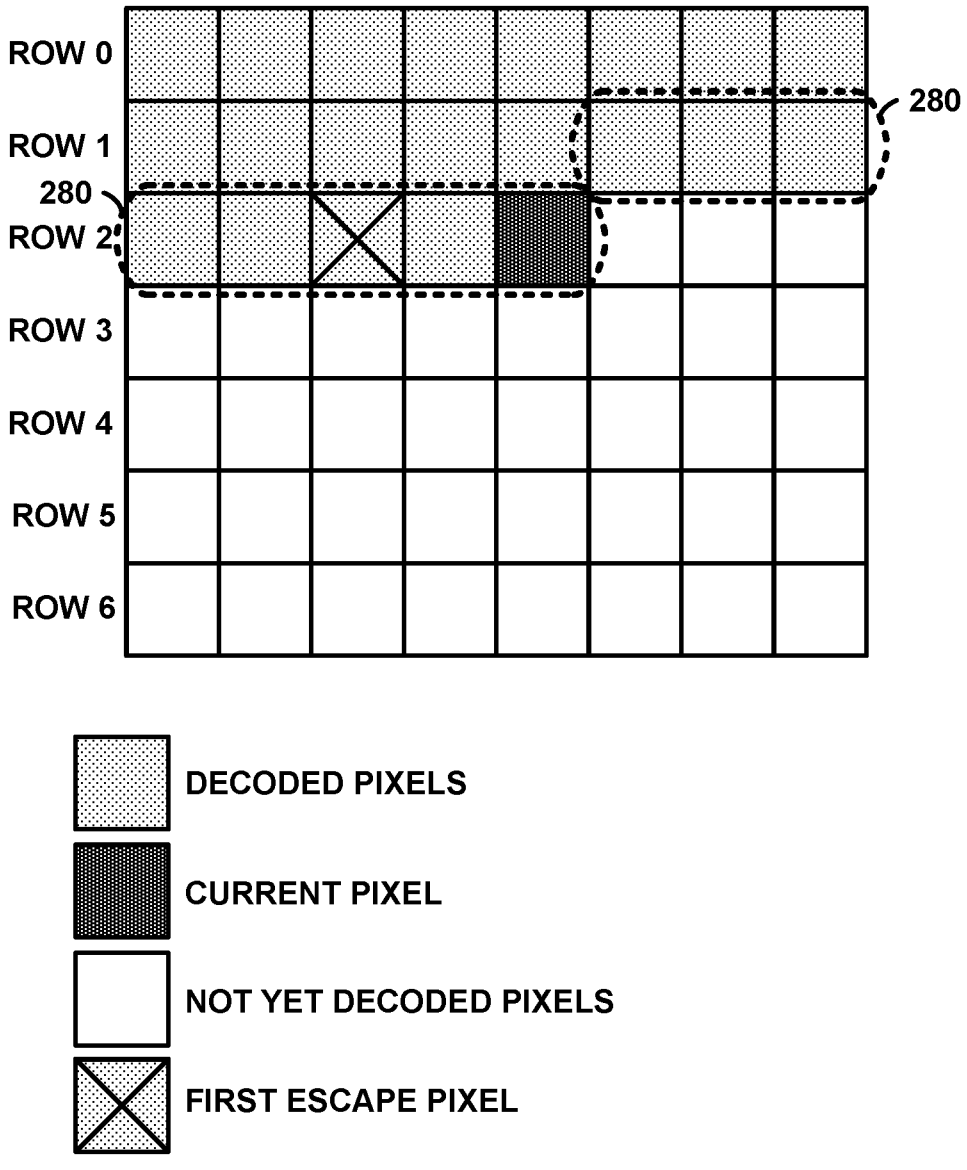

ROW 0
ROW 1
280
ROW 2
ROW 3
ROW 4
ROW 5
ROW 6
280
DECODED PIXELS
CURRENT PIXEL
NOT YET DECODED PIXELS
FIRST ESCAPE PIXEL

FIG. 6

...	
if(currentPaletteSize != 0)	
palette_escape_val_present_flag	ae(v)
indices_idc_coding() (Relocate all occurrence of **palette_index_idc** to here)	
if(palette_escape_val_present_flag) {	
if(cu_qp_delta_enabled_flag && !IsCuQpDeltaCoded) {	
cu_qp_delta_palette_abs	ae(v)
if(cu_qp_delta_palette_abs)	
cu_qp_delta_palette_sign_flag	ae(v)
}	
if(cu_chroma_qp_offset_enabled_flag && !IsCuChromaQpOffsetCoded) {	
cu_chroma_qp_palette_offset_flag	ae(v)
if(cu_chroma_qp_offset_flag && chroma_qp_offset_list_len_minus1 > 0)	
cu_chroma_qp_palette_offset_idx	ae(v)
}	
}	
if(indexMax > 0)	
palette_transpose_flag	ae(v)
scanPos = 0	
while(scanPos < nCbS * nCbS) {	
...	
if(palette_run_type_flag[xC][yC] = = COPY_INDEX_MODE && adjustedIndexMax > 0)	
palette_index_idc	ae(v)
if(indexMax > 0) {	
maxPaletteRun = nCbS * nCbS – scanPos – 1	
if(maxPaletteRun > 0) {	
palette_run_msb_id_plus1	ae(v)
if(palette_run_msb_id_plus1 > 1)	
palette_run_refinement_bits	ae(v)
}	
} else	
...	

FIG. 7

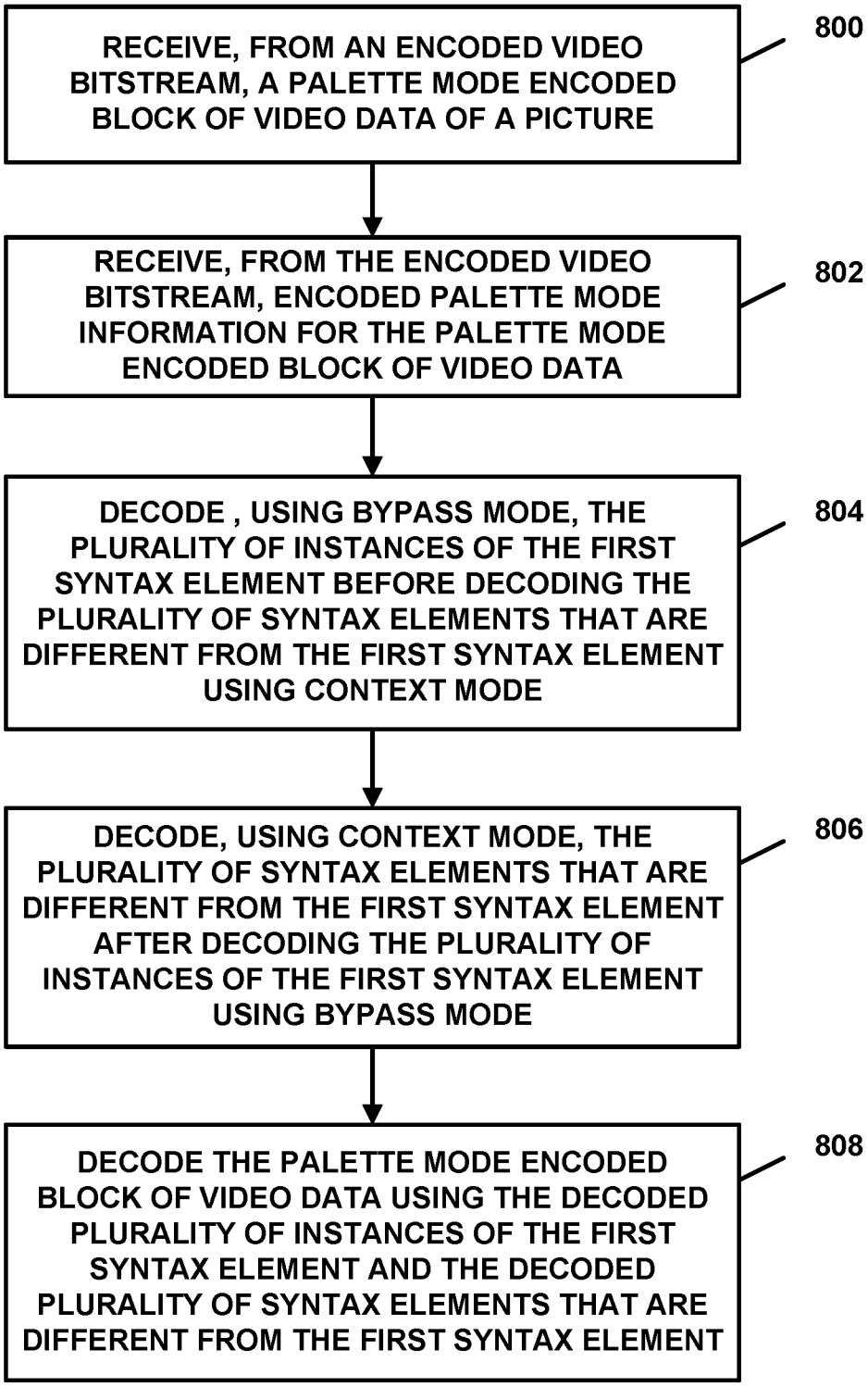

RECEIVE, FROM AN ENCODED VIDEO BITSTREAM, A PALETTE MODE ENCODED BLOCK OF VIDEO DATA OF A PICTURE
800
RECEIVE, FROM THE ENCODED VIDEO BITSTREAM, ENCODED PALETTE MODE INFORMATION FOR THE PALETTE MODE ENCODED BLOCK OF VIDEO DATA
802
DECODE , USING BYPASS MODE, THE PLURALITY OF INSTANCES OF THE FIRST SYNTAX ELEMENT BEFORE DECODING THE PLURALITY OF SYNTAX ELEMENTS THAT ARE DIFFERENT FROM THE FIRST SYNTAX ELEMENT USING CONTEXT MODE
804
DECODE, USING CONTEXT MODE, THE PLURALITY OF SYNTAX ELEMENTS THAT ARE DIFFERENT FROM THE FIRST SYNTAX ELEMENT AFTER DECODING THE PLURALITY OF INSTANCES OF THE FIRST SYNTAX ELEMENT USING BYPASS MODE
806
DECODE THE PALETTE MODE ENCODED BLOCK OF VIDEO DATA USING THE DECODED PLURALITY OF INSTANCES OF THE FIRST SYNTAX ELEMENT AND THE DECODED PLURALITY OF SYNTAX ELEMENTS THAT ARE DIFFERENT FROM THE FIRST SYNTAX ELEMENT
808

FIG. 8

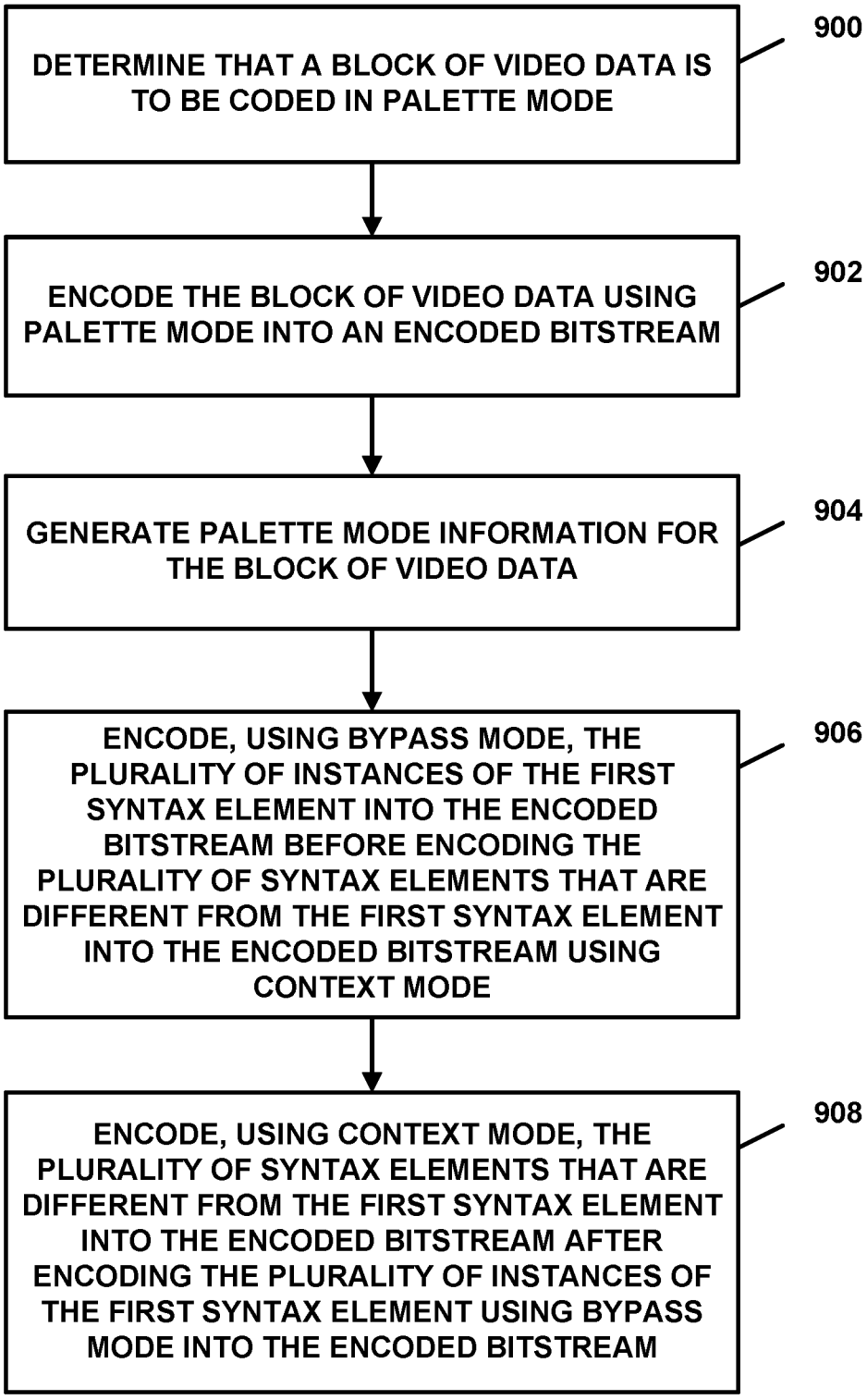

DETERMINE THAT A BLOCK OF VIDEO DATA IS TO BE CODED IN PALETTE MODE
900
ENCODE THE BLOCK OF VIDEO DATA USING PALETTE MODE INTO AN ENCODED BITSTREAM
902
GENERATE PALETTE MODE INFORMATION FOR THE BLOCK OF VIDEO DATA
904
ENCODE, USING BYPASS MODE, THE PLURALITY OF INSTANCES OF THE FIRST SYNTAX ELEMENT INTO THE ENCODED BITSTREAM BEFORE ENCODING THE PLURALITY OF SYNTAX ELEMENTS THAT ARE DIFFERENT FROM THE FIRST SYNTAX ELEMENT INTO THE ENCODED BITSTREAM USING CONTEXT MODE
906
ENCODE, USING CONTEXT MODE, THE PLURALITY OF SYNTAX ELEMENTS THAT ARE DIFFERENT FROM THE FIRST SYNTAX ELEMENT INTO THE ENCODED BITSTREAM AFTER ENCODING THE PLURALITY OF INSTANCES OF THE FIRST SYNTAX ELEMENT USING BYPASS MODE INTO THE ENCODED BITSTREAM
908

FIG. 9

INTERNATIONAL SEARCH REPORT

International application No
PCT/US2016/015663

A. CLASSIFICATION OF SUBJECT MATTER
INV. H04N19/70 H04N19/13 H04N19/593 H04N19/463 H04N19/42
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

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
Y	CHEN J ET AL: "Description of screen content coding technology proposal by Qualcomm", 17. JCT-VC MEETING; 27-3-2014 - 4-4-2014; VALENCIA; (JOINT COLLABORATIVE TEAM ON VIDEO CODING OF ISO/IEC JTC1/SC29/WG11 AND ITU-T SG.16); URL: HTTP://WFTP3.ITU.INT/AV-ARCH/JCTVC-SITE/,, no. JCTVC-Q0031-v3, 28 March 2014 (2014-03-28), XP030115916, Section 2.7.1 with sub-section 2.7.1.1 ----- -/--	1-45

[X] 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
13 April 2016	20/04/2016

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	Colesanti, Carlo

1

Form PCT/ISA/210 (second sheet) (April 2005)

C(Continuation). DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
Y	Rajan Joshi ET AL: "High Efficiency Video Coding (HEVC) Screen Content Coding: Draft 2, Document JCTVC-S1005", , 12 October 2014 (2014-10-12), pages 1-350, XP55263982, 18th Meeting: Sapporo, JP, 30 June - 9 July 2014 Retrieved from the Internet: URL:http://phenix.int-evry.fr/jct/ [retrieved on 2016-04-08] cited in the application Sections 7.3.8.8, 7.4.9.6, 8.4.5.2.8, 9.3.3.13, 9.4.1.2.8; tables 9-43	1-45
Y	Vivienne Sze ET AL: "High Efficiency Video Coding (HEVC) : Algorithms and Architectures - Chapter 8: Entropy Coding in HEVC" In: "High Efficiency Video Coding (HEVC) : Algorithms and Architectures - Chapter 8: Entropy Coding in HEVC", 1 January 2014 (2014-01-01), Springer International Publishing, XP55263413, ISBN: 978-3-319-06894-7 pages 209-269, abstract Sections 8.2.3.2, 8.3 with sub-sections 8.3.3, 8.3.3.2, and 8.8.2 with sub-sections 8.8.2.2	1-45
X,P	KARCZEWICZ M ET AL: "Non CE1: Grouping Palette Indices At Front", 20. JCT-VC MEETING; 10-2-2105 - 18-2-2015; GENEVA; (JOINT COLLABORATIVE TEAM ON VIDEO CODING OF ISO/IEC JTC1/SC29/WG11 AND ITU-T SG.16); URL: HTTP://WFTP3.ITU.INT/AV-ARCH/JCTVC-SITE/,, no. JCTVC-T0065-v5, 11 February 2015 (2015-02-11), XP030117191, the whole document	1-45
X,P	JOSHI R ET AL: "Screen Content Coding Test Model 4 Encoder Description (SCM 4)", 20. JCT-VC MEETING; 10-2-2105 - 18-2-2015; GENEVA; (JOINT COLLABORATIVE TEAM ON VIDEO CODING OF ISO/IEC JTC1/SC29/WG11 AND ITU-T SG.16); URL: HTTP://WFTP3.ITU.INT/AV-ARCH/JCTVC-SITE/,, no. JCTVC-T1014, 1 June 2015 (2015-06-01), XP030117420, Section 3.4 with sub-sections	1-45

1

(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 107211138 A
(43)申请公布日 2017.09.26

(21)申请号 201680006977.1

(22)申请日 2016.01.29

(30)优先权数据
62/110,302 2015.01.30 US
15/009,477 2016.01.28 US

(85)PCT国际申请进入国家阶段日
2017.07.24

(86)PCT国际申请的申请数据
PCT/US2016/015663 2016.01.29

(87)PCT国际申请的公布数据
WO2016/123488 EN 2016.08.04

(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 马尔塔·卡切维奇 濮伟
瑞珍·雷克斯曼·乔许
瓦迪姆·谢廖金

(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 杨林勳

(51)Int.Cl.
H04N 19/13(2014.01)
H04N 19/42(2014.01)
H04N 19/463(2014.01)
H04N 19/593(2014.01)
H04N 19/70(2014.01)
H04N 19/93(2014.01)

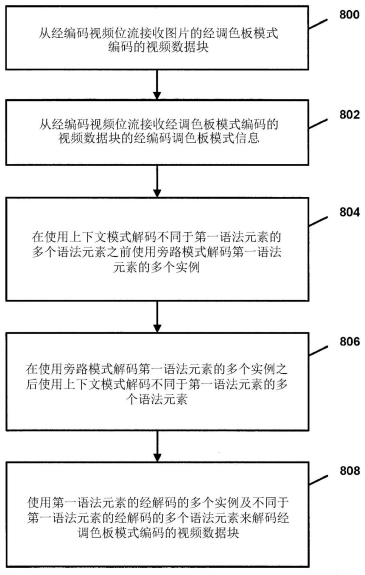
权利要求书6页 说明书40页 附图9页

(54)发明名称

用于高吞吐量CABAC译码的调色板索引分组

(57)摘要

在实例中,一种解码视频数据的方法可包含接收图片的经调色板模式编码的视频数据块。所述方法可包含接收用于所述经调色板模式编码的视频数据块的经编码调色板模式信息。所述经编码调色板模式信息可包含第一语法元素的多个实例及不同于所述第一语法元素的多个语法元素。所述方法可包含在使用上下文模式解码不同于所述第一语法元素的所述多个语法元素之前使用旁路模式解码所述第一语法元素的所述多个实例。所述方法可包含在使用旁路模式解码所述第一语法元素的所述多个实例之后使用上下文模式解码不同于所述第一语法元素的所述多个语法元素。



CN 107211138 A

1.一种解码视频数据的方法,所述方法包括:

从经编码视频位流接收图片的经调色板模式编码的视频数据块;

从所述经编码视频位流接收用于所述经调色板模式编码的视频数据块的经编码调色板模式信息,其中所述经编码调色板模式信息包含第一语法元素的多个实例及不同于所述第一语法元素的多个语法元素;

在使用上下文模式解码不同于所述第一语法元素的所述多个语法元素之前使用旁路模式解码所述第一语法元素的所述多个实例;

在使用旁路模式解码所述第一语法元素的所述多个实例之后使用上下文模式解码不同于所述第一语法元素的所述多个语法元素;及

使用所述第一语法元素的所述经解码的多个实例及不同于所述第一语法元素的所述经解码的多个语法元素来解码所述经调色板模式编码的视频数据块。

2.根据权利要求1所述的方法,其中所述第一语法元素的所述多个实例包含用于所述经调色板模式编码的视频数据块的所述第一语法元素的所有实例。

3.根据权利要求1所述的方法,其中所述第一语法元素为对调色板条目的阵列的索引的指示或指定用于对应于逸出样本的色彩分量的经量化的逸出经译码样本值,且其中不同于所述第一语法元素的所述多个语法元素包含指定表示游程长度的变量的二进制表示中的最高有效位的索引的语法元素及指定游程类型模式的语法元素。

4.根据权利要求1所述的方法,其中所述第一语法元素为palette_index_idc或palette_escape_val,且其中不同于所述第一语法元素的所述多个语法元素包含palette_run_msb_id_plus1语法元素。

5.根据权利要求1所述的方法,其中所述第一语法元素的所述多个实例被分组在一起,以使得当解码所述经调色板模式编码的视频数据块时在旁路模式与上下文模式之间的切换减少。

6.根据权利要求1所述的方法,其中所述经编码调色板模式信息包含指示所述第一语法元素的实例的数目的第二语法元素,其中不同于所述第一语法元素的所述多个语法元素不同于所述第二语法元素,且其中所述方法进一步包括:

在不同于所述第一语法元素及所述第二语法元素的所述多个语法元素的所述解码之前使用旁路模式解码所述第二语法元素。

7.根据权利要求6所述的方法,其中在用于所述经调色板模式编码的视频数据块的所述第一语法元素的任何两个实例之间无所述第二语法元素的实例交错。

8.根据权利要求6所述的方法,其进一步包括:

在解码等于由所述第二语法元素指示的所述数目的所述第一语法元素的实例的数目之后确定在所述经编码视频位流中的在所述第一语法元素的实例的所述数目之后的后续数据对应于不同于所述第一语法元素及所述第二语法元素的所述多个语法元素。

9.根据权利要求6所述的方法,其中所述经编码调色板模式信息包含第三语法元素及第四语法元素,其中所述方法进一步包括:

解码所述第三语法元素以确定对应于所述第三语法元素的值指示所述经调色板模式编码的视频数据块是否包含逸出样本;

解码所述第四语法元素以确定对应于所述第四语法元素的值指示调色板大小;及

在使用旁路模式解码所述第一语法元素及所述第二语法元素的所述多个实例之后基于分别对应于所述第三语法元素及所述第四语法元素的所述所确定的值使用上下文模式来解码不同于所述第一语法元素及所述第二语法元素的所述多个语法元素。

10.根据权利要求6所述的方法，其中所述经编码调色板模式信息包含第三语法元素，其中所述方法进一步包括：

解码所述第三语法元素以确定对应于所述第三语法元素的值，所述第三语法元素指定调色板索引针对所述经调色板模式编码的视频数据块所具有的不同值的数目；及

在使用旁路模式解码所述第一语法元素及所述第二语法元素的所述多个实例之后基于对应于所述第三语法元素的所述所确定的值使用上下文模式解码不同于所述第一语法元素及所述第二语法元素的所述多个语法元素。

11.根据权利要求6所述的方法，其中所述经编码调色板模式信息包含第三语法元素，其中所述方法进一步包括：

解码所述第三语法元素以确定对应于所述第三语法元素的值指示用于所述经调色板模式编码的视频数据块的palette_run_type_flag[xC][yC]的语法元素的最末实例。

