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

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
13 June 2024 (13.06.2024)

WIPO | PCT

(10) International Publication Number
WO 2024/123906 A1

(51) International Patent Classification:
H04N 19/119 (2014.01)

(21) International Application Number:
PCT/US2023/082724

(22) International Filing Date:
06 December 2023 (06.12.2023)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
63/430,648 06 December 2022 (06.12.2022) US

(71) Applicant: **GOOGLE LLC** [US/US]; 1600 Amphitheatre Parkway, Mountain View, California 94043 (US).

(72) Inventors: **MUKHERJEE, Debargha**; c/o Google LLC, 1600 Amphitheatre Parkway, Mountain View, California 94043 (US). **TSAI, Chi Yo**; c/o Google LLC, 1600 Amphitheatre Parkway, Mountain View, California 94043 (US). **CHEN, Yue**; c/o Google LLC, 1600 Amphitheatre Parkway, Mountain View, California 94043 (US). **CHEN, Jianle**; c/o Google LLC, 1600 Amphitheatre Parkway, Mountain View, California 94043 (US). **LI, Xiang**; c/o Google LLC, 1600 Amphitheatre Parkway, Mountain View, California 94043 (US). **JOSHI, Urvang**; c/o Google LLC, 1600 Amphitheatre Parkway, Mountain View, California 94043 (US).

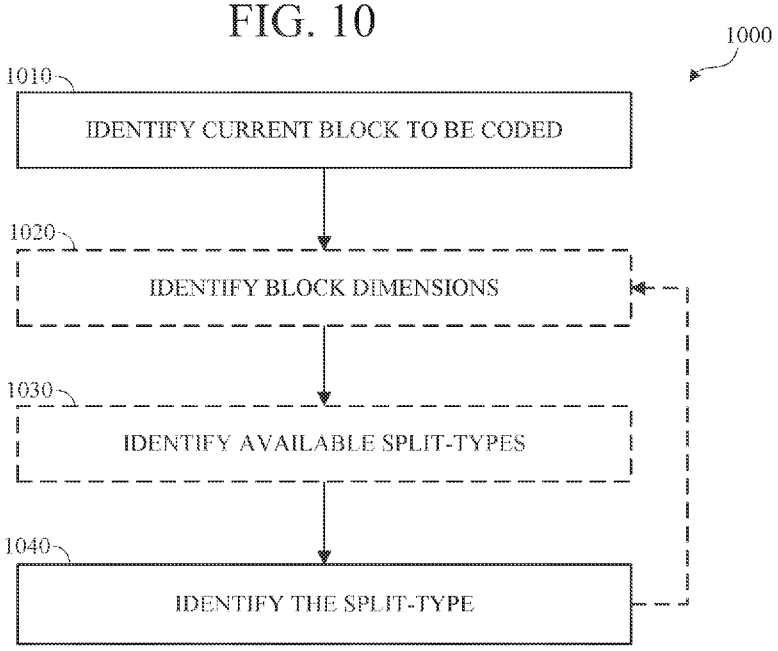
(74) Agent: **KNIGHT, Michelle** et al.; 3001 West Big Beaver Rd., Suite 624, Troy, Michigan 48084 (US).

(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, CV, CZ, DE, DJ, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IQ, IR, IS, IT, JM, JO, JP, KE, KG, KH, KN, KP, KR, KW, KZ, LA, LC, LK, LR, LS, LU, LY, MA, MD, MG, MK, MN, MU, 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, ST, SV, SY, TH,

WO 2024/123906 A1

(54) Title: RECURSIVE BLOCK PARTITIONING FOR IMAGE AND VIDEO COMPRESSION BASED ON POWER-OF-TWO SIZES

FIG. 10



(57) Abstract: Encoding and decoding a block using power-of-two sizes is described. At a decoder, a current block to be decoded is identified from an encoded bitstream. The current block was partitioned by an encoder using a recursive partitioning scheme. The recursive partitioning scheme includes only a no split split-type, binary split-types, and at least two quad split-types that each result in at least two partitions of different dimensions. One or more sub-blocks are decoded to reconstruct the current block. The recursive partitioning scheme may be one of multiple recursive partitioning schemes. The recursive partitioning scheme, the available split-types of a recursive partitioning scheme, or both, may be identified or limited by the block size.

[Continued on next page]

TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, WS, ZA, ZM, ZW.

(84) **Designated States** *(unless otherwise indicated, for every kind of regional protection available)*: ARIPO (BW, CV, GH, GM, KE, LR, LS, MW, MZ, NA, RW, SC, SD, SL, ST, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, RU, TJ, TM), European (AL, AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HR, HU, IE, IS, IT, LT, LU, LV, MC, ME, 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))*

RECURSIVE BLOCK PARTITIONING FOR IMAGE AND VIDEO COMPRESSION BASED ON POWER-OF-TWO SIZES

BACKGROUND

[0001] Digital images and video can be used, for example, on the internet, for remote business meetings via video conferencing, high-definition video entertainment, video advertisements, or sharing of user-generated content. Due to the large amount of data involved in transferring and processing image and video data, high-performance compression may be advantageous for transmission and storage. Accordingly, it would be advantageous to provide high-resolution image and video transmitted over communications channels having limited bandwidth, such as image and video coding using inter-intra prediction with implicit models.

SUMMARY

[0002] This application relates to encoding and decoding of image data, video stream data, or both, for transmission or storage. Disclosed herein are aspects of systems, methods, and apparatuses for encoding and decoding image data that is coded using recursive block partitioning based on power-of-two sizes.

[0003] A decoding method according to an aspect of the teachings herein includes identifying, from an encoded bitstream, a current block to be decoded, identifying one or more sub-blocks resulting from partitioning the current block using a recursive partitioning scheme, wherein the recursive partitioning scheme includes only a no split split-type, binary split-types, and at least two quad split-types that each result in at least two partitions of different dimensions, and decoding the one or more sub-blocks to reconstruct the current block.

[0004] An encoding method according to an aspect of the teachings herein includes identifying a current block to be encoded, identifying one or more sub-blocks by partitioning the current block using a recursive partitioning scheme, wherein the recursive partitioning scheme includes only a no split split-type, binary split-types, and at least two quad split-types that each result in at least two partitions of different dimensions, and encoding the one or more sub-blocks.

[0005] Another aspect of the teachings herein is a method for decoding image data. The method can include identifying, from an encoded bitstream, a current block to be decoded, identifying block dimensions of the current block, identifying available split-types for a recursive partitioning scheme using the block dimensions, wherein the recursive partitioning scheme includes only a no split split-type, binary split-types, and at least two quad split-types that each result in at least two partitions of different dimensions, identifying the split-type for the current block from the available split-types, and decoding one or more partitions corresponding to the current block based on the split-type.

[0006] Aspects of encoding and decoding apparatuses are also described.

[0007] Other aspects of the teachings herein include bitstreams determined according to the encoding techniques described herein and bitstreams that can be decoded according to the decoding techniques described herein. The bitstreams may be stored in a non-tangible computer-readable storage medium.

[0008] Variations in these and other aspects will be described in additional detail hereafter.

BRIEF DESCRIPTION OF THE DRAWINGS

[0009] The description herein makes reference to the accompanying drawings wherein like reference numerals refer to like parts throughout the several views unless otherwise noted or otherwise clear from context.

[0010] FIG. 1 is a diagram of a computing device in accordance with implementations of this disclosure.

[0011] FIG. 2 is a diagram of a computing and communications system in accordance with implementations of this disclosure.

[0012] FIG. 3 is a diagram of a video stream for use in encoding and decoding in accordance with implementations of this disclosure.

[0013] FIG. 4 is a block diagram of an encoder in accordance with implementations of this disclosure.

[0014] FIG. 5 is a block diagram of a decoder in accordance with implementations of this disclosure.

[0015] FIG. 6 is a block diagram of a representation of a portion of an image or frame explaining coding using recursive block partitioning.

[0016] FIG. 7 is a diagram of a 7-ary partitioning scheme in accordance with implementations of this disclosure.

[0017] FIG. 8 is a diagram of a 5-ary partitioning scheme in accordance with implementations of this disclosure.

[0018] FIG. 9 is a diagram of a 9-ary partitioning scheme in accordance with implementations of this disclosure.

[0019] FIG. 10 is a flowchart diagram of an example of coding a block using recursive block partitioning based on power-of-two sizes.

DETAILED DESCRIPTION

[0020] Image and video compression schemes may include breaking an image, or frame, into smaller portions, such as blocks, and generating an output bitstream using techniques to minimize the bandwidth utilization of the information included for each block in the output. In some implementations, the information included for each block in the output may be limited by reducing spatial redundancy, reducing temporal redundancy, or a combination thereof. For example, temporal or spatial redundancies may be reduced by predicting a frame, or a portion thereof, based on information available to both the encoder and decoder, and including information representing a difference, or residual, between the predicted frame and the original frame in the encoded bitstream. The residual information may be further compressed by transforming the residual information into transform coefficients, quantizing the transform coefficients, and entropy coding the quantized transform coefficients. Other coding information, such as motion information, may be included in the encoded bitstream, which may include transmitting differential information based on predictions of the encoding information, which may be entropy coded to further reduce the corresponding bandwidth utilization. An encoded bitstream can be decoded to reconstruct the blocks and the source images from the limited information. In some implementations, the accuracy, efficiency, or both, of coding a block using either inter-prediction or intra-prediction may be limited. When the image data or information comprises an image instead of a video frame, inter-prediction is not used.

[0021] Splitting or breaking the image data into blocks for coding may be performed using a recursive block partitioning scheme. Efficient block partitioning is a major contributor to the performance of a codec. For example, breaking a larger block into blocks with consistent image data can provide for better prediction of the blocks. However, practical reasons often result in a preference to use only power-of-two block sizes for prediction and transformation. The blocks resulting from current partitioning schemes designed to defer to this preference

may not represent the most efficient partition of the block.

[0022] Partitioning schemes described herein provide for more flexible partitioning choices than conventional schemes by incorporating new quad split types that can be used with binary split types. Implementations of these teachings are described below after an initial description of the environment in which the teachings may be implemented.

[0023] FIG. 1 is a diagram of a computing device 100 in accordance with implementations of this disclosure. The computing device 100 shown includes a memory 110, a processor 120, a user interface (UI) 130, an electronic communication unit 140, a sensor 150, a power source 160, and a bus 170. As used herein, the term "computing device" includes any unit, or a combination of units, capable of performing any method, or any portion or portions thereof, disclosed herein.

[0024] The computing device 100 may be a stationary computing device, such as a personal computer (PC), a server, a workstation, a minicomputer, or a mainframe computer; or a mobile computing device, such as a mobile telephone, a personal digital assistant (PDA), a laptop, or a tablet PC. Although shown as a single unit, any one element or elements of the computing device 100 can be integrated into any number of separate physical units. For example, the user interface 130 and processor 120 can be integrated in a first physical unit and the memory 110 can be integrated in a second physical unit.

[0025] The memory 110 can include any non-transitory computer-usable or computer-readable medium, such as any tangible device that can, for example, contain, store, communicate, or transport data 112, instructions 114, an operating system 116, or any information associated therewith, for use by or in connection with other components of the computing device 100. The non-transitory computer-usable or computer-readable medium can be, for example, a solid state drive, a memory card, removable media, a read-only memory (ROM), a random-access memory (RAM), any type of disk including a hard disk, a floppy disk, an optical disk, a magnetic or optical card, an application-specific integrated circuits (ASICs), or any type of non-transitory media suitable for storing electronic information, or any combination thereof.

[0026] Although shown a single unit, the memory 110 may include multiple physical units, such as one or more primary memory units, such as random-access memory units, one or more secondary data storage units, such as disks, or a combination thereof. For example, the data 112, or a portion thereof, the instructions 114, or a portion thereof, or both, may be stored in a secondary storage unit and may be loaded or otherwise transferred to a primary

storage unit in conjunction with processing the respective data 112, executing the respective instructions 114, or both. In some implementations, the memory 110, or a portion thereof, may be removable memory.

