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

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
01 February 2024 (01.02.2024)

WIPO | PCT

(10) International Publication Number
WO 2024/025638 A1

(51) International Patent Classification:
G06T 17/20 (2006.01) *G06T 9/00* (2006.01)
G06T 17/00 (2006.01) *G06T 19/00* (2011.01)

(21) International Application Number:
PCT/US2023/023296

(22) International Filing Date:
24 May 2023 (24.05.2023)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
63/392,409 26 July 2022 (26.07.2022) US
18/312,338 04 May 2023 (04.05.2023) US

(71) Applicant: **TENCENT AMERICA LLC** [US/US]; 2747 Park Boulevard, Palo Alto, California 94306 (US).

(72) Inventors: **NGUYEN CANH, Thuong**; c/o TENCENT AMERICA LLC, 2747 Park Boulevard, Palo Alto, California 94306 (US). **ZHANG, Xiang**; c/o TENCENT AMERICA LLC, 2747 Park Boulevard, Palo Alto, California 94306 (US). **XU, Xiaozhong**; c/o TENCENT AMERICA LLC, 2747 Park Boulevard, Palo Alto, California 94306 (US). **LIU, Shan**; c/o TENCENT AMERICA LLC, 2747 Park Boulevard, Palo Alto, California 94306 (US).

(74) Agent: **RABENA, John F.** et al.; SUGHRUE MION, PLLC, 2000 Pennsylvania Ave., N.W., Suite 9000, Washington, District of Columbia 20006 (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, 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))*

WO 2024/025638 A1

(54) Title: METHODS FOR INSTANCE-BASED MESH CODING

(57) Abstract: A method and apparatus comprising computer code configured to cause a processor or processors to obtain an input mesh corresponding to volumetric data of at least one three-dimensional (3D) visual content, partitioning the input mesh into a plurality of submeshes, each of the submeshes comprising a respective one of instances of objects, determine a similarity between at least two of the instance of objects, grouping the submeshes based on the determined similarity, and code the volumetric data based on grouping the submeshes.

METHODS FOR INSTANCE-BASED MESH CODING

CROSS REFERENCE TO RELATED APPLICATION

[1] The present application claims priority to provisional application US 63/392,409 filed on July 26, 2022, and U.S. Application No. 18/312,338, filed on May 4, 2023, the disclosures of which are incorporated herein by reference in their entirety.

BACKGROUND

1. Field

[2] The present disclosure is directed a set of advanced video coding technologies including both lossless and lossy mesh coding technologies based on instances of meshes.

2. Description of Related Art

[3] The advances in 3D capture, modeling, and rendering have promoted the ubiquitous presence of 3D contents across several platforms and devices. Nowadays, it is possible to capture a baby's first step in one continent and allow the grandparents to see (and maybe interact) and enjoy a full immersive experience with the child in another continent. Nevertheless, in order to achieve such realism, models are becoming ever more sophisticated, and a significant amount of data is linked to the creation and consumption of those models.

[4] VMesh is an ongoing MPEG standard to compress the static and dynamic mesh. VMesh separates the input mesh into a simplified base mesh and a residual mesh. The base mesh may be encoded at high quality while the remainder mesh may be encoded with subdivision surface fitting and displacement encoding to exploit local characteristic.

[5] However, a complex mesh often contains information about multiple instances to relate associate texture maps. This information is available at the encoding time. On the other hand, a mesh could be segmented into several parts based on their characteristics. For example, there are more polygons in the face region of a human mesh.

[6] As such, a constant quantization step size applied for all instances, objects, parts in mesh leads to a large quantization error, mesh regions may not be not equally important, the number of faces may be varied significantly in different parts of a mesh, and a base mesh could be simpler than original mesh and the displacement, thus could require less accuracy in bitdepth. And for any of those reasons there is therefore a desire for technical solutions to such problems that arose in video coding technology.

SUMMARY

[7] There is included a method and apparatus comprising memory configured to store computer program code and a processor or processors configured to access the computer program code and operate as instructed by the computer program code. The computer program is configured to cause the processor implement obtaining code configured to cause the at least one processor to obtain in input mesh corresponding to volumetric data of at least one three-dimensional (3D) visual content; partitioning code configured to cause the at least one processor to partition the input mesh corresponding to the volumetric data into a plurality of submeshes, each of the submeshes comprising a respective one of instances of objects; determining code configured to cause the at least one processor to determine a similarity between at least two of the instances of objects; grouping code configured to cause at least one processor to group submeshes based on the determined similarity; and coding code configured to cause the at least one processor to code the volumetric data based on the grouping of the plurality of submeshes.

[8] According to exemplary embodiments, determining the similarity may include comparing bounding boxes of the at least two of the submeshes.

[9] According to exemplary embodiments, comparing the bounding boxes may include comparing ratios of dimensions of the bounding boxes.

[10] According to exemplary embodiments, coding the volumetric data may include signaling different ones of codecs respective to ones of the at least two of the submeshes.

[11] According to exemplary embodiments, coding the volumetric data may include signaling prediction of a first one of the at least two of the submeshes based on a prediction of a second one of the at least two of the submeshes.

[12] According to exemplary embodiments, coding the volumetric data may include signaling only a single one prediction index for an entirety of the first one of the at least two of the submeshes.

[13] According to exemplary embodiments, coding the volumetric data may include signaling only a plurality of prediction indices for the first one of the at least two of the submeshes.

[14] According to exemplary embodiments, a first one of the plurality of prediction indices may be based on a first prediction index, a second one of the plurality of prediction indices may be based on a second prediction index, the first prediction index may be based on a first other prediction index which is of the second one of the at least two submeshes, and the second one of the plurality of prediction indices is based on a second other prediction index which is of a third one of the at least two submeshes.