12.根据权利要求6所述的方法，其进一步包括：

使用截断莱斯码与指数哥伦布码的级联来解码所述第二语法元素。

13.根据权利要求1所述的方法，其进一步包括：

确定所述经编码的视频数据块具有一或多个逸出样本；

解码所述经编码的视频数据块中的所述一或多个逸出样本中的最末逸出样本；

推断适用于所述经编码的视频数据块的在所述最末逸出样本之后的样本的索引值；及

使用在所述最末逸出样本之后的所述样本中的每一样本的所述所推断索引值来解码所述经编码的视频数据块的在所述最末逸出样本之后的所述样本。

14.根据权利要求6所述的方法，其进一步包括：

确定所接收的调色板索引的数目；

基于所接收的调色板索引的所述数目及所述第一语法元素的实例的所述数目确定剩余调色板索引的数目；及

基于所接收的调色板索引的所述数目及所述第一语法元素的实例的所述数目确定用于所述经编码的视频数据块的最大可能游程值。

15.根据权利要求14所述的方法，其进一步包括：

根据nCbS*nCbS-scanPos-1-paletteIndicesLeft确定用于所述经编码的视频数据块的所述最大可能游程值，其中nCbS指定所述经编码的视频数据块的大小，scanPos指定扫描位置，且paletteIndicesLeft指定剩余调色板索引的所述数目。

16.一种用于解码视频数据的装置，所述装置包括：

经配置以存储所述视频数据的存储器；及

与所述存储器通信的视频解码器，所述视频解码器经配置以：

从所述存储器接收图片的经调色板模式编码的视频数据块；

接收用于所述经调色板模式编码的视频数据块的经编码调色板模式信息，其中所述经编码调色板模式信息包含第一语法元素的多个实例及不同于所述第一语法元素的多个语法元素；

在使用上下文模式解码不同于所述第一语法元素的所述多个语法元素之前使用旁路模式解码所述第一语法元素的所述多个实例；

在使用旁路模式解码所述第一语法元素的所述多个实例之后使用上下文模式解码不同于所述第一语法元素的所述多个语法元素；及

使用所述第一语法元素的所述经解码的多个实例及不同于所述第一语法元素的所述经解码的多个语法元素来解码所述经调色板模式编码的视频数据块。

17.根据权利要求16所述的装置，其中所述第一语法元素的所述多个实例包含用于所述经调色板模式编码的视频数据块的所述第一语法元素的所有实例。

18.根据权利要求16所述的装置，其中所述第一语法元素为对调色板条目的阵列的索引的指示或指定用于对应于逸出样本的色彩分量的经量化的逸出经译码样本值，且其中不同于所述第一语法元素的所述多个语法元素包含指定表示游程长度的变量的二进制表示中的最高有效位的索引的语法元素及指定游程类型模式的语法元素。

19.根据权利要求16所述的装置，其中所述第一语法元素为palette_index_idc或palette_escape_val，且其中不同于所述第一语法元素的所述多个语法元素包含palette_run_msb_id_plus1语法元素。

20.根据权利要求16所述的装置，其中所述第一语法元素的所述多个实例被分组在一起，以使得当解码所述经调色板模式编码的视频数据块时在旁路模式与上下文模式之间的切换减少。

21.根据权利要求16所述的装置，其中所述经编码调色板模式信息包含指示所述第一语法元素的实例的数目的第二语法元素，其中不同于所述第一语法元素的所述多个语法元素不同于所述第二语法元素，且其中所述视频解码器经进一步配置以：

在不同于所述第一语法元素及所述第二语法元素的所述多个语法元素的所述解码之前使用旁路模式解码所述第二语法元素。

22.根据权利要求21所述的装置，其中在用于所述经调色板模式编码的视频数据块的所述第一语法元素的任何两个实例之间无所述第二语法元素的实例交错。

23.根据权利要求21所述的装置，其中所述视频解码器经进一步配置以在解码等于由所述第二语法元素指示的所述数目的所述第一语法元素的实例的数目之后确定在所述经编码视频位流中的在所述第一语法元素的实例的所述数目之后的后续数据对应于不同于所述第一语法元素及所述第二语法元素的所述多个语法元素。

24.根据权利要求21所述的装置，其中所述经编码调色板模式信息包含第三语法元素及第四语法元素，其中所述视频解码器经进一步配置以：

解码所述第三语法元素以确定对应于所述第三语法元素的值指示所述经调色板模式编码的视频数据块是否包含逸出样本；

解码所述第四语法元素以确定对应于所述第四语法元素的值指示调色板大小；及

在使用旁路模式解码所述第一语法元素及所述第二语法元素的所述多个实例之后基于分别对应于所述第三语法元素及所述第四语法元素的所述所确定的值使用上下文模式来解码不同于所述第一语法元素及所述第二语法元素的所述多个语法元素。

25.根据权利要求21所述的装置，其中所述经编码调色板模式信息包含第三语法元素，其中所述视频解码器经进一步配置以：

解码所述第三语法元素以确定对应于所述第三语法元素的值，所述第三语法元素指定调色板索引针对所述经调色板模式编码的视频数据块所具有的不同值的数目；及

在使用旁路模式解码所述第一语法元素及所述第二语法元素的所述多个实例之后基于对应于所述第三语法元素的所述所确定的值使用上下文模式解码不同于所述第一语法元素及所述第二语法元素的所述多个语法元素。

26.根据权利要求21所述的装置，其中所述经编码调色板模式信息包含第三语法元素，其中所述视频解码器经进一步配置以：

解码所述第三语法元素以确定对应于所述第三语法元素的值指示用于所述经调色板模式编码的视频数据块的palette_run_type_flag[xC][yC]的语法元素的最末实例。

27.根据权利要求21所述的装置，其中所述视频解码器经进一步配置以：

使用截断莱斯码与指数哥伦布码的串联来解码所述第二语法元素。

28.根据权利要求16所述的装置，其中所述视频解码器经进一步配置以：

确定所述经编码的视频数据块具有一或多个逸出样本；

解码所述经编码的视频数据块中的所述一或多个逸出样本中的最末逸出样本；

推断适用于所述经编码的视频数据块的在所述最末逸出样本之后的样本的索引值；及

使用在所述最末逸出样本之后的所述样本中的每一样本的所述所推断索引值来解码所述经编码的视频数据块的在所述最末逸出样本之后的所述样本。

29.根据权利要求21所述的装置，其中所述视频解码器经进一步配置以：

确定所接收的调色板索引的数目；

基于所接收的调色板索引的所述数目及所述第一语法元素的实例的所述数目确定剩余调色板索引的数目；及

基于所接收的调色板索引的所述数目及所述第一语法元素的实例的所述数目确定用于所述经编码的视频数据块的最大可能游程值。

30.根据权利要求29所述的装置，其中所述视频解码器经进一步配置以：

根据nCbS*nCbS-scanPos-1-paletteIndicesLeft确定用于所述经编码的视频数据块的所述最大可能游程值，其中nCbS指定所述经编码的视频数据块的大小，scanPos指定扫描位置，且paletteIndicesLeft指定剩余调色板索引的所述数目。

31.一种非暂时性计算机可读存储媒体，其上存储有指令，所述指令在执行时使得一或多个处理器：

从存储器接收图片的经调色板模式编码的视频数据块；

接收用于所述经调色板模式编码的视频数据块的经编码调色板模式信息，其中所述经编码调色板模式信息包含第一语法元素的多个实例及不同于所述第一语法元素的多个语法元素；

在使用上下文模式解码不同于所述第一语法元素的所述多个语法元素之前使用旁路模式解码所述第一语法元素的所述多个实例；

在使用旁路模式解码所述第一语法元素的所述多个实例之后使用上下文模式解码不同于所述第一语法元素的所述多个语法元素；及

使用所述第一语法元素的所述经解码的多个实例及不同于所述第一语法元素的所述经解码的多个语法元素来解码所述经调色板模式编码的视频数据块。

32.一种编码视频数据的方法,所述方法包括:

确定视频数据块将以调色板模式经译码;

使用调色板模式将所述视频数据块编码成经编码位流,其中使用调色板模式编码所述视频数据块包括:

产生用于所述视频数据块的调色板模式信息,其中所述调色板模式信息包含第一语法元素的多个实例及不同于所述第一语法元素的多个语法元素;

在使用上下文模式将不同于所述第一语法元素的所述多个语法元素编码成所述经编码位流之前使用旁路模式将所述第一语法元素的所述多个实例编码成所述经编码位流;及

在使用旁路模式将所述第一语法元素的所述多个实例编码成所述经编码位流之后使用上下文模式将不同于所述第一语法元素的所述多个语法元素编码成所述经编码位流。

33.根据权利要求32所述的方法,其中所述第一语法元素的所述多个实例包含用于所述视频数据块的所述第一语法元素的所有实例。

34.根据权利要求32所述的方法,其中所述第一语法元素为对调色板条目的阵列的索引的指示或指定用于对应于逸出样本的色彩分量的经量化的逸出经译码样本值,且其中不同于所述第一语法元素的所述多个语法元素包含指定表示游程长度的变量的二进制表示中的最高有效位的索引的语法元素及指定游程类型模式的语法元素。

35.根据权利要求32所述的方法,其中所述第一语法元素为palette_index_idc或palette_escape_val,且其中不同于所述第一语法元素的所述多个语法元素包含palette_run_msb_id_plus1语法元素。

36.根据权利要求32所述的方法,其中所述第一语法元素的所述多个实例被分组在一起,以使得当编码所述经调色板模式编码的视频数据块时在旁路模式与上下文模式之间的切换减少。

37.根据权利要求32所述的方法,其中所述调色板模式信息包含指示所述第一语法元素的实例的数目的第二语法元素,其中不同于所述第一语法元素的所述多个语法元素不同于所述第二语法元素,且其中所述方法进一步包括:

在不同于所述第一语法元素及所述第二语法元素的所述多个语法元素的所述编码之前使用旁路模式将所述第二语法元素编码成所述经编码位流。

38.根据权利要求37所述的方法,其中在用于所述视频数据块的所述第一语法元素的任何两个实例之间无所述第二语法元素的实例交错。

39.根据权利要求37所述的方法,其进一步包括:

在所述经编码位流中的所述第一语法元素的所述经编码的多个实例之后将所述第二语法元素编码成所述经编码位流。

40.根据权利要求37所述的方法,其中所述调色板模式信息包含第三语法元素及第四语法元素,其中所述方法进一步包括:

将对应于所述第三语法元素的指示所述视频数据块是否包含逸出样本的值编码成所述经编码位流;及

将对应于所述第四语法元素的指示调色板大小的值编码成所述经编码位流。

41.根据权利要求37所述的方法,其中所述调色板模式信息包含第三语法元素,其中所述方法进一步包括:

将对应于所述第三语法元素的值编码成所述经编码位流，所述第三语法元素指定调色板索引针对所述视频数据块所具有的不同值的数目。

42.根据权利要求37所述的方法，其中所述调色板模式信息包含第三语法元素，其中所述方法进一步包括：

编码对应于所述第三语法元素的指示用于所述视频数据块的palette_run_type_flag[xC][yC]的语法元素的最末实例的值。

43.根据权利要求37所述的方法，其进一步包括：

使用截断莱斯码与指数哥伦布码的级联来编码所述第二语法元素。

44.根据权利要求32的方法，其进一步包括：

编码所述视频数据块中的所述一或多个逸出样本中的最末逸出样本；

推断适用于所述视频数据块的在所述最末逸出样本之后的样本的索引值；及

使用在所述最末逸出样本之后的所述样本中的每一样本的所述所推断索引值来编码所述视频数据块的在所述最末逸出样本之后的所述样本。

45.一种用于编码视频数据的装置，所述装置包括：

经配置以存储所述视频数据的存储器；及

与所述存储器通信的视频编码器，所述视频编码器经配置以：

确定存储于所述存储器中的视频数据块将以调色板模式经编码；

使用调色板模式将所述视频数据块编码成经编码位流，其中经配置以使用调色板模式编码所述视频数据块的所述视频编码器包括经配置以进行以下各者的所述视频编码器：

产生用于所述视频数据块的调色板模式信息，其中所述调色板模式信息包含第一语法元素的多个实例及不同于所述第一语法元素的多个语法元素；

在使用上下文模式将不同于所述第一语法元素的所述多个语法元素编码成所述经编码位流之前使用旁路模式将所述第一语法元素的所述多个实例编码成所述经编码位流；及

在使用旁路模式将所述第一语法元素的所述多个实例编码成所述经编码位流之后使用上下文模式将不同于所述第一语法元素的所述多个语法元素编码成所述经编码位流。

用于高吞吐量CABAC译码的调色板索引分组

[0001] 本申请案主张2015年1月30日申请的美国临时专利申请案第62/110,302号的权益,所述申请案的全部内容特此以引用的方式并入本文中。

技术领域

[0002] 本发明涉及编码及解码内容,且更特定地说涉及根据基于调色板的译码模式来编码及解码内容。

背景技术

[0003] 数字视频能力可并入到广泛范围的装置中,包含数字电视、数字直播系统、无线广播系统、个人数字助理(PDA)、膝上或台式计算机、平板计算机、电子书阅读器、数码相机、数字记录装置、数字媒体播放器、视频游戏装置、视频游戏控制台、蜂窝式或卫星无线电电话、所谓的“智能电话”、视频电话会议装置、视频流式传输装置及其类似者。数字视频装置实施视频压缩技术,例如由MPEG-2、MPEG-4、ITU-T H.263、ITU-T H.264/MPEG-4、第10部分先进视频译码(AVC)、ITU-T H.265、高效率视频译码(HEVC)所界定的标准及这些标准的扩展中所描述的那些技术。视频装置通过实施这些视频压缩技术可较有效地发射、接收、编码、解码及/或存储数字视频信息。

[0004] 视频压缩技术执行空间(图片内)预测及/或时间(图片间)预测来减少或移除视频序列中固有的冗余。对于基于块的视频译码,可将视频切片(即,视频帧或视频帧的一部分)分割成视频块。使用关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经帧内译码(I)的切片中的视频块。图片的经帧间译码(P或B)切片中的视频块可使用关于同一图片中的相邻块中的参考样本的空间预测或关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] 空间或时间预测产生待译码的块的预测性块。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来编码帧间译码块,且残余数据指示经译码块与预测性块之间的差异。根据帧内译码模式及残余数据来编码帧内译码块。为了进一步压缩,可将残余数据从像素域变换到变换域,从而产生残余系数,接着可对残余系数进行量化。最初布置成二维阵列的经量化系数可经扫描以便产生系数的一维向量,且可应用熵译码以达成甚至更多压缩。

[0006] 可使用调色板模式编码并解码例如图像的内容。大体来说,调色板模式是涉及使用调色板表示内容的技术。内容可经编码以使得由包含对应于调色板的值的索引映射来表示内容。可解码索引映射以重建内容。

发明内容

[0007] 本发明的技术涉及基于调色板的内容译码。举例来说,在基于调色板的内容译码中,内容译码器(例如,例如视频编码器或视频解码器的内容译码器)可形成作为色彩表的“调色板”以用于表示特定区域(例如,给定块)的视频数据。基于调色板的内容译码可(例

如)尤其可用于具有相对较小数目的色彩的视频数据的译码区域。内容译码器可针对像素中的一或多者而译码使像素与表示像素的色彩的调色板中的条目相关的调色板索引(例如,索引值)而非译码实际像素值(或其残差)。本发明中描述的技术可包含用于信号传送基于调色板的译码模式、发射调色板、导出调色板、导出非发射语法元素的值、发射基于调色板的译码映射及其它语法元素、预测调色板条目、调色板索引的译码游程、熵译码调色板信息及各种其它调色板译码技术中的一或多者的各种组合的技术。

[0008] 在一个实例中,本发明描述解码视频数据的方法,其包括从经编码视频位流接收图片的经调色板模式编码的视频数据块;从经编码视频位流接收用于经调色板模式编码的视频数据块的经编码调色板模式信息,其中经编码调色板模式信息包含第一语法元素的多个实例及不同于第一语法元素的多个语法元素;在使用上下文模式解码不同于第一语法元素的多个语法元素之前使用旁路模式解码第一语法元素的多个实例;在使用旁路模式解码第一语法元素的多个实例之后使用上下文模式解码不同于第一语法元素的多个语法元素;及使用第一语法元素的经解码的多个实例及不同于第一语法元素的经解码的多个语法元素来解码经调色板模式编码的视频数据块。

[0009] 在另一实例中,本发明描述用于解码视频数据的装置,所述装置包括经配置以存储视频数据的存储器;及与存储器通信的视频解码器,所述视频解码器经配置以:从经编码视频位流接收图片的经调色板模式编码的视频数据块;从经编码视频位流接收用于经调色板模式编码的视频数据块的经编码调色板模式信息,其中经编码调色板模式信息包含第一语法元素的多个实例及不同于第一语法元素的多个语法元素;在使用上下文模式解码不同于第一语法元素的多个语法元素之前使用旁路模式解码第一语法元素的多个实例;在使用旁路模式解码第一语法元素的多个实例之后使用上下文模式解码不同于第一语法元素的多个语法元素;及使用第一语法元素的经解码的多个实例及不同于第一语法元素的经解码的多个语法元素来解码经调色板模式编码的视频数据块。

[0010] 在另一实例中,本发明描述其上存储有指令的非暂时性计算机可读存储媒体,所述指令在执行时使得一或多个处理器从经编码视频位流接收图片的经调色板模式编码的视频数据块;从经编码视频位流接收用于经调色板模式编码的视频数据块的经编码调色板模式信息,其中经编码调色板模式信息包含第一语法元素的多个实例及不同于第一语法元素的多个语法元素;在使用上下文模式解码不同于第一语法元素的多个语法元素之前使用旁路模式解码第一语法元素的多个实例;在使用旁路模式解码第一语法元素的多个实例之后使用上下文模式解码不同于第一语法元素的多个语法元素;及使用第一语法元素的经解码的多个实例及不同于第一语法元素的经解码的多个语法元素来解码经调色板模式编码的视频数据块。

[0011] 在另一实例中,本发明描述编码视频数据的方法,所述方法包括确定视频数据块将以调色板模式译码;使用调色板模式将视频数据块编码成经编码位流,其中使用调色板模式编码视频数据块包括:产生用于视频数据块的调色板模式信息,其中调色板模式信息包含第一语法元素的多个实例及不同于第一语法元素的多个语法元素;在使用上下文模式将不同于第一语法元素的多个语法元素编码成经编码位流之前使用旁路模式将第一语法元素的多个实例编码成经编码位流;及在使用旁路模式将第一语法元素的多个实例编码成经编码位流之后使用上下文模式将不同于第一语法元素的多个语法元素编码成经编码位

流。

[0012] 在另一实例中，本发明描述用于编码视频数据的装置，所述装置包括经配置以存储视频数据的存储器；及与所述存储器通信的视频编码器，所述视频编码器经配置以：确定存储于存储器中的视频数据的块将以调色板模式编码；使用调色板模式将视频数据块编码成经编码位流，其中经配置以使用调色板模式编码视频数据块的视频编码器包括经配置以进行以下各者的视频编码器：产生用于视频数据块的调色板模式信息，其中调色板模式信息包含第一语法元素的多个实例及不同于第一语法元素的多个语法元素；在使用上下文模式将不同于第一语法元素的多个语法元素编码成经编码位流之前使用旁路模式将第一语法元素的多个实例编码成经编码位流；及在使用旁路模式将第一语法元素的多个实例编码成经编码位流之后使用上下文模式将不同于第一语法元素的多个语法元素编码成经编码位流。

[0013] 下文在随附图式及描述中阐述本发明的一或多个实例的细节。本发明的其它特征、目标及优点将从描述及图式及权利要求书显而易见。

附图说明

[0014] 图1是说明可利用本发明中所描述的技术的实例视频译码系统的框图。

[0015] 图2是说明可执行本发明中所描述的技术的实例视频编码器的框图。

[0016] 图3是说明可执行本发明中所描述的技术的实例视频解码器的框图。

[0017] 图4是说明符合本发明的技术的确定用于基于调色板的视频译码的调色板条目的实例的概念图。

[0018] 图5是说明符合本发明的技术的确定至用于像素的块的调色板的索引的实例的概念图。

[0019] 图6是说明符合本发明的技术的确定最大上方复制游程长度、假设光栅扫描次序的实例的概念图。

[0020] 图7是说明用于调色板模式的语法元素的译码次序的变化的表。

[0021] 图8是说明符合本发明的用于基于调色板的视频译码的技术的用于解码视频数据的实例过程的流程图。

[0022] 图9是说明符合本发明的用于基于调色板的视频译码的技术的用于编码视频数据的实例过程的流程图。

具体实施方式

[0023] 本发明的方面是针对用于内容译码(例如，视频译码)的技术。特定地说，本发明描述用于内容数据(例如，视频数据)的基于调色板的译码的技术及用于调色板译码信息的基于上下文的自适应二进制算术译码(CABAC)的技术。在本发明的各种实例中，如下文更详细地描述，本发明的技术可针对以调色板模式预测或译码块以改良译码效率及/或降低编解码器复杂度的过程。举例来说，本发明描述关于调色板索引分组(例如先进的调色板索引分组)的技术。

[0024] 在(例如，如D.Marpe、H.Schwarz及T.Wiegand，IEEE Trans.Cir.&Sys.Video Tech.，“H.264/AVC视频压缩标准中的基于上下文的自适应二进制算术译码”编号2003年7

月7日，卷13中所描述的)CABAC过程中，存在两个模式：(1)旁路模式及(2)上下文模式。在旁路模式中，不存在上下文更新过程。因此，旁路模式可通过采用硬件或ISA级并行度实现比基于上下文的模式更高的数据吞吐量。旁路模式的这一益处随着可经一起处理的旁路二进制数的数目的增大而变得更大。

[0025] 在当前调色板模式译码设计中，如R.Joshi及J.Xu的“高效率视频译码(HEVC)屏幕内容译码：草案2”JCTVC-S1005中所描述的，在屏幕内容译码中，palette_index_idc及palette_escape_val的语法元素经CABAC旁路模式译码且与经CABAC上下文模式译码的其它语法元素(例如，palette_run_msb_id_plus1)交错。本发明描述将经旁路模式译码的语法元素分组在一起的技术。如本文所使用，“经旁路模式译码”及“经上下文模式译码”分别可与“经旁路译码”及“经上下文译码”互换。

[0026] 如本文所使用，术语“内容”的实例可改变成术语“视频”，且术语“视频”的实例可改变成术语“内容”。无论术语“内容”或“视频”是被用作形容词、名词还是词类的其它部分，情况都如此。举例来说，对“内容译码器”的参考还包含对“视频译码器”的参考，且对“视频译码器”的参考还包含对“内容译码器”的参考。类似地，对“内容”的参考还包含对“视频”的参考，且对“视频”的参考还包含对“内容”的参考。

[0027] 如本文所使用，“内容”指代任何类型的内容。举例来说，“内容”可指代视频、屏幕内容、图像、任何图形内容、任何可显示内容或与其相对应的任何数据(例如，视频数据、屏幕内容数据、图像数据、图形内容数据、可显示内容数据及其类似者)。

[0028] 如本文所使用，术语“视频”可指代屏幕内容、可移动内容、可以序列呈现的多个图像或与其相对应的任何数据(例如，屏幕内容数据、可移动内容数据、视频数据、图像数据及其类似者)。

[0029] 如本文所使用，术语“图像”可指代单个图像、一或多个图像、对应于视频的多个图像中的一或多个图像、不对应于视频的多个图像中的一或多个图像、对应于视频的多个图像(例如，对应于视频的全部图像或对应于视频的不到全部图像)、单个图像的子部分、单个图像的多个子部分、对应于多个图像的多个子部分、一或多个图形基元、图像数据、图形数据及其类似者。

[0030] 在传统视频译码中，假设图像为连续色调且在空间上平滑。基于这些假设，已开发例如基于块的变换、滤波及其它译码工具的各种工具，且这些工具已针对天然内容视频展示出良好效能。然而，在类似远程台式计算机、协同工作及无线显示器的应用中，计算机产生的屏幕内容可为待压缩的主要内容。这一类型的屏幕内容往往会具有离散色调、陡线及高对比度物件边界。可不再应用连续色调及平滑度的假设，且因此传统视频译码技术可在压缩内容(例如，屏幕内容)方面效率低下。

[0031] 在基于调色板的视频译码的一个实例中，视频编码器可通过确定用于块的调色板(例如，明确地译码调色板、预测调色板或其组合)，定位调色板中的条目以表示一或多个像素的值，及使用指示用于表示块的像素值的调色板中的条目的索引值来编码调色板及块两者而编码视频数据块。在一些实例中，视频编码器可在经编码位流中用信号发送调色板及/或索引值。继而，视频解码器可从经编码位流获得用于块的调色板，以及用于所述块的个别像素的索引值。视频解码器可使像素的索引值与调色板的条目相关以重建块的各种像素值。

[0032] 举例来说，可假设视频数据的特定区域具有相对较小数目的色彩。视频译码器(例如，视频编码器或视频解码器)可译码(例如，编码或解码)所谓的“调色板”以表示特定区域的视频数据。调色板可表达为表示特定区域(例如，给定块)的视频数据的色彩或像素值的索引(例如，表)。视频译码器可译码索引，所述索引使一或多个像素值与调色板中的适当值相关。每一像素可与调色板中表示像素的色彩的条目相关联。举例来说，调色板可包含给定块中的最主要像素值。在一些情况下，最主要像素值可包含在所述块内最频繁地出现的一或多个像素值。另外，在一些情况下，视频译码器可应用临限值以确定是否应将像素值包含作为所述块中的最主要像素值中的一者。根据基于调色板的译码的各种方面，视频译码器可对指示当前块的像素值中的一或多者的索引值进行译码，而不是针对视频数据的当前块对实际像素值或其残余部分进行译码。在基于调色板的译码的情况下，索引值指示调色板中被用于表示当前块的个别像素值的对应条目。以上描述意欲提供基于调色板的视频译码的概述。