[0027] The data 112 can include information, such as input audio data, encoded audio data, decoded audio data, or the like. The instructions 114 can include directions, such as code, for performing any method, or any portion or portions thereof, disclosed herein. The instructions 114 can be realized in hardware, software, or any combination thereof. For example, the instructions 114 may be implemented as information stored in the memory 110, such as a computer program, that may be executed by the processor 120 to perform any of the respective methods, algorithms, aspects, or combinations thereof, as described herein.

[0028] Although shown as included in the memory 110, in some implementations, the instructions 114, or a portion thereof, may be implemented as a special purpose processor, or circuitry, that can include specialized hardware for carrying out any of the methods, algorithms, aspects, or combinations thereof, as described herein. Portions of the instructions 114 can be distributed across multiple processors on the same machine or different machines or across a network such as a local area network, a wide area network, the Internet, or a combination thereof.

[0029] The processor 120 can include any device or system capable of manipulating or processing a digital signal or other electronic information now-existing or hereafter developed, including optical processors, quantum processors, molecular processors, or a combination thereof. For example, the processor 120 can include a special purpose processor, a central processing unit (CPU), a digital signal processor (DSP), a plurality of microprocessors, one or more microprocessor in association with a DSP core, a controller, a microcontroller, an Application Specific Integrated Circuit (ASIC), a Field Programmable Gate Array (FPGA), a programmable logic array, programmable logic controller, microcode, firmware, any type of integrated circuit (IC), a state machine, or any combination thereof. As used herein, the term "processor" includes a single processor or multiple processors.

[0030] The user interface 130 can include any unit capable of interfacing with a user, such as a virtual or physical keypad, a touchpad, a display, a touch display, a speaker, a microphone, a video camera, a sensor, or any combination thereof. For example, the user interface 130 may be an audio-visual display device, and the computing device 100 may present audio, such as decoded audio, using the user interface 130 audio-visual display device, such as in conjunction with displaying video, such as decoded video. Although shown

as a single unit, the user interface 130 may include one or more physical units. For example, the user interface 130 may include an audio interface for performing audio communication with a user, and a touch display for performing visual and touch-based communication with the user.

[0031] The electronic communication unit 140 can transmit, receive, or transmit and receive signals via a wired or wireless electronic communication medium 180, such as a radio frequency (RF) communication medium, an ultraviolet (UV) communication medium, a visible light communication medium, a fiber optic communication medium, a wireline communication medium, or a combination thereof. For example, as shown, the electronic communication unit 140 is operatively connected to an electronic communication interface 142, such as an antenna, configured to communicate via wireless signals.

[0032] Although the electronic communication interface 142 is shown as a wireless antenna in FIG. 1, the electronic communication interface 142 can be a wireless antenna, as shown, a wired communication port, such as an Ethernet port, an infrared port, a serial port, or any other wired or wireless unit capable of interfacing with a wired or wireless electronic communication medium 180. Although FIG. 1 shows a single electronic communication unit 140 and a single electronic communication interface 142, any number of electronic communication units and any number of electronic communication interfaces can be used.

[0033] The sensor 150 may include, for example, an audio-sensing device, a visible light-sensing device, a motion sensing device, or a combination thereof. For example, 100the sensor 150 may include a sound-sensing device, such as a microphone, or any other sound-sensing device now existing or hereafter developed that can sense sounds in the proximity of the computing device 100, such as speech or other utterances, made by a user operating the computing device 100. In another example, the sensor 150 may include a camera, or any other image-sensing device now existing or hereafter developed that can sense an image such as the image of a user operating the computing device. Although a single sensor 150 is shown, the computing device 100 may include a number of sensors 150. For example, the computing device 100 may include a first camera oriented with a field of view directed toward a user of the computing device 100 and a second camera oriented with a field of view directed away from the user of the computing device 100.

[0034] The power source 160 can be any suitable device for powering the computing device 100. For example, the power source 160 can include a wired external power source interface; one or more dry cell batteries, such as nickel-cadmium (NiCd), nickel-zinc (NiZn),

nickel metal hydride (NiMH), lithium-ion (Li-ion); solar cells; fuel cells; or any other device capable of powering the computing device 100. Although a single power source 160 is shown in FIG. 1, the computing device 100 may include multiple power sources 160, such as a battery and a wired external power source interface.

[0035] Although shown as separate units, the electronic communication unit 140, the electronic communication interface 142, the user interface 130, the power source 160, or portions thereof, may be configured as a combined unit. For example, the electronic communication unit 140, the electronic communication interface 142, the user interface 130, and the power source 160 may be implemented as a communications port capable of interfacing with an external display device, providing communications, power, or both.

[0036] One or more of the memory 110, the processor 120, the user interface 130, the electronic communication unit 140, the sensor 150, or the power source 160, may be operatively coupled via a bus 170. Although a single bus 170 is shown in FIG. 1, a computing device 100 may include multiple buses. For example, the memory 110, the processor 120, the user interface 130, the electronic communication unit 140, the sensor 150, and the bus 170 may receive power from the power source 160 via the bus 170. In another example, the memory 110, the processor 120, the user interface 130, the electronic communication unit 140, the sensor 150, the power source 160, or a combination thereof, may communicate data, such as by sending and receiving electronic signals, via the bus 170.

[0037] Although not shown separately in FIG. 1, one or more of the processor 120, the user interface 130, the electronic communication unit 140, the sensor 150, or the power source 160 may include internal memory, such as an internal buffer or register. For example, the processor 120 may include internal memory (not shown) and may read data 112 from the memory 110 into the internal memory (not shown) for processing.

[0038] Although shown as separate elements, the memory 110, the processor 120, the user interface 130, the electronic communication unit 140, the sensor 150, the power source 160, and the bus 170, or any combination thereof can be integrated in one or more electronic units, circuits, or chips.

[0039] FIG. 2 is a diagram of a computing and communications system 200 in accordance with implementations of this disclosure. The computing and communications system 200 shown includes computing and communication devices 100A, 100B, 100C, access points 210A, 210B, and a network 220. For example, the computing and communication system 200 can be a multiple access system that provides communication, such as voice, audio, data,

video, messaging, broadcast, or a combination thereof, to one or more wired or wireless communicating devices, such as the computing and communication devices 100A, 100B, 100C. Although, for simplicity, FIG. 2 shows three computing and communication devices 100A, 100B, 100C, two access points 210A, 210B, and one network 220, any number of computing and communication devices, access points, and networks can be used.

[0040] A computing and communication device 100A, 100B, 100C can be, for example, a computing device, such as the computing device 100 shown in FIG. 1. For example, the computing and communication devices 100A, 100B may be user devices, such as a mobile computing device, a laptop, a thin client, or a smartphone, and the computing and communication device 100C may be a server, such as a mainframe or a cluster. Although the computing and communication device 100A and the computing and communication device 100B are described as user devices, and the computing and communication device 100C is described as a server, any computing and communication device may perform some or all of the functions of a server, some or all of the functions of a user device, or some or all of the functions of a server and a user device. For example, the server computing and communication device 100C may receive, encode, process, store, transmit, or a combination thereof audio data and one or both of the computing and communication device 100A and the computing and communication device 100B may receive, decode, process, store, present, or a combination thereof the audio data.

[0041] Each computing and communication device 100A, 100B, 100C, which may include a user equipment (UE), a mobile station, a fixed or mobile subscriber unit, a cellular telephone, a personal computer, a tablet computer, a server, consumer electronics, or any similar device, can be configured to perform wired or wireless communication, such as via the network 220. For example, the computing and communication devices 100A, 100B, 100C can be configured to transmit or receive wired or wireless communication signals. Although each computing and communication device 100A, 100B, 100C is shown as a single unit, a computing and communication device can include any number of interconnected elements.

[0042] Each access point 210A, 210B can be any type of device configured to communicate with a computing and communication device 100A, 100B, 100C, a network 220, or both via wired or wireless communication links 180A, 180B, 180C. For example, an access point 210A, 210B can include a base station, a base transceiver station (BTS), a Node-B, an enhanced Node-B (eNode-B), a Home Node-B (HNode-B), a wireless router, a wired router, a hub, a relay, a switch, or any similar wired or wireless device. Although each access

point 210A, 210B is shown as a single unit, an access point can include any number of interconnected elements.

[0043] The network 220 can be any type of network configured to provide services, such as voice, data, applications, voice over internet protocol (VoIP), or any other communications protocol or combination of communications protocols, over a wired or wireless communication link. For example, the network 220 can be a local area network (LAN), wide area network (WAN), virtual private network (VPN), a mobile or cellular telephone network, the Internet, or any other means of electronic communication. The network can use a communication protocol, such as the transmission control protocol (TCP), the user datagram protocol (UDP), the internet protocol (IP), the real-time transport protocol (RTP) the HyperText Transport Protocol (HTTP), or a combination thereof.

[0044] The computing and communication devices 100A, 100B, 100C can communicate with each other via the network 220 using one or more a wired or wireless communication links, or via a combination of wired and wireless communication links. For example, as shown the computing and communication devices 100A, 100B can communicate via wireless communication links 180A, 180B, and computing and communication device 100C can communicate via a wired communication link 180C. Any of the computing and communication devices 100A, 100B, 100C may communicate using any wired or wireless communication link, or links. For example, a first computing and communication device 100A can communicate via a first access point 210A using a first type of communication link, a second computing and communication device 100B can communicate via a second access point 210B using a second type of communication link, and a third computing and communication device 100C can communicate via a third access point (not shown) using a third type of communication link. Similarly, the access points 210A, 210B can communicate with the network 220 via one or more types of wired or wireless communication links 230A, 230B. Although FIG. 2 shows the computing and communication devices 100A, 100B, 100C in communication via the network 220, the computing and communication devices 100A, 100B, 100C can communicate with each other via any number of communication links, such as a direct wired or wireless communication link.

[0045] In some implementations, communications between one or more of the computing and communication device 100A, 100B, 100C may omit communicating via the network 220 and may include transferring data via another medium (not shown), such as a data storage device. For example, the server computing and communication device 100C may store audio

data, such as encoded audio data, in a data storage device, such as a portable data storage unit, and one or both of the computing and communication device 100A or the computing and communication device 100B may access, read, or retrieve the stored audio data from the data storage unit, such as by physically disconnecting the data storage device from the server computing and communication device 100C and physically connecting the data storage device to the computing and communication device 100A or the computing and communication device 100B.

[0046] Other implementations of the computing and communications system 200 are possible. For example, in an implementation, the network 220 can be an ad-hoc network and can omit one or more of the access points 210A, 210B. The computing and communications system 200 may include devices, units, or elements not shown in FIG. 2. For example, the computing and communications system 200 may include many more communicating devices, networks, and access points.

[0047] FIG. 3 is a diagram of a video stream 300 for use in encoding and decoding in accordance with implementations of this disclosure. A video stream 300, such as a video stream captured by a video camera or a video stream generated by a computing device, may include a video sequence 310. The video sequence 310 may include a sequence of adjacent frames 320. Although three adjacent frames 320 are shown, the video sequence 310 can include any number of adjacent frames 320.

[0048] Each frame 330 from the adjacent frames 320 may represent a single image from the video stream. Although not shown in FIG. 3, a frame 330 may include one or more segments, tiles, or planes, which may be coded, or otherwise processed, independently, such as in parallel. A frame 330 may include one or more tiles 340. Each of the tiles 340 may be a rectangular region of the frame that can be coded independently. Each of the tiles 340 may include respective blocks 350. Although not shown in FIG. 3, a block can include pixels. For example, a block can include a 16×16 group of pixels, an 8×8 group of pixels, an 8×16 group of pixels, or any other group of pixels. Unless otherwise indicated herein, the term 'block' can include a superblock, a macroblock, a segment, a slice, or any other portion of a frame. A frame, a block, a pixel, or a combination thereof can include display information, such as luminance information, chrominance information, or any other information that can be used to store, modify, communicate, or display the video stream or a portion thereof.