[15] According to exemplary embodiments, wherein signaling prediction of the first one of the at least two of the submeshes may include signaling a first prediction index and a second prediction index, and the first prediction index may be a single index that is collectively set to

each of a plurality of vertices of the first one of the at least two of the submeshes, and the second prediction index may be set to at least one other vertex of the first one of the at least two of the submeshes.

[16] According to exemplary embodiments, at least one of the first prediction index and the second prediction index may be based on a prediction index set to at least one vertex of the second one of the at least two of the submeshes.

BRIEF DESCRIPTION OF THE DRAWINGS

[17] Further features, nature, and various advantages of the disclosed subject matter will be more apparent from the following detailed description and the accompanying drawings in which:

[18] Fig. 1 is a schematic illustrations of a diagram in accordance with embodiments.

[19] Fig. 2 is a simplified block diagram in accordance with embodiments.

[20] Fig. 3 is a simplified illustration in accordance with embodiments.

[21] Fig. 4 is a simplified illustration in accordance with embodiments.

[22] Fig. 5 is a simplified illustration in accordance with embodiments.

[23] Fig. 6 is a simplified illustration in accordance with embodiments.

[24] Fig. 7 is a simplified illustration in accordance with embodiments.

[25] Fig. 8 is a simplified illustration in accordance with embodiments.

[26] Fig. 9 is a simplified illustration in accordance with embodiments.

[27] Fig. 10 is a simplified illustration in accordance with embodiments.

[28] Fig. 11 is a simplified illustration in accordance with embodiments.

[29] Fig. 12 is a simplified illustration in accordance with embodiments.

[30] Fig. 13 is a simplified flow diagram in accordance with embodiments.

[31] Fig. 14 is a simplified flow diagram in accordance with embodiments.

[32] Fig. 15 is a simplified flow diagram in accordance with embodiments.

[33] Fig. 16 is a simplified illustration in accordance with embodiments.

[34] Fig. 17 is a simplified illustration in accordance with embodiments.

[35] Fig. 18 is a simplified illustration in accordance with embodiments.

[36] Fig. 19 is a simplified illustration in accordance with embodiments.

[37] Fig. 20 is a simplified flow diagram in accordance with embodiments.

[38] Fig. 21 is a simplified illustration in accordance with embodiments.

[39] Fig. 22 is a simplified flow diagram in accordance with embodiments.

[40] Fig. 23 is a simplified flow diagram in accordance with embodiments.

[41] Fig. 24 is a simplified diagram in accordance with embodiments.

DETAILED DESCRIPTION

[42] The proposed features discussed below may be used separately or combined in any order. Further, the embodiments may be implemented by processing circuitry (e.g., one or more processors or one or more integrated circuits). In one example, the one or more processors execute a program that is stored in a non-transitory computer-readable medium.

[43] Fig. 1 illustrates a simplified block diagram of a communication system 100 according to an embodiment of the present disclosure. The communication system 100 may include at least

two terminals 102 and 103 interconnected via a network 105. For unidirectional transmission of data, a first terminal 103 may code video data at a local location for transmission to the other terminal 102 via the network 105. The second terminal 102 may receive the coded video data of the other terminal from the network 105, decode the coded data and display the recovered video data. Unidirectional data transmission may be common in media serving applications and the like.

[44] Fig. 1 illustrates a second pair of terminals 101 and 104 provided to support bidirectional transmission of coded video that may occur, for example, during videoconferencing. For bidirectional transmission of data, each terminal 101 and 104 may code video data captured at a local location for transmission to the other terminal via the network 105. Each terminal 101 and 104 also may receive the coded video data transmitted by the other terminal, may decode the coded data and may display the recovered video data at a local display device.

[45] In Fig. 1, the terminals 101, 102, 103 and 104 may be illustrated as servers, personal computers and smart phones but the principles of the present disclosure are not so limited. Embodiments of the present disclosure find application with laptop computers, tablet computers, media players and/or dedicated video conferencing equipment. The network 105 represents any number of networks that convey coded video data among the terminals 101, 102, 103 and 104, including for example wireline and/or wireless communication networks. The communication network 105 may exchange data in circuit-switched and/or packet-switched channels. Representative networks include telecommunications networks, local area networks, wide area networks and/or the Internet. For the purposes of the present discussion, the architecture and topology of the network 105 may be immaterial to the operation of the present disclosure unless explained herein below.

[46] Fig. 2 illustrates, as an example for an application for the disclosed subject matter, the placement of a video encoder and decoder in a streaming environment. The disclosed subject matter can be equally applicable to other video enabled applications, including, for example, video conferencing, digital TV, storing of compressed video on digital media including CD, DVD, memory stick and the like, and so on.

[47] A streaming system may include a capture subsystem 203, that can include a video source 201, for example a digital camera, creating, for example, an uncompressed video sample stream 213. That sample stream 213 may be emphasized as a high data volume when compared to encoded video bitstreams and can be processed by an encoder 202 coupled to the video source 201, which may be for example a camera as discussed above. The encoder 202 can include hardware, software, or a combination thereof to enable or implement aspects of the disclosed subject matter as described in more detail below. The encoded video bitstream 204, which may be emphasized as a lower data volume when compared to the sample stream, can be stored on a streaming server 205 for future use. One or more streaming clients 212 and 207 can access the streaming server 205 to retrieve copies 208 and 206 of the encoded video bitstream 204. A client 212 can include a video decoder 211 which decodes the incoming copy of the encoded video bitstream 208 and creates an outgoing video sample stream 210 that can be rendered on a display 209 or other rendering device (not depicted). In some streaming systems, the video bitstreams 204, 206 and 208 can be encoded according to certain video coding/compression standards. Examples of those standards are noted above and described further herein.