[0033] 基于调色板的译码可特别适合于屏幕产生的内容译码或其中一或多个传统译码工具效率低下的其它内容。用于视频数据的基于调色板译码的技术可与一或多个其它译码技术(例如用于帧间或帧内预测性译码的技术)一起使用。举例来说，如下文更详细地描述，编码器或解码器或组合式编码器解码器(编解码器)可经配置以执行帧间及帧内预测性译码，以及基于调色板的译码。

[0034] 在一些实例中，基于调色板的译码技术可经配置以与一或多个视频译码标准一起使用。举例来说，高效率视频译码(HEVC)为由ITU-T视频译码专家组(VCEG)及ISO/IEC运动图片专家组(MPEG)的视频译码联合合作小组(JCT-VC)开发的新的视频译码标准。定案的HEVC标准文件于2013年4月由国际电信联盟(ITU)的电信标准化部门公开为“ITU-T H.265，系列H:视听服务的视听及多媒体系统基础设施——运动视频的译码——高效率视频译码(SERIES H:AUDIOVISUAL AND MULTIMEDIA SYSTEMS Infrastructure of audiovisual services-Coding of moving video-High efficiency video coding)”。

[0035] 为了提供屏幕产生内容的更有效译码，JCT-VC将开发延伸到HEVC标准(被称作HEVC屏幕内容译码(SCC)标准)。被称作“HEVC SCC草案2”或“WD2”的HEVC SCC标准的新工作草案描述于R.Joshi及J.Xu的文件JCTVC-S1005“HEVC屏幕内容译码草案文字2”(ITU-T SG 16WP 3及ISO/IEC JTC 1/SC 29/WG 11的关于视频译码的联合合作小组(JCT-VC)，第19次会议:法国，斯特拉斯堡，2014年10月17日到24日)。

[0036] 就HEVC框架来说，作为实例，基于调色板的译码技术可经配置以用作译码单元(CU)模式。在其它实例中，基于调色板的译码技术可经配置以用作HEVC的框架中的预测单元(PU)模式。因此，在CU模式的上下文中描述的所有以下所公开的过程可另外或替代地适用于PU。然而，这些基于HEVC的实例不应被视为约束或限制本文中所描述的基于调色板的译码技术，因而，这些技术可经应用以独立地或作为其它现有或尚待开发的系统/标准的部分工作。在这些情况下，用于调色板译码的单元可为方形块、矩形块或甚至非矩形形状的区域。

[0037] 在一些实例中，调色板可由一或多个CU、PU或数据的任何区域(例如，数据的任何块)导出。举例来说，调色板可包括(及可由以下者组成):当前CU中的最主要像素值，其中针对这一特定实例，CU为数据的区域。将调色板的大小及元件首先从视频编码器发射到视频

解码器。可使用相邻CU(例如,上方及/或左方的经译码CU)中的调色板的大小及/或元件来直接译码或预测性地译码调色板的大小及/或元件。此后,根据特定扫描次序基于调色板来编码CU中的像素值。对于CU中的每一像素位置,旗标(例如,palette_flag或escape_flag)可首先被发射以指示像素值是否包含于调色板中。针对映射到调色板中的条目的那些像素值,针对CU中的给定像素位置用信号发送与那一条目相关联的调色板索引。对于调色板中并不存在的那些像素值,特殊索引可分配给像素且实际像素值(可能以经量化形式)可经发射以用于CU中的给定像素位置,而非发送旗标(例如,palette_flag或escape_flag)。这些像素被称作"逸出像素"。可使用任何现有熵译码方法(例如,固定长度译码,一元译码等)来译码逸出像素。在一些实例中,本文所描述的一或多个技术可利用例如palette_flag或escape_flag的旗标。在其它实例中,本文所描述的一或多种技术可不利用例如palette_flag或escape_flag的旗标。

[0038] 视频数据块中的样本可使用水平光栅扫描次序或其它扫描次序进行处理(例如,扫描)。举例来说,视频编码器可通过使用水平光栅扫描次序扫描调色板索引而将调色板索引的二维块转换成一维阵列。同样,视频解码器可使用水平光栅扫描次序重建调色板索引的块。因此,本发明可将先前样本称作按扫描次序在块中当前经译码的样本之前的样本。应了解,除水平光栅扫描以外的扫描(例如,竖直光栅扫描次序)也可为可适用的。以上实例以及本发明中阐述的其它实例意图提供对基于调色板的视频译码的概述。

[0039] 图1为说明可利用本发明的技术的实例视频译码系统10的框图。如本文所使用,术语"视频译码器"一般是指视频编码器及视频解码器两者。在本发明中,术语"视频译码"或"译码"一般可指视频编码或视频解码。视频译码系统10的视频编码器20及视频解码器30表示可经配置以执行根据本发明中描述的各种实例的用于基于调色板的视频译码及熵译码(例如,CABAC)的装置的实例。举例来说,视频编码器20及视频解码器30可经配置以使用基于调色板的译码或非基于调色板的译码选择地译码视频数据的各种块,例如HEVC译码中的CU或PU。非基于调色板的译码模式可指代各种帧间预测性时间译码模式或帧内预测性空间译码模式,例如由HEVC标准指定的各种译码模式。

[0040] 如图1中所展示,视频译码系统10包含源装置12及目的地装置14。源装置12产生经编码视频数据。因此,源装置12可被称作视频编码装置或视频编码设备。目的地装置14可解码由源装置12所产生的经编码视频数据。因此,目的地装置14可被称作视频解码装置或视频解码设备。源装置12及目的地装置14可为视频译码装置或视频译码设备的实例。

[0041] 源装置12及目的地装置14可包括广泛范围的装置,包含台式计算机、移动计算装置、笔记型(例如,膝上)计算机、平板计算机、机顶盒、例如所谓的"智能"电话的手持电话、电视、相机、显示装置、数字媒体播放器、视频游戏控制台、车载计算机(in-carcomputer)或其类似者。

[0042] 目的地装置14可经由信道16从源装置12接收经编码视频数据。信道16可包括能够将经编码视频数据从源装置12移动到目的地装置14的一或多个媒体或装置。在一个实例中,信道16可包括使源装置12能够实时地将经编码视频数据直接发射到目的地装置14的一或多个通信媒体。在此实例中,源装置12可根据例如无线通信协议的通信标准来调制经编码视频数据,且可将经调制视频数据发射到目的地装置14。一或多个通信媒体可包含无线及/或有线通信媒体,例如射频(RF)频谱或一或多个物理发射线。一或多个通信媒体可形成

基于分组的网络(例如局域网、广域网或全球网络(例如,因特网)的部分。一或多个通信媒体可包含路由器、交换器、基站,或促进从源装置12到目的地装置14的通信的其它设备。

[0043] 在另一实例中,信道16可包含存储由源装置12所产生的经编码视频数据的存储媒体。在此实例中,目的地装置14可(例如)经由磁盘存取或卡存取而存取存储媒体。存储媒体可包含多种本地存取的数据存储媒体,例如蓝光光盘、DVD、CD-ROM、闪速存储器,或用于存储经编码视频数据的其它合适的数字存储媒体。

[0044] 在又一实例中,信道16可包含存储由源装置12产生的经编码视频数据的文件服务器或另一中间存储装置。在此实例中,目的地装置14可经由流式传输或下载而存取存储于文件服务器或其它中间存储装置处的经编码视频数据。文件服务器可为能够存储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的类型的服务器。实例文件服务器包含网页服务器(例如,用于网站)、文件传送协议(FTP)服务器、网络附接存储(NAS)装置及本地磁盘驱动器。

[0045] 目的地装置14可经由标准数据连接(例如,因特网连接)来存取经编码视频数据。数据连接的实例类型可包含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、电缆调制解调器等),或两者的组合。经编码视频数据从文件服务器的发射可为流式传输发射、下载发射或两者的组合。

[0046] 源装置12及目的地装置14可经配置以符合本发明执行基于调色板的译码及熵译码(例如,CABAC)。然而,本发明用于基于调色板的译码或CABAC的技术不限于无线应用或设定。所述技术可适用于(例如)经由因特网支持多种多媒体应用(例如空中电视广播、有线电视发射、卫星电视发射、流式传输视频发射)的视频译码、用于存储于数据存储媒体上的视频数据的编码、存储于数据存储媒体上的视频数据的解码,或其它应用。在一些实例中,视频译码系统10可经配置以支持单向或双向视频发射从而支持例如视频流式传输、视频播放、视频广播及/或视频电话的应用。

[0047] 图1中所示的视频译码系统10仅为实例,且本发明的技术可适用于未必包含编码装置与解码装置之间的任何数据通信的视频译码设定(例如,视频编码或视频解码)。在其它实例中,从经由网络流式传输的本地存储器或类似者检索数据。视频编码装置可编码数据及将数据存储到存储器,及/或视频解码装置可从存储器检索数据及解码数据。在许多实例中,由彼此不通信但仅将数据编码到存储器及/或从存储器检索且解码数据的装置来执行编码及解码。

[0048] 在图1的实例中,源装置12包含视频源18、视频编码器20及输出接口22。在一些实例中,输出接口22可包含调制器/解调器(调制解调器)及/或发射器。视频源18可包含例如摄像机的视频捕捉装置、含有先前所捕捉的视频数据的视频档案库、用以从视频内容提供者处接收视频数据的视频馈入接口及/或用于产生视频数据的计算机图形系统,或视频数据的这些来源的组合。

[0049] 视频编码器20可编码来自视频源18的视频数据。在一些实例中,源装置12经由输出接口22将经编码视频数据直接发射到目的地装置14。在其它实例中,经编码视频数据还可存储于存储媒体或文件服务器上,以供目的地装置14稍后存取以用于解码及/或播放。

[0050] 在图1的实例中,目的地装置14包含输入接口28、视频解码器30及显示装置32。在一些实例中,输入接口28包含接收器及/或调制解调器。输入接口28可经由信道16接收经编

码视频数据。显示装置32可与目的地装置14集成或在目的地装置14外部。一般来说，显示装置32显示经解码视频数据。显示装置32可包括各种显示装置，例如液晶显示器(LCD)、等离子显示器、有机发光二极管(OLED)显示器，或另一类型的显示装置。

[0051] 本发明通常可指代视频编码器20将某些信息“用信号发送”或“发射”到另一装置，例如，视频解码器30。术语“用信号发送”或“发射”可大体上指代用于解码经压缩视频数据的语法元素及/或其它数据的通信。此通信可实时地或近实时地发生。替代地，可历时一时间跨度而发生此通信，例如此通信可在编码时间处将语法元素以经编码位流存储到计算机可读存储媒体时发生，所述语法元素随后可由解码装置在存储于此媒体之后的任何时间进行检索。因此，虽然视频解码器30可被称作“接收”某些信息，但信息的接收未必实时或接近实时发生且可在存储之后在某一时间从媒体检索。

[0052] 视频编码器20及视频解码器30各自可实施为多种合适电路中的任一者，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、硬件或其任何组合。如果部分地以软件来实施技术，那么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存储媒体中，且可使用一或多个处理器在硬件中执行所述指令以执行本发明的技术。可将上述内容(包含硬件、软件、硬件与软件的组合等)中的任一者视为一或多个处理器。视频编码器20及视频解码器30中的每一者可包含于一或多个编码器或解码器中，编码器或解码器中的任一者可在对应装置中集成为组合式编码器/解码器(编解码器(CODEC))的部分。

[0053] 在一些实例中，视频编码器20及视频解码器30根据例如上文所提及的HEVC标准且在HEVC标准中描述的视频压缩标准操作。除了基本HEVC标准之外，正持续努力产生用于HEVC的可伸缩视频译码、多视图视频译码及3D译码扩展。另外，可提供基于调色板的译码模式(例如，如本发明中所描述)以用于扩展HEVC标准。在一些实例中，本发明中针对基于调色板的译码而描述的技术可适用于经配置以根据其它视频译码标准操作的编码器及解码器。因此，出于实例的目的而描述用于HEVC编解码器中的译码单元(CU)或预测单元(PU)的译码的基于调色板的译码模式的应用。

[0054] 在HEVC及其它视频译码标准中，视频序列通常包括一系列图片。图片还可被称作“帧”。图片可包括三个样本阵列，标示为S_L、S_{Cb}及S_{Cr}。S_L为明度样本的二维阵列(即，块)。S_{Cb}为Cb彩度样本的二维阵列。S_{Cr}为Cr彩度样本的二维阵列。彩度样本在本文中还可被称作“色度”样本。在其它情况下，图片可为单色的，且可仅包括明度样本阵列。

[0055] 为产生图片的经编码表示，视频编码器20可产生译码树型单元(CTU)的集合。CTU中的每一者可为明度样本的译码树型块、色度样本的两个对应译码树型块，及用以对所述译码树型块的样本进行译码的语法结构。译码树型块可为样本的N×N块。CTU还可被称作“树型块”或“最大译码单元”(LCU)。HEVC的CTU可广泛地类似于例如H.264/AVC的其它标准的宏块。然而，CTU未必限于特定大小且可包含一或多个译码单元(CU)。切片可包含在光栅扫描中连续排序的整数数目个CTU。经译码切片可包括切片标头及切片数据。切片的切片标头可为包含提供关于切片的信息的语法元素的语法结构。切片数据可包含切片的经译码CTU。

[0056] 本发明可使用术语“视频单元”或“视频块”或“块”以指代一或多个样本块及用于对样本的所述一或多个块的样本进行译码的语法结构。视频单元或块的实例类型可包含

CTU、CU、PU、变换单元(TU)、宏块、宏块分区等等。在一些情形中，PU的论述可与宏块或宏块分区的论述互换。

[0057] 为产生经译码CTU，视频编码器20可对CTU的译码树型块递归地执行四分树分割，以将译码树型块划分成译码块，因此命名为“译码树型单元”。译码块为样本的N×N块。CU可为图片的明度样本的译码块及色度样本的两个对应译码块，所述图片具有明度样本阵列、Cb样本阵列及Cr样本阵列，以及用以译码所述译码块的样本的语法结构。视频编码器20可将CU的译码块分割成一或多个预测块。预测块可为应用相同预测的样本的矩形(即，正方形或非正方形)块。CU的预测单元(PU)可为图片的明度样本的预测块，图片的色度样本的两个对应预测块，及用以对预测块样本进行预测的语法结构。视频编码器20可针对CU的每一PU的明度预测块、Cb预测块及Cr预测块产生预测性明度块、预测性Cb块及预测性Cr块。

[0058] 视频编码器20可使用帧内预测或帧间预测，以产生PU的预测性块。如果视频编码器20使用帧内预测产生PU的预测性块，那么视频编码器20可基于与PU相关联的图片的经解码样本而产生PU的预测性块。

[0059] 如果视频编码器20使用帧间预测以产生PU的预测性块，那么视频编码器20可基于除与PU相关联的图片外的一或多个图片的经解码样本来产生PU的预测性块。视频编码器20可使用单向预测或双向预测以产生PU的预测性块。当视频编码器20使用单向预测来产生PU的预测性块时，PU可具有单一运动向量(MV)。当视频编码器20使用双向预测来产生PU的预测性块时，PU可具有两个MV。

[0060] 在视频编码器20产生CU的一或多个PU的预测性块(例如，预测性明度块、预测性Cb块及预测性Cr块)之后，视频编码器20可产生CU的残余块。CU的残余块中的每一样本可指示CU的PU的预测性块中的样本与CU的译码块中的对应样本之间的差异。举例来说，视频编码器20可产生CU的明度残余块。CU的明度残余块中的每一样本指示CU的预测性明度块中的一者中的明度样本与CU的原始明度译码块中的对应样本之间的差异。另外，视频编码器20可产生用于CU的Cb残余块。CU的Cb残余块中的每一样本可指示CU的预测性Cb块中的一者中的Cb样本与CU的原始Cb译码块中的对应样本之间的差异。视频编码器20还可产生用于CU的Cr残余块。CU的Cr残余块中的每一样本可指示CU的预测性Cr块中的一者中的Cr样本与CU的原始Cr译码块中的对应样本之间的差异。

[0061] 此外，视频编码器20可使用四分树分割将CU的残余块(例如，明度残余块、Cb残余块及Cr残余块)分解成一或多个变换块(例如，明度变换块、Cb变换块及Cr变换块)。变换块可为应用相同变换的样本的矩形块。CU的变换单元(TU)可为明度样本的变换块、色度样本的两个对应变换块，及用以对变换块样本进行变换的语法结构。因此，CU的每一TU可与明度变换块、Cb变换块及Cr变换块相关联。与TU相关联的明度变换块可为CU的明度残余块的子块。Cb变换块可为CU的Cb残余块的子块。Cr变换块可为CU的Cr残余块的子块。

[0062] 视频编码器20可将一或多个变换应用于变换块以产生TU的系数块。系数块可为变换系数的二维阵列。变换系数可为纯量。举例来说，视频编码器20可将一或多个变换应用于TU的明度变换块以产生TU的明度系数块。视频编码器20可将一或多个变换应用于TU的Cb变换块以产生TU的Cb系数块。视频编码器20可将一或多个变换应用于TU的Cr变换块以产生TU的Cr系数块。

[0063] 在产生系数块(例如，明度系数块、Cb系数块或Cr系数块)之后，视频编码器20可量

化所述系数块。量化通常指代对变换系数进行量化以可能减少用以表示变换系数的数据的量，从而提供进一步压缩的过程。在视频编码器20量化系数块之后，视频编码器20可熵编码指示经量化的变换系数的语法元素。举例来说，视频编码器20可对指示经量化变换系数的语法元素执行上下文自适应二进制算术译码(CABAC)。

[0064] 就CABAC来说，作为实例，视频编码器20及视频解码器30可选择概率模型(也被称作上下文模型)以基于上下文对与视频数据的块相关联的符号进行译码。举例来说，上下文模型(Ctx)可为应用于选择多个不同上下文中的一者的索引或差量，所述上下文中的每一者可对应于特定概率模型。因此，不同概率模型通常针对每一上下文来定义。在编码或解码二进制之后，概率模型基于二进制的值经进一步更新以反映对二进制的最新概率评估。举例来说，概率模型可作为有限状态机中的状态得以维持。每一特定状态可对应于特定可能性值。对应于概率模型的更新的下一状态可取决于当前二进制(例如，当前经译码的二进制)的值。因此，概率模型的选择可受先前经译码二进制的值的影响，因为所述值至少部分指示二进制具有给定值的概率。上文所描述的上下文译码过程可通常被称作上下文自适应译码模式。

[0065] 因此，视频编码器20可使用概率模型对目标符号进行编码。同样地，视频解码器30可使用概率模型剖析目标符号。在一些情况下，视频编码器20可使用上下文自适应译码与非上下文自适应译码的组合来译码语法元素。举例来说，视频编码器20可通过选择对上下文操作以对一些二进制进行译码的概率模型或“上下文模型”来二进制进行上下文译码。对比来说，对于其它二进制，视频编码器20可通过当译码二进制时绕过或省略常规算术译码过程而对二进制进行旁路译码。在这些实例中，视频编码器20可使用固定概率模型来对二进制进行旁路译码。也就是说，经旁路译码的二进制不包含上下文或概率更新。

[0066] 视频编码器20可输出包括经熵编码的语法元素的位流。位流还可包含未经熵编码的语法元素。所述位流可包含形成经译码图片及相关联数据的表示的位的序列。位流可包括网络抽象层(NAL)单元的序列。NAL单元中的每一者包含NAL单元标头，且封装原始字节序列有效负载(RBSP)。NAL单元标头可包含指示NAL单元类型码的语法元素。通过NAL单元的NAL单元标头指定的NAL单元类型码指示NAL单元的类型。RBSP可为含有封装于NAL单元内的整数数目个字节的语法结构。在一些情况下，RBSP包含零位。

[0067] 不同类型的NAL单元可封装不同类型的RBSP。举例来说，第一类型的NAL单元可封装图片参数集(PPS)的RBSP，第二类型的NAL单元可封装经译码切片的RBSP，第三类型的NAL单元可封装补充增强信息(SEI)的RBSP，等等。封装视频译码数据的RBSP(如与参数集及SEI讯息的RBSP相对)的NAL单元可被称作视频译码层(VCL)NAL单元。

[0068] 视频解码器30可接收由视频编码器20产生的位流。此外，视频解码器30可剖析位流以从位流解码语法元素。视频解码器30可至少部分基于从位流解码的语法元素而重建视频数据的图片。重建视频数据的过程可大体上与由视频编码器20执行的过程互逆。举例来说，视频解码器30可使用PU的MV来确定当前CU的经帧间预测PU的预测性块。同样地，视频解码器30可产生当前CU的PU的经帧内预测块。另外，视频解码器30可对与当前CU的TU相关联的变换系数块进行反量化。视频解码器30可对变换系数块执行反变换，以重建与当前CU的TU相关联的变换块。视频解码器30可通过将当前CU的PU的预测性块的样本添加到从当前CU的TU的变换块的反量化及反变换所获得的对应残余值来重建当前CU的译码块。通过重建图

片的每一CU的译码块，视频解码器30可重建图片。

[0069] 在一些实例中，视频编码器20及视频解码器30可经配置以执行基于调色板的译码。举例来说，在基于调色板的译码中，视频编码器20及视频解码器30可将所谓的调色板译码为表示特定区域(例如，给定块)的视频数据的色彩或像素值的表，而不是执行上文所描述的帧内预测或帧间预测译码技术。以此方式，视频译码器可对当前块的像素值中的一或多者的索引值进行译码，而不是对视频数据的当前块的实际像素值或其残差进行译码，其中所述索引值指示调色板中用于表示当前块的像素值的条目。

[0070] 举例来说，视频编码器20可通过确定用于块的调色板、定位调色板中的条目以表示每一像素的值及对调色板及像素的使像素值与调色板相关的索引值进行编码而对视频数据的块进行编码。视频解码器30可从经编码位流获得用于块的调色板，以及块的像素的索引值。视频解码器30可将个别像素的索引值与调色板的条目匹配以重建块的像素值。在与个别像素相关联的索引值不匹配块的对应调色板的任何索引值的情况下，出于基于调色板的译码的目的，视频解码器30可将这一像素识别为逸出像素。

[0071] 如下文更详细地描述，基于调色板的译码的基本构想为：对于待译码的视频数据的给定块，视频编码器20可导出包含当前块中的最主要像素值的调色板。举例来说，调色板可指经确定或假设为当前CU的主要及/或代表的数个像素值。视频编码器20可首先将调色板的大小和元件发射到视频解码器30。另外，视频编码器20可根据特定扫描次序对给定块中的像素值进行编码。对于包含于给定块中的每一像素，视频编码器20可用信号发送将像素值映射到调色板中的对应条目的索引值。如果像素值并不包含于调色板中(即，不存在指定经调色板译码的块的特定像素值的调色板条目)，那么所述像素被定义为“逸出像素”。根据基于调色板的译码，视频编码器20可编码且用信号发送经保留以用于逸出像素的索引值。在一些实例中，视频编码器20还可编码及用信号发送包含于给定块中的逸出像素的像素值(或其经量化版本)。举例来说，视频解码器30可经配置以基于失真度量(例如，MSE，SAD及其类似者)确定像素值是否匹配或以其它方式接近调色板条目。

[0072] 一旦接收到由视频编码器20用信号发送的经编码视频位流，视频解码器30可首先基于从视频编码器20接收的信息确定调色板。视频解码器30可接着将与给定块中的像素位置相关联的所接收到的索引值映射到调色板的条目，以重建给定块的像素值。在一些情况下，视频解码器30可确定经调色板译码块的像素为逸出像素，例如，通过确定像素由经保留以用于逸出像素的索引值而经调色板译码。在视频解码器30识别经调色板译码块中的逸出像素的情况下，视频解码器30可接收包含于给定块中的逸出像素的像素值(或其经量化版本)。视频解码器30可通过将个别像素值映射到对应调色板条目且通过使用像素值(或其经量化版本)而重建经调色板译码块，以重建包含于经调色板译码块中的任何逸出像素。

[0073] 如上文所陈述，在实例调色板译码模式中，调色板可包含由索引编号的条目。每一条目可表示色彩分量值或强度(例如，在例如YCbCr、RGB、YUV、CMYK或其它格式的色彩空间中)，所述值或强度可用作块的预测符或经最终重建的块样本。如标准提交文件JCTVC-Q0094(Wei Pu等人，“AHG10:用于基于REext6.0的调色板译码的建议软件(Suggested Software for Palette Coding based on REext6.0)”JCTVC-Q0094，西班牙巴伦西亚市，2014年3月27日到2014年4月4日)中所描述，调色板可包含从预测符调色板复制的条目。预测符调色板可包含来自先前使用调色板模式译码的块或来自其它经重建样本的调色板条

目。对于预测符调色板中的每一条目，发送二进制旗标以指示条目是否被复制到当前调色板(由旗标＝1指示)。此被称作二进制调色板预测向量。另外，当前调色板可包括经明确用信号发送的新条目(例如，由经明确用信号发送的新条目组成)。还可用信号发送新条目的数目。

[0074] 作为另一实例，在调色板模式中，调色板可包含由表示色彩分量值的索引编号的条目，所述色彩分量值可被用作块样本的预测符或经最终重建的块样本。调色板中的每一条目可含有(例如)一个明度分量(例如，明度值)、两个色度分量(例如，两个色度值)或三个色彩分量(例如，RGB、YUV等)。先前经解码的调色板条目可存储于列表中。举例来说，此列表可用于预测在当前调色板模式CU中的调色板条目。二进制预测向量可以位流经用信号发送以指示列表中的哪些条目再用于当前调色板中。在一些实例中，游程长度译码可用于压缩二进制调色板预测符。举例来说，可使用0阶指数哥伦布码(Exp-Golomb code)来对游程长度值进行译码。