[0049] FIG. 4 is a block diagram of an encoder 400 in accordance with implementations of this disclosure. Encoder 400 can be implemented in a device, such as the computing device

100 shown in FIG. 1 or the computing and communication devices 100A, 100B, 100C shown in FIG. 2, as, for example, a computer software program stored in a data storage unit, such as the memory 110 shown in FIG. 1. The computer software program can include machine instructions that may be executed by a processor, such as the processor 120 shown in FIG. 1, and may cause the device to encode video data as described herein. The encoder 400 can be implemented as specialized hardware included, for example, in computing device 100.

[0050] The encoder 400 can encode an input video stream 402, such as the video stream 300 shown in FIG. 3, to generate an encoded (compressed) bitstream 404. In some implementations, the encoder 400 may include a forward path for generating the compressed bitstream 404. The forward path may include an intra/inter prediction unit 410, a transform unit 420, a quantization unit 430, an entropy encoding unit 440, or any combination thereof. In some implementations, the encoder 400 may include a reconstruction path (indicated by the broken connection lines) to reconstruct a frame for encoding of further blocks. The reconstruction path may include a dequantization unit 450, an inverse transform unit 460, a reconstruction unit 470, a filtering unit 480, or any combination thereof. Other structural variations of the encoder 400 can be used to encode the video stream 402.

[0051] For encoding the video stream 402, each frame within the video stream 402 can be processed in units of blocks. Thus, a current block may be identified from the blocks in a frame, and the current block may be encoded.

[0052] At the intra/inter prediction unit 410, the current block can be encoded using either intra-frame prediction, which may be within a single frame, or inter-frame prediction, which may be from frame to frame. Intra-prediction may include generating a prediction block from samples in the current frame that have been previously encoded and reconstructed. Inter-prediction may include generating a prediction block from samples in one or more previously constructed reference frames. Generating a prediction block for a current block in a current frame may include performing motion estimation to generate a motion vector indicating an appropriate reference portion of the reference frame.

[0053] The intra/inter prediction unit 410 may subtract the prediction block from the current block (raw block) to produce a residual block. The transform unit 420 may perform a block-based transform, which may include transforming the residual block into transform coefficients in, for example, the frequency domain. Examples of block-based transforms include the Karhunen-Loève Transform (KLT), the Discrete Cosine Transform (DCT), the Singular Value Decomposition Transform (SVD), and the Asymmetric Discrete Sine

Transform (ADST). In an example, the DCT may include transforming a block into the frequency domain. The DCT may include using transform coefficient values based on spatial frequency, with the lowest frequency, i.e., the direct current (DC) coefficient at the top-left of the matrix and the highest frequency coefficient at the bottom-right of the matrix.

[0054] The quantization unit 430 may convert the transform coefficients into discrete quantum values, which may be referred to as quantized transform coefficients or quantization levels. The quantized transform coefficients can be entropy encoded by the entropy encoding unit 440 to produce entropy-encoded coefficients. Entropy encoding can include using a probability distribution metric. The entropy-encoded coefficients and information used to decode the block, which may include the type of prediction used, motion vectors, and quantizer values, can be output to the compressed bitstream 404. The compressed bitstream 404 can be formatted using various techniques, such as run-length encoding (RLE) and zero-run coding.

[0055] The reconstruction path can be used to maintain reference frame synchronization between the encoder 400 and a corresponding decoder, such as the decoder 500 shown in FIG. 5. The reconstruction path may be similar to the decoding process discussed below and may include decoding the encoded frame, or a portion thereof, which may include decoding an encoded block, which may include dequantizing the quantized transform coefficients at the dequantization unit 450 and inverse transforming the dequantized transform coefficients at the inverse transform unit 460 to produce a derivative residual block. The reconstruction unit 470 may add the prediction block generated by the intra/inter prediction unit 410 to the derivative residual block to create a decoded block. The filtering unit 480 can be applied to the decoded block to generate a reconstructed block, which may reduce distortion, such as blocking artifacts. Although one filtering unit 480 is shown in FIG. 4, filtering the decoded block may include loop filtering, deblocking filtering, or other types of filtering or combinations of types of filtering. The reconstructed block may be stored or otherwise made accessible as a reconstructed block, which may be a portion of a reference frame, for encoding another portion of the current frame, another frame, or both, as indicated by the broken line at 482. Coding information, such as deblocking threshold index values, for the frame may be encoded, included in the compressed bitstream 404, or both, as indicated by the broken line at 484.

[0056] Other variations of the encoder 400 can be used to encode the compressed bitstream 404. For example, a non-transform-based encoder 400 can quantize the residual

block directly without the transform unit 420. In some implementations, the quantization unit 430 and the dequantization unit 450 may be combined into a single unit.

[0057] FIG. 5 is a block diagram of a decoder 500 in accordance with implementations of this disclosure. The decoder 500 can be implemented in a device, such as the computing device 100 shown in FIG. 1 or the computing and communication devices 100A, 100B, 100C shown in FIG. 2, as, for example, a computer software program stored in a data storage unit, such as the memory 110 shown in FIG. 1. The computer software program can include machine instructions that may be executed by a processor, such as the processor 120 shown in FIG. 1, and may cause the device to decode video data as described herein. The decoder 500 can be implemented as specialized hardware included, for example, in computing device 100.

[0058] The decoder 500 may receive a compressed bitstream 502, such as the compressed bitstream 404 shown in FIG. 4, and may decode the compressed bitstream 502 to generate an output video stream 504. The decoder 500 may include an entropy decoding unit 510, a dequantization unit 520, an inverse transform unit 530, an intra/inter prediction unit 540, a reconstruction unit 550, a filtering unit 560, or any combination thereof. Other structural variations of the decoder 500 can be used to decode the compressed bitstream 502.

[0059] The entropy decoding unit 510 may decode data elements within the compressed bitstream 502 using, for example, Context Adaptive Binary Arithmetic Decoding, to produce a set of quantized transform coefficients. The dequantization unit 520 can dequantize the quantized transform coefficients, and the inverse transform unit 530 can inverse transform the dequantized transform coefficients to produce a derivative residual block, which may correspond to the derivative residual block generated by the inverse transform unit 460 shown in FIG. 4. Using header information decoded from the compressed bitstream 502, the intra/inter prediction unit 540 may generate a prediction block corresponding to the prediction block created in the encoder 400. At the reconstruction unit 550, the prediction block can be added to the derivative residual block to create a decoded block. The filtering unit 560 can be applied to the decoded block to reduce artifacts, such as blocking artifacts, which may include loop filtering, deblocking filtering, or other types of filtering or combinations of types of filtering, and which may include generating a reconstructed block, which may be output as the output video stream 504.

[0060] Other variations of the decoder 500 can be used to decode the compressed bitstream 502. For example, the decoder 500 can produce the output video stream 504 without the deblocking filtering unit 570.

[0061] FIG. 6 is a block diagram of a representation of a portion 600 of a frame, such as the frame 330 shown in FIG. 3. FIG. 6 is used to describe using recursive partitioning to code a sequence of blocks.

[0062] As shown, the portion 600 of the frame includes four 64×64 blocks 610, in two rows and two columns in a matrix or Cartesian plane. In some implementations, a 64×64 block may be a maximum coding unit, N=64. Each 64×64 block may include four 32×32 blocks 620. Each 32×32 block may include four 16×16 blocks 630. Each 16×16 block may include four 8×8 blocks 640. Each 8×8 block 640 may include four 4×4 blocks 650. Each 4×4 block 650 may include 16 pixels, which may be represented in four rows and four columns in each respective block in the Cartesian plane or matrix. The pixels may include information representing an image captured in the frame, such as luminance information, color information, and location information. In some implementations, a block, such as a 16×16 pixel block as shown, may include a luminance block 660, which may include luminance pixels 662; and two chrominance blocks 670, 680, such as a U or Cb chrominance block 670, and a V or Cr chrominance block 680. The chrominance blocks 670, 680 may include chrominance pixels 690. For example, the luminance block 660 may include 16×16 luminance pixels 662 and each chrominance block 670, 680 may include 8×8 chrominance pixels 690 as shown. Although one arrangement of blocks is shown, any arrangement may be used.

[0063] In some implementations, video coding may include ordered block-level coding. Ordered block-level coding may include coding blocks of a frame in an order, such as raster-scan order, wherein blocks may be identified and processed starting with a block in the upper left corner of the frame, or portion of the frame, and proceeding along rows from left to right and from the top row to the bottom row, identifying each block in turn for processing. For example, the 64×64 block in the top row and left column of a frame may be the first block coded and the 64×64 block immediately to the right of the first block may be the second block coded. The second row from the top may be the second row coded, such that the 64×64 block in the left column of the second row may be coded after the 64×64 block in the rightmost column of the first row.

[0064] In some implementations, coding a block may include using quad-tree coding, which may include coding smaller block units within a block in raster-scan order. For example, the 64×64 block shown in the bottom left corner of the portion of the frame shown in FIG. 6, may be coded using quad-tree coding wherein the top left 32×32 block may be

coded, then the top right 32×32 block may be coded, then the bottom left 32×32 block may be coded, and then the bottom right 32×32 block may be coded. Each 32×32 block may be coded using quad-tree coding wherein the top left 16×16 block may be coded, then the top right 16×16 block may be coded, then the bottom left 16×16 block may be coded, and then the bottom right 16×16 block may be coded. Each 16×16 block may be coded using quad-tree coding wherein the top left 8×8 block may be coded, then the top right 8×8 block may be coded, then the bottom left 8×8 block may be coded, and then the bottom right 8×8 block may be coded. Each 8×8 block may be coded using quad-tree coding wherein the top left 4×4 block may be coded, then the top right 4×4 block may be coded, then the bottom left 4×4 block may be coded, and then the bottom right 4×4 block may be coded. In some implementations, 8×8 blocks may be omitted for a 16×16 block, and the 16×16 block may be coded using quad-tree coding wherein the top left 4×4 block may be coded, then the other 4×4 blocks in the 16×16 block may be coded in raster-scan order.

[0065] In some implementations, video coding may include compressing the information included in an original, or input, frame by, for example, omitting some of the information in the original frame from a corresponding encoded frame. For example, coding may include reducing spectral redundancy, reducing spatial redundancy, reducing temporal redundancy, or a combination thereof.

[0066] In some implementations, reducing spectral redundancy may include using a color model based on a luminance component (Y) and two chrominance components (U and V or Cb and Cr), which may be referred to as the YUV or YCbCr color model, or color space. Using the YUV color model may include using a relatively large amount of information to represent the luminance component of a portion of a frame and using a relatively small amount of information to represent each corresponding chrominance component for the portion of the frame. For example, a portion of a frame may be represented by a high-resolution luminance component, which may include a 16×16 block of pixels, and by two lower resolution chrominance components, each of which represents the portion of the frame as an 8×8 block of pixels. A pixel may indicate a value, for example, a value in the range from 0 to 255, and may be stored or transmitted using, for example, eight bits. Although this disclosure is described in reference to the YUV color model, any color model may be used.

[0067] In some implementations, reducing spatial redundancy may include transforming a block into the frequency domain using, for example, a discrete cosine transform (DCT). For example, a unit of an encoder, such as the transform unit 420 shown in FIG. 4, may perform a

DCT using transform coefficient values based on spatial frequency.

[0068] In some implementations, reducing temporal redundancy may include using similarities between frames to encode a frame using a relatively small amount of data based on one or more reference frames, which may be previously encoded, decoded, and reconstructed frames of the video stream. For example, a block or pixel of a current frame may be similar to a spatially corresponding block or pixel of a reference frame. In some implementations, a block or pixel of a current frame may be similar to block or pixel of a reference frame at a different spatial location and reducing temporal redundancy may include generating motion information indicating the spatial difference, or translation, between the location of the block or pixel in the current frame and corresponding location of the block or pixel in the reference frame.

[0069] In some implementations, reducing temporal redundancy may include identifying a portion of a reference frame that corresponds to a current block or pixel of a current frame. For example, a reference frame, or a portion of a reference frame, which may be stored in memory, may be searched to identify a portion for generating a prediction to use for encoding a current block or pixel of the current frame with maximal efficiency. For example, the search may identify a portion of the reference frame for which the difference in pixel values between the current block and a prediction block generated based on the portion of the reference frame is minimized and may be referred to as motion searching. In some implementations, the portion of the reference frame searched may be limited. For example, the portion of the reference frame searched, which may be referred to as the search area, may include a limited number of rows of the reference frame. In an example, identifying the portion of the reference frame for generating a prediction may include calculating a cost function, such as a sum of absolute differences (SAD), between the pixels of portions of the search area and the pixels of the current block.