[48] Fig. 3 may be a functional block diagram of a video decoder 300 according to an embodiment of the present invention.

[49] A receiver 302 may receive one or more codec video sequences to be decoded by the decoder 300; in the same or another embodiment, one coded video sequence at a time, where the decoding of each coded video sequence is independent from other coded video sequences. The coded video sequence may be received from a channel 301, which may be a hardware/software link to a storage device which stores the encoded video data. The receiver 302 may receive the encoded video data with other data, for example, coded audio data and/or ancillary data streams, that may be forwarded to their respective using entities (not depicted). The receiver 302 may separate the coded video sequence from the other data. To combat network jitter, a buffer memory 303 may be coupled in between receiver 302 and entropy decoder / parser 304 ("parser" henceforth). When receiver 302 is receiving data from a store/forward device of sufficient bandwidth and controllability, or from an isosychronous network, the buffer 303 may not be needed, or can be small. For use on best effort packet networks such as the Internet, the buffer 303 may be required, can be comparatively large and can advantageously of adaptive size.

[50] The video decoder 300 may include a parser 304 to reconstruct symbols 313 from the entropy coded video sequence. Categories of those symbols include information used to manage operation of the decoder 300, and potentially information to control a rendering device such as a display 312 that is not an integral part of the decoder but can be coupled to it. The control information for the rendering device(s) may be in the form of Supplementary Enhancement Information (SEI messages) or Video Usability Information (VUI) parameter set fragments (not depicted). The parser 304 may parse / entropy-decode the coded video sequence received. The coding of the coded video sequence can be in accordance with a video coding technology or standard, and can follow principles well known to a person skilled in the art, including variable length coding, Huffman coding, arithmetic coding with or without context sensitivity, and so

forth. The parser 304 may extract from the coded video sequence, a set of subgroup parameters for at least one of the subgroups of pixels in the video decoder, based upon at least one parameters corresponding to the group. Subgroups can include Groups of Pictures (GOPs), pictures, tiles, slices, macroblocks, Coding Units (CUs), blocks, Transform Units (TUs), Prediction Units (PUs) and so forth. The entropy decoder / parser may also extract from the coded video sequence information such as transform coefficients, quantizer parameter values, motion vectors, and so forth.

[51] The parser 304 may perform entropy decoding / parsing operation on the video sequence received from the buffer 303, so to create symbols 313. The parser 304 may receive encoded data, and selectively decode particular symbols 313. Further, the parser 304 may determine whether the particular symbols 313 are to be provided to a Motion Compensation Prediction unit 306, a scaler / inverse transform unit 305, an Intra Prediction Unit 307, or a loop filter 311.

[52] Reconstruction of the symbols 313 can involve multiple different units depending on the type of the coded video picture or parts thereof (such as: inter and intra picture, inter and intra block), and other factors. Which units are involved, and how, can be controlled by the subgroup control information that was parsed from the coded video sequence by the parser 304. The flow of such subgroup control information between the parser 304 and the multiple units below is not depicted for clarity.

[53] Beyond the functional blocks already mentioned, decoder 300 can be conceptually subdivided into a number of functional units as described below. In a practical implementation operating under commercial constraints, many of these units interact closely with each other and can, at least partly, be integrated into each other. However, for the purpose of describing the

disclosed subject matter, the conceptual subdivision into the functional units below is appropriate.

[54] A first unit is the scaler / inverse transform unit 305. The scaler / inverse transform unit 305 receives quantized transform coefficient as well as control information, including which transform to use, block size, quantization factor, quantization scaling matrices, etc. as symbol(s) 313 from the parser 304. It can output blocks comprising sample values, that can be input into aggregator 310.

[55] In some cases, the output samples of the scaler / inverse transform 305 can pertain to an intra coded block; that is: a block that is not using predictive information from previously reconstructed pictures, but can use predictive information from previously reconstructed parts of the current picture. Such predictive information can be provided by an intra picture prediction unit 307. In some cases, the intra picture prediction unit 307 generates a block of the same size and shape of the block under reconstruction, using surrounding already reconstructed information fetched from the current (partly reconstructed) picture 309. The aggregator 310, in some cases, adds, on a per sample basis, the prediction information the intra prediction unit 307 has generated to the output sample information as provided by the scaler / inverse transform unit 305.

[56] In other cases, the output samples of the scaler / inverse transform unit 305 can pertain to an inter coded, and potentially motion compensated block. In such a case, a Motion Compensation Prediction unit 306 can access reference picture memory 308 to fetch samples used for prediction. After motion compensating the fetched samples in accordance with the symbols 313 pertaining to the block, these samples can be added by the aggregator 310 to the output of the scaler / inverse transform unit (in this case called the residual samples or residual

signal) so to generate output sample information. The addresses within the reference picture memory form where the motion compensation unit fetches prediction samples can be controlled by motion vectors, available to the motion compensation unit in the form of symbols 313 that can have, for example X, Y, and reference picture components. Motion compensation also can include interpolation of sample values as fetched from the reference picture memory when sub-sample exact motion vectors are in use, motion vector prediction mechanisms, and so forth.

[57] The output samples of the aggregator 310 can be subject to various loop filtering techniques in the loop filter unit 311. Video compression technologies can include in-loop filter technologies that are controlled by parameters included in the coded video bitstream and made available to the loop filter unit 311 as symbols 313 from the parser 304, but can also be responsive to meta-information obtained during the decoding of previous (in decoding order) parts of the coded picture or coded video sequence, as well as responsive to previously reconstructed and loop-filtered sample values.