[0075] 在本发明中，将假设每一调色板条目指定样本的所有色彩分量的值。然而，本发明的概念适用于使用每一色彩分量的独立调色板及/或独立调色板条目。并且，假设使用水平光栅扫描次序处理块中的样本。然而，例如竖直光栅扫描次序的其它扫描也可适用。如上文所提及，调色板可含有经预测调色板条目((例如)从用于对前述块进行译码的调色板预测的)及对当前块具专一性且明确地经用信号发送的新条目。编码器及解码器可知晓经预测及新调色板条目的数目且其总和可指示块中的总调色板大小。

[0076] 如上文所引用的JCTVC-Q0094的实例中所提出的，使用调色板译码的块中的每一样本可属于三个模式中的一者，如下文所阐述：

[0077] ●逸出模式。在此模式中，样本值并未作为调色板条目包含到调色板中，且对于所有色彩分量，明确地用信号发送经量化的样本值。所述情形类似于新调色板条目的用信号发送，尽管对于新调色板条目，并不将色彩分量值量化。

[0078] ●CopyAbove模式(还被称作CopyFromTop模式)。在此模式中，从位于样本的块中的当前样本正上方的样本复制当前样本的调色板条目索引。在其它实例中，对于上方复制模式，视频数据块可经转置以使得块上方的样本实际上为块左边的样本。

[0079] ●值模式(还被称作索引模式)。在此模式中，明确地用信号发送调色板条目索引的值。

[0080] 如本文所描述，调色板条目索引可被称作调色板索引或简称为索引。这些术语可互换地使用以描述本发明的技术。另外，如下文更详细地描述，调色板索引可具有一或多个相关联的色彩或强度值。举例来说，调色板索引可具有与像素的单一色彩或强度分量(例如，RGB数据的红色分量、YUV数据的Y分量，或其类似者)相关联的单一相关联的色彩或强度值。在另一实例中，调色板索引可具有多个相关联的色彩或强度值。在一些情况下，可应用基于调色板的视频译码来对单色视频进行译码。因此，“色彩值”大体上可指用以产生像素值的任何色彩或非彩色分量。

[0081] 游程值可指示使用相同的调色板译码模式译码的调色板索引的游程。举例来说，关于值模式，视频译码器(例如，视频编码器20或视频解码器30)可对索引值及指示扫描次序中具有相同索引值且使用调色板索引进行译码的多个连续后续样本的游程值进行译码。关于CopyAbove模式，视频译码器可对当前样本值的索引值与上方相邻样本的索引值相同

(例如,定位于所述样本上方的样本当前在块中经译码)的指示及指示扫描次序中也从上方相邻样本复制索引值的多个连续后续样本的游程值进行译码。因此,在上述实例中,调色板索引值的游程是指具有相同值的调色板值的游程或从上方相邻样本复制的索引值的游程。

[0082] 因此,游程可针对给定模式指定属于相同模式的后续样本的数目。在一些情况下,用信号发送索引值及游程值可与游程长度译码相似。在出于说明的目的的实例中,对应于视频数据块的索引块的连续调色板索引值的字串可为0,2,2,2,2,5。每一索引值对应于视频数据块中的样本。在此实例中,视频译码器可使用值模式对第二样本(例如,“2”的第一调色板索引值)进行译码。在对2的索引值进行译码之后,视频译码器可对3的游程进行译码,所述游程指示三个后续样本也具有相同的2的调色板索引值。以类似方式,在使用CopyAbove模式对索引进行译码之后对四个调色板索引的游程进行译码可指示:从当前经译码的样本位置上方的行中的对应调色板索引值复制总共五个调色板索引。

[0083] 使用调色板,视频编码器20及/或视频解码器30可经配置以将样本的块(例如,视频数据的块)译码成索引块,其中索引块为包含映射到一或多个调色板条目的索引值及在一些实例中的一或多个逸出像素值的块。视频编码器20可经配置以对索引块进行熵编码以压缩索引块。类似地,视频解码器30可经配置以对经编码索引块进行熵解码以产生索引块,视频解码器30可从所述索引块产生样本的块(例如,由编码器20编码的视频数据块)。举例来说,基于游程长度的熵译码可用于压缩与解压缩索引块。在一些实例中,视频编码器20及视频解码器30可经配置以分别使用CABAC来对索引块进行熵编码及解码。

[0084] 为将CABAC译码应用于信息(例如,语法元素、例如索引块的索引值的索引块或其它信息),视频译码器(例如,视频编码器20及视频解码器30)可对信息执行二进制化。二进制化指代将信息转换成一系列一或多个位的过程。一或多个位的每个系列可被称为“二进制”。二进制化为无损过程且可包含以下译码技术中的一个或组合:固定长度译码、一元译码、截断的一元译码、截断的莱斯(Rice)译码、哥伦布译码、指数哥伦布译码、哥伦布-莱斯译码、哥伦布译码的任一形式、莱斯译码的任一形式及熵译码的任一形式。举例来说,二进制化可包含使用8位固定长度技术将整数值5表示成00000101,或者使用一元译码技术将整数值5表示成11110。

[0085] 在二进制化之后,视频译码器可识别译码上下文。译码上下文可识别译码具有特定值的二进制数的概率。举例来说,译码上下文可指示对0值二进制进行译码的0.7概率,以及对1值二进制进行译码的0.3概率。在识别译码上下文之后,视频译码器可基于上下文算术地译码二进制,此已知为上下文模式译码。使用CABAC上下文模式译码而译码的二进制可被称作“上下文二进制”。

[0086] 此外,视频译码器(例如,视频编码器20及视频解码器30)可使用旁路CABAC译码(例如,旁路模式译码)来译码一些二进制,而非对所有二进制执行上下文模式译码。旁路模式译码指代不使用自适应上下文(例如,译码上下文)而算术地译码二进制的过程。也就是说,旁路译码引擎不选择上下文,并且可假设两个符号(0和1)的概率都为0.5。虽然旁路模式译码的带宽效率可不如上下文模式译码,但是在对二进制执行旁路模式译码而不是对二进制执行上下文模式译码时可能在计算方面成本较低。此外,执行旁路模式译码可允许较高的并行化度及吞吐量。使用旁路模式译码而译码的二进制可被称作“旁路二进制”。

[0087] 视频编码器20及视频解码器30可经配置具有CABAC译码器(例如,分别为CABAC编

码器及CABAC解码器)。CABAC译码器可包含用以执行CABAC上下文模式译码的上下文模式译码引擎及用以执行旁路模式译码的旁路模式译码引擎。如果二进制经上下文模式译码,那么上下文模式译码引擎用于对此二进制进行译码。上下文模式译码引擎可需要多于两个处理周期来对单个二进制进行译码。然而,由于恰当的管线设计,上下文模式译码引擎可仅需要n+M周期以对n个二进制进行编码,其中M为起始管线的额外负荷。M通常大于0。

[0088] 在CABAC译码过程开始时(即,从旁路模式到上下文模式的每次转换且反之亦然),引入管线额外负荷。如果二进制经旁路模式译码,那么旁路模式译码引擎用于对此二进制进行译码。旁路模式译码引擎可预计仅需要一个周期来对n位信息进行,其中n可大于一。因此,如果旁路二进制及上下文二进制的集合内的所有旁路二进制经一起译码且所述集合内的所有上下文二进制经一起译码,那么对旁路二进制及上下文二进制的集合进行译码的周期的总数可减少。特定来说,在转变到上下文模式译码之前或之后将旁路二进制一起译码可节省重新启动上下文模式译码引擎所需要的额外负荷。举例来说,当分别使用调色板模式对视频数据块进行编码或解码时,视频编码器20及视频解码器30可经配置以在旁路模式到上下文模式之间切换。在另一实例中,视频编码器20及视频解码器30可经配置以减少当使用调色板模式对视频数据块进行编码或解码时编码或解码过程在旁路模式到上下文模式之间切换的次数。

[0089] 本发明中描述的技术可包含用于用信号发送基于调色板的视频译码模式,发射调色板,导出调色板,用信号发送扫描次序,导出扫描次序,及发射基于调色板的视频译码映射及其它语法元素中的一或多者的各种组合的技术。举例来说,本发明的技术可针对熵译码调色板信息。在一些实例中,本发明的技术可尤其用于提高译码效率并降低与基于调色板的视频译码相关联的译码低效率。因此,如下文更详细描述,在一些情况下,当使用调色板模式译码视频数据时,本发明的技术可改良效率及改良位率。

[0090] 如上文所描述,在屏幕内容译码中的当前调色板模式设计中,palette_index_idc及palette_escape_val的语法元素经CABAC旁路译码,且与经CABAC上下文译码的其它语法元素(例如,palette_run_msb_id_plus1)交错。然而,将经旁路译码的信息(例如,语法元素)分组在一起可为有益的,这可改良译码效率及/或降低编解码器复杂度。

[0091] 例如如JCTVC-S1005中所定义,palette_index_idc的语法元素可为对表示为currentPaletteEntries的阵列的索引的指示。palette_index_idc的值可在0到(adjustedIndexMax-1)(包含性的)的范围内。例如如JCTVC-S1005中所定义,palette_escape_val的语法元素可指定分量的经量化逸出经译码样本值。例如如JCTVC-S1005中所定义,palette_run_msb_id_plus1减1可指定paletteRun的二进制表示中的最高有效位的索引。例如如JCTVC-S1005中所定义,当palette_run_type_flag等于COPY_ABOVE_MODE时,可变paletteRun可指定具有与上述行中位置相同的调色板索引的连续位置减1的数目,或当palette_run_type_flag等于COPY_INDEX_MODE时指定具有相同调色板索引的连续位置减1的数目。关于palette_index_idc、palette_escape_val、palette_run_msb_id_plus1、currentPaletteEntries、adjustedIndexMax及paletteRun的额外细节可见于JCTVC-S1005中。

[0092] 在一些实例中,本发明描述在调色板索引块译码部分的前面分组所有语法元素palette_index_idc以改良CABAC吞吐量的方法。举例来说,视频编码器20可经配置以在调

色板索引块译码部分前面编码所有语法元素palette_index_idc。举例来说，视频编码器20可经配置以在对待经上下文模式编码的语法元素进行编码之前对所有语法元素palette_index_idc进行编码。类似地，视频解码器30可经配置以在调色板索引块译码部分前面对所有语法元素palette_index_idc进行解码。举例来说，视频解码器30可经配置以在对经上下文模式编码的语法元素进行解码之前对所有语法元素palette_index_idc进行解码。

[0093] 作为另一实例，视频编码器20可经配置以在调色板索引块译码部分前面对所有语法元素palette_index_idc进行旁路模式编码，以使得在对关于调色板游程类型(例如，CopyAbove模式或索引模式)及/或游程长度(例如，palette_run_msb_id_plus1)的语法元素进行编码之前对所有语法元素palette_index_idc进行编码。类似地，视频解码器30可经配置以在块的调色板索引块译码部分前面对块的所有语法元素palette_index_idc进行解码，以使得在对关于调色板游程类型(例如，CopyAbove模式或索引模式)及/或游程长度(例如，palette_run_msb_id_plus1)的语法元素进行解码之前对所有语法元素palette_index_idc进行解码。

[0094] 关于调色板游程类型(例如，CopyAbove模式或索引模式)及/或游程长度(例如，palette_run_msb_id_plus1)的语法元素

[0095] 作为另一实例，实例视频编码器20可经配置以在对关于调色板游程类型(例如，CopyAbove模式或索引模式)及/或游程长度(例如，palette_run_msb_id_plus1)的语法元素进行上下文编码之前对所有语法元素palette_index_idc进行编码。类似地，视频解码器30可经配置以在对关于调色板游程类型(例如，CopyAbove模式或索引模式)及/或游程长度(例如，palette_run_msb_id_plus1)的语法元素进行上下文解码之前对所有语法元素palette_index_idc进行解码。

[0096] 作为另一实例，视频编码器20可经配置以在对待经上下文模式编码的语法元素进行编码之前对调色板块译码部分内的所有语法元素palette_index_idc进行编码。类似地，视频解码器30可经配置以在解码经上下文模式编码的语法元素之前解码调色板块译码部分内的所有语法元素palette_index_idc。作为另一实例，视频编码器20可经配置以在对关于调色板游程类型(例如，CopyAbove模式或索引模式)及/或游程长度(例如，palette_run_msb_id_plus1)的语法元素进行上下文编码之前对调色板块译码部分内的所有语法元素palette_index_idc进行编码。类似地，视频解码器30可经配置以在对关于调色板游程类型(例如，CopyAbove模式或索引模式)及/或游程长度(例如，palette_run_msb_id_plus1)的语法元素进行上下文解码之前对调色板块译码部分内的所有语法元素palette_index_idc进行解码。

[0097] 大体来说，视频编码器20及视频解码器30可经配置以使以分别使用上下文模式编码或解码语法元素的旁路模式而编码或解码palette_index_idc不交错。举例来说，视频编码器20及视频解码器30可经配置以使以分别使用上下文模式编码或解码关于调色板游程类型(例如，CopyAbove模式或索引模式)及/或游程长度(例如，palette_run_msb_id_plus1)的语法元素的旁路模式而编码或解码palette_index_idc不交错。作为另一实例，视频编码器20可经配置以在对需要上下文模式的语法元素进行上下文编码之前对palette_index_idc语法元素的所有实例进行旁路编码。类似地，视频解码器30可经配置以在对需要上下文模式的语法元素进行上下文解码之前对palette_index_idc语法元素的所有实例进

行旁路解码。作为另一实例，视频编码器20可经配置以在对关于调色板游程类型(例如，CopyAbove模式或索引模式)及/或游程长度(例如，palette_run_msb_id_plus1)的语法元素进行上下文编码之前对语法元素palette_index_idc的所有实例进行旁路编码。类似地，视频解码器30可经配置以在对关于调色板游程类型(例如，CopyAbove模式或索引模式)及/或游程长度(例如，palette_run_msb_id_plus1)的语法元素进行上下文解码之前对palette_index_idc语法元素的所有实例进行旁路解码。

[0098] 视频编码器20及视频解码器30还可分别编码及解码表示palette_index_idc的出现次数的值。视频编码器20及视频解码器30可使用表示palette_index_idc的出现次数的值以分别编码或解码语法元素palette_index_idc中的每一者。本发明中描述的技术还可消除调色板游程长度相关的语法元素的冗余，且消除palette_run_type_flag及palette_index_idc的冗余。

[0099] 在一些实例中，本发明描述在块(例如，PU或CU)的调色板索引块译码部分前面分组所有语法元素palette_escape_val以改良CABAC吞吐量的方法。举例来说，视频编码器20可经配置以在块的调色板索引块译码部分前面编码所有语法元素palette_escape_val。举例来说，视频编码器20可经配置以在调色板索引块译码部分前面对所有语法元素palette_escape_val进行旁路模式编码以使得在对关于调色板游程类型(例如，CopyAbove模式或索引模式)及/或游程长度(例如，palette_run_msb_id_plus1)的语法元素进行编码之前对所有语法元素palette_escape_val进行编码。类似地，视频解码器30可经配置以在块的调色板索引块译码部分前面对块的所有语法元素palette_escape_val进行解码以使得在对关于调色板游程类型(例如，CopyAbove模式或索引模式)及/或游程长度(例如，palette_run_msb_id_plus1)的语法元素进行解码之前对所有语法元素palette_escape_val进行解码。作为另一实例，视频编码器20可经配置以在对待经上下文模式编码的语法元素进行编码之前对所有语法元素palette_escape_val进行编码。举例来说，视频编码器20可经配置以在对关于调色板游程类型(例如，CopyAbove模式或索引模式)及/或游程长度(例如，palette_run_msb_id_plus1)的语法元素进行上下文编码之前对所有语法元素palette_escape_val进行编码。类似地，视频解码器30可经配置以在块的调色板索引块译码部分前面对所有语法元素palette_escape_val进行解码。举例来说，视频解码器30可经配置以在对块中的经上下文模式编码的语法元素进行解码之前对所有语法元素palette_escape_val进行解码。

[0100] 作为另一实例，视频编码器20可经配置以在对待经上下文模式编码的语法元素进行编码之前对块的调色板块译码部分内的所有语法元素palette_escape_val进行编码。类似地，视频解码器30可经配置以在对块的经上下文模式编码的语法元素进行解码之前对块的调色板块译码部分内的所有语法元素palette_escape_val进行解码。

[0101] 大体来说，视频编码器20及视频解码器30可经配置以使以分别针对块使用上下文模式编码或解码语法元素的旁路模式而编码或解码块(例如，PU或CU)的palette_escape_val不交错。举例来说，视频编码器20及视频解码器30可经配置以使以分别使用上下文模式编码或解码关于调色板游程类型(例如，CopyAbove模式或索引模式)及/或游程长度(例如，palette_run_msb_id_plus1)的语法元素的旁路模式而编码或解码palette_escape_val不交错。作为另一实例，视频编码器20可经配置以在对需要上下文模式的语法元素进行上下文编码之前对块的palette_escape_val语法元素的所有实例进行旁路编码。类似地，视频

解码器30可经配置以在对需要块的上下文模式的语法元素进行上下文解码之前对块(例如,PU或CU)的palette_escape_val语法元素的所有实例进行旁路解码。

[0102] 视频编码器20及视频解码器30还可分别对表示块的palette_escape_val的出现次数的值进行编码及解码。视频编码器20及视频解码器30可使用表示palette_escape_val的出现次数的值以分别对块的语法元素palette_escape_val中的每一者进行编码或解码。本发明中描述的技术可减小块的palette_index_idc的动态范围,这可产生改良的译码效率。

[0103] 可以任何组合或与彼此独立而结合彼此从而利用本文所描述的技术、方面及/或实例。举例来说,视频编码器20及视频解码器30可经配置以执行本文所描述的技术、方面及/或实例中的一或多者的任一个或任何合适的组合。

[0104] 在一些实例中,如上文所描述,为改良CABAC吞吐量,视频译码器(例如,视频编码器20)可经配置以分组语法元素palette_index_idc的所有出现。举例来说,视频译码器(例如,视频编码器20)可经配置以在当前块的索引译码部分前面分组当前块(例如,PU或CU)中的语法元素palette_index_idc的所有出现。类似地,如上文所描述,视频解码器(例如,视频解码器30)可经配置以解码所有语法元素palette_index_idc。图7说明其中视频编码器20可经配置以(例如)关于R.Joshi及J.Xu“高效率视频译码(HEVC)屏幕内容译码:草案2”JCTVC-S1005,7.3.3.8部分,在索引译码块前面分组当前块(例如,CU)中的语法元素palette_index_idc的所有出现的一个实例。本发明的此方面被称作方面1。具体来说,图7说明将语法元素palette_index_idc的实例重新定位于索引译码块前面(也可被称作调色板块译码部分或索引译码块的前面)的视频编码器20的实例。通过重新定位所说明的语法元素palette_index_idc的所说明的实例,视频编码器20可经配置以通过使用旁路模式语法元素palette_index_idc的所有实例进行译码及切换到上下文模式以对在索引译码块中的语法元素palette_index_idc的所有实例经旁路模式编码之后发生的调色板信息进行译码从而改良CABAC吞吐量。

[0105] 根据JCTVC-S1005的公开,将以旁路模式译码palette_index_idc的一个实例,随后将以上下文模式译码关于调色板游程类型的语法元素的一个实例及palette_run_msb_id_plus1的一个实例,且所述过程将重复while(scanPos<nCbS*nCbS),意味着视频编码器将在旁路模式译码与上下文模式译码之间来回切换,因为待使用旁路模式译码的语法元素未被分组在一起。这在图7中被描绘,其中椭圆在“while(scanPos<nCbS*nCbS)”的循环正下方(即,椭圆不包括展示关于调色板游程类型的语法元素使用上下文模式经编码的信息),围绕随之而来的palette_index_idc语法元素的if语句的方块在“while(scanPos<nCbS*nCbS)”的循环及后续伪码之下。然而,如上文所描述,图7还描绘本发明的方面1,其为将语法元素palette_index_idc的一或多个实例分组(也可被称作重新定位)到(例如)索引译码块前面。通过重新定位待使用旁路模式编码的一或多个语法元素(例如,或其它调色板信息),视频编码器(例如,视频编码器20)可通过减少视频编码器或视频解码器必须在旁路模式编码与上下文模式编码之间切换的次数而增加熵译码的吞吐量。类似地,通过以此方式重新定位一或多个语法元素,视频解码器(例如,视频解码器30)的吞吐量可增大,因为视频解码器必须在旁路模式解码与上下文模式解码之间切换的次数减少。在本发明中描述的技术的一些实例中,在palette_run_msb_id_plus1的实例将以上下文模式来译码之前将以旁

路模式译码palette_index_idc语法元素的所有实例。

[0106] 在一些实例中，视频编码器20可经配置以使用称为(例如)num_palette_index的语法元素用信号发送语法元素palette_index_idc的出现次数(例如，实例)。举例来说，视频编码器20可以位流用信号发送num_palette_index的值，其中所述值表示语法元素palette_index_idc的出现次数。在一些实例中，视频编码器20可经配置以不用信号发送如palette_index_idc的索引值。在这些实例中，视频解码器30可经配置以推断索引值。举例来说，可以num_palette_index计数palette_index_idc的发生，所述计数可等于游程类型(例如，COPY_INDEX_MODE)在特定块中发生的次数。即使当推断游程类型(例如，COPY_INDEX_MODE)或推断palette_index_idc时，其仍计入num_palette_index。如本文所使用，在一些实例中，经剖析、解码或保持待解码的多个索引的参考可指代与模式或索引是否被推断无关的COPY_INDEX_MODE的数目。视频解码器30可经配置以通过(例如)解码来自位流的对应于num_palette_index语法元素的经编码值而确定语法元素palette_index_idc的出现次数(例如，实例)。本发明的这一方面被称作方面2。视频编码器20和视频解码器30可经配置以使用方面2或不使用方面2实施方面1。在语法方面，根据一些实例，方面2可被定义为：

[0107]

indices_idc_coding(){	
num_palette_index	ae(v)
for(i＝0;i<num_palette_index;i++)	
palette_index_idc	ae(v)
}	

[0108] 在一些实例中，仅当可变indexMax大于1时，视频编码器20及视频解码器30可经配置以实施(例如，通过启用)方面1及方面2。本发明的这一方面被称作方面3。可变indexMax可指定调色板索引具有的用于当前译码单元的不同值的数目。在一些实例中，indexMax可指代(调色板大小+palette_escape_val_present_flag)的数量。

[0109] 在一些实例中，当以下情况时可禁用方面1及方面2：(a)当前块中不存在逸出像素(即palette_escape_val_present_flag＝＝0)且调色板大小小于2；或(b)当前块中可存在至少一个逸出像素(即，palette_escape_val_present_flag＝＝1)且调色板大小等于0。在其它实例中，仅当可变indexMax大于2时，视频编码器20及视频解码器30可经配置以实施(例如，通过启用)方面1及方面2。类似地，在其中indexMax等于(调色板大小+palette_escape_val_present_flag)的实例中，当indexMax大于1时可启用(例如，实施)方面1及方面2。举例来说，如果调色板大小为0且palette_escape_val_present_flag为1，那么块中的所有像素为逸出像素；及，由此已经已知索引。作为另一实例，如果palette_escape_val_present_flag为0且调色板大小为1，那么，同样地，每一像素具有索引0；及，由此可不必要用信号发送索引。

[0110] 在一些实例中，视频编码器20可经配置以实施方面1及方面2以使得语法元素palette_run_type_flag[xC][yC]的最末发生(例如，实例)通过视频编码器20在调色板索引块译码部分前面被用信号发送。本发明的这一方面被称作方面4。具体来说，根据一些实例，如下文，可通过添加新的语法元素palette_last_run_type_flag来更新语法表：

[0111]

indices_idc_coding(){	
num_palette_index	ae(v)
for(i=0;i<num_palette_index;i++)	
palette_index_idc	ae(v)
palette_last_run_type_flag	ae(v)
}	

[0112] 视频解码器30可经配置以通过(例如)解码来自位流的经编码palette_last_run_type_flag语法元素来确定语法元素palette_run_type_flag[xC][yC]的最末发生(例如,实例)。palette_last_run_type_flag的语法元素可在(例如)CABAC中经旁路模式译码或经上下文模式译码。在其中palette_last_run_type_flag语法元素经上下文模式译码的实例中,palette_last_run_type_flag语法元素可与palette_run_type_flag[xC][yC]共享相同上下文,或palette_last_run_type_flag语法元素可具有独立于palette_run_type_flag[xC][yC]的上下文的其自身上下文。

[0113] 在一些实例中,视频解码器30可经配置以解码语法元素palette_index_idc,以使得针对palette_index_idc语法元素的第一发生(例如,实例)禁用动态范围调节过程。本发明的这一方面被称作方面5。具体来说,使用与JCTVC-S1005部分7.4.9.6中指定的adjustedIndexMax可变的导出过程极类似的过程。出于对比的目的,JCTVC-S1005描述可变adjustedIndexMax可如下导出:

[0114] adjustedIndexMax=indexMax

[0115] if(scanPos>0)

[0116] adjustedIndexMax-=1

[0117] 然而,根据本发明的方面5,可变adjustIndexMax可如下文所阐述而导出。举例来说,对于每一块,在剖析之前,可变isFirstIndex经初始化为1。在一些实例中,可变adjustedIndexMax可如下导出:

[0118] adjustedIndexMax=indexMax

[0119] palette_index_idc

[0120] if(isFirstIndex){

[0121] adjustedIndexMax-=isFirstIndex

[0122] isFirstIndex=0

[0123] }

[0124] 在一些实例中,在剖析及解码paletteRun之前,视频解码器30可经配置以检查一或多个条件。本发明的这一方面被称作方面6。例如由JCTVC-S1005中所公开,当palette_run_type_flag等于COPY_ABOVE_MODE时,可变paletteRun可指定具有与上述行中位置相同的调色板索引的连续位置减1的数目,或当palette_run_type_flag等于COPY_INDEX_MODE时指定具有相同调色板索引的连续位置减1的数目。

[0125] 参考视频解码器30可经配置以检查的一或多个条件,如果视频解码器30确定满足所述条件中的一或多者,那么视频解码器30可经配置以针对关于当前paletteRun(即,palette_run_msb_id_plus1及palette_run_refinement_bits)的语法元素旁路剖析及解

码过程。在此实例中，视频解码器30可经配置以随着运行到当前块的末端(即，等于maxPaletteRun)隐含地推导当前paletteRun。与方面6相关的一或多个条件的列表包括：(i)等于num_palette_index的经剖析/经解码palette_index_idc语法元素的数目；或，替代地，可定义可变paletteIndicesLeft等于num_palette_index减所接收的索引的数目，且使用所述定义，可将此条件陈述为paletteIndicesLeft等于零；及/或(ii)当前调色板游程类型palette_run_type_flag[xC][yC]等于最末调色板游程类型palette_last_run_type_flag。

[0126] 在一些实例中，如果不同时满足上文针对方面6所阐述的条件(i)及(ii)，那么视频编码器20可经配置以将调色板游程长度译码到位流中。本发明的此方面被称作方面7。在其它实例中，如果不同时满足上文针对方面6所阐述的条件(i)及(ii)，那么视频编码器20可经配置以将调色板游程长度译码到位流中。根据当前草案说明书JCTVC-S1005，需要指定最大可达成游程长度的参数为输入，其中参数等于maxPaletteRun＝nCbS*nCbS-scanPos-1。然而，根据本发明，视频编码器20可经配置以将指定最大可达成游程长度的参数减小到maxPaletteRun＝nCbS*nCbS-scanPos-1-paletteIndicesLeft以改良译码效率。如本文所使用，nCbS指定当前块的大小。

[0127] 在一些实例中，如果块并不在调色板共享模式中(即，palette_share_flag[x0][y0]＝＝0)，那么可施加规范性约束条件于需要其从未用信号发送具有未使用条目的调色板的视频编码器20上。本发明的这一方面被称作方面8。

[0128] 在一些实例中，对于不使用调色板共享的调色板模式，当满足以下条件中的一或多者时：其中num_palette_index等于indexMax的条件1及其中paletteIndicesLeft＝＝1的条件2，视频解码器30可经配置以旁路对语法元素palette_index_idc的当前发生(例如，实例)的解码。在这些实例中，视频解码器30可经配置以将语法元素palette_index_idc的当前发生的值隐含地导出为调色板中的索引，但已在解码过程期间出现于索引映射中(例如，直到此点未在解码过程中出现于索引图中)。本发明的这一方面被称作方面9。

[0129] 视频解码器30可经配置以导出上文针对方面9所阐述的语法元素palette_index_idc的当前发生的值，因为条件1需要0与(indexMax-1)之间的每一索引(包含性地)经用信号发送且仅用信号发送一次。因此，在第一(indexMax-1)索引值被用信号发送之后，视频解码器30可经配置以导出最末索引值为0与(indexMax-1)之间的数目，所述值已在当前索引映射图的解码过程期间出现。

[0130] 在一些实例中，当满足以下条件中的一者或两者时，视频解码器30可经配置以旁路对语法元素palette_run_type_flag[xC][yC]的当前发生(例如，实例)的解码：条件1，其中paletteIndicesLeft等于0，及条件2，其中当前像素处于扫描次序中的块的最末位置。在这些实例中，视频解码器30可经配置以隐含地导出语法元素palette_run_type_flag[xC][yC]的当前发生的值。举例来说，当满足条件1时，palette_run_type_flag[xC][yC]视频解码器30可经配置以导出语法元素palette_run_type_flag[xC][yC]的当前发生的值为COPY_ABOVE_MODE。作为另一实例，当满足条件1时，如果paletteIndicesLeft>0，那么palette_run_type_flag[xC][yC]视频解码器30可经配置以导出语法元素palette_run_type_flag[xC][yC]的当前发生的值为COPY_INDEX_MODE，且如果paletteIndicesLeft＝0，那么导出为COPY_ABOVE_MODE。本发明的这一方面被称作方面10。

[0131] 如本文中所描述，视频编码器20及视频解码器30可经配置以确定何时满足条件。举例来说，关于方面10，视频解码器30可经配置以确定是否满足条件1。类似地，视频解码器30可经配置以确定是否满足条件2。回应于确定满足条件1或条件2，如上文所阐述，视频解码器30可经配置以导出语法元素palette_run_type_flag[xC][yC]的当前发生的值。

[0132] 在一些实例中，视频编码器20及视频解码器30可经配置以使用任一哥伦布码族分别对num_palette_index语法元素进行编码或解码。举例来说，视频编码器20及视频解码器30可经配置以使用(例如)哥伦布莱斯码、指数哥伦布码、截断莱斯码、一元码或哥伦布莱斯码与指数哥伦布码的级联来分别对num_palette_index语法元素进行编码或解码。本发明的这一方面被称作方面11。

[0133] 在其它实例中，视频编码器20及视频解码器30可经配置以使用任一哥伦布码族中得任一截断版本来分别对num_palette_index语法元素进行编码或解码。举例来说，视频编码器20及视频解码器30可经配置以使用(例如)截断哥伦布莱斯码、截断指数哥伦布码、截断截断莱斯码、截断一元码或截断莱斯码与指数哥伦布码的级联(例如，用于译码coeff_abs_level_remaining语法元素的程序代码)来分别对num_palette_index语法元素进行编码或解码。本发明这一方面被称作方面12。

[0134] 在一些实例中，与方面11或方面12相关的任意哥伦布参数取决于CU大小、indexMax、调色板大小及/或palette_escape_val_present_flag。所述相依性可表达为方程或查找表。在一些实例中，视频编码器20可经配置以用信号发送查找表或方程中的参数，以使得其由(例如)SPS/PPS/切片标头中的视频解码器30所接收。替代地或另外，可根据逐块基础适应性地更新参数。本发明的这一方面被称作方面13。在一些实例中，哥伦布参数cRiceParam可取决于indexMax、调色板大小及/或palette_escape_val_present_flag。哥伦布参数cRiceParam可从块到块而改变。

[0135] 在一些实例中，视频编码器20可经配置以通过用信号发送num_palette_index的值与差量值之间的差异而预测性地编码num_palette_index，所述值可由称为(例如)numPaletteIndexCoded的语法元素来表达。本发明的这一方面被称作方面14。举例来说，视频编码器20可经配置以通过用信号发送numPaletteIndexCoded的值而预测性地编码num_palette_index，其中numPaletteIndexCoded＝num_palette_index-IndexOffsetValue。类似地，视频解码器30可经配置以通过(例如)确定来自位流的numPaletteIndexCoded的值而预测性地解码num_palette_index。因为numPaletteIndexCoded＝num_palette_index-IndexOffsetValue，视频解码器30可经配置以基于所确定的numPaletteIndexCoded的值及IndexOffsetValue的值而确定num_palette_index的值。

[0136] 在一些实例中，可变IndexOffsetValue可为常数。举例来说，IndexOffsetValue可等于调色板共享模式的X的常数值或可等于非调色板共享模式的Y的常数值，其中X及Y为整数。在一些实例中，X及Y可相同(例如，X等于Y，例如等于1)。在其它实例中，X及Y可不同(例如，X不等于Y)。举例来说，当使用调色板共享模式时IndexOffsetValue可等于9，且当使用非共享模式时IndexOffsetValue可等于33。在一些实例中，可变IndexOffsetValue可取决于语法元素palette_share_flag[x0][y0]。在其它实例中，可变IndexOffsetValue可取决于可变indexMax。举例来说，IndexOffsetValue可等于indexMax。在一些实例中，视频编码器20可经配置以用信号发送SPS/PPS/切片标头中的IndexOffsetValue。替代地或另外，可

变IndexOffsetValue可经适应性地逐块更新，意味着对应于可变IndexOffsetValue的值可经适应性地逐块更新。

[0137] 在一些实例中，视频编码器20及视频解码器30可经配置以分别编码或解码可使用任意哥伦布码族或任意截断哥伦布族(例如哥伦布莱斯码与指数哥伦布码的级联)译码的numPaletteIndexCoded。举例来说，当IndexOffsetValue等于1时，numPaletteIndexCoded等于num_palette_index-1。

[0138] 在一些实例中，视频编码器20及视频解码器30可经配置以使用任意哥伦布码族分别编码或解码numPaletteIndexCoded。举例来说，视频编码器20及视频解码器30可经配置以使用(例如)哥伦布莱斯码、指数哥伦布码、截断莱斯码、一元码或哥伦布莱斯码与指数哥伦布码的级联来分别编码或解码numPaletteIndexCoded。

[0139] 在其它实例中，视频编码器20及视频解码器30可经配置以使用任意哥伦布码族的任意截断版本来分别编码或解码numPaletteIndexCoded。举例来说，视频编码器20及视频解码器30可经配置以使用(例如)截断哥伦布莱斯码、截断指数哥伦布码、截断截断莱斯码、截断一元码或截断莱斯码与指数哥伦布码的级联(例如，用于译码coeff_abs_level_remaining语法元素的代码)来分别编码或解码numPaletteIndexCoded。

[0140] 为译码numPaletteIndexCoded，视频编码器20可经配置以确定numPaletteIndexCoded的正负号。视频编码器20可经配置以用信号发送指示numPaletteIndexCoded的值为负值与否(例如，经确定正负号是正的还是负的)的旗标。本发明的这一方面被称作方面15。在一些实例中，视频编码器20可经配置以用信号发送旗标，且随后用信号发送numPaletteIndexCoded的值。在其它实例中，视频编码器20可经配置以用信号发送numPaletteIndexCoded的值，且随后用信号发送旗标。视频编码器20可经配置以使用旁路模式或上下文模式编码旗标。如果经上下文译码，那么上下文可取决于CU大小、indexMax、调色板大小及/或palette_escape_val_present_flag。

[0141] 如上文所描述，视频编码器20可经配置以根据一些实例确定numPaletteIndexCoded的正负号。如果所确定numPaletteIndexCoded的正负号为负的，那么视频编码器20可经配置以将(1-numPaletteIndexCoded)的值编码到位流中。如果所确定numPaletteIndexCoded的正负号为正的，那么视频编码器20可经配置以将numPaletteIndexCoded的值编码到位流中。视频编码器20可经配置以使用取决于(例如)numPaletteIndexCoded的正负号、CU大小、indexMax、调色板大小及/或palette_escape_val_present_flag的不同哥伦布码参数来编码(1-numPaletteIndexCoded)的值或numPaletteIndexCoded)值。

[0142] 在一些实例中，视频编码器20可经配置以使用映射操作来表示numPaletteIndexCoded的负部，此可除方面15外还可替代方面15。本发明的这一方面被称作方面16。举例来说，可引入映射间隔，且其被定义为可变mapInterval。视频编码器20可经配置以使用可变mapInterval将numPaletteIndexCoded的负值映射到等间距的等于：mapInterval×(-numPaletteIndexCoded)-1的正值。numPaletteIndexCoded的相对应正值可因此移位以适应由经映射负值所获得的位置。

[0143] 举例来说，如果mapInterval＝2，且numPaletteIndexCoded是选自{-3,-2,-1,0,1,2,3}，那么所述映射可如下文表I中所说明。在此实例中，视频编码器20可经配置以使用

表I中的经映射值来编码numPaletteIndexCode的值。举例来说，视频编码器20可经配置以将经映射值熵编码成二进制形式。

[0144] 表I.码字映射实例

[0145]

numPaletteIndexCoded	经映射值
-3	5
-2	3
-1	1
0	0
1	2
2	4
3	6

[0146] 在一些实例中，视频编码器20可经配置以使用关于方面16所描述的映射操作来表示numPaletteIndexCoded的负部。视频编码器20还可经配置以移除当实施方面16时可出现的一或多个冗余。本发明的这一方面被称作方面17。举例来说，numPaletteIndexCoded的负值的数目可在A＝{-1,-2,...,-IndexOffsetValue+1}范围内。作为另一实例，numPaletteIndexCode的负值的数目可在A＝{-1,-2,...,-IndexOffsetValue+1,IndexOffsetValue}范围内。在这些实例中的任一者中，经映射值仅需要保留负numPaletteIndexCoded值的(IndexOffsetValue-1)或IndexOffsetValue位置。举例来说，如果mapInterval＝2，且numPaletteIndexCoded是选自{-3,-2,-1,0,1,2,3,4,5,6,7,8}，那么所述映射可如下文表II中所说明。在此实例中，视频编码器20可经配置以使用表II中的经映射值来编码numPaletteIndexCode的值。举例来说，视频编码器20可经配置以将经映射值熵编码成二进制形式。

[0147] 表II.码字映射实例

[0148]

numPaletteIndexCoded	经映射值
-3	5
-2	3
-1	1
0	0
1	2
2	4
3	*6*
4	*7*
5	*8*
6	*9*
7	*10*
8	*11*

[0149] 如上文表II中所展示，视频编码器20可经配置以编码对应于numPaletteIndexCode的值的经映射值，以使得numPaletteIndexCode的负值与正值在某一值之后不交错。举例来说，在上文表II的实例中，不存在经由以numPaletteIndexCoded的值3开始的经映射值的numPaletteIndexCoded的正值与负值的交错(即，numPaletteIndexCoded的正值3-8映射到经映射值6-11)。

[0150] 如上文所描述，视频编码器20还可经配置以移除当实施方面16时可出现的一或多

个冗余。不同于上文所描述的冗余实例的另一冗余实例包含:由于num_palette_index以当前块中的像素的总数为上限,numPaletteIndexCoded也有上限。因此,在以正码字的所有概率分配位置之后,负值可经映射到以下位置而不交错。举例来说,如果mapInterval＝2,且numPaletteIndexCoded是选自{-5,-4,-3,-2,-1,0,1,2,3},那么所述映射可如下文表III中所说明。在此实例中,视频编码器20可经配置以使用表III中的经映射值来编码numPaletteIndexCode的值。举例来说,视频编码器20可经配置以将经映射值熵编码成二进制形式。

[0151] 表III.码字映射实例

[0152]

numPaletteIndexCoded	经映射值
-5	*8*
-4	*7*
-3	5
-2	3
-1	1
0	0
1	2
2	4
3	6

[0153] 如上文表III中所展示,视频编码器20可经配置以编码对应于numPaletteIndexCode的值的经映射值,以使得numPaletteIndexCode的负值与正值在某一值之后不交错。举例来说,在上文表III的实例中,不存在经由以numPaletteIndexCoded的值4开始的经映射值的numPaletteIndexCoded的正值与负值的交错(即,numPaletteIndexCoded的负值-4及-5映射到经映射值7及8)。

[0154] 在一些实例中,视频编码器20可经配置以进一步解耦调色板索引与调色板游程之间的关系。本发明的这一方面被称作方面18。举例来说,视频编码器20可经配置以使得调色板游程译码的上下文取决于前述调色板游程长度或取决于前述游程的palette_run_msb_id_plus1、indexMax及/或CU大小,而不是允许调色板游程译码的上下文取决于经剖析或经解码索引。

[0155] 在一些实例中,为进一步分组旁路二进制,视频编码器20可经配置以如下用信号发送调色板游程类型(即palette_run_type_flag[xC][yC])之前用信号发送调色板块中的逸出索引的数目以及逸出值。本发明的这一方面被称作方面19。斜体部分说明相对于JCT-VC S1005之前述版本的变化,且加粗部分以及右列中的“ae(v)”指示语法元素的用信号发送。

[0156]

...	
if(currentPaletteSize != 0)	
palette_escape_val_present_flag	ae(v)
if(palette_escape_val_present_flag || (indexMax > 0))	
escape_idc_coding()	
if(palette_escape_val_present_flag) {	
if(cu_qp_delta_enabled_flag && !IsCuQpDeltaCoded) {	
cu_qp_delta_palette_abs	ae(v)

[0157]

if(cu_qp_delta_palette_abs)	
cu_qp_delta_palette_sign_flag	ae(v)
}	
if(cu_chroma_qp_offset_enabled_flag && !IsCuChromaQpOffsetCoded) {	
cu_chroma_qp_palette_offset_flag	ae(v)
if(cu_chroma_qp_offset_flag && chroma_qp_offset_list_len_minus1 > 0)	
cu_chroma_qp_palette_offset_idx	ae(v)
}	
}	
if(indexMax > 0)	
palette_transpose_flag	ae(v)
scanPos = 0	
while(scanPos < nCbS * nCbS) {	
…	

[0158] 在上文的实例中，escape_idc_coding()由用信号发送逸出索引的数目及对应于每一逸出索引的逸出值构成。如果palette_escape_val_present_flag为0或如果indexMax等于0，那么可不用信号发送调色板块中的逸出索引的数目。在前者情形中，逸出索引的数目经推断为0，且不用信号发送逸出值。在indexMax等于0的后者情形中，当palette_escape_val_present_flag等于1且用信号发送逸出值时，逸出索引的数目经推断等于块大小，或当palette_escape_val_present_flag等于0时逸出索引的数目经推断为零。

[0159] 在一些实例中，视频编码器20可经配置以使用哥伦布码族用信号发送逸出索引的数目。本发明的这一方面被称作方面20。举例来说，视频编码器20可经配置以使用(例如)哥伦布莱斯码、指数哥伦布码、截断莱斯码、一元码或哥伦布莱斯码与指数哥伦布码的级联来用信号发送逸出索引的数目。以上代码的截断版本可与等于块大小的最大集合一起使用。

[0160] 在一些实例中，提出对palette_escape_val_present_flag实施规范性限制，当palette_escape_val_present_flag等于0时，当前块中不存在逸出像素。本发明的这一方面被称作方面21。当palette_escape_val_present_flag等于1时，当前块中存在至少一个逸出像素。由于此限制，在escape_idc_coding()中，可译码逸出索引减1的数目，而不是逸出索引的数目以改良译码效率。在那种情况下，截断哥伦布码族的最大值可因此经调整到(blockSize-1)。

[0161] 在一些实例中，当在译码索引图块之前用信号发送逸出索引的数目时且当已经译码所有逸出索引时，那么indexMax可减小1。此外，如果indexMax变成1，那么索引、游程及模式译码经终止，因为可推断所有剩余样本的索引。本发明的这一方面被称作方面22。作为方面22的一个实例，假设调色板大小等于1且palette_escape_val_present_flag等于1。通常，可能的索引值为0及1，其中1用于逸出样本。在方面22下，视频编码器20可经配置以用信号发送逸出值/样本的数目。随后，当用信号发送索引且遇到最末逸出值/样本时，视频编码器20及/或视频解码器30两者可经配置以推断(例如，确定)不再存在逸出值/样本。由此，视频编码器20及/或视频解码器30可经配置以确定从最末逸出值/样本到块的末端可产生的仅有的索引值为0，意味着视频编码器20可经配置以不将模式、索引值及/或游程值从最末逸出值/样本用信号发送到块的末端。

[0162] 在一些实例中，escape_idc_coding()与indices_idc_coding()组合使用。本发明的这一方面被称作方面23。在一个实例中，可在用信号发送索引的数目之前用信号发送逸出索引的数目。在这种情形下，仅需要以indices_idc_coding()用信号发送非逸出索引的数目。在一个实例中，可在用信号发送索引的数目之后用信号发送逸出索引的数目。在这种情形下，截断哥伦布码族的最大值可因此经调整到num_palette_index。

[0163] 视频编码器20及/或视频解码器30可经配置以根据本发明中描述的技术来操作。大体来说，视频编码器20及/或视频解码器30可经配置以确定当前块以调色板模式来译码，对第一语法元素的多个实例进行旁路模式译码以用于重建当前块，且在对第一语法元素的多个实例进行旁路模式译码之后，对第二语法元素的多个实例进行上下文模式解码以用于重建当前块。

[0164] 图2为说明可实施本发明的技术的实例视频编码器20的框图。出于解释的目的而提供图2，且不应将其视为对如本发明中广泛例示及描述的技术的限制。出于解释的目的，本发明在HEVC译码及例如HEVC的SCC扩展的上下文中描述视频编码器20。然而，本发明的技术可适用于其它译码标准或方法。

[0165] 视频编码器20表示根据本发明中描述的各种实例的可经配置以执行用于基于调色板的译码及熵译码(例如，CABAC)的技术的装置的实例。

[0166] 在图2的实例中，视频编码器20包含块编码单元100、视频数据存储器101、残余产生单元102、变换处理单元104、量化单元106、反量化单元108、反变换处理单元110、重建单元112、滤波器单元114、经解码图片缓冲器116及熵编码单元118。块编码单元100包含帧间预测处理单元120及帧内预测处理单元126。帧间预测处理单元120包含运动估计单元及运动补偿单元(未展示)。视频编码器20还包含经配置以执行本发明中所描述的基于调色板的译码技术的各种方面的基于调色板的编码单元122。在其它实例中，视频编码器20可包含更多、更少或不同功能性组件。

[0167] 视频数据存储器101可存储待由视频编码器20的组件编码的视频数据。可(例如)从视频源18获得存储于视频数据存储器101中的视频数据。经解码图片缓冲器116可为参考图片存储器，其存储参考视频数据以供视频编码器20(例如)以帧内或帧间译码模式)编码视频数据时使用。视频数据存储器101及经解码图片缓冲器116可由多种存储器装置中的任一者形成，例如，动态随机存取存储器(DRAM)(包含同步DRAM(SDRAM))、磁阻式RAM(MRAM)、电阻式RAM(RRAM)或其它类型的存储器装置。可由同一存储器装置或独立存储器装置提供视频数据存储器101及经解码图片缓冲器116。在各种实例中，视频数据存储器101可与视频编码器20的其它组件一起在芯片上，或相对于那些组件在芯片外。

[0168] 视频编码器20可接收视频数据。视频编码器20可编码视频数据的图片的切片中的每一CTU。CTU中的每一者可与具有相等大小的明度译码树型块(CTB)及图片的对应CTB相关联。作为编码CTU的部分，块编码单元100可执行四分树分割以将CTU的CTB划分成逐渐更小的块。较小块可为CU的译码块。举例来说，块编码单元100可将与CTU相关联的CTB分割成四个相等大小的子块，将子块中的一或多者分割成四个相等大小的子子块，等等。

[0169] 视频编码器20可编码CTU的CU以产生CU的经编码表示(即，经译码CU)。作为编码CU的部分，块编码单元100可分割CU的一或多个PU中的与CU相关联的译码块。因此，每一PU可与明度预测块及对应色度预测块相关联。视频编码器20及视频解码器30可支持具有各种大

小的PU。如上文所指示，CU的大小可指代CU的明度译码块的大小，且PU的大小可指代PU的明度预测块的大小。假设特定CU的大小为2N×2N，那么视频编码器20及视频解码器30可支持用于帧内预测的2N×2N或N×N的PU大小，及用于帧间预测的2N×2N、2N×N、N×2N、N×N或类似者的对称PU大小。视频编码器20及视频解码器30还可支持用于帧间预测的2N×nU、2N×nD、nL×2N及nR×2N的PU大小的不对称分割。

[0170] 帧间预测处理单元120可通过对CU的每一PU执行帧间预测而产生用于PU的预测性数据。用于PU的预测性数据可包含PU的预测性块及PU的运动信息。取决于PU是在I切片中、P切片中还是B切片中，帧间预测单元121可针对CU的PU执行不同操作。在I切片中，所有PU经帧内预测。因此，如果PU在I切片中，那么帧间预测单元121不对PU执行帧间预测。因此，对于I模式中编码的块，经预测块是使用空间预测从同一帧内的先前经编码的相邻块而形成。

[0171] 如果PU是在P切片中，那么帧间预测处理单元120的运动估计单元可在PU的参考区域的参考图片列表(例如，“RefPicList0”)中搜索参考图片。PU的参考区域可为参考图片内含有最紧密地对应于PU的样本块的样本块的区域。帧间预测处理单元120的运动估计单元可产生参考索引，所述参考索引指示含有PU的参考区域的参考图片在RefPicList0中的位置。另外，运动估计单元可产生指示PU的译码块与相关联于参考区域的参考位置之间的空间移位的MV。举例来说，MV可为提供从当前经解码图片中的坐标到参考图片中的坐标的偏移的二维向量。运动估计单元可将参考索引及MV输出为PU的运动信息。帧间预测处理单元120的运动补偿单元可基于在由PU的运动向量指示的参考位置处的实际或内插样本而产生PU的预测性块。

[0172] 如果PU在B切片中，那么运动估计单元可针对PU执行单向预测或双向预测。为针对PU执行单向预测，运动估计单元可针对PU的参考区域搜索RefPicList0或第二参考图片列表(“RefPicList1”)中的参考图片。运动估计单元可输出以下各者作为PU的运动信息：指示含有参考区域的参考图片在RefPicList0或RefPicList1中的位置的参考索引、指示PU的预测块与相关联于参考区域的参考位置之间的空间位移的MV，及指示参考图片是在RefPicList0还是在RefPicList1中的一或多个预测方向指示符。帧间预测处理单元120的运动补偿单元可至少部分地基于由PU的运动向量指示的参考区域处的实际或内插样本而产生PU的预测性块。