[0070] In some implementations, the spatial difference between the location of the portion of the reference frame for generating a prediction in the reference frame and the current block in the current frame may be represented as a motion vector. The difference in pixel values between the prediction block and the current block may be referred to as differential data, residual data, a prediction error, or as a residual block. In some implementations, generating motion vectors may be referred to as motion estimation, and a pixel of a current block may be indicated based on location using Cartesian coordinates as $f_{x,y}$. Similarly, a pixel of the search area of the reference frame may be indicated based on location using Cartesian coordinates as

$r_{x,y}$. A motion vector (MV) for the current block may be determined based on, for example, a SAD between the pixels of the current frame and the corresponding pixels of the reference frame.

[0071] Although described herein with reference to matrix or Cartesian representation of a frame for clarity, a frame may be stored, transmitted, processed, or any combination thereof, in any data structure such that pixel values may be efficiently represented for a frame or image. For example, a frame may be stored, transmitted, processed, or any combination thereof, in a two-dimensional data structure such as a matrix as shown, or in a one-dimensional data structure, such as a vector array. In an implementation, a representation of the frame, such as a two-dimensional representation as shown, may correspond to a physical location in a rendering of the frame as an image. For example, a location in the top left corner of a block in the top left corner of the frame may correspond with a physical location in the top left corner of a rendering of the frame as an image.

[0072] In some implementations, block-based coding efficiency may be improved by partitioning input blocks into one or more prediction partitions, which may be rectangular, including square, partitions for prediction coding. In some implementations, video coding using prediction partitioning may include selecting a prediction partitioning scheme from among multiple candidate prediction partitioning schemes. For example, in some implementations, candidate prediction partitioning schemes for a 64×64 coding unit may include rectangular size prediction partitions ranging in sizes from 4×4 to 64×64, such as 4×4, 4×8, 8×4, 8×8, 8×16, 16×8, 16×16, 16×32, 32×16, 32×32, 32×64, 64×32, or 64×64. In some implementations, video coding using prediction partitioning may include a full prediction partition search, which may include selecting a prediction partitioning scheme by encoding the coding unit using each available candidate prediction partitioning scheme and selecting the best scheme, such as the scheme that produces the least rate-distortion error.

[0073] In some implementations, encoding a video frame may include identifying a prediction partitioning scheme for encoding a current block, such as block 610. In some implementations, identifying a prediction partitioning scheme may include determining whether to encode the block as a single prediction partition of maximum coding unit size, which may be 64×64 as shown, or to partition the block into multiple prediction partitions, which may correspond with the sub-blocks, such as the 32×32 blocks 620 the 16×16 blocks 630, or the 8×8 blocks 640, as shown, and may include determining whether to partition into one or more smaller prediction partitions. For example, a 64×64 block may be partitioned

into four 32×32 prediction partitions. Three of the four 32×32 prediction partitions may be encoded as 32×32 prediction partitions and the fourth 32×32 prediction partition may be further partitioned into four 16×16 prediction partitions. Three of the four 16×16 prediction partitions may be encoded as 16×16 prediction partitions and the fourth 16×16 prediction partition may be further partitioned into four 8×8 prediction partitions, each of which may be encoded as an 8×8 prediction partition. In some implementations, identifying the prediction partitioning scheme may include using a prediction partitioning decision tree.

[0074] In some implementations, video coding for a current block may include identifying an optimal prediction coding mode from multiple candidate prediction coding modes, which may provide flexibility in handling video signals with various statistical properties and may improve the compression efficiency. For example, a video coder may evaluate each candidate prediction coding mode to identify the optimal prediction coding mode, which may be, for example, the prediction coding mode that minimizes an error metric, such as a rate-distortion cost, for the current block. In some implementations, the complexity of searching the candidate prediction coding modes may be reduced by limiting the set of available candidate prediction coding modes based on similarities between the current block and a corresponding prediction block. In some implementations, the complexity of searching each candidate prediction coding mode may be reduced by performing a directed refinement mode search. For example, metrics may be generated for a limited set of candidate block sizes, such as 16×16, 8×8, and 4×4, the error metric associated with each block size may be in descending order, and additional candidate block sizes, such as 4×8 and 8×4 block sizes, may be evaluated.

[0075] In some implementations, block-based coding efficiency may be improved by partitioning a current residual block into one or more transform partitions, which may be rectangular, including square, partitions for transform coding. In some implementations, video coding using transform partitioning may include selecting a uniform transform partitioning scheme. For example, a current residual block, such as block 610, may be a 64×64 block and may be transformed without partitioning using a 64×64 transform.

[0076] Although not expressly shown in FIG. 6, a residual block may be transform partitioned using a uniform transform partitioning scheme. For example, a 64×64 residual block may be transform partitioned using a uniform transform partitioning scheme including four 32×32 transform blocks, using a uniform transform partitioning scheme including sixteen 16×16 transform blocks, using a uniform transform partitioning scheme including

sixty-four 8×8 transform blocks, or using a uniform transform partitioning scheme including 256 4×4 transform blocks.

[0077] In some implementations, video coding using transform partitioning may include identifying multiple transform block sizes for a residual block using multiform transform partition coding. In some implementations, multiform transform partition coding may include recursively determining whether to transform a current block using a current block size transform or by partitioning the current block and multiform transform partition coding each partition. For example, the bottom left block 610 shown in FIG. 6 may be a 64×64 residual block, and multiform transform partition coding may include determining whether to code the current 64×64 residual block using a 64×64 transform or to code the 64×64 residual block by partitioning the 64×64 residual block into partitions, such as four 32×32 blocks 620, and multiform transform partition coding each partition. In some implementations, determining whether to transform partition the current block may be based on comparing a cost for encoding the current block using a current block size transform to a sum of costs for encoding each partition using partition size transforms.

[0078] As initially described, and although not shown in FIG. 6, a block may be partitioned using a binary split. For example, the bottom-right of the portion 600 may be subject to a binary horizontal split such that two 32x64 blocks (one on top of the other) are formed or may be subject to a binary vertical split such that two 64x32 blocks (one on top of the other) are formed. This binary split type may be used as a partition choice in the description above. along with the two-way quad split. Stated more generally, if a block comprises a 2N x 2N block, the partition choices for the recursive partitioning may be no partition (e.g., 2N x 2N), a quad type partition N x N, a horizontal type partition N x 2N, or a vertical type partition 2N x N. A ternary type partition that results in three blocks, namely 2N x N/2, 2N x N, and 2N x N/2 arranged vertically in sequence or N/2 x 2N, N x 2N, and N/2 x N arranged horizontally in sequence, is also possible. This partition type, while adding some flexibility to the partitioning, can cause a level of tree-entanglement with binary horizontal and vertical splits that can potentially result in efficiency loss.

[0079] The set of alternate partitioning schemes described herein increases the variations in the sizes and/or shapes of the resulting blocks while reducing the probability of entanglement by using only binary and quad type splits.

[0080] FIG. 7 is a diagram of a proposed partitioning scheme 700 using a 7-ary split-type. Each sub-partition can be recursively split until a minimum block-size constraint is reached.

Further, and depending on the size of the parent block (whether the original block or a partition of the original block), some split-types may be disallowed (not allowed to be used) if one or more of the resulting sub-partitions is not a valid or supported block size. For example, a split-type for a block that results in a sub-block having horizontal or vertical dimension of less than 4 pixels may not be used in the recursive partitioning.

[0081] The proposed partitioning scheme 700 may be referred to as a 7-ary scheme as, absent a minimum block-size constraint and/or an invalid or unsupported block size, each block and each partition (i.e., each sub-block) of the block may be partitioned using one of 7 available partition types or split-types (e.g., during recursive partitioning).

[0082] The split-types include, from left to right, a no split-type, four uneven unidirectional quad split-types labeled 1:2:4:1, in particular two uneven vertical quad split-types (two vertical split-types) and two uneven horizontal quad split-types (two horizontal split-types), and two even binary split-types labeled 1:1. A no split-type means that the current block is not (further) partitioned. The uneven unidirectional quad split-types use a vertical or horizontal partition only partition where the first 1/8 of the N pixels along the first axis are assigned to a first partition (or first sub-block), the next 2/8 of the N pixels along first axis are assigned to a second partition (or second sub-block), the next 4/8 of the N pixels along first axis are assigned to a third partition (or third sub-block), and the final 1/8 N of the pixels along the first axis are assigned to a fourth partition (or fourth sub-block). The pixel dimension along the second axis orthogonal to the first axis for each sub-block is the original dimension of the block. Examples using a 64 x 64 (N x N) block for the block to be partitioned in FIG. 7 follow. The sub-blocks of the first uneven vertical quad split-type would have the dimensions 8 x 64 (2^3 x 2^6) pixels, 16 x 64 (2^4 x 2^6) pixels, 32 x 64 (2^5 x 2^6) pixels, and 8 x 64 (2^3 x 2^6) pixels (more generally 1/8 * N x N, 2/8 * N x N, 4/8 * N x N, and 1/8 * N x N). The sub-blocks of the second uneven vertical quad split-type would be similarly determined and have the dimensions 8 x 64 (2^3 x 2^6) pixels, 32 x 64 (2^5 x 2^6) pixels, 16 x 64 (2^4 x 2^6) pixels, and 8 x 64 (2^3 x 2^6) pixels. Similarly, the sub-blocks of the first uneven horizontal quad split-type would have the dimensions 64 x 8 (2^6 x 2^3) pixels, 64 x 16 (2^6 x 2^4) pixels, 64 x 32 (2^6 x 2^5) pixels, and 64 x 8 (2^6 x 2^3) pixels. The sub-blocks of the second uneven horizontal quad split-type would have the dimensions 64 x 8 (2^6 x 2^3) pixels, 64 x 16 (2^6 x 2^4) pixels, 64 x 32 (2^6 x 2^5) pixels, and 64 x 8 (2^6 x 2^3) pixels.

[0083] The two even binary split-types labeled 1:1 shown in FIG. 7 (representing an even split between the block) correspond to respectively to the horizontal binary split-type and the

vertical binary split-type described above where the first 1/2 of the N pixels along the first axis are assigned to a first partition (or first sub-block), and the second 1/2 of the N pixels along first axis are assigned to a second partition (or second sub-block). The pixel dimension along the second axis orthogonal to the first axis for each sub-block is the original dimension of the block.

[0084] FIG. 8 is a diagram of a proposed partitioning scheme 800 using a 5-ary split-type. Each sub-partition can be recursively split until a minimum block-size constraint is reached. Further, and depending on the size of the parent block (whether the original block or a partition of the original block), some split-types may be disallowed (not allowed to be used) if one or more of the resulting sub-partitions is not a valid or supported block size. For example, a split-type for a block that results in a sub-block having horizontal or vertical dimension of less than 4 pixels may not be used in the recursive partitioning.

[0085] The proposed partitioning scheme 800 may be referred to as a 5-ary scheme as, absent a minimum block-size constraint and/or an invalid or unsupported block size, each block and each partition (i.e., each sub-block) of the block may be partitioned using one of 5 available partition types or split-types (e.g., during recursive partitioning).

[0086] The split-types include, from left to right, a no split-type, two H-type quad split-types, and two even binary split-types labeled 1:1. The no split-type and the two even binary split-types labeled 1:1 are the same as described with regards to FIG. 7, so further description is omitted.

[0087] The two H-type quad split-types are referred to as such because the partition forms an H shape due to the two square-shaped partitions centered in the block with two unidirectional split-types on opposing sides. For the H-type split-type, the first 1/4 of the pixels along the first axis form a first partition (or first sub-block) with a dimension along the second axis (e.g., N or 2N) the same as the original block. The second 1/2 of the pixels along the first axis are split in parallel with the first axis along the center to form two partitions (second and third sub-blocks) of equal size. The final 1/4 of the pixels along the first axis form a fourth partition (or fourth sub-block) with a dimension along the second axis the same as the original block (e.g., N or 2N). That is, two unidirectional partitions (the first and fourth) have the same dimensions / size.