[58] The output of the loop filter unit 311 can be a sample stream that can be output to the render device 312 as well as stored in the reference picture memory 557 for use in future inter-picture prediction.

[59] Certain coded pictures, once fully reconstructed, can be used as reference pictures for future prediction. Once a coded picture is fully reconstructed and the coded picture has been identified as a reference picture (by, for example, parser 304), the current reference picture 309 can become part of the reference picture buffer 308, and a fresh current picture memory can be reallocated before commencing the reconstruction of the following coded picture.

[60] The video decoder 300 may perform decoding operations according to a predetermined video compression technology that may be documented in a standard, such as ITU-T Rec. H.265.

The coded video sequence may conform to a syntax specified by the video compression technology or standard being used, in the sense that it adheres to the syntax of the video compression technology or standard, as specified in the video compression technology document or standard and specifically in the profiles document therein. Also necessary for compliance can be that the complexity of the coded video sequence is within bounds as defined by the level of the video compression technology or standard. In some cases, levels restrict the maximum picture size, maximum frame rate, maximum reconstruction sample rate (measured in, for example megasamples per second), maximum reference picture size, and so on. Limits set by levels can, in some cases, be further restricted through Hypothetical Reference Decoder (HRD) specifications and metadata for HRD buffer management signaled in the coded video sequence.

[61] In an embodiment, the receiver 302 may receive additional (redundant) data with the encoded video. The additional data may be included as part of the coded video sequence(s). The additional data may be used by the video decoder 300 to properly decode the data and/or to more accurately reconstruct the original video data. Additional data can be in the form of, for example, temporal, spatial, or signal-to-noise ratio (SNR) enhancement layers, redundant slices, redundant pictures, forward error correction codes, and so on.

[62] Fig. 4 may be a functional block diagram of a video encoder 400 according to an embodiment of the present disclosure.

[63] The encoder 400 may receive video samples from a video source 401 (that is not part of the encoder) that may capture video image(s) to be coded by the encoder 400.

[64] The video source 401 may provide the source video sequence to be coded by the encoder (303) in the form of a digital video sample stream that can be of any suitable bit depth (for example: 8 bit, 10 bit, 12 bit, ...), any colorspace (for example, BT.601 Y CrCB, RGB, ...) and

any suitable sampling structure (for example Y CrCb 4:2:0, Y CrCb 4:4:4). In a media serving system, the video source 401 may be a storage device storing previously prepared video. In a videoconferencing system, the video source 401 may be a camera that captures local image information as a video sequence. Video data may be provided as a plurality of individual pictures that impart motion when viewed in sequence. The pictures themselves may be organized as a spatial array of pixels, wherein each pixel can comprise one or more samples depending on the sampling structure, color space, etc. in use. A person skilled in the art can readily understand the relationship between pixels and samples. The description below focuses on samples.

[65] According to an embodiment, the encoder 400 may code and compress the pictures of the source video sequence into a coded video sequence 410 in real time or under any other time constraints as required by the application. Enforcing appropriate coding speed is one function of Controller 402. Controller controls other functional units as described below and is functionally coupled to these units. The coupling is not depicted for clarity. Parameters set by controller can include rate control related parameters (picture skip, quantizer, lambda value of rate-distortion optimization techniques, ...), picture size, group of pictures (GOP) layout, maximum motion vector search range, and so forth. A person skilled in the art can readily identify other functions of controller 402 as they may pertain to video encoder 400 optimized for a certain system design.

[66] Some video encoders operate in what a person skilled in the art readily recognizes as a "coding loop." As an oversimplified description, a coding loop can consist of the encoding part of an encoder 400 ("source coder" henceforth) (responsible for creating symbols based on an input picture to be coded, and a reference picture(s)), and a (local) decoder 406 embedded in the encoder 400 that reconstructs the symbols to create the sample data that a (remote) decoder also

would create (as any compression between symbols and coded video bitstream is lossless in the video compression technologies considered in the disclosed subject matter). That reconstructed sample stream is input to the reference picture memory 405. As the decoding of a symbol stream leads to bit-exact results independent of decoder location (local or remote), the reference picture buffer content is also bit exact between local encoder and remote encoder. In other words, the prediction part of an encoder "sees" as reference picture samples exactly the same sample values as a decoder would "see" when using prediction during decoding. This fundamental principle of reference picture synchronicity (and resulting drift, if synchronicity cannot be maintained, for example because of channel errors) is well known to a person skilled in the art.

[67] The operation of the "local" decoder 406 can be the same as of a "remote" decoder 300, which has already been described in detail above in conjunction with Fig. 3. Briefly referring also to Fig. 4, however, as symbols are available and en/decoding of symbols to a coded video sequence by entropy coder 408 and parser 304 can be lossless, the entropy decoding parts of decoder 300, including channel 301, receiver 302, buffer 303, and parser 304 may not be fully implemented in local decoder 406.

[68] An observation that can be made at this point is that any decoder technology except the parsing/entropy decoding that is present in a decoder also necessarily needs to be present, in substantially identical functional form, in a corresponding encoder. The description of encoder technologies can be abbreviated as they are the inverse of the comprehensively described decoder technologies. Only in certain areas a more detail description is required and provided below.

[69] As part of its operation, the source coder 403 may perform motion compensated predictive coding, which codes an input frame predictively with reference to one or more

previously-coded frames from the video sequence that were designated as "reference frames." In this manner, the coding engine 407 codes differences between pixel blocks of an input frame and pixel blocks of reference frame(s) that may be selected as prediction reference(s) to the input frame.