[0173] 为针对PU执行双向帧间预测，运动估计单元可针对PU的参考区域在RefPicList0中搜索参考图片，且还可针对PU的另一参考区域在RefPicList1中搜索参考图片。运动估计单元可产生指示含有参考区域的参考图片在RefPicList0及RefPicList1中的位置的参考图片索引。另外，运动估计单元可产生指示与参考区域相关联的参考位置与PU的样本块之间的空间移位的MV。PU的运动信息可包含PU的参考索引及MV。运动补偿单元可至少部分地基于由PU的运动向量指示的参考区域处的实际或内插样本而产生PU的预测性块。

[0174] 根据本发明的各种实例，视频编码器20可经配置以执行基于调色板的译码。关于HEVC构架，作为一实例，基于调色板的译码技术可经配置以在CU层级使用。在其它实例中，基于调色板的视频译码技术可经配置以在PU层级使用。在其它实例中，基于调色板的译码技术可经配置以在子预测单元(子PU)层级(例如，预测单元的子块)使用。因此，本文中(贯穿本发明)所描述的在CU层级的上下文中的所有所公开的过程可另外或替代地应用于PU层级或子PU层级。然而，这些基于HEVC的实例不应被视为约束或限制本文中所描述的基于调

色板的视频译码技术，因这些技术可适用于独立地或作为其它现有或尚待开发的系统/标准的部分而工作。在这些情况下，用于调色板译码的单元可为方形块、矩形块或甚至非矩形形状的区域。

[0175] 当(例如)针对CU或PU选择基于调色板的编码模式时，基于调色板的编码单元122(例如)可执行基于调色板的解码。举例来说，基于调色板的编码单元122可经配置以产生具有指示像素值的条目的调色板，选择调色板中的像素值以表示视频数据块中的至少一些位置的像素值，及用信号发送使视频数据块的位置中的至少一些与调色板中分别对应于所选择像素值的条目相关联的信息。尽管将各种功能描述为通过基于调色板的编码单元122执行，但这些功能中的一些或全部可通过其它处理单元或不同处理单元的组合执行。

[0176] 根据本发明的方面，基于调色板的编码单元122可经配置以执行本文中所描述的用于调色板译码的技术的任何组合。

[0177] 帧内预测处理单元126可通过对PU执行帧内预测而产生用于PU的预测性数据。用于PU的预测性数据可包含PU的预测性块及各种语法元素。帧内预测处理单元126可对I切片中、P切片中及B切片中的PU执行帧内预测。

[0178] 为对PU执行帧内预测，帧内预测处理单元126可使用多个帧内预测模式，以产生用于PU的预测性数据的多个集合。帧内预测处理单元126可使用来自相邻PU的样本块的样本来产生用于PU的预测性块。对于PU、CU及CTU，假设从左到右从上到下的编码次序，那么相邻PU可在PU上方、右上方、左上方或左边。帧内预测处理单元126可使用各种数目的帧内预测模式，例如，33个方向帧内预测模式。在一些实例中，帧内预测模式的数目可取决于与PU相关联的区域的大小。

[0179] 块编码单元100可从通过帧间预测处理单元120所产生的用于PU的预测性数据或通过帧内预测处理单元126所产生的用于PU的预测性数据中选择用于CU的PU的预测性数据。在一些实例中，块编码单元100基于预测性数据集合的速率/失真量度而选择用于CU的PU的预测性数据。所选预测性数据的预测性块在本文中可被称作所选预测性块。

[0180] 残余产生单元102可基于CU的明度译码块、Cb译码块及Cr译码块及CU的PU的所选择预测性明度块、预测性Cb块及预测性Cr块而产生CU的明度残余块、Cb残余块及Cr残余块。举例来说，残余产生单元102可产生CU的残余块，以使得残余块中的每一样本具有等于CU的译码块中的样本与CU的PU的相对应所选预测性块中的相对应样本之间的差的值。

[0181] 变换处理单元104可执行四分树分割以将与CU相关联的残余块分割成与CU的TU相关联的变换块。因此，在一些实例中，TU可与明度变换块及两个色度变换块相关联。CU的TU的明度变换块及色度变换块的大小及位置可基于或可不基于CU的PU的预测块的大小及位置。被称为“残余四分树”(RQT)的四分树结构可包含与区域中的每一者相关联的节点。CU的TU可对应于RQT的叶节点。

[0182] 变换处理单元104可通过将一或多个变换应用于TU的变换块而产生CU的每一TU的变换系数块。变换处理单元104可将各种变换应用于与TU相关联的变换块。举例来说，变换处理单元104可将离散余弦变换(DCT)、定向变换或概念上类似的变换应用于变换块。在一些实例中，变换处理单元104并不将变换应用于变换块。在这些实例中，变换块可经处理为变换系数块。

[0183] 量化单元106可量化系数块中的变换系数。量化过程可降低与变换系数中的一些

或全部相关联的位深度。举例来说，在量化期间，可将n位变换系数下舍入到m位变换系数，其中n大于m。量化单元106可基于相关联于CU的量化参数(QP)值来量化与CU的TU相关联的系数块。视频编码器20可通过调整与CU相关联的QP值来调整应用于相关联于CU的系数块的量化程度。量化可引入信息丢失，因此经量化变换系数可具有比原始变换系数低的精度。

[0184] 反量化单元108及反变换处理单元110可分别将反量化及反变换应用于系数块，以从系数块重建残余块。重建单元112可将经重建的残余块添加到由块编码单元100产生的一或多个预测性块的对应样本，以产生与TU相关联的经重建的变换块。通过以此方式重建CU的每一TU的变换块，视频编码器20可重建CU的译码块。

[0185] 滤波器单元114可执行一或多个解块操作以减少与CU相关联的译码块中的块伪影。滤波器单元114可执行其它滤波操作，包括样本适应性偏移(SAO)滤波及/或适应性回路滤波(adaptive loop filtering;ALF)。经解码图片缓冲器116可在滤波器单元114对经重建译码块执行一或多个解块操作之后，存储经重建译码块。帧间预测处理单元120可使用含有经重建的译码块的参考图片来对其它图片的PU执行帧间预测。另外，帧内预测处理单元126可使用经解码图片缓冲器116中的经重建译码块来对与CU相同的图片中的其它PU执行帧内预测。

[0186] 熵编码单元118可从视频编码器20的其它功能性组件接收数据。举例来说，熵编码单元118可从量化单元106接收系数块，且可从块编码单元100接收语法元素。熵编码单元118可对数据执行一或多个熵编码操作，以产生经熵编码数据。举例来说，熵编码单元118可对数据执行上下文自适应译码操作(例如，CABAC操作)、上下文自适应可变长度译码(CAVLC)操作、可变至可变(variable-to-variable;V2V)长度译码操作、基于语法的上下文自适应二进制算术译码(SBAC)操作、概率区间分割熵(PIPE)译码操作、指数哥伦布编码操作或另一类型的熵编码操作。视频编码器20可输出包含由熵编码单元118产生的经熵编码数据的位流。举例来说，位流可包含表示用于CU的RQT的数据。

[0187] 在一些实例中，残余译码并不与调色板译码一起执行。因此，当使用调色板译码模式译码时，视频编码器20可不执行变换或量化。另外，视频编码器20可对单独使用调色板译码模式从残余数据产生的数据进行熵编码。

[0188] 根据本发明的技术中的一或多者，视频编码器20，且具体地说基于调色板的编码单元122，可执行经预测视频块的基于调色板的视频译码。如上文所描述，由视频编码器20产生的调色板可明确地经编码并发送到视频解码器30，从先前调色板条目预测，从先前像素值预测，或其组合。

[0189] 根据本发明的一或多种技术，视频编码器20可经配置以确定以调色板模式译码当前块，对第一语法元素的多个实例进行旁路模式编码以用于重建当前块，且对第一语法元素的多个实例进行旁路模式编码之后，(例如)使用CABAC译码过程对第二语法元素的多个实例进行上下文模式编码以用于重建当前块。视频编码器20可经配置以(例如)使用CABAC译码过程的旁路模式来对第一语法元素的多个实例中的任意两个实例进行旁路模式编码，而不与第二语法元素的多个实例中的任一实例的上下文模式编码交错。在一个实例中，第一语法元素包括palette_index_idc语法元素或palette_escape_val语法元素中的一者，且第二语法元素包括palette_run_msb_id_plus1语法元素。视频编码器20可经配置以在用于当前块的索引块译码部分前面对第一语法元素的多个实例进行旁路编码。

[0190] 视频编码器20可经配置以编码指示第一语法元素的实例的数目的第三语法元素，其中对第一语法元素的多个实例进行旁路模式编码包括基于第三语法元素对第一语法元素的多个实例进行旁路模式编码。视频编码器20可使用哥伦布莱斯码、指数哥伦布码、截断莱斯码、一元码或哥伦布莱斯码与指数哥伦布码的级联或前述码中的任一者的截断版本来编码第三语法元素。

[0191] 图3为说明可经配置以执行本发明的技术的实例视频解码器30的框图。出于解释的目的而提供图3，且其并不限制如本发明中广泛例示及描述的技术。出于解释的目的，本发明在HEVC译码的上下文中描述视频解码器30。然而，本发明的技术可适用于其它译码标准或方法。

[0192] 上文关于编码器20所描述的调色板译码的细节在此关于解码器30不作重复，但应理解，解码器30可相对于本文关于编码器20所描述的任何编码过程而执行互逆解码过程。

[0193] 视频解码器30表示根据本发明中描述的各种实例的可经配置以执行用于基于调色板的译码及熵译码(例如，CABAC)的技术的装置的实例。

[0194] 在图3的实例中，视频解码器30包含熵解码单元150、视频数据存储器151、块解码单元152、反量化单元154、反变换处理单元156、重建单元158、滤波器单元160，及经解码图片缓冲器162。块解码单元152包含运动补偿单元164及帧内预测处理单元166。视频解码器30还包含基于调色板的解码单元165，其经配置以执行本发明中所描述的基于调色板的译码技术的各种方面。在其它实例中，视频解码器30可包含更多、更少或不同功能性组件。

[0195] 视频数据存储器151可存储待由视频解码器30的组件解码的视频数据，例如经编码视频位流。可(例如)从计算机可读媒体16(例如，从本地视频源(例如，照相机))经由视频数据的有线或无线网络通信或者通过存取物理数据存储媒体而获得存储于视频数据存储器151中的视频数据。视频数据存储器151可形成存储来自经编码视频位流的经编码视频数据的经译码图片缓冲器(CPB)。经解码图片缓冲器162可为存储用于通过视频解码器30解码视频数据(例如，以帧内或帧间译码模式)时使用的参考视频数据的参考图片存储器。视频数据存储器151及经解码图片缓冲器162可由多种存储器装置中的任一者形成，例如，动态随机存取存储器(DRAM)(包含同步DRAM(SDRAM))、磁阻式RAM(MRAM)、电阻式RAM(RRAM)或其它类型的存储器装置。可通过相同存储器装置或独立存储器装置来提供视频数据存储器151及经解码图片缓冲器162。在各种实例中，视频数据存储器151可与视频解码器30的其它组件一起在芯片上，或相对于那些组件在芯片外。

[0196] 可由视频数据存储器151提供的经译码图片缓冲器(CPB)可接收并存储位流的经编码视频数据(例如，NAL单元)。熵解码单元150可从CPB接收经编码视频数据(例如，NAL单元)并剖析NAL单元以解码语法元素。熵解码单元150可对所述NAL单元中的经熵编码语法元素进行熵解码。块解码单元152、反量化单元154、反变换处理单元156、重建单元158及滤波器单元160可基于从位流提取的语法元素而产生经解码视频数据。

[0197] 视频解码器30可经配置以执行与本文中所描述的视频编码器20的所述过程大体上互逆的过程。类似地，视频编码器20可经配置以执行与本文中描述的视频解码器30的所述过程大体上互逆的过程。举例来说，视频解码器30可经配置以解码位流中的经编码语法元素的公开内容同样必定公开视频编码器20可经配置以将语法元素编码到位流中。

[0198] 作为另一实例，熵解码单元150可经配置以执行与本文中所描述的熵编码单元118

的所述过程大体上互逆的过程。根据本发明的方面，熵解码单元150可经配置以对通过熵编码单元118产生的任何码字进行熵解码。举例来说，熵解码单元150可经配置以对均匀的及不均匀第k阶截断指数哥伦布(TEGk)编码的值(例如用于CU的二进制调色板预测向量及/或调色板映射)进行熵解码。作为另一实例，熵解码单元150可经配置以熵解码第k阶指数哥伦布(EGk)码字、第k阶截断指数哥伦布(TEGk)码字、第k阶不均匀的截断指数哥伦布(TEGk)码字或其任何组合。

[0199] 位流的NAL单元可包含经译码切片NAL单元。作为解码位流的部分，熵解码单元150可从经译码切片NAL单元提取语法元素且对所述语法元素进行熵解码。经译码切片中的每一者可包含切片标头及切片数据。切片标头可含有关于切片的语法元素。切片标头中的语法元素可包含识别与含有切片的图片相关联的PPS的语法元素。

[0200] 除解码来自位流的语法元素以外，视频解码器30可对未经分割的CU执行重建操作。为了对未分割的CU执行重建操作，视频解码器30可对CU的每一TU执行重建操作。通过针对CU的每一TU执行重建操作，视频解码器30可重建CU的残余块。

[0201] 作为对CU的TU执行重建操作的部分，反量化单元154可对与TU相关联的系数块进行反量化(即，解量化)。反量化单元154可使用与TU的CU相关联的QP值来确定反量化单元154应用的量化程度及同样地反量化程度。也就是说，可通过调整在量化变换系数时使用的QP值来控制压缩比，即，用于表示原始序列及经压缩序列的位数目的比率。压缩比还可取决于所应用的熵译码的方法。

[0202] 在反量化单元154反量化系数块之后，反变换处理单元156可将一或多个反变换应用于系数块以便产生与TU相关联的残余块。举例来说，反变换处理单元156可将反DCT、反整数变换、反Karhunen-Loeve变换(KLT)、反旋转变换、反定向变换或另一反变换应用于系数块。

[0203] 如果使用帧内预测来编码PU，那么帧内预测处理单元166可执行帧内预测以产生PU的预测性块。帧内预测处理单元166可使用帧内预测模式以基于空间相邻的PU的预测块而产生PU的预测性明度块、预测性Cb块及预测性Cr块。帧内预测处理单元166可基于从位流解码的一或多个语法元素而确定用于PU的帧内预测模式。

[0204] 块解码单元152可基于从位流提取的语法元素建构第一参考图片列表(RefPicList0)及第二参考图片列表(RefPicList1)。此外，如果使用帧间预测来编码PU，那么熵解码单元150可提取用于PU的运动信息。运动补偿单元164可基于PU的运动信息而确定PU的一或多个参考区域。运动补偿单元164可基于PU的一或多个参考块处的样本块产生PU的预测性明度块、预测性Cb块及预测性Cr块。

[0205] 重建单元158可在适当时使用与CU的TU相关联的明度变换块、Cb变换块及Cr变换块以及CU的PU的预测性明度块、预测性Cb块及预测性Cr块(即，帧内预测数据或帧间预测数据)来重建CU的明度译码块、Cb译码块以及Cr译码块。举例来说，重建单元158可将明度变换块、Cb变换块及Cr变换块的样本添加到预测性明度块、预测性Cb块及预测性Cr块的对应样本以重建CU的明度译码块、Cb译码块及Cr译码块。

[0206] 滤波器单元160可执行解块操作以减少与CU的明度译码块、Cb译码块及Cr译码块相关联的块伪影。视频解码器30可将CU的明度译码块、Cb译码块及Cr译码块存储于经解码图片缓冲器162中。经解码图片缓冲器162可提供参考图片以用于后续运动补偿、帧内预测

及在显示装置(例如图1的显示装置32)上的呈现。举例来说,视频解码器30可基于经解码图片缓冲器162中的明度块、Cb块及Cr块对其它CU的PU执行帧内预测操作或帧间预测操作。

[0207] 根据本发明的各种实例,视频解码器30可经配置以执行基于调色板的译码。当选定基于调色板的解码模式(例如)以用于CU或PU时,基于调色板的解码单元165(例如)可执行基于调色板的解码。举例来说,基于调色板的解码单元165可经配置以产生具有指示像素值的条目的调色板,接收使视频数据块中的至少一些像素位置与调色板中的条目相关联的信息,基于信息选择调色板中的像素值,及基于调色板中的选定像素值重建块的像素值。尽管各种功能经描述为通过基于调色板的解码单元165执行,但这些功能中的一些或全部可通过其它处理单元或不同处理单元的组合来执行。

[0208] 基于调色板的解码单元165可接收调色板译码模式信息,并在调色板译码模式信息指示调色板译码模式适用于块时执行上述操作。在调色板译码模式信息指示调色板译码模式不适用于块时,或在其它模式信息指示不同模式的使用时,基于调色板的解码单元165使用非基于调色板的译码模式(例如,HEVC帧间预测译码模式或HEVC帧内预测译码模式)解码视频数据块。视频数据的块可为(例如)根据HEVC译码过程产生的CU或PU。基于调色板的译码模式可包括多个不同基于调色板的译码模式中的一者,或可存在单一基于调色板的译码模式。

[0209] 根据本发明的方面,基于调色板的解码单元165可经配置以执行本文中所描述的用于调色板译码的技术的任何组合。上文关于编码器20所描述的调色板译码的细节在此关于解码器30不重复,但应理解,解码器30可相对于本文关于编码器20所描述的任何基于调色板的编码过程而执行互逆的基于调色板的解码过程。

[0210] 视频解码器30可经配置以确定当前块以调色板模式经译码,对第一语法元素的多个实例进行旁路模式解码以用于(例如)使用CABAC译码过程的旁路模式来重建当前块,且在对第一语法元素的多个实例进行旁路模式解码之后,对第二语法元素的多个实例进行上下文模式解码以用于(例如)使用CABAC译码过程重建当前块。视频解码器30可对第一语法元素的多个实例中的任何两个实例进行旁路模式解码,而不与第二语法元素的多个实例中的任一个实例的上下文模式解码相交错。在一些实例中,第一语法元素包括palette_index_idc语法元素或palette_escape_val语法元素中的一者,且第二语法元素包括palette_run_msb_id_plus1语法元素。视频解码器30可在用于当前块的索引块译码部分前面对第一语法元素的多个实例进行旁路解码。

[0211] 视频解码器30可解码指示第一语法元素的实例的数目的第三语法元素,其中对第一语法元素的多个实例进行旁路模式解码包括基于第三语法元素对第一语法元素的多个实例进行旁路模式解码。视频解码器30可使用哥伦布莱斯码、指数哥伦布码、截断莱斯码、一元码或哥伦布莱斯码与指数哥伦布码的级联或前述码中的任一者的截断版本来解码第三语法元素。

[0212] 图4为说明符合本发明的技术的确定用于译码视频数据的调色板的实例的概念图。图4的实例包含具有与第一调色板184相关联的第一PAL(调色板)译码单元(CU)180及与第二调色板192相关联的第二PAL CU 188的图片178。如下文将较详细描述且根据本发明的技术,第二调色板192是基于第一调色板184。图片178还包含通过帧内预测译码模式译码的块196及通过帧间预测译码模式译码的块200。

[0213] 出于解释的目的，在视频编码器20(图1及图2)及视频解码器30(图1及图3)的上下文中并关于HEVC视频译码标准描述图4的技术。然而，应理解本发明的技术不限于此方式，且可通过其它视频译码处理器及/或装置应用于其它视频译码过程及/或标准中。

[0214] 大体来说，调色板指代对于当前经译码的CU(在图4的实例中为CU 188)来说为主要的及/或代表性的多个像素值。第一调色板184(还可被称作索引184)及第二调色板192(还可被称作索引192)经展示为包含多个调色板(还可被称作多个索引)。在一些实例中，根据本发明的方面，视频译码器(例如视频编码器20或视频解码器30)可针对CU的每一色彩分量单独地译码调色板(例如，索引)。举例来说，视频编码器20可编码用于CU的明度(Y)分量的调色板、用于CU的色度(U)分量的另一调色板及用于CU的色度(V)分量的又一调色板。在此实例中，Y调色板的条目可表示CU的像素的Y值，U调色板的条目可表示CU的像素的U值，且V调色板的条目可表示CU的像素的V值。

[0215] 在其它实例中，视频编码器20可编码用于CU的全部色彩分量的单一调色板。在此实例中，视频编码器20可编码具有为三重值(包含Yi、Ui及Vi)的第i个条目的调色板。在这种情况下，调色板包含用于像素的分量中的每一者的值。因此，作为具有多个独立调色板的调色板集合的调色板184及192的表示仅为一个实例且不意欲为限制性的。

[0216] 在图4的实例中，第一调色板184包含分别具有条目索引值1、条目索引值2及条目索引值3的三个条目202到206。第一调色板184使得索引值(例如，第一调色板184的左列中展示的值)与像素值相关。举例来说，如图4中所展示，第一调色板184中的一者使索引值1、2及3分别与像素值A、B及C相关。如本文中所描述，视频译码器(例如，视频编码器20或视频解码器30)可使用基于调色板的译码对使用索引1到3(还可表达为索引值1到3)的块的像素进行译码，而不是译码第一CU 180的实际像素值。也就是说，对于第一CU 180的每一像素位置，视频编码器20可编码用于像素的索引值，其中索引值与第一调色板184中的一或多者中的像素值相关联。视频解码器30可从位流获得索引值并使用索引值及第一调色板184中的一或多者重建像素值。因此，第一调色板184是通过视频编码器20在经编码视频数据位流中发射以供视频解码器30在基于调色板的解码时使用。

[0217] 在一些实例中，视频编码器20及视频解码器30可基于第一调色板184来确定第二调色板192。举例来说，视频编码器20及/或视频解码器30可定位确定预测性调色板(在此实例中，第一调色板184)所依据的一或多个块。在一些实例(例如图4中所说明的实例)中，视频编码器20及/或视频解码器30可在确定第二CU 188的预测性调色板时定位先前经译码的CU，例如左侧相邻CU(第一CU 180)。

[0218] 在图4的实例中，第二调色板192包含分别具有条目索引值1、条目索引值2及条目索引值3的三个条目208到212。第二调色板192使索引值(例如，第一调色板192的左列中展示的值)与像素值相关。举例来说，如图4中所展示，第二调色板192中的一者使索引值1、2及3分别与像素值A、B及D相关。在此实例中，视频编码器20可译码指示第一调色板184的哪些条目包含于第二调色板192中的一或多个语法元素。在图4的实例中，一或多个语法元素作为向量216说明。向量216具有多个相关联二进制(或位)，其中每一二进制指示与所述二进制相关联的调色板预测符是否用于预测当前调色板的条目。举例来说，向量216指示第一调色板184中的前两个条目(202及204)包含于第二调色板192中(向量216中的值“1”)，而第一调色板184中的第三条目不包含于第二调色板192中(向量216中的值“0”)。在图4的实例中，

向量为布林(Boolean)向量。

[0219] 在一些实例中，视频编码器20及视频解码器30可在执行调色板预测时确定调色板预测符列表(其还可被称作调色板预测符表)。调色板预测符列表可包含来自用以预测用于译码当前块的调色板的一或多个条目的一或多个相邻块的调色板的条目。视频编码器20及视频解码器30可以相同方式来建构列表。视频编码器20及视频解码器30可译码数据(例如向量216)以指示调色板预测符列表的哪些条目将包含于用于译码当前块的调色板中。

[0220] 图5为说明符合本发明的技术的确定至用于像素的块的调色板的索引的实例的概念图。举例来说，图5包含索引块240(还可被称作映射240或索引映射240)，所述索引块包含使与索引值相关联的像素的对应位置相关于调色板244的条目的索引值(例如，索引值1、2及3)。

[0221] 虽然索引块240在图5的实例中说明为包含每一像素位置的索引值，但应理解在其它实例中，并非全部像素位置可与使像素值与调色板244的条目相关的索引值相关联。也就是说，如上文所提到，在一些实例中，如果像素值并不包含于调色板244中，那么视频编码器20可编码(且视频解码器30可从经编码位流获得)用于索引块240中的位置的实际像素值(或其经量化版本)的指示。

[0222] 在一些实例中，视频编码器20及视频解码器30可经配置以译码指示哪些像素位置与哪些索引值相关联的额外映射。举例来说，假设索引块240中的(i,j)条目对应于CU的(i,j)位置。视频编码器20可编码用于指示条目是否具有相关联索引值的索引块(即，每一像素位置)的每一条目的一或多个语法元素。举例来说，视频编码器20可编码具有值为一的旗标以指示在CU中的(i,j)位置处的像素值为调色板244中的值中的一者。

[0223] 在此实例中，视频编码器20还可编码调色板(图5的实例中展示为244)。在调色板244包含单个条目及相关联像素值的情况下，视频编码器20可跳过索引值的用信号发送。视频编码器20可编码具有零值的旗标以指示在CU中的(i,j)位置处的像素值并非调色板244中的值中的一者。在此实例中，视频编码器20还可编码像素值的指示以供视频解码器30用于重构像素值。在一些情况下，可以有损方式译码像素值。

[0224] CU的一个位置中的像素的值可提供CU的其它位置中的一或多个其它像素的值的指示。举例来说，可存在CU的相邻像素位置将具有相同像素值或可映射到相同索引值(在有损译码情况下，其中多于一个像素值可映射到单个索引值)的相对高概率。