[0088] Examples using a 64 x 64 (N x N) (2^6 x 2^6) block for the block to be partitioned in FIG. 8 follow. The sub-blocks of the first H-type quad split-type would have the dimensions 16 x 64 (N/4 x N) (2^4 x 2^6) pixels, 32 x 32 (N/2 x N/2) (2^5 x 2^5) pixels, 32 x 32 (N/2 x N/2)

(2^5 x 2^5) pixels, and 16 x 64 (N/4 x N) (2^4 x 2^6) pixels. The sub-blocks of the second H-type quad split-type would have the dimensions 64 x 16 (N x N/4) (2^6 x 2^4) pixels, 32 x 32 (N/2 x N/2) (2^5 x 2^5) pixels, 32 x 32 (N/2 x N/2) (2^5 x 2^5) pixels, and 64 x 16 (N x N/4) (2^6 x 2^4) pixels.

[0089] FIG. 9 is a diagram of a proposed partitioning scheme 900 using a 9-ary split-type. Each sub-partition can be recursively split until a minimum block-size constraint is reached. Further, and depending on the size of the parent block (whether the original block or a partition of the original block), some split-types may be disallowed (not allowed to be used) if one or more of the resulting sub-partitions is not a valid or supported block size. For example, a split-type for a block that results in a sub-block having horizontal or vertical dimension of less than 4 pixels may not be used in the recursive partitioning.

[0090] The proposed partitioning scheme 900 may be referred to as a 9-ary scheme as, absent a minimum block-size constraint and/or an invalid or unsupported block size, each block and each partition (i.e., each sub-block) of the block may be partitioned using one of 9 available partition types (e.g., during recursive partitioning). Because the scheme 900 is a combination of the 7-ary scheme 700 and the 5-ary scheme 800, the description of the split-types is omitted here.

[0091] FIG. 10 is a flowchart diagram of an example of coding a block using split-types in accordance with implementations of this disclosure. Coding the block according to the method or technique 1000 of FIG. 10 may be implemented in an encoder, a decoder, or both, such as the encoder 400 shown in FIG. 4 or the decoder 500 shown in FIG. 5. The technique 1000 can be implemented, for example, as a software program that may be executed by computing devices such as a computing device 100. The software program can include machine-readable instructions that may be stored in a memory such as the memory 110, and that, when executed by a processor, such as the processor 120, may cause the computing device to perform the technique 1000. The technique 1000 can be implemented using specialized hardware or firmware. Multiple processors, memories, or both, may be used.

[0092] The technique 1000 starts with identifying a coding unit / superblock / block (original or parent) at operation 1010, also referred to as a current block to be coded (encoded or decoded). An encoder may identify the current block to be coded using a scan order, such as raster scan order. A decoder may identify the current block to be coded from an encoded bitstream.

[0093] The split-type from the partitioning scheme next identified. Optionally, the

partitioning scheme may be identified. Identifying the split-type may include identifying the block size (e.g., dimensions) at 1020 and using the dimensions of the current block at 1030 to identify the available split-types, also referred to as the set of split-types supported. The initial size or pixel dimensions of each coding unit / superblock used to code the frame may comprise 64x64 pixels or 128x128 pixels at operation 1020. At operation 1030, for example, all split-types of a particular partitioning scheme may be available where the current block comprises 64x64 pixels. Some split-types may be excluded because of block size restrictions, e.g., one of the sub-partitions in the split-type may be unsupported size (such as below a minimum size in one dimension).

[0094] The encoder, such as the encoder 400, would identify the split-type at operation 1040 from the available split-types at operation 1030. For example, the encoder would select the partitioning scheme from the available partitioning schemes including one or more of the partitioning schemes 700, 800, or 900 according to the techniques described above with regards to FIG. 6. The encoder can then signal a symbol (e.g., an index) with a block (such as in the block header) to indicate what split-type was used for partitioning, where each split-type of the partitioning scheme (or a sub-set of those split-types—such as those available based on the block size) has a unique index. In some implementations, an index may not be transmitted. For example, a split-type may conform to a default split type such that the decoder knows to select the default split type when no index is received. In some implementations, identifying the available split-types at operation 1030 may include selecting one of multiple available sets of partitioning schemes. For example, and depending on block size, one or more of the available partitioning schemes may be excluded. The encoder may signal, when more than one partitioning scheme is available, an index including the partitioning scheme as well as the index for the split-type if sent by the encoder (e.g., instead of inferred by the decoder).

[0095] For a decoder, the block dimensions may be optionally identified at operation 1020 so as to optionally identify the available split-types at operation 1030. For example, if multiple partitioning schemes are available to both the encoder and decoder, the block dimensions may be used to infer the partitioning scheme used for partitioning the block (e.g., the available split-types identified at operation 1030). In some implementations where multiple partitioning schemes are available, the decoder may not identify the block dimensions at operation 1020 to identify the available split-types at operation 1030. Instead, the decoder may identify the available split-types at operation 1030 by decoding a symbol

from the encoded bitstream representing the partitioning scheme used by the encoder. When only one partitioning scheme is available, operations 1020 and 1030 may not be required.

[0096] For a decoder, identifying the split-type at 1040 may include decoding a symbol (e.g., from the encoded bitstream from the block header) identifying a split-type from the available split-types (e.g., from the partitioning scheme). In some implementations (e.g., for certain block sizes), only one split-type may be available (generally the no-split option). In such implementations, that split-type may be automatically chosen at 1040. That is, there may be a default split-type when no split-type is signaled.

[0097] Subsequently, each sub-partition in the chosen split-type is processed recursively in depth-first order in the same manner. In the example of FIG. 10, the (partition) block size is identified at 1020, the available split-types are identified at 1030 after eliminating invalid ones from the partitioning scheme, and the split-type is identified at 1040 (e.g., by rate-distortion optimization at an encoder or decoding a symbol indicating chosen split-type at a decoder). If the no split split-type is signaled or chosen for a partition, then the recursive decomposition terminates. The encoder encodes respective sub-partition(s) of the current block as described above with regards to the encoder 400 (e.g., prediction, transformation, quantization, entropy coding), along with information needed by the decoder to decode the current block, and the decoder decodes other symbols indicating the prediction mode, motion vector(s) if any, coded prediction residues, etc., for the end or terminal block as described above with regards to the decoder 500. Note that because of the way the split-types are designed, for any parent power-of-two sized blocks, the end blocks are also guaranteed to be power-of-two blocks (2^m x 2^n, where m and n are integers equal to or greater than 1 or, in some implementations, are integers equal to or greater than 2).

[0098] The technique 1000 is one example of how to implement the teachings herein. In some examples, decoding does not require an identification of the block dimensions and the available split-types. This is because the decoder may directly receive the identity of the split-type in the bitstream (operation 1040) for the current block to be coded (operation 1010). In some examples, an encoder may not identify the available split-types at operation 1030 based on the identified block dimension (operation 1020). Instead, for example, the block (or sub-block) to be coded that is identified at 1010 may be tested against all available split-types of one or more of the partitioning schemes and the split-type(s) resulting in one or more partitions (sub-blocks) below a defined size (e.g., below a minimum block size) are eliminated from selection. In other words, operations 1020 and 1030 may be swapped.

[0099] Stated generally, for a current block to be encoded, an encoder determines sub-blocks of the current block using a recursive partitioning scheme that includes only a no split split-type, binary split-types, and at least two quad split-types that each result in at least two partitions of different dimensions, and then encodes the sub-blocks. For a current block to be decoded, a decoder identifies sub-blocks of the current block partitioned using a recursive partitioning scheme that includes only a no split split-type, binary split-types, and at least two quad split-types that each result in at least two partitions of different dimensions, and then decodes the sub-blocks to reconstruct the current block.

[0100] According to the disclosure herein, improvements to compression efficiency may be achieved using one or more unique block-partitioning schemes that are limited to a no split split-type, binary split-types, and quad split-types that result in at least two partitions of different dimensions.

[0101] As used herein, the terms "optimal", "optimized", "optimization", or other forms thereof, are relative to a respective context and are not indicative of absolute theoretic optimization unless expressly specified herein.

[0102] The words "example" or "embodiment" are used herein to mean serving as an example, instance, implementation, or illustration. Any aspect or design described herein as using these terms is not necessarily to be construed as preferred or advantageous over other aspects or designs. Rather, use of the words is intended to present concepts in a concrete fashion. Relatedly, use of the term "an embodiment" or "one embodiment" or "an implementation" or "one implementation" throughout is not intended to mean the same embodiment or implementation unless described as such. As used herein, the terms "determine" and "identify", or any variations thereof, includes selecting, ascertaining, computing, looking up, receiving, determining, establishing, obtaining, or otherwise identifying or determining in any manner whatsoever using one or more of the devices described herein.

[0103] As used in this application, the term "or" is intended to mean an inclusive "or" rather than an exclusive "or". That is, unless specified otherwise, or clear from context, "X includes A or B" is intended to mean any of the natural inclusive permutations. That is, if X includes A; X includes B; or X includes both A and B, then "X includes A or B" is satisfied under any of the foregoing instances. In addition, the articles "a" and "an" as used in this application and the appended claims should generally be construed to mean "one or more" unless specified otherwise or clear from context to be directed to a singular form.

[0104] Further, for simplicity of explanation, although the figures and descriptions herein may include sequences or series of steps or stages, elements of the methods disclosed herein can occur in various orders and/or concurrently. Additionally, elements of the methods disclosed herein may occur with other elements not explicitly presented and described herein. Furthermore, one or more elements of the methods described herein may be omitted from implementations of methods in accordance with the disclosed subject matter.

[0105] The implementations of the transmitting computing and communication device 100A and/or the receiving computing and communication device 100B (and the algorithms, methods, instructions, etc. stored thereon and/or executed thereby) can be realized in hardware, software, or any combination thereof. The hardware can include, for example, computers, intellectual property (IP) cores, application-specific integrated circuits (ASICs), programmable logic arrays, optical processors, programmable logic controllers, microcode, microcontrollers, servers, microprocessors, digital signal processors or any other suitable circuit. In the claims, the term "processor" should be understood as encompassing any of the foregoing hardware, either singly or in combination. The terms "signal" and "data" are used interchangeably. Further, portions of the transmitting computing and communication device 100A and the receiving computing and communication device 100B do not necessarily have to be implemented in the same manner.

[0106] Further, in one implementation, for example, the transmitting computing and communication device 100A or the receiving computing and communication device 100B can be implemented using a computer program that, when executed, carries out any of the respective methods, algorithms and/or instructions described herein. In addition, or alternatively, for example, a special purpose computer/processor can be utilized which can contain specialized hardware for carrying out any of the methods, algorithms, or instructions described herein.

[0107] The transmitting computing and communication device 100A and receiving computing and communication device 100B can, for example, be implemented on computers in a real-time video system. Alternatively, the transmitting computing and communication device 100A can be implemented on a server and the receiving computing and communication device 100B can be implemented on a device separate from the server, such as a hand-held communications device. In this instance, the transmitting computing and communication device 100A can encode content using an encoder 400 into an encoded video signal and transmit the encoded video signal to the communications device. In turn, the

communications device can then decode the encoded video signal using a decoder 500. Alternatively, the communications device can decode content stored locally on the communications device, for example, content that was not transmitted by the transmitting computing and communication device 100A. Other suitable transmitting computing and communication device 100A and receiving computing and communication device 100B implementation schemes are available. For example, the receiving computing and communication device 100B can be a generally stationary personal computer rather than a portable communications device and/or a device including an encoder 400 may also include a decoder 500.

[0108] Further, all or a portion of implementations can take the form of a computer program product accessible from, for example, a tangible computer-usable or computer-readable medium. A computer-usable or computer-readable medium can be any device that can, for example, tangibly contain, store, communicate, or transport the program for use by or in connection with any processor. The medium can be, for example, an electronic, magnetic, optical, electromagnetic, or a semiconductor device. Other suitable mediums are also available.