[70] The local video decoder 406 may decode coded video data of frames that may be designated as reference frames, based on symbols created by the source coder 403. Operations of the coding engine 407 may advantageously be lossy processes. When the coded video data may be decoded at a video decoder (not shown in Fig. 4), the reconstructed video sequence typically may be a replica of the source video sequence with some errors. The local video decoder 406 replicates decoding processes that may be performed by the video decoder on reference frames and may cause reconstructed reference frames to be stored in the reference picture memory 405, which may be for example a cache. In this manner, the encoder 400 may store copies of reconstructed reference frames locally that have common content as the reconstructed reference frames that will be obtained by a far-end video decoder (absent transmission errors).

[71] The predictor 404 may perform prediction searches for the coding engine 407. That is, for a new frame to be coded, the predictor 404 may search the reference picture memory 405 for sample data (as candidate reference pixel blocks) or certain metadata such as reference picture motion vectors, block shapes, and so on, that may serve as an appropriate prediction reference for the new pictures. The predictor 404 may operate on a sample block-by-pixel block basis to find appropriate prediction references. In some cases, as determined by search results obtained by the predictor 404, an input picture may have prediction references drawn from multiple reference pictures stored in the reference picture memory 405.

[72] The controller 402 may manage coding operations of the source coder 403, which may be for example a video coder, including, for example, setting of parameters and subgroup parameters used for encoding the video data.

[73] Output of all aforementioned functional units may be subjected to entropy coding in the entropy coder 408. The entropy coder translates the symbols as generated by the various functional units into a coded video sequence, by loss-less compressing the symbols according to technologies known to a person skilled in the art as, for example Huffman coding, variable length coding, arithmetic coding, and so forth.

[74] The transmitter 409 may buffer the coded video sequence(s) as created by the entropy coder 408 to prepare it for transmission via a communication channel 411, which may be a hardware/software link to a storage device which would store the encoded video data. The transmitter 409 may merge coded video data from the source coder 403 with other data to be transmitted, for example, coded audio data and/or ancillary data streams (sources not shown).

[75] The controller 402 may manage operation of the encoder 400. During coding, the controller 402 may assign to each coded picture a certain coded picture type, which may affect the coding techniques that may be applied to the respective picture. For example, pictures often may be assigned as one of the following frame types:

[76] An Intra Picture (I picture) may be one that may be coded and decoded without using any other frame in the sequence as a source of prediction. Some video codecs allow for different types of Intra pictures, including, for example Independent Decoder Refresh Pictures. A person skilled in the art is aware of those variants of I pictures and their respective applications and features.

[77] A Predictive picture (P picture) may be one that may be coded and decoded using intra prediction or inter prediction using at most one motion vector and reference index to predict the sample values of each block.

[78] A Bi-directionally Predictive Picture (B Picture) may be one that may be coded and decoded using intra prediction or inter prediction using at most two motion vectors and reference indices to predict the sample values of each block. Similarly, multiple-predictive pictures can use more than two reference pictures and associated metadata for the reconstruction of a single block.

[79] Source pictures commonly may be subdivided spatially into a plurality of sample blocks (for example, blocks of 4 x 4, 8 x 8, 4 x 8, or 16 x 16 samples each) and coded on a block-by-block basis. Blocks may be coded predictively with reference to other (already coded) blocks as determined by the coding assignment applied to the blocks' respective pictures. For example, blocks of I pictures may be coded non-predictively or they may be coded predictively with reference to already coded blocks of the same picture (spatial prediction or intra prediction). Pixel blocks of P pictures may be coded non-predictively, via spatial prediction or via temporal prediction with reference to one previously coded reference pictures. Blocks of B pictures may be coded non-predictively, via spatial prediction or via temporal prediction with reference to one or two previously coded reference pictures.

[80] The encoder 400, which may be for example a video coder, may perform coding operations according to a predetermined video coding technology or standard, such as ITU-T Rec. H.265. In its operation, the encoder 400 may perform various compression operations, including predictive coding operations that exploit temporal and spatial redundancies in the input

video sequence. The coded video data, therefore, may conform to a syntax specified by the video coding technology or standard being used.

[81] In an embodiment, the transmitter 409 may transmit additional data with the encoded video. The source coder 403 may include such data as part of the coded video sequence. Additional data may comprise temporal/spatial/SNR enhancement layers, other forms of redundant data such as redundant pictures and slices, Supplementary Enhancement Information (SEI) messages, Visual Usability Information (VUI) parameter set fragments, and so on.

[82] Fig. 5 illustrates a simplified block-style workflow diagram 500 of exemplary view-port dependent processing an in Omnidirectional Media Application Format (OMAF) that may allow for 360-degree virtual reality (VR360) streaming described in OMAF.

[83] At acquisition block 1001, video data A is acquired, such as data of multiple images and audio of same time instances in a case that the image data may represent scenes in VR360. At processing block 1003, the images B_i of the same time instance are processed by one or more of being stitched, mapped onto a projected picture with respect to one or more virtual reality (VR) angles or other angles/viewpoint(s) and region-wise packed. Additionally, metadata may be created indicating any of such processed information and other information so as to assist in delivering and rendering processes.

[84] With respect to data D, at image encoding block 1005, the projected pictures are encoded to data E_i and composed into a media file, and in viewport-independent streaming, and at video encoding block 1004, the video pictures are encoded as data E_v as a single-layer bitstream, for example, and with respect to data B_a the audio data may also be encoded into data E_a at audio encoding block 1002.