[0225] 因此，视频编码器20可编码指示按给定扫描次序的具有相同像素值或索引值的多个连续像素或索引值的一或多个语法元素。如上文所提到，相同值像素或索引值的字串可在本文中称为游程。在出于说明的目的的实例中，如果按给定扫描次序的两个连续像素或索引具有不同值，那么游程等于零。如果按给定扫描次序的两个连续像素或索引具有相同值但按扫描次序的第三像素或索引具有不同值，那么游程等于一。对于具有相同值的三个连续索引或像素，游程为二，等等。视频解码器30可从经编码的位流获得指示游程的语法元素，并可使用数据以确定具有相同像素或索引值的连续位置的数目。

[0226] 在根据本发明的技术的一些实例中，熵编码单元118及熵解码单元150可经配置以对索引块240进行熵译码。举例来说，熵编码单元118及熵解码单元150可经配置以对游程长度(例如，游程长度值或游程长度码)及/或与调色板模式中的索引块相关的二进制调色板预测向量进行熵译码。

[0227] 图6为符合本发明的技术的说明确定最大上方复制游程长度、假设光栅扫描次序的实例的概念图。在图6的实例中，如果由虚线280涵盖的像素中无一者经译码为逸出样本，那么最大可能游程长度为35(即，无阴影的像素位置的数目)。如果在虚线280内的像素中的一或多者经译码为逸出样本，假设标记为逸出像素(具有“X”的像素位置)的像素为在虚线280内的在扫描次序中的第一逸出像素，那么最大可能经译码的上方复制游程长度为五。

[0228] 在一些实例中，视频解码器30可仅针对在虚线280内的像素确定游程模式(例如，其中像素经译码的调色板模式)。因此，在最差的情况下，视频解码器30为BlockWidth-1像素做出确定。在一些实例中，视频解码器30可经配置以实施关于像素(为其检查游程模式)的最大数目的某些限制。举例来说，如果像素与当前像素处于相同的行，那么视频解码器30可仅检查在虚线280内的像素。视频解码器30可推断在虚线280内的所有其它像素未经译码为逸出样品。图6中的实例假设光栅扫描次序。然而，所述技术可应用于其它扫描次序，例如竖直、水平横移，及竖直横移。

[0229] 根据本发明的实例，如果当前游程模式为‘上方复制’，那么用于当前像素的游程长度的上下文可取决于相对于当前像素的上方相邻像素的索引的索引值。在此实例中，如果相对于当前像素的上方相邻像素在当前CU的外侧，那么视频解码器假设相对应的索引等于预定义常数k。在一些实例中，k＝0。

[0230] 在熵译码期间，熵编码器或解码器可将待编码或解码的符号的位安置为一或多个二进制。二进制可指示符号的值是否等于零。熵译码器或熵解码器可使用二进制的值来调整熵译码过程。在一些实例中，熵编码器或熵解码器还可使用二进制来指示值是否大于特定值，例如，大于零、大于一，等等。

[0231] 在一些实例中，如果当前模式为‘上方复制’，那么游程长度码字的第一二进制基于相对于当前样本(例如，像素)的上方相邻样本(例如，像素)是否等于0而选择两个候选CABAC上下文中的一者。

[0232] 作为另一实例，如果当前模式为‘先前复本’，那么游程长度码字的第一二进制基于索引值是否等于0、等于1、等于2或大于2来选择四个候选CABAC上下文中的一者。

[0233] 图8为说明符合本发明的技术的用于解码视频数据的实例过程的流程图。出于说明的目的，通常将图8的过程描述为由视频解码器30执行，但多种其它处理器也可执行图8中所展示的过程。在一些实例中，块解码单元152、基于调色板的解码单元165及/或熵解码单元150可执行图8中展示的一或多个过程。

[0234] 在图8的实例中，视频解码器30可经配置以从经编码视频位流接收图片的经调色板模式编码的视频数据块(800)。视频解码器30可经配置以从经编码视频位流接收用于经调色板模式编码的视频数据块的经编码调色板模式信息(802)。在一些实例中，经编码调色板模式信息可包含第一语法元素的多个实例及不同于第一语法元素的多个语法元素。举例来说，第一语法元素可包含palette_index_idc或palette_escape_val，且不同于第一语法元素的多个语法元素可包含palette_run_msb_id_plus1语法元素。作为另一实例，第一语法元素可为对调色板条目的阵列的索引的指示，或第一语法元素可指定用于对应于逸出样本的色彩分量的经量化的逸出经译码样本值。不同于第一语法元素的多个语法元素可包含指定表示游程长度的变量的二进制表示中的最高有效位的索引的语法元素及指定游程类型模式的语法元素。

[0235] 作为另一实例，不同于第一语法元素的多个语法元素可为不同于第一语法元素的任何语法元素及所有语法元素。如本文中关于一些实例所描述，不同于第一语法元素的多个语法元素还可不同于第二语法元素、第三语法元素及/或第四语法元素。在这些实例中，不同于第一语法元素、第二语法元素、第三语法元素及第四语法元素的多个语法元素可为不同于第一语法元素、第二语法元素、第三语法元素及/或第四语法元素的任何语法元素及所有语法元素。在一些实例中，不同于第一语法元素的多个语法元素可为不被旁路模式解码及/或不被旁路模式解码的任何语法元素及所有语法元素。

[0236] 视频解码器30可经配置以在使用上下文模式解码不同于第一语法元素的多个语法元素之前使用旁路模式(例如，CABAC译码过程的旁路模式)解码第一语法元素的多个实例(804)。视频解码器30可经配置以在使用旁路模式解码第一语法元素的多个实例之后使用上下文模式(例如，常规CABAC模式(而不是旁路模式))来解码不同于第一语法元素的多个语法元素(806)。在一些实例中，第一语法元素的多个实例包含用于经调色板模式编码的视频数据块的第一语法元素的所有实例。在这些实例中，在解码任何后续数据(例如，不同于第一语法元素的多个语法元素)之前使用旁路模式解码第一语法元素的所有实例。以其它方式陈述，视频解码器30可经配置以在使用旁路模式解码用于经调色板模式编码的视频数据块的第一语法元素的所有实例之后使用上下文模式解码不同于第一语法元素的多个语法元素。

[0237] 视频解码器30可经配置以使用第一语法元素的经解码的多个实例及不同于第一语法元素的经解码的多个语法元素来解码经调色板模式编码的视频数据块(808)。在一些实例中，第一语法元素的多个实例被分组在一起，以使得当解码经调色板模式编码的视频数据块时在旁路模式与上下文模式之间的切换减少。

[0238] 在一些实例中，经编码调色板模式信息可包含指示第一语法元素的实例的数目(例如，指示存在多少第一语法元素的实例用于经调色板模式编码的视频数据块)的第二语法元素。不同于第一语法元素的多个语法元素还可不同于第二语法元素。在这些实例中，视频解码器30可经配置以在解码不同于第一语法元素及第二语法元素的多个语法元素之前使用旁路模式解码第二语法元素。在一些实例中，无第二语法元素的实例在用于经调色板模式编码的视频数据块的第一语法元素的任何两个实例之间交错。在一些实例中，视频解码器30可经配置以在解码等于由第二语法元素指示的数目的第一语法元素的多个实例之后，确定在经编码视频位流中在第一语法元素的实例的数目之后的后续数据对应于不同于第一语法元素及第二语法元素的多个语法元素。在一些实例中，视频解码器30可经配置以使用截断莱斯码与指数哥伦布码的级联来解码第二语法元素。

[0239] 在一些实例中，经编码调色板模式信息可包含第三语法元素及第四语法元素。在这些实例中，视频解码器30可经配置以解码第三语法元素从而确定对应于第三语法元素的值指示经调色板模式编码的视频数据块是否包含逸出像素。视频解码器30可经配置以解码第四语法元素从而确定对应于第四语法元素的值指示调色板大小。视频解码器30可经配置以在使用旁路模式解码第一语法元素及第二语法元素的多个实例之后，基于所确定的分别对应于第三语法元素及第四语法元素的值使用上下文模式解码不同于第一语法元素及第二语法元素的多个语法元素。

[0240] 在一些实例中，经编码调色板模式信息可包含另一语法元素，且视频解码器30可

经配置以解码此其它语法元素以确定对应于此其它语法元素的值，此其它语法元素指定调色板索引针对经调色板模式编码的视频数据块所具有的不同值的数目。视频解码器30可经配置以在使用旁路模式解码第一语法元素及第二语法元素的多个实例之后基于对应于此其它语法元素的所确定的值使用上下文模式解码不同于第一语法元素及第二语法元素的多个语法元素。

[0241] 在一些实例中，经编码调色板模式信息可包含另一语法元素，且视频解码器30可经配置以解码此其它语法元素以确定对应于此其它语法元素的值指示用于经调色板模式编码的视频数据块的palette_run_type_flag[xC][yC]的语法元素的最末实例。

[0242] 在一些实例中，视频解码器30可经配置以确定经编码视频数据块具有一或多个逸出样本。在这些实例中，视频解码器30可经配置以解码经编码视频数据块中的一或多个逸出样本中的最末逸出样本。视频解码器30可经配置以推断适用于经编码视频数据块的在最末逸出样本之后的样本的索引值。视频解码器30可经配置以使用用于最末逸出样本之后的样本的每一样本的所推断索引值来解码经编码视频数据块的最末逸出样本之后的样本。

[0243] 在一些实例中，视频解码器30可经配置以确定所接收的调色板索引的数目。在这些实例中，视频解码器30可经配置以基于所接收的调色板索引的数目及第一语法元素的实例的数目而确定剩余调色板索引的数目。视频解码器30可经配置以基于所接收的调色板索引的数目及第一语法元素的实例的数目而确定用于经编码视频数据块的最大可能游程值。在一些实例中，视频解码器30可经配置以根据nCbS*nCbS-scanPos-1-paletteIndicesLeft确定用于经编码视频数据块的最大可能游程值，其中nCbS指定经编码视频数据块的大小，scanPos指定扫描位置，且paletteIndicesLeft指定剩余调色板索引的数目。

[0244] 图9为说明符合本发明的技术的用于编码视频数据的实例过程的流程图。出于说明的目的，通常将图9的过程描述为由视频编码器20执行，但多种其它处理器也可执行图9中所展示的过程。在一些实例中，块编码单元100、基于调色板的编码单元122及/或熵编码单元118可执行图9中展示的一或多个过程。

[0245] 在图9的实例中，视频编码器20可经配置以确定视频数据块将以调色板模式编码(900)。视频编码器20可经配置以使用调色板模式将视频数据块编码成经编码位流(902)。在一些实例中，视频编码器20可经配置以产生用于视频数据块的调色板模式信息(904)。调色板模式信息可包含第一语法元素的多个实例及不同于第一语法元素的多个语法元素。举例来说，第一语法元素可包含palette_index_idc或palette_escape_val，且不同于第一语法元素的多个语法元素可包含palette_run_msb_id_plus1语法元素。作为另一实例，第一语法元素可为对调色板条目的阵列的索引的指示，或第一语法元素可指定用于对应于逸出样本的色彩分量的经量化的逸出经译码样本值。不同于第一语法元素的多个语法元素可包含指定表示游程长度的变量的二进制表示中的最高有效位的索引的语法元素及指定游程类型模式的语法元素。

[0246] 作为另一实例，不同于第一语法元素的多个语法元素可为不同于第一语法元素的任何语法元素及所有语法元素。如本文中关于一些实例所描述，不同于第一语法元素的多个语法元素还可不同于第二语法元素、第三语法元素及/或第四语法元素。在这些实例中，不同于第一语法元素、第二语法元素、第三语法元素及第四语法元素的多个语法元素可为不同于第一语法元素、第二语法元素、第三语法元素及/或第四语法元素的任何语法元素及

所有语法元素。在一些实例中，不同于第一语法元素的多个语法元素可为不被旁路模式编码及/或不被旁路模式编码的任何语法元素及所有语法元素。

[0247] 视频编码器20可经配置以在使用上下文模式将不同于第一语法元素的多个语法元素编码成经编码位流之前使用旁路模式(例如，CABAC译码过程的旁路模式)来将第一语法元素的多个实例编码成经编码位流(906)。视频编码器20可经配置以在使用旁路模式将第一语法元素的多个实例编码成经编码位流之后使用上下文模式(例如，常规的基于CABAC上下文的模式)将不同于第一语法元素的多个语法元素编码成经编码位流(908)。在一些实例中，第一语法元素的多个实例被分组在一起，以使得当编码经调色板模式编码的视频数据块时在旁路模式与上下文模式之间的切换减少。

[0248] 在一些实例中，第一语法元素的多个实例包含用于视频数据块的第一语法元素的所有实例。在这些实例中，在编码任何后续数据(例如，不同于第一语法元素的多个语法元素)之前使用旁路模式编码第一语法元素的所有实例。以其它方式陈述，视频编码器20可经配置以在使用旁路模式编码用于视频数据块的第一语法元素的所有实例之后使用上下文模式编码不同于第一语法元素的多个语法元素。

[0249] 在一些实例中，调色板模式信息可包含指示第一语法元素的实例的数目的第二语法元素(例如，指示存在用于视频数据块的第一语法元素的实例的数目)。不同于第一语法元素的多个语法元素还可不同于第二语法元素。在这些实例中，视频编码器20可经配置以在编码不同于第一语法元素及第二语法元素的多个语法元素之前使用旁路模式将第二语法元素编码成经编码位流。在一些实例中，视频编码器20可经配置以编码第一语法元素的多个实例，以使得无第二语法元素的实例在用于经编码位流中的经调色板模式编码的视频数据块的第一语法元素的任何两个实例之间交错。在一些实例中，视频编码器20可经配置以在经编码位流中的第一语法元素的经编码的多个实例之后将第二语法元素编码成经编码位流。举例来说，视频编码器20可经配置以首先编码第一语法元素的所有实例，且随后将第二语法元素编码成经编码位流。在一些实例中，视频编码器20可经配置以使用截断莱斯码与指数哥伦布码的级联来编码第二语法元素。

[0250] 在一些实例中，调色板模式信息可包含第三语法元素及第四语法元素。在这些实例中，视频编码器20可经配置以将对应于第三语法元素的值编码成经编码位流，所述第三语法元素指示视频数据块是否包含逸出像素。视频编码器20可经配置以使对应于第四语法元素的指示调色板大小的值成为经编码位流。在一些实例中，调色板模式信息可包含另一语法元素，且视频编码器20可经配置以将对应于指定调色板索引针对视频数据块所具有的不同值的数目的此其它语法元素的值编码成经编码位流。

[0251] 在一些实例中，经编码调色板模式信息可包含另一语法元素，且视频编码器20可经配置以编码对应于此其它语法元素的值，此其它语法元素指示用于视频数据块的palette_run_type_flag[xC][yC]的语法元素的最末实例。

[0252] 在一些实例中，视频编码器20可经配置以编码视频数据块中的一或多个逸出样本中的最末逸出样本。在这些实例中，视频编码器20可经配置以推断适用于视频数据块的在最末逸出样本之后的样本的索引值。视频编码器20可经配置以使用用于最末逸出样本之后的样本中的每一样本的所推断索引值来编码视频数据块的在最末逸出样本之后的样本。

[0253] 应理解，本文所描述的所有技术可单独地或以组合方式使用。举例来说，视频编码

器20及/或其一或多个组件及视频解码器30及/或其一或多个组件可以任何组合执行本发明中所描述的技术。

[0254] 应认识到，取决于实例，本文中所描述的技术中的任一者的某些动作或事件可以不同序列执行、可添加、合并或完全省略(例如，并非所有所描述动作或事件对于所述技术的实是必要的)。此外，在某些实例中，可(例如)经由多线程处理、中断处理或多个处理器同时而非依序执行动作或事件。另外，尽管出于清晰的目的，本发明的某些方面被描述为由单一模块或单元执行，但应理解，本发明的技术可通过与视频译码器相关联的单元或模块的组合来执行。

[0255] 出于说明的目的，已关于发展中HEVC标准而描述本发明的某些方面。然而，本发明中所描述的技术可适用于其它视频译码过程，包含尚未开发的其它标准或专属视频译码过程。

[0256] 上文所描述的技术可通过视频编码器20(图1及2)及/或视频解码器30(图1及3)执行，两者通常可称作视频译码器。同样地，视频译码在适用时可指代视频编码或视频解码。

[0257] 根据本发明，术语“或”可解译为“及/或”，其中上下文并不以其它方式指示。另外，虽然例如“一或多个”或“至少一个”或其类似者的短语可已被用于本文中所公开的一些特征(但并非其它者)；并未使用此语言的特征可解释为具有暗示上下文并不以其它方式指示的这种含义。

[0258] 虽然在上文所描述技术的各种方面的特定组合，但提供这些组合仅为了说明本发明中描述的技术的实例。因此，本发明的技术不应限于这些实例组合且可涵盖本发明中描述的技术的各种方面的任何可设想组合。

[0259] 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，那么所述功能可作为一或多个指令或码而存储于计算机可读媒体上或经由计算机可读媒体发射，且由基于硬件的处理单元执行。计算机可读媒体可包含计算机可读存储媒体(其对应于有形媒体(例如数据存储媒体))，或包含有助于(例如)根据通信协议将计算机程序从一处传送到另一处的任何媒体的通信媒体。以此方式，计算机可读媒体大体可对应于(1)非暂时性的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用于实施本发明中所描述的技术的指令、程序代码及/或数据结构的任何可用媒体。计算机程序产品可包含计算机可读媒体。

[0260] 借助于实例而非限制，此计算机可读存储媒体可包括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、闪速存储器，或可用以存储呈指令或数据结构形式的所要程序代码且可由计算机存取的任何其它媒体。并且，将任何连接恰当地称为计算机可读媒体。举例来说，如果使用同轴电缆、光纤电缆、双绞线、数字用户线(DSL)或例如红外线、无线电及微波的无线技术从网站、服务器或其它远程源发射指令，那么同轴电缆、光纤电缆、双绞线、DSL或例如红外线、无线电及微波的无线技术包含于媒体的定义中。然而，应理解，计算机可读存储媒体及数据存储媒体不包含连接、载波、信号或其它暂时性媒体，而实际上有关于非暂时性有形存储媒体。如本文所使用的磁盘及光盘包含紧密光盘(CD)、激光光盘、光学光盘、数字多功能光盘(DVD)、软盘及蓝光光盘，其中磁盘通常以磁性方式再生数据，而光盘用激光以光学方式再生数据。以上各物的组合也应包含于计

算机可读媒体的范围内。

[0261] 可通过一或多个处理器来执行指令，所述一或多个处理器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)、现场可编程门阵列(FPGA)或其它等效的集成或离散逻辑电路。因此，如本文中所使用的术语“处理器”可指前述结构或适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，可在经配置用于编码及解码的专用硬件及/或软件模块内提供本文中所描述的功能性，或将本文中所描述的功能性并入于组合式编解码器中。并且，所述技术可完全实施于一或多个电路或逻辑元件中。

[0262] 本发明的技术可以多种装置或设备予以实施，所述装置或设备包含无线手机、集成电路(IC)或IC的集合(例如，芯片组)。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公开技术的装置的功能性方面，但未必要求由不同硬件单元来实现。实情为，如上文所描述，各种单元可组合于编解码器硬件单元中，或结合合适软件及/或固件通过互操作性硬件单元(包含如上所述的一或多个处理器)的集合来提供。

[0263] 本文中已描述各种实例。涵盖所描述的系统、操作、功能或实例的任何组合。这些及其它实例处于以下权利要求书的范围内。

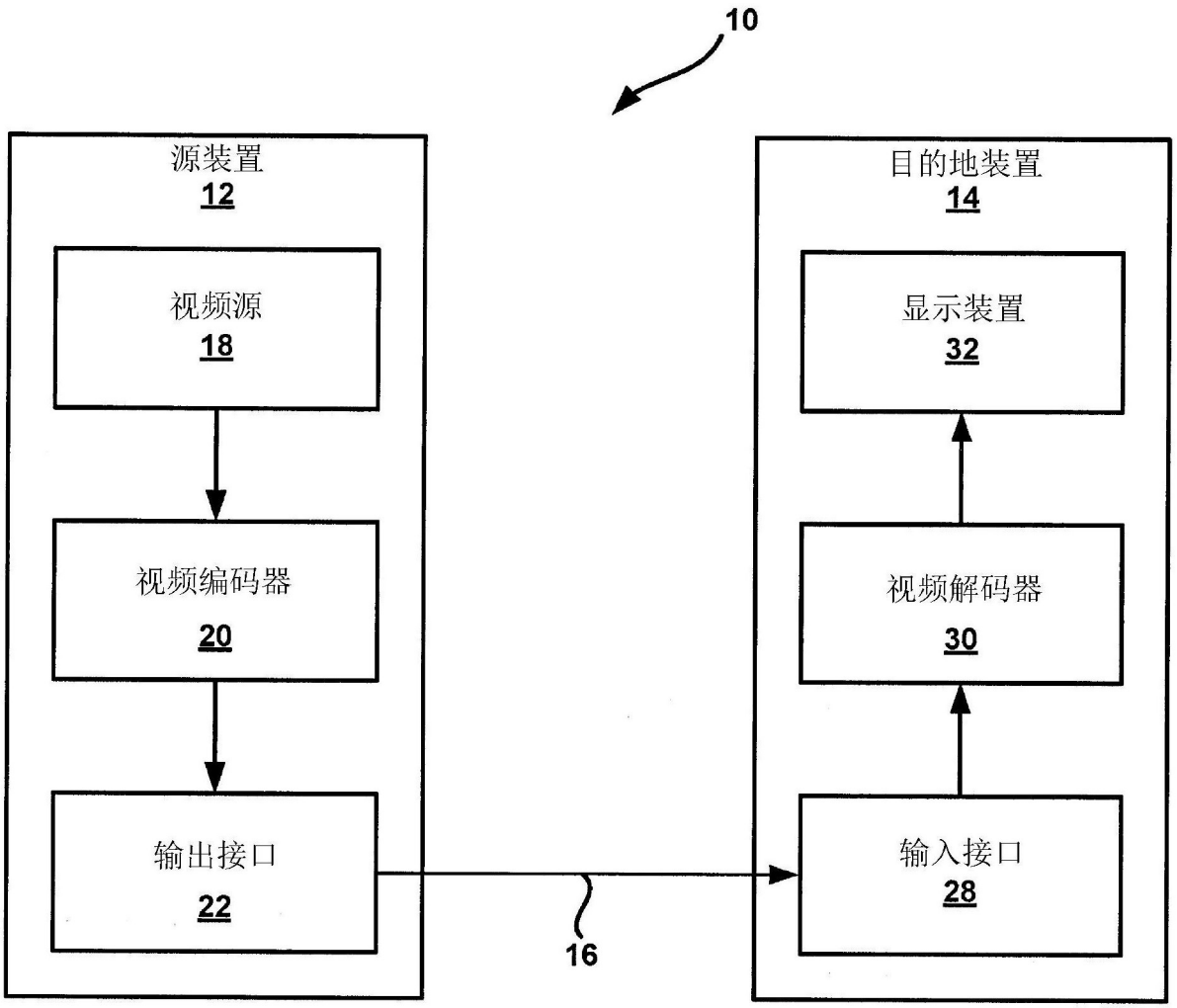

10
源装置
12
视频源
18
视频编码器
20
输出接口
22
16
目的地装置
14
显示装置
32
视频解码器
30
输入接口
28

图1

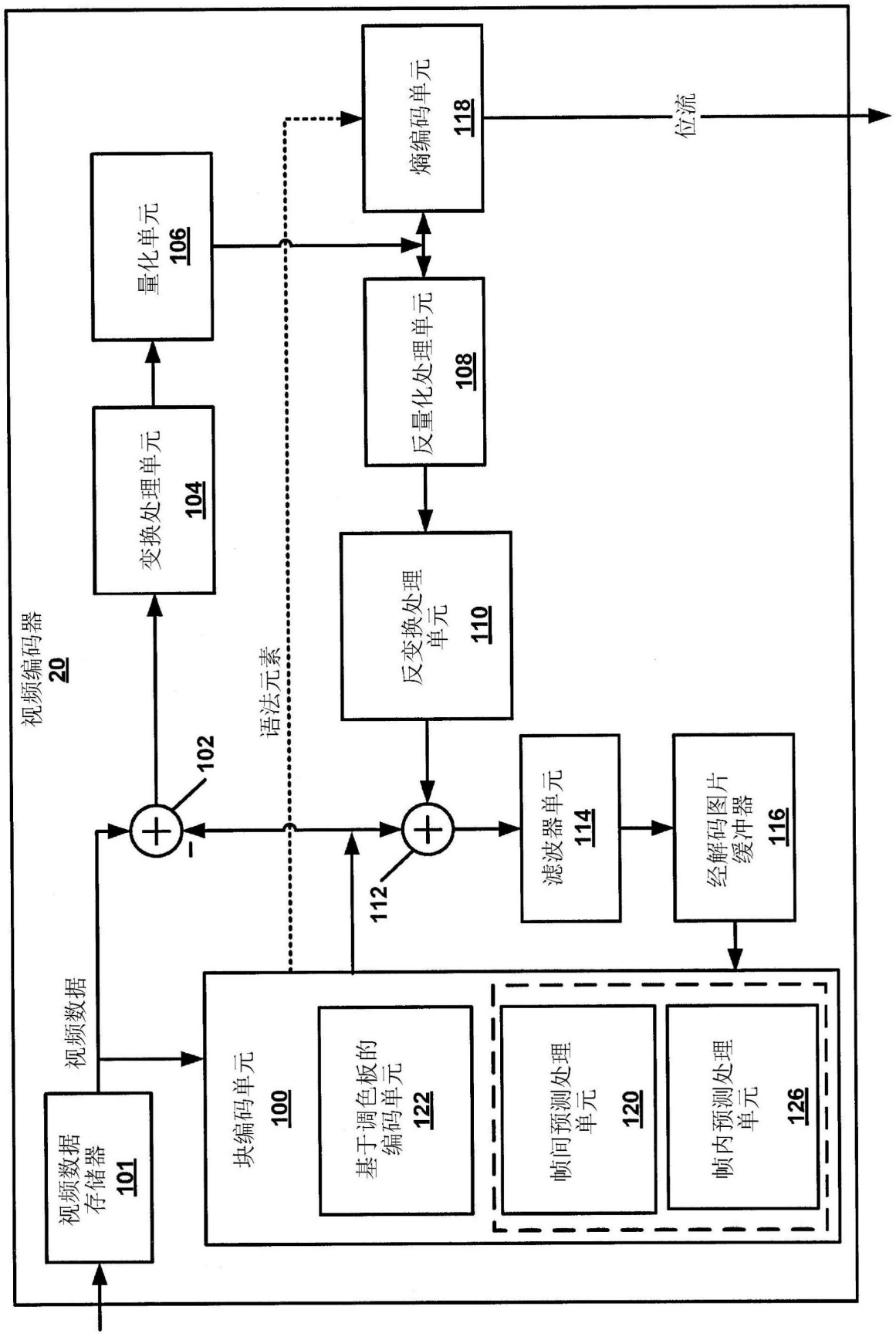

视频编码器
20
视频数据
存储器
101
视频数据
块编码单元
100
基于调色板的
编码单元
122
帧间预测处理
单元
120
帧内预测处理
单元
126
102
变换处理单元
104
量化单元
106
熵编码单元
118
位流
反量化处理单元
108
反变换处理
单元
110
112
滤波器单元
114
经解码图片
缓冲器
116
语法元素