[0109] It will be appreciated that aspects can be implemented in any convenient form. For example, aspects may be implemented by appropriate computer programs which may be carried on appropriate carrier media which may be tangible carrier media (e.g. disks) or intangible carrier media (e.g. communications signals). Aspects may also be implemented using suitable apparatus which may take the form of programmable computers running computer programs arranged to implement the methods and/or techniques disclosed herein. Aspects can be combined such that features described in the context of one aspect may be implemented in another aspect.

[0110] The above-described implementations have been described to allow easy understanding of the application are not limiting. On the contrary, the application covers various modifications and equivalent arrangements included within the scope of the appended claims, which scope is to be accorded the broadest interpretation to encompass all such modifications and equivalent structure as is permitted under the law.

What is claimed is:

1. A method, comprising:

identifying, from an encoded bitstream, a current block to be decoded;

identifying one or more sub-blocks resulting from partitioning the current block using a recursive partitioning scheme, wherein the recursive partitioning scheme includes only a no split split-type, binary split-types, and at least two quad split-types that each result in at least two partitions of different dimensions; and

decoding the one or more sub-blocks to reconstruct the current block.

2. A method of claim 1, comprising:

identifying block dimensions of the current block; and

identifying the recursive partitioning scheme using the block dimensions, wherein identifying the one or more sub-blocks comprises identifying the split-type for the current block from available split-types of the recursive partitioning scheme.

3. The method of one of claim 1 or claim 2, wherein each partition of the recursive partitioning scheme has dimensions of powers-of-two.

4. The method of any one of claims 1 to 3, wherein the at least two quad split-types includes two horizontal split-types and two vertical split-types.

5. The method of any one of claims 1 to 4, wherein the recursive partitioning scheme is a 7-ary partitioning scheme.

6. The method of any one of claims 1 to 4, wherein the recursive partitioning scheme is a 9-ary partitioning scheme.

7. The method of claim 6, wherein the at least two quad split-types includes two H-type quad split-types.

8. The method of any one of claims 1 to 3, wherein the recursive partitioning scheme is a 5-ary partitioning scheme.

9. The method of claim 8, wherein the at least two quad split-types includes two H-type quad split-types.

10. A method, comprising:

identifying a current block to be encoded;

identifying one or more sub-blocks resulting from partitioning the current block using a recursive partitioning scheme, wherein the recursive partitioning scheme includes only a no split split-type, binary split-types, and at least two quad split-types that each result in at least two partitions of different dimensions; and

encoding the one or more sub-blocks.

11. The method of claim 10, comprising:

identifying block dimensions of the current block; and

identifying available split-types for the recursive partitioning scheme using the block dimensions, wherein identifying the one or more sub-blocks comprises identifying the split-type for the current block from the available split-types.

12. The method of claim 11, wherein the available split-types excludes any split-type of the recursive partitioning scheme that would result in a sub-block below a minimum block size.

13. The method of any one of claims 10 to 12, wherein the recursive partitioning scheme is a 9-ary partitioning scheme.

14. The method of any one of claims 10 to 12, wherein the recursive partitioning scheme is a 5-ary partitioning scheme.

15. The method of any one of claims 10 to 12, wherein the recursive partitioning scheme is a 7-ary partitioning scheme.

16. An apparatus, comprising:

a processor configured to perform the method of any one of claims 1 to 15.

17. An apparatus, comprising:

a processor; and

a memory storing instructions that cause the processor to perform the method of any one of claims 1 to 15.

18. A non-transitory computer-readable storage medium storing instructions for performing the method of any one of claims 1 to 15.

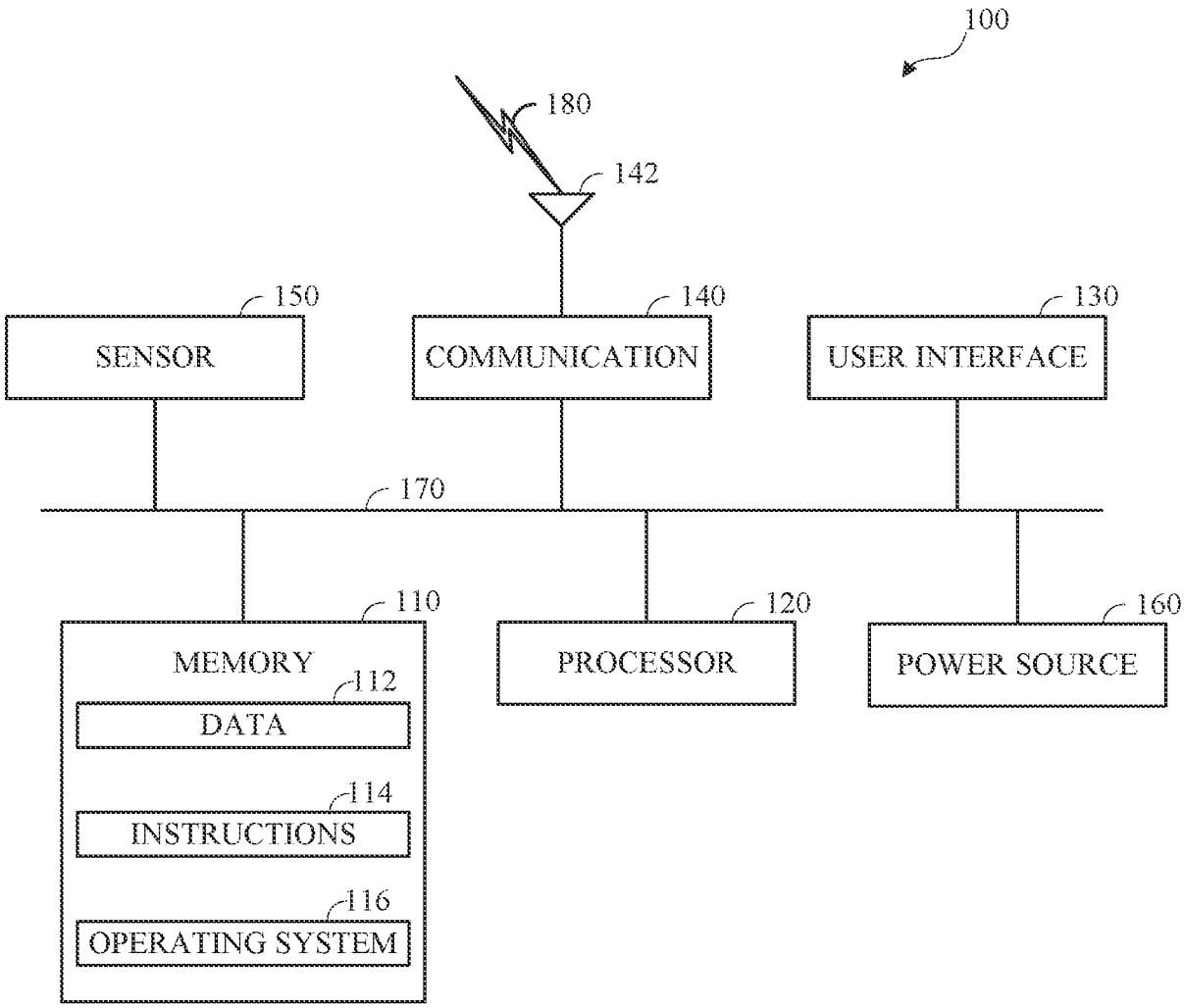

100
180
142
150
SENSOR
140
COMMUNICATION
130
USER INTERFACE
170
110
MEMORY
112
DATA
114
INSTRUCTIONS
116
OPERATING SYSTEM
120
PROCESSOR
160
POWER SOURCE

FIG. 1

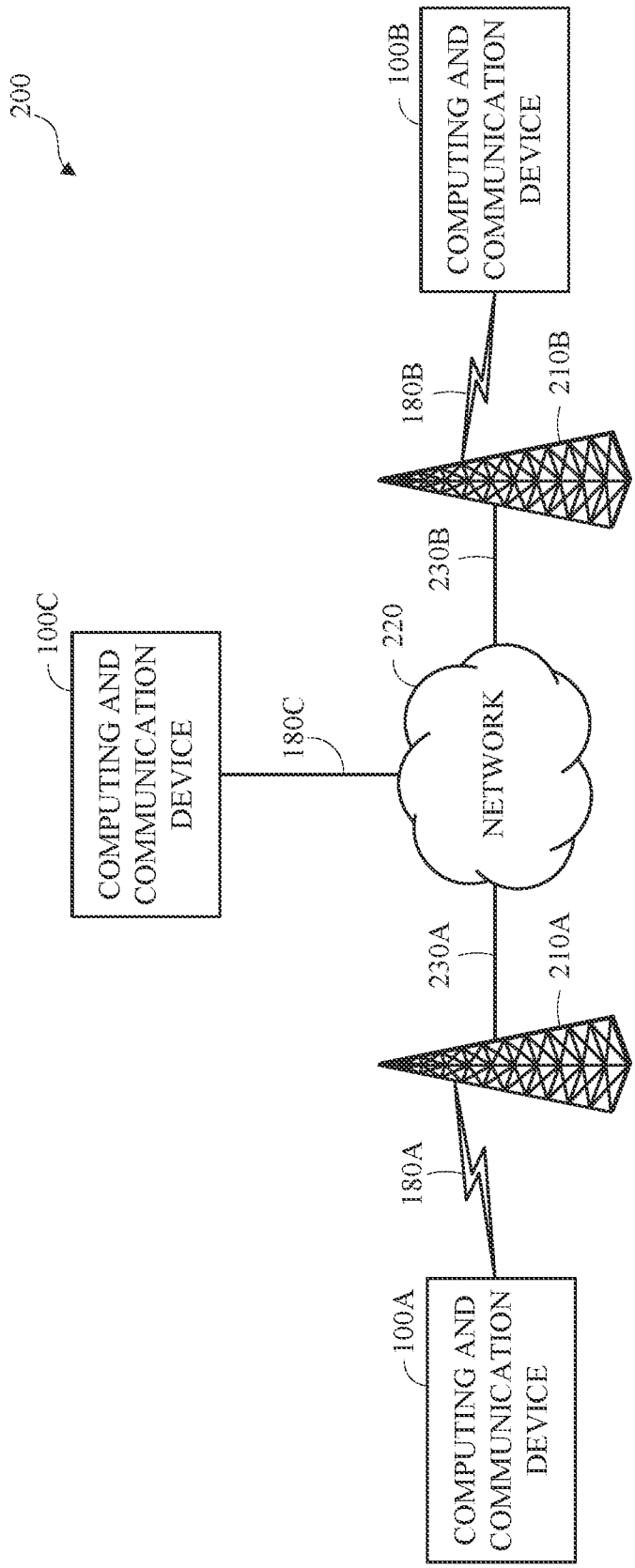

200
100B
COMPUTING AND COMMUNICATION DEVICE
180B
210B
230B
100C
COMPUTING AND COMMUNICATION DEVICE
180C
220
NETWORK
230A
210A
180A
100A
COMPUTING AND COMMUNICATION DEVICE