[85] The data E_a, E_v, and E_i, the entire coded bitstream F_i and/or F may be stored at a (content delivery network (CDN)/cloud) server, and typically may be fully transmitted, such as at delivery block 1007 or otherwise, to an OMAF player 1020 and may be fully decoded by a decoder such that at least an area of a decoded picture corresponding to a current viewport is rendered to the user at display block 1016 with respect to the various metadata, file playback, and orientation/viewport metadata, such as an angle at which a user may be looking through a VR image device with respect to viewport specifications of that device, from the head/eye tracking block 1008. A distinct feature of VR360 is that only a viewport may be displayed at any particular time, and such feature may be utilized to improve the performance of omnidirectional video systems, through selective delivery depending on the user's viewport (or any other criteria, such as recommended viewport timed metadata). For example, viewport-dependent delivery may be enabled by tile-based video coding according to exemplary embodiments.

[86] As with the encoding blocks described above, the OMAF player 1020 according to exemplary embodiments may similarly reverse one or more facets of such encoding with respect to the file/segment decapsulation of one or more of the data F' and/or F'_i and metadata, decode the audio data E'_i at audio decoding block 1010, the video data E'_v at video decoding block 1013, and the image data E'_i at image decoding block 1014 to proceed with audio rendering of the data B'_a at audio rendering block 1011 and image rendering of the data D' at image rendering block 1015 so as to output, in a VR360 format according to various metadata such as the orientation/viewport metadata, display data A'_i at display block 1016 and audio data A'_s at the loudspeakers/headphones block 1012. The various metadata may influence ones of the data decoding and rendering processes depending on various tracks, languages, qualities, views, that

may be selected by or for a user of the OMAF player 1020, and it is to be understood that the order of processing described herein is presented for exemplary embodiments and may be implemented in other orders according to other exemplary embodiments.

[87] Fig. 6 illustrates a simplified block-style content flow process diagram 600 for (coded) point cloud data with view-position and angle dependent processing of point cloud data (herein "V-PCC") with respect to capturing/generating/(de)coding/rendering/displaying 6 degree-of-freedom media. It is to be understood that the described features may be used separately or combined in any order and elements such as for encoding and decoding, among others illustrated, may be implemented by processing circuitry (*e.g.*, one or more processors or one or more integrated circuits), and the one or more processors may execute a program that is stored in a non-transitory computer-readable medium according to exemplary embodiments.

[88] The diagram 600 illustrates exemplary embodiments for streaming of coded point cloud data according to V-PCC.

[89] At the volumetric data acquisition block 1101, a real-world visual scene or a computer-generated visual scene (or combination of them) may be captured by a set of camera devices or synthesized by a computer as a volumetric data, and the volumetric data, which may have an arbitrary format, may be converted to a (quantized) point cloud data format, through image processing at the converting to point cloud block 1102. For example, data from the volumetric data may be area data by area data converted into ones of points of the point cloud by pulling one or more of the values described below from the volumetric data and any associated data into a desired point cloud format according to exemplary embodiments. According to exemplary embodiments, the volumetric data may be a 3D data set of 2D images, such as slices from which a 2D projection of the 3D data set may be projected for example. According to exemplary

embodiments, point cloud data formats include representations of data points in one or more various spaces and may be used to represent the volumetric data and may offer improvements with respect to sampling and data compression, such as with respect to temporal redundancies, and, for example, a point cloud data in an x, y, z, format representing, at each point of multiple points of the cloud data, color values (e.g., RGB, etc.), luminance, intensity, etc. and could be used with progressive decoding, polygon meshing, direct rendering, octree 3D representations of 2D quadtree data.

[90] At projection to images block 1103, the acquired point cloud data may be projected onto 2D images and encoded as image/video pictures with video-based point cloud coding (V-PCC). The projected point cloud data may be composed of attributes, geometry, occupancy map, and other metadata used for point cloud data reconstruction such as with painter's algorithms, ray casting algorithms, (3D) binary space partition algorithms, among others for example.

[91] At the scene generator block 1109, on the other hand, a scene generator may generate some metadata to be used for rendering and displaying 6 degrees-of-freedom (DoF) media, by a director's intention or a user's preference for example. Such 6 DoF media may include the 360VR like 3D viewing of a scene from rotational changes on 3D axis X, Y, Z in addition to additional dimension allowing for movement front/back, up/down, and left/right with respect to a virtual experience within or at least according to point cloud coded data. The scene description metadata defines one or more scene composed of the coded point cloud data and other media data, including VR360, light field, audio, etc. and may be provided to one or more cloud servers and or file/segment encapsulation/decapsulation processing as indicated in Fig. 6 and related descriptions.

[92] After video encoding block 1104 and image encoding block 1105 similar to the video and image encoding described above (and as will be understood, audio encoding also may be provided

as described above), file/segment encapsulation block 1106 processes such that the coded point cloud data are composed into a media file for file playback or a sequence of an initialization segment and media segments for streaming according to a particular media container file format such as one or more video container formats and such as may be used with respect to DASH described below, among others as such descriptions represent exemplary embodiments. The file container also may include the scene description metadata, such as from the scene generator block 1109, into the file or the segments.

[93] According to exemplary embodiments, the file is encapsulated depending on the scene description metadata to include at least one view position and at least one or more angle views at that/those view position(s) each at one or more times among the 6DoF media such that such file may be transmitted on request depending on user or creator input. Further, according to exemplary embodiments, a segment of such file may include one or more portions of such file such as a portion of that 6DoF media indicating a single viewpoint and angle thereat at one or more times; however, these are merely exemplary embodiments and may be changed depending on various conditions such as network, user, creator capabilities and inputs.