图2

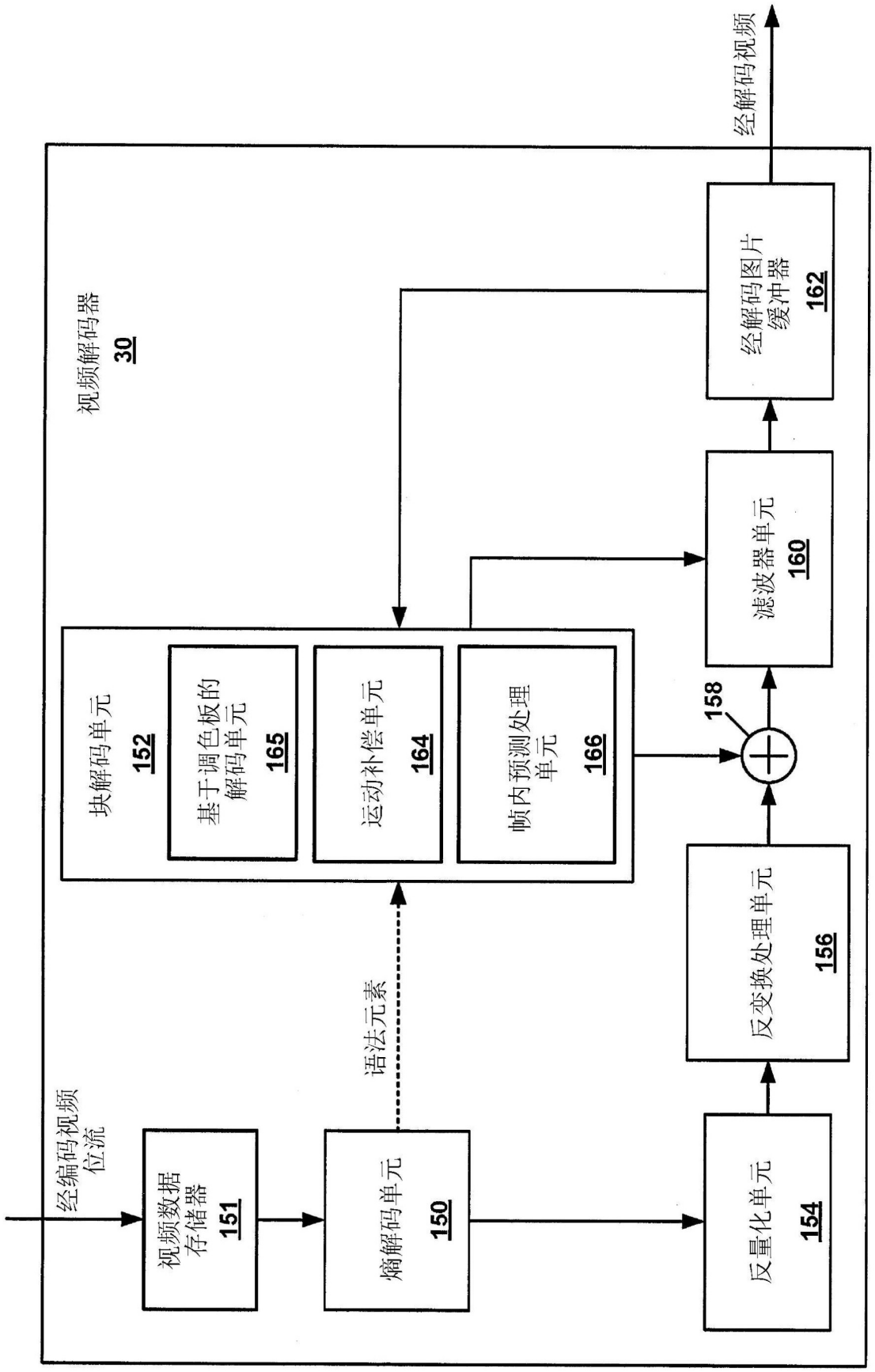

经编码视频位流
视频数据存储器 151
熵解码单元 150
语法元素
视频解码器 30
块解码单元 152
基于调色板的解码单元 165
运动补偿单元 164
帧内预测处理单元 166
158
反量化单元 154
反变换处理单元 156
滤波器单元 160
经解码图片缓冲器 162
经解码视频

图3

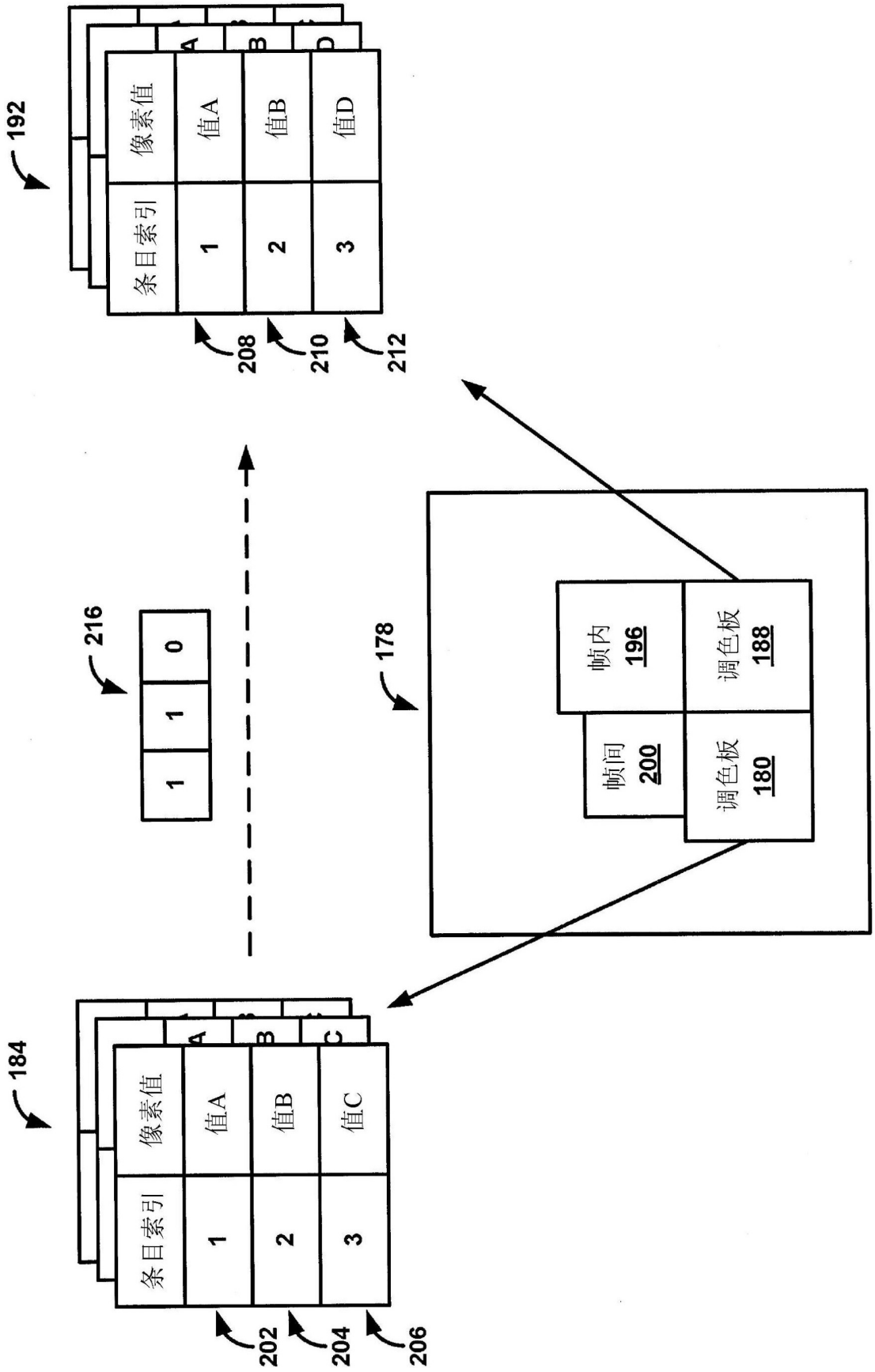

192
条目索引
像素值
A
B
D
1
2
3
值A
值B
值D
208
210
212
216
1
1
0
178
帧内 196
调色板 188
帧间 200
调色板 180
184
条目索引
像素值
A
B
C
1
2
3
值A
值B
值C
202
204
206

图4

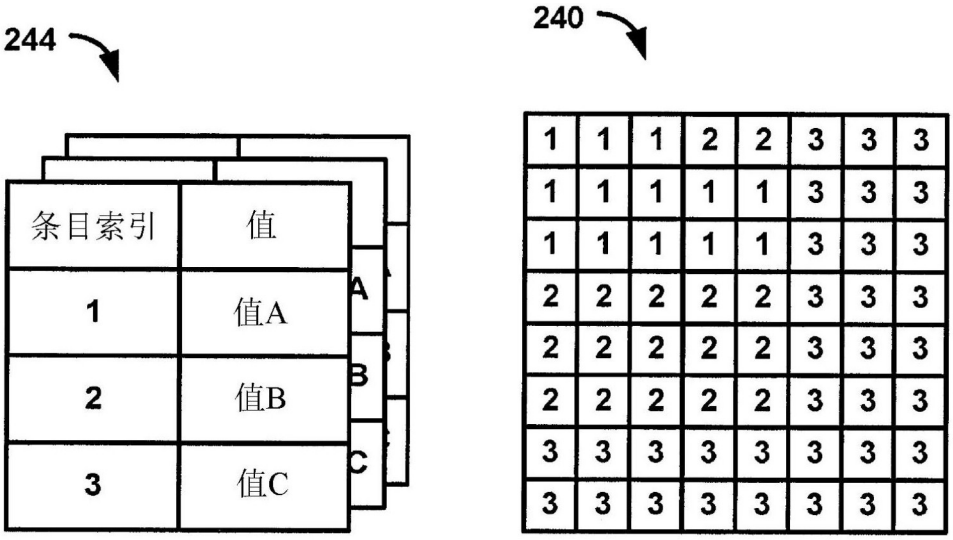

244
240
条目索引
值
1
值A
2
值B
3
值C
A
B
C
1 1 1 2 2 3 3 3
1 1 1 1 1 3 3 3
1 1 1 1 1 3 3 3
2 2 2 2 2 3 3 3
2 2 2 2 2 3 3 3
2 2 2 2 2 3 3 3
3 3 3 3 3 3 3 3
3 3 3 3 3 3 3 3

图5

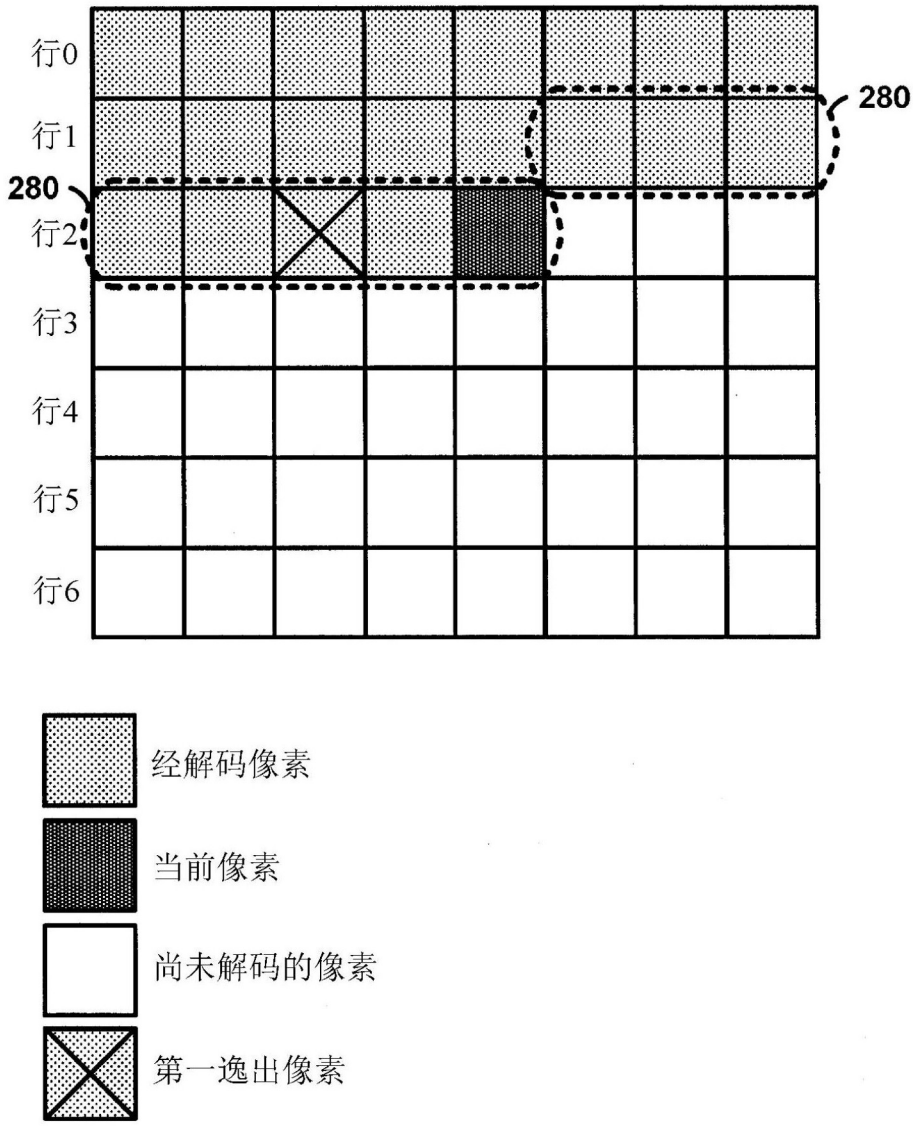


图6

...	
if(currentPaletteSize != 0)	
palette_escape_val_present_flag	ae(v)
indices_idc_coding() (Relocate all occurrence of **palette_index_idc** to here)	
if(palette_escape_val_present_flag) {	
if(cu_qp_delta_enabled_flag && !IsCuQpDeltaCoded) {	
cu_qp_delta_palette_abs	ae(v)
if(cu_qp_delta_palette_abs)	
cu_qp_delta_palette_sign_flag	ae(v)
}	
if(cu_chroma_qp_offset_enabled_flag && !IsCuChromaQpOffsetCoded) {	
cu_chroma_qp_palette_offset_flag	ae(v)
if(cu_chroma_qp_offset_flag && chroma_qp_offset_list_len_minus1 > 0)	
cu_chroma_qp_palette_offset_idx	ae(v)
}	
}	
if(indexMax > 0)	
palette_transpose_flag	ae(v)
scanPos = 0	
while(scanPos < nCbS * nCbS) {	
...	
if(palette_run_type_flag[xC][yC] = = COPY_INDEX_MODE && adjustedIndexMax > 0)	
palette_index_idc	ae(v)
if(indexMax > 0) {	
maxPaletteRun = nCbS * nCbS – scanPos – 1	
if(maxPaletteRun > 0) {	
palette_run_msb_id_plus1	ae(v)
if(palette_run_msb_id_plus1 > 1)	
palette_run_refinement_bits	ae(v)
}	
} else	
...	

图7

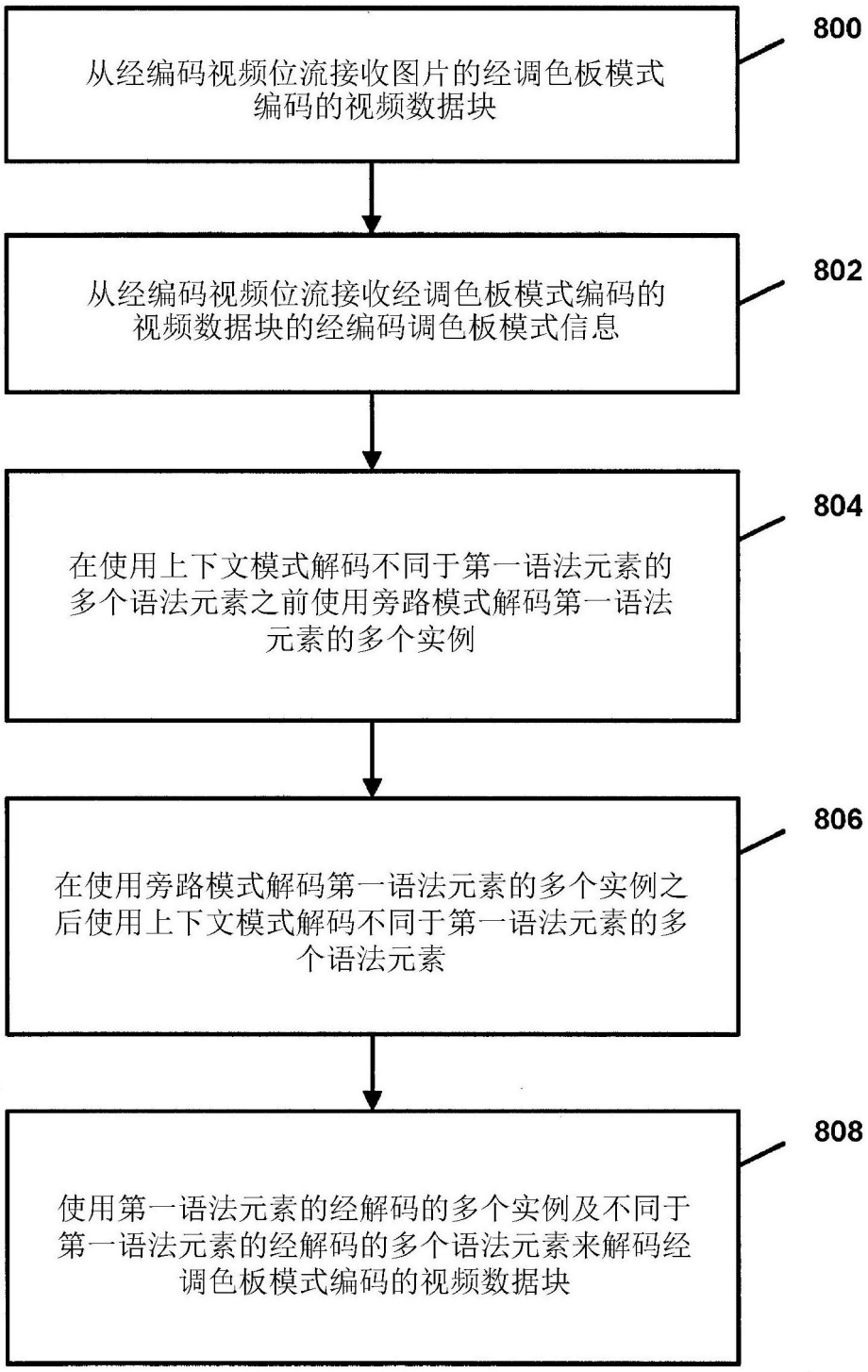

800
从经编码视频位流接收图片的经调色板模式编码的视频数据块
802
从经编码视频位流接收经调色板模式编码的视频数据块的经编码调色板模式信息
804
在使用上下文模式解码不同于第一语法元素的多个语法元素之前使用旁路模式解码第一语法元素的多个实例
806
在使用旁路模式解码第一语法元素的多个实例之后使用上下文模式解码不同于第一语法元素的多个语法元素
808
使用第一语法元素的经解码的多个实例及不同于第一语法元素的经解码的多个语法元素来解码经调色板模式编码的视频数据块

图8

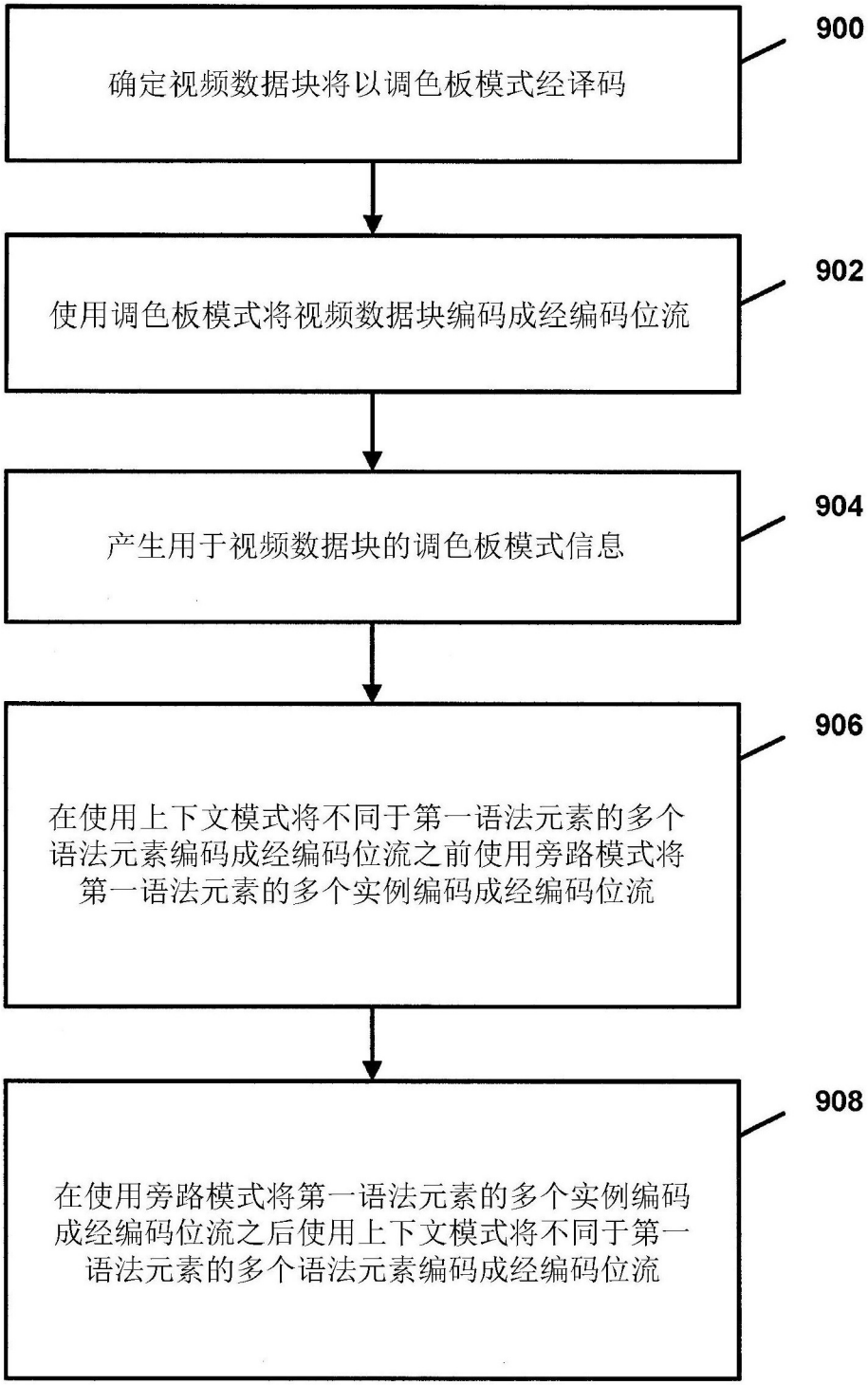

900
确定视频数据块将以调色板模式经译码
902
使用调色板模式将视频数据块编码成经编码位流
904
产生用于视频数据块的调色板模式信息
906
在使用上下文模式将不同于第一语法元素的多个语法元素编码成经编码位流之前使用旁路模式将第一语法元素的多个实例编码成经编码位流
908
在使用旁路模式将第一语法元素的多个实例编码成经编码位流之后使用上下文模式将不同于第一语法元素的多个语法元素编码成经编码位流

图9