FIG. 2

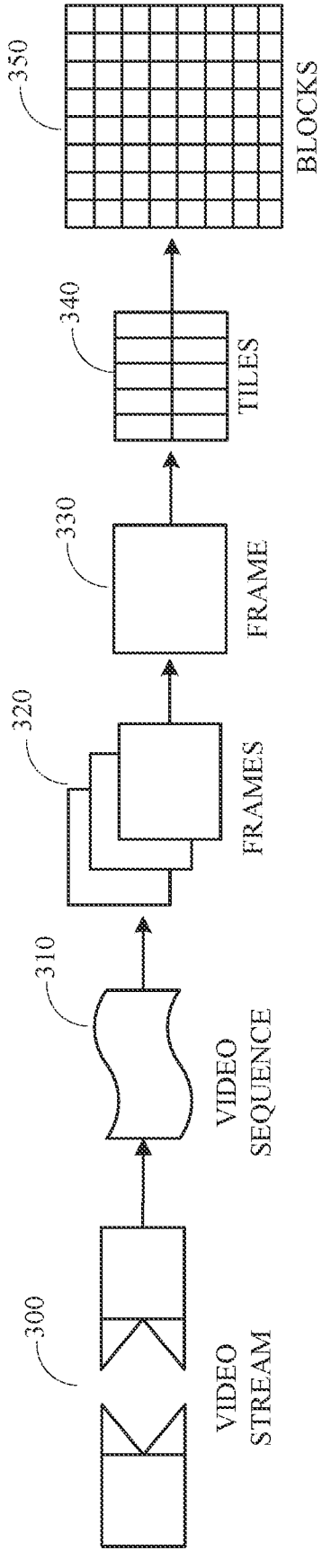

300
VIDEO STREAM
310
VIDEO SEQUENCE
320
FRAMES
330
FRAME
340
TILES
350
BLOCKS

FIG. 3

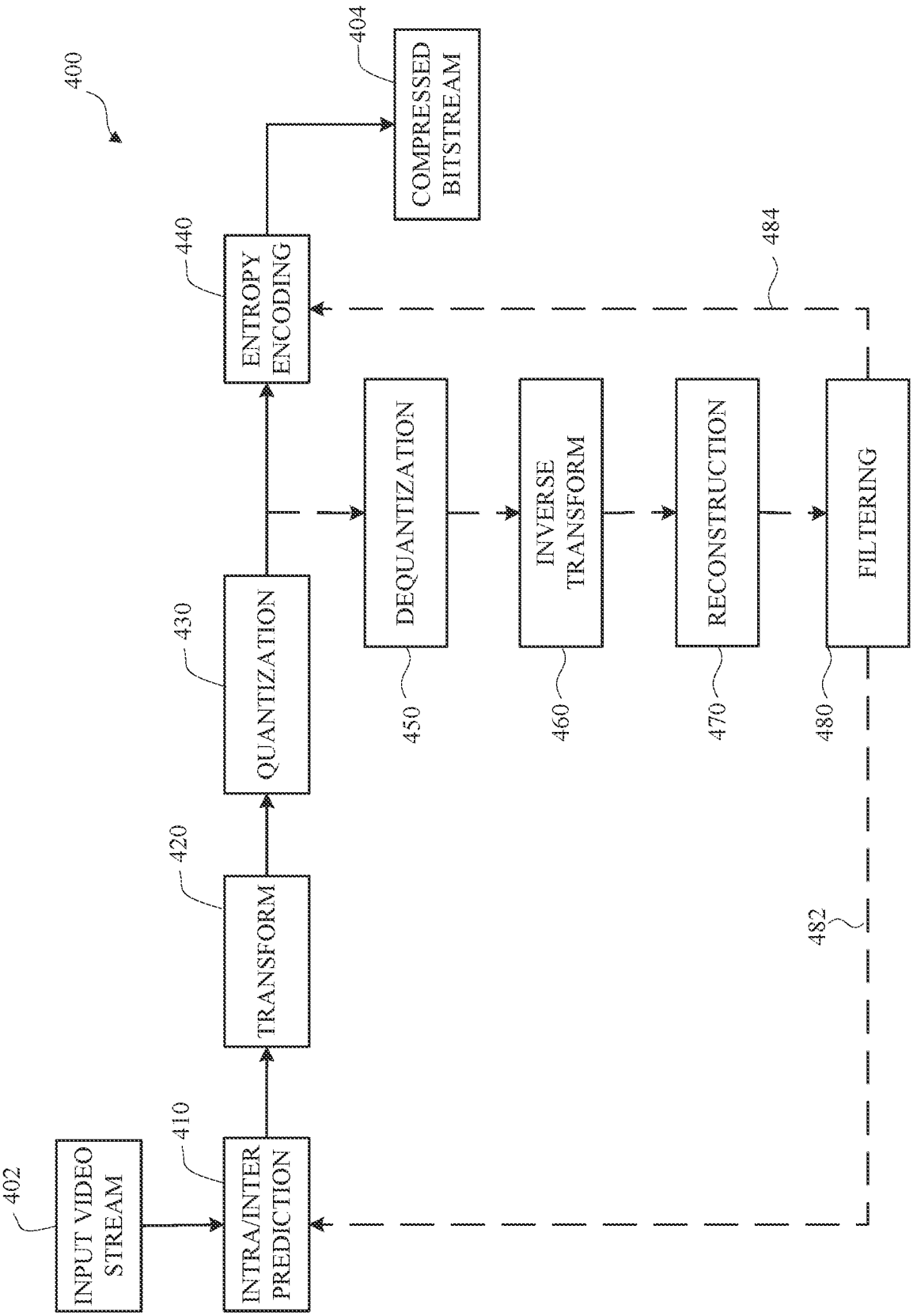

400
402
INPUT VIDEO STREAM
410
INTRA/INTER PREDICTION
420
TRANSFORM
430
QUANTIZATION
440
ENTROPY ENCODING
404
COMPRESSED BITSTREAM
450
DEQUANTIZATION
460
INVERSE TRANSFORM
470
RECONSTRUCTION
480
FILTERING
482
484

FIG. 4

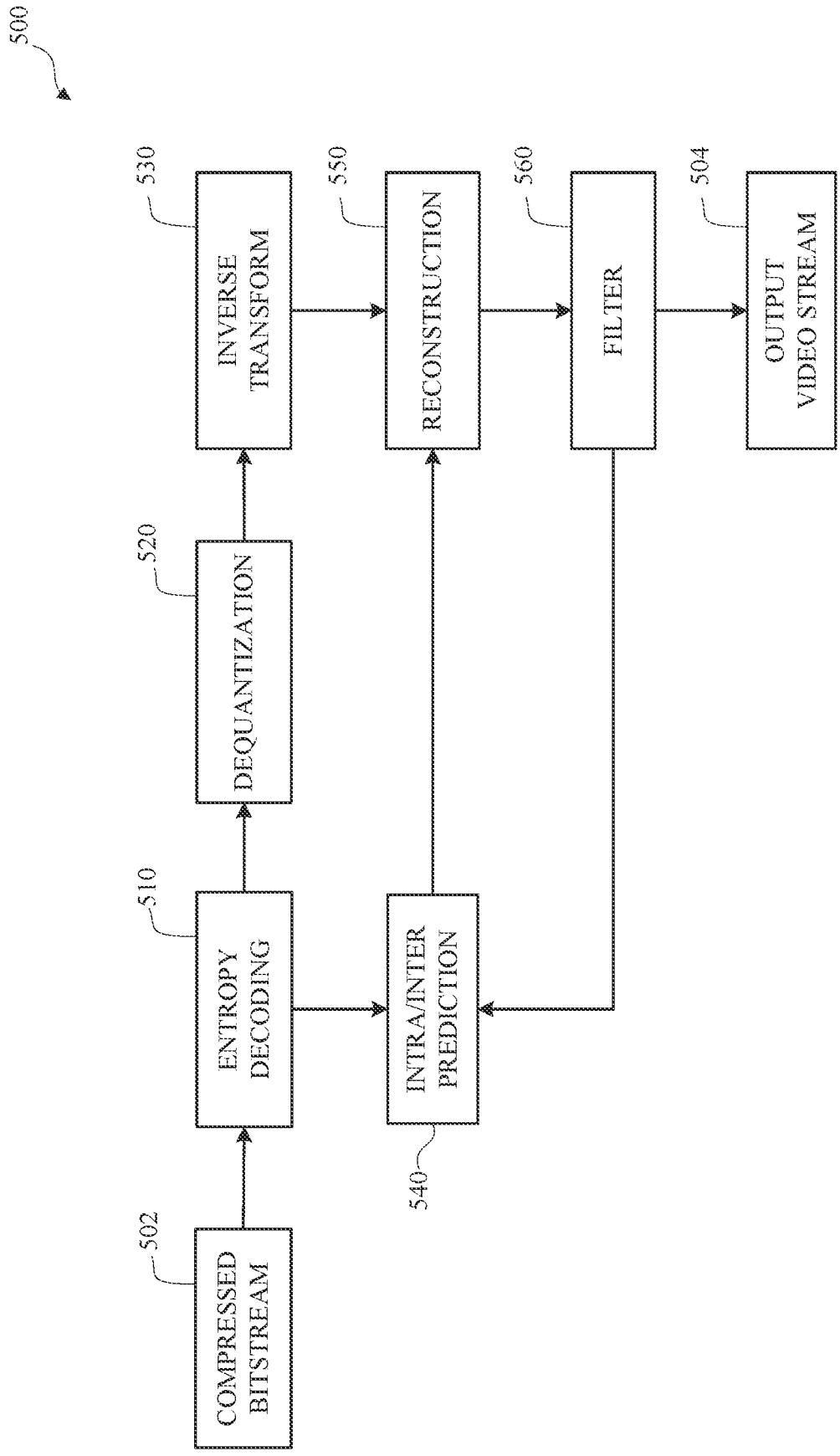

500
502
COMPRESSED BITSTREAM
510
ENTROPY DECODING
520
DEQUANTIZATION
530
INVERSE TRANSFORM
540
INTRA/INTER PREDICTION
550
RECONSTRUCTION
560
FILTER
504
OUTPUT VIDEO STREAM

FIG. 5

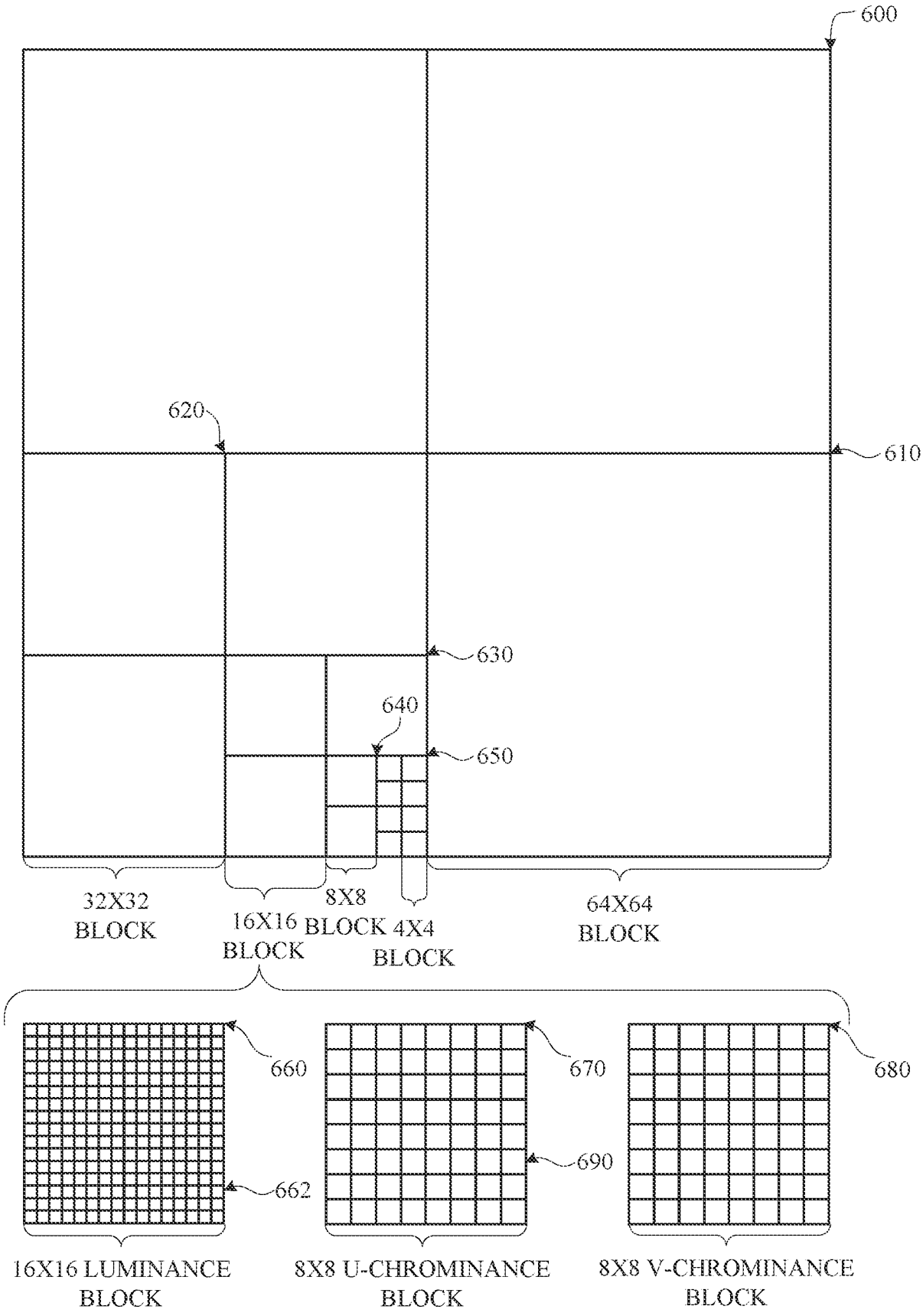

600
620
610
630
640
650
32X32 BLOCK
16X16 BLOCK
8X8 BLOCK
4X4 BLOCK
64X64 BLOCK
660
662
670
690
680
16X16 LUMINANCE BLOCK
8X8 U-CHROMINANCE BLOCK
8X8 V-CHROMINANCE BLOCK