[94] According to exemplary embodiments, the point cloud data is partitioned into multiple 2D/3D regions, which are independently coded such as at one or more of video encoding block 1104 and image encoding block 1105. Then, each independently coded partition of point cloud data may encapsulated at file/segment encapsulation block 1106 as a track in a file and/or segement. According to exemplary embodiments, each point cloud track and/or a metadata track may include some useful metadata for view-position/angle dependent processing.

[95] According to exemplary embodiments, the metadata, such as included in a file and/or segment encapsulated with respect to the file/segment encapsulation block, useful for the view-

position/angle dependent processing includes one or more of the following: layout information of 2D/3D partitions with indices, (dynamic) mapping information associating a 3D volume partition with one or more 2D partitions (e.g. any of a tile/tile group/slice/sub-picture), 3D positions of each 3D partition on a 6DoF coordinate system, representative view position/angle lists, selected view position/angle lists corresponding to a 3D volume partition, indices of 2D/3D partitions corresponding to a selected view position/angle, quality (rank)information of each 2D/3D partition, and rendering information of each 2D/3D partition for example depending on each view position/angle. Calling on such metadata when requested, such as by a user of the V-PCC player or as directed by a content creator for the user of the V-PCC player, may allow for more efficient processing with respect to specific portions of the 6DoF media desired with respect to such metadata such that the V-PCC player may deliver higher quality images of focused on portions of the 6DoF media than other portions rather than delivering unused portions of that media.

[96] From the file/segment encapsulation block 1106, the file or one or more segments of the file may be delivered using a delivery mechanism (*e.g.*, by Dynamic Adaptive Streaming over HTTP (DASH)) directly to any of the V-PCC player 1125 and a cloud server, such as at the cloud server block 1107 at which the cloud server can extract one or more tracks and/or one or more specific 2D/3D partitions from a file and may merge multiple coded point cloud data into one data.

[97] According to data such as with the position/viewing angle tracking block 1108, if the current viewing position and angle(s) is/are defined on a 6DoF coordinate system, at a client system, then the view-position/angle metadata may be delivered, from the file/segment encapsulation block 1106 or otherwise processed from the file or segments already at the cloud server, at cloud server block 1107 such that the cloud sever may extract appropriate partition(s) from the store file(s) and merge them (if necessary) depending on the metadata from the client

system having the V-PCC player 1125 for example, and the extracted data can be delivered to the client, as a file or segments.

[98] With respect to such data, at the file/segment decapsulation block 1115, a file decapsulator processes the file or the received segments and extracts the coded bitstreams and parses the metadata, and at video decoding and image decoding blocks, the coded point cloud data are then decoded into decoded and reconstructed, at point cloud reconstruction block 1112, to point cloud data, and the reconstructed point cloud data can be displayed at display block 1114 and/or may first be composed depending on one or more various scene descriptions at scene composition block 1113 with respect to scene description data according to the scene generator block 1109.

[99] In view of the above, such exemplary V-PCC flow represents advantages with respect to a V-PCC standard including one or more of the described partitioning capabilities for multiple 2D/3D areas, a capability of a compressed domain assembly of coded 2D/3D partitions into a single conformant coded video bitstream, and a bitstream extraction capability of coded 2D/3D of a coded picture into conformant coded bitstreams, where such V-PCC system support is further improved by including container formation for a VVC bitstream to support a mechanism to contain metadata carrying one or more of the above-described metadata.

[100] In that light and according to exemplary embodiments further described below, the term "mesh" indicates a composition of one or more polygons that describe the surface of a volumetric object. Each polygon is defined by its vertices in 3D space and the information of how the vertices are connected, referred to as connectivity information. Optionally, vertex attributes, such as colors, normals, etc., could be associated with the mesh vertices. Attributes could also be associated with the surface of the mesh by exploiting mapping information that parameterizes the mesh with 2D attribute maps. Such mapping may be described by a set of parametric coordinates, referred to as

UV coordinates or texture coordinates, associated with the mesh vertices. 2D attribute maps are used to store high resolution attribute information such as texture, normals, displacements etc. Such information could be used for various purposes such as texture mapping and shading according to exemplary embodiments.

[101] Nonetheless, a dynamic mesh sequence may require a large amount of data since it may consist of a significant amount of information changing over time. Therefore, efficient compression technologies are required to store and transmit such contents. Mesh compression standards IC, MESHGRID, FAMC were previously developed by MPEG to address dynamic meshes with constant connectivity and time varying geometry and vertex attributes. However, these standards do not take into account time varying attribute maps and connectivity information. DCC (Digital Content Creation) tools usually generate such dynamic meshes. In counterpart, it is challenging for volumetric acquisition techniques to generate a constant connectivity dynamic mesh, especially under real time constraints. This type of contents is not supported by the existing standards. According to exemplary embodiments herein, there is described aspects of a new mesh compression standards to directly handle dynamic meshes with time varying connectivity information and optionally time varying attribute maps, this standard targets lossy, and lossless compression for various applications, such as real-time communications, storage, free viewpoint video, AR and VR. Functionalities such as random access and scalable/progressive coding are also considered. A “mesh”, or an “input mesh” or the like, is data of multiple, at least two, vertices.

[102]

[103] Fig. 7 represents an example framework 700 of one dynamic mesh compression such as for a 2D atlas sampling based method. Each frame of the input meshes 1201 can be preprocessed by a series of operations, e.g., tracking, remeshing, parameterization, voxelization. Note that, these

operations can be encoder-only, meaning they might not be part of the decoding process and such possibility may be signaled in metadata by a flag such as indicating 0 for encoder only and 1 for other. After that, one can get the meshes with 2D UV atlases 1202, where each vertex of the mesh has one or more associated UV coordinates on the 2D atlas. Then, the meshes can be converted to multiple maps, including the geometry maps and attribute maps, by sampling on the 2D atlas. Then these 2D maps can be coded by video/image codecs, such as HEVC, VVC, AV1, AVS3, etc. On the decoder 1203 side, the meshes can be reconstructed from the decoded 2D maps. Any post-processing and filtering can also be applied on the reconstructed meshes 1204. Note that other metadata might be signaled to the decoder side for the purpose of 3D mesh reconstruction. Note that the chart boundary information, including the uv and xyz coordinates, of the boundary vertices can be predicted, quantized and entropy coded in the bitstream. The quantization step size can be configured in the encoder side to tradeoff between the quality and the bitrates.

[104] In some implementations, a 3D mesh can be partitioned into several segments (or patches/charts). Each segment is composed of a set of connected vertices associated with their geometry, attribute, and connectivity information. As illustrated in the example 800 of volumetric data in Fig. 8, the UV parameterization process 1302 of mapping from 3D mesh segments onto 2D charts, such as to the above noted 2D UV atlases 1202 block, maps one or more mesh segments 1301 onto a 2D chart 1303 in the 2D UV atlas 1304. Each vertex (v_n) in the mesh segment will be assigned with a 2D UV coordinates in the 2D UV atlas. Note that the vertices (v_n) in a 2D chart form a connected component as their 3D counterpart. The geometry, attribute, and connectivity information of each vertex can be inherited from their 3D counterpart as well. For example, information may be indicated that vertex v_4 connects directly to vertices v_0, v_5, v_1, and v_3, and similarly information of each of the other vertices may also be likewise indicated. Further, such

2D texture mesh would, according to exemplary embodiments, further indicate information, such as color information, in a patch-by-patch basis such as by patches of each triangle, *e.g.*, v_2, v_5, v_3.

[105] For example, further to the features of the example 800 of Fig. 8, see the example 900 of Fig. 9 where the 3D mesh segment 1301 can be also mapped to multiple separate 2D charts 1401 and 1402. In this case, a vertex in 3D could corresponds to multiple vertices in 2D UV atlas. As shown in Fig. 9, the same 3D mesh segment is mapped to multiple 2D charts, instead of a single chart as in Fig. 8, in the 2D UV atlas. For example, 3D vertices v_1 and v_4 each have two 2D correspondences v_1,v_1', and v_4, v_4', respectively. As such, a general 2D UV atlas of a 3D mesh may consist of multiple charts as shown in Figure 14, where each chart may contain multiple (usually more than or equal to 3) vertices associated with their 3D geometry, attribute, and connectivity information.

[106] Fig. 10 shows an example 1000 illustrating a derived triangulation in a chart with boundary vertices B_0, B_1, B_2, B_3, B_4, B_5, B_6, B_7. When presented with such information, any triangulation method can be applied to create connectivity among the vertices (including boundary vertices and sampled vertices). For example, for each vertex, find the closest two vertices. Or for all vertices, continuously generate triangles until a minimum number of triangles is achieved after a set number of tries. As shown in the example 1000, there are various regularly shaped, repeating triangles and various oddly shaped triangles, generally closest to the boundary vertices, having their own unique dimensions that may or may not be shared with any other of the triangles. The connectivity information can be also reconstructed by explicit signaling. If a polygon cannot be recovered by implicit rules, the encoder can signal the connectivity information in the bitstream according to exemplary embodiments.

[107] Viewing the above-described patches, the example 1000 may represent one such patch, such a patch formed of vertices v_3, v_2, v_5 shown in any of Figs. 14 and 15.

[108] Boundary vertices B_0, B_1, B_2, B_3, B_4, B_5, B_6, B_7 are defined in the 2D UV space. As shown in Figure 15, the filled vertices are boundary vertices because they are on the boundary edges of a connected component (a patch/chart). A boundary edge can be determined by checking if the edge is only appeared in one triangle. The following information of boundary vertices is significant and should be signaled in the bitstream according to exemplary embodiments: geometry information, *e.g.*, the 3D XYZ coordinates even though currently in the 2D UV parametric form, and the 2D UV coordinates.

[109] For a case in which a boundary vertex in 3D corresponds to multiple vertices in 2D UV atlas, such as shown in Fig. 9, the mapping from 3D XUZ to 2D UV can be one-to-multiple. Therefore, a UV-to-XYZ (or referred to as UV2XYZ) index can be signaled to indicate the mapping function. UV2XYZ may be a 1D-array of indices that correspond each 2D UV vertex to a 3D XYZ vertex.

[110] According to exemplary embodiments, to represent a mesh signal efficiently, a subset of the mesh vertices may be coded first, together with the connectivity information among them. In the original mesh, the connection among these vertices may not exist as they are subsampled from the original mesh. There are different ways to signal the connectivity information among the vertices, and such subset is therefore referred to as the base mesh or as base vertices.

[111] Other vertices, however, can be predicted by applying interpolation between two or more already decoded mesh vertices. A predictor vertex will have its geometry location along the edge of two connected existing vertices, so the geometry information of the predictor can be calculated based on the neighboring decoded vertices. In some cases, the displacement vector or prediction

error, from the to-be-coded vertex to the vertex predictor, is to be further coded. For example, see the example 1100 of Fig. 11, an example of such edge-based vertex prediction is shown or more specifically of vertex geometry prediction using intra prediction by extrapolation by extending a triangle to a parallelogram, as shown on the left, and interpolation by weighted averaging of two existing vertices as shown in the right. After decoding the base vertices (i.e. the solid triangle 1801 in Fig. 11 left), interpolation among these base vertices can be done along the connected edges. For example, the middle point of each edge can be generated as predictors. The geometry locations of these interpolated points are therefore (