FIG. 6

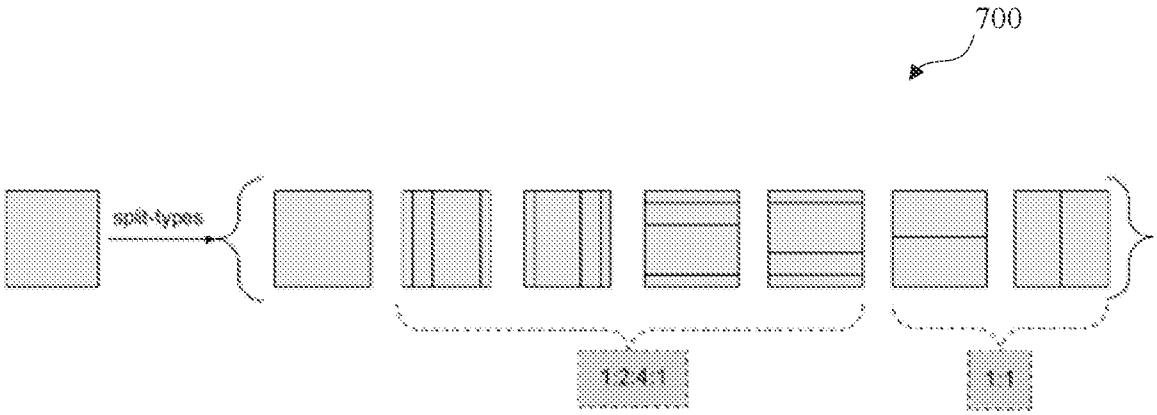


FIG. 7

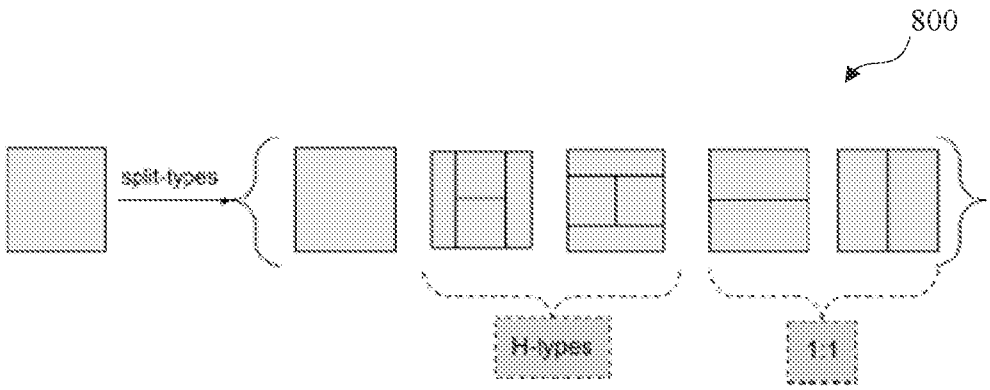


FIG. 8

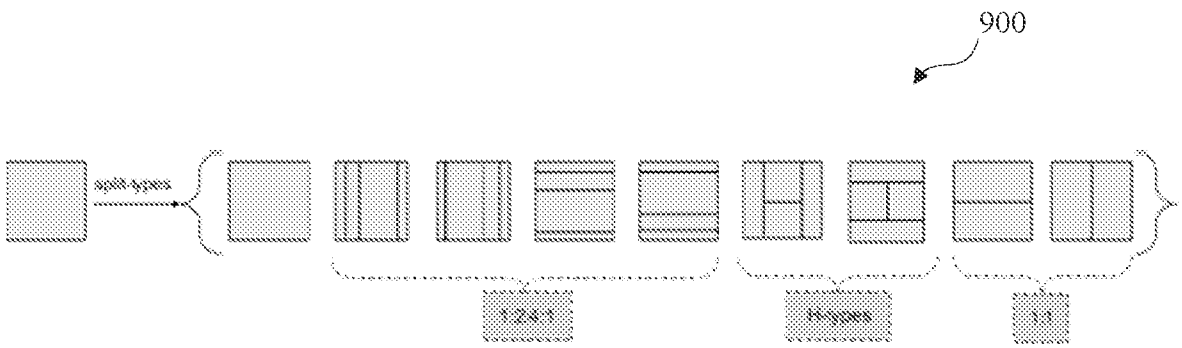


FIG. 9

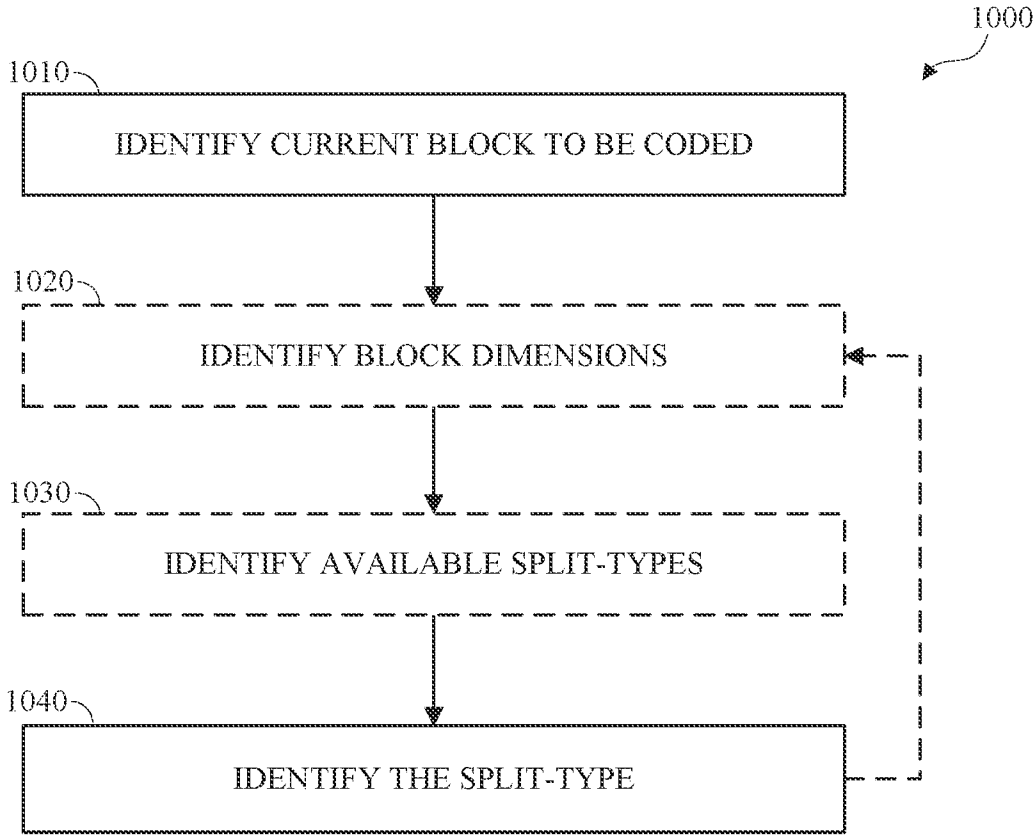

1000
1010
IDENTIFY CURRENT BLOCK TO BE CODED
1020
IDENTIFY BLOCK DIMENSIONS
1030
IDENTIFY AVAILABLE SPLIT-TYPES
1040
IDENTIFY THE SPLIT-TYPE

FIG. 10

INTERNATIONAL SEARCH REPORT

International application No
PCT/US2023/082724

A. CLASSIFICATION OF SUBJECT MATTER
INV. H04N19/119
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.
X	WO 2020/156572 A1 (BEIJING BYTEDANCE NETWORK TECH CO LTD [CN]; BYTEDANCE INC [US]) 6 August 2020 (2020-08-06)	1-11, 13-18
Y	the whole document	12

T	ZHANG KAI ET AL: "Advanced Block Partitioning Methods Beyond VVC", 2022 IEEE INTERNATIONAL SYMPOSIUM ON CIRCUITS AND SYSTEMS (ISCAS), IEEE, 27 May 2022 (2022-05-27), pages 1928-1932, XP034224539, DOI: 10.1109/ISCAS48785.2022.9937574 [retrieved on 2022-11-11] the whole document	

	-/--	

[X] Further documents are listed in the continuation of Box C.

[X] 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
7 March 2024	15/03/2024

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	Raeymaekers, Peter

1

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

INTERNATIONAL SEARCH REPORT

International application No
PCT/US2023/082724

C(Continuation). DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	ZHANG JIAQI ET AL: "Recent Development of AVS Video Coding Standard: AVS3", 2019 PICTURE CODING SYMPOSIUM (PCS), IEEE, 12 November 2019 (2019-11-12), pages 1-5, XP033688131, DOI: 10.1109/PCS48520.2019.8954503 [retrieved on 2020-01-08] the whole document	1,8,9
Y	WO 2020/048466 A1 (HUAWEI TECH CO LTD [CN]) 12 March 2020 (2020-03-12) page 1	12
A	BROSS B ET AL: "Versatile Video Coding (Draft 10)", 19. JVET MEETING; 20200622 - 20200701; TELECONFERENCE; (THE JOINT VIDEO EXPLORATION TEAM OF ISO/IEC JTC1/SC29/WG11 AND ITU-T SG.16), , no. JVET-S2001 4 September 2020 (2020-09-04), XP030289618, Retrieved from the Internet: URL:http://phenix.int-evry.fr/jvet/doc_end_user/documents/19_Teleconference/wg11/JVET-S2001-v17.zip JVET-S2001-vH.docx [retrieved on 2020-09-04] 6.4 Availability processes 7.4.3.4 Sequence parameter set RBSP semantics, p.107, sps_log2_diff_min_qt_min_intra_slice_luma	12

1

INTERNATIONAL SEARCH REPORT

Information on patent family members

International application No
PCT/US2023/082724

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
WO 2020156572 A1	06-08-2020	CN 113273197 A	17-08-2021
		CN 113273217 A	17-08-2021
		CN 113366855 A	07-09-2021
		US 2021360242 A1	18-11-2021
		US 2021360243 A1	18-11-2021
		US 2021360244 A1	18-11-2021
		WO 2020156572 A1	06-08-2020
		WO 2020156573 A1	06-08-2020
		WO 2020156574 A1	06-08-2020
WO 2020048466 A1	12-03-2020	AU 2019334017 A1	08-04-2021
		AU 2023202264 A1	04-05-2023
		BR 112021003999 A2	25-05-2021
		CA 3111043 A1	12-03-2020
		CN 112673626 A	16-04-2021
		CN 116208767 A	02-06-2023
		EP 3808081 A1	21-04-2021
		JP 7286757 B2	05-06-2023
		JP 2021535645 A	16-12-2021
		JP 2023111927 A	10-08-2023
		KR 20210024114 A	04-03-2021
		KR 20230054917 A	25-04-2023
		US 2021258618 A1	19-08-2021
		US 2023171437 A1	01-06-2023
		WO 2020048466 A1	12-03-2020
		ZA 202101704 B	28-04-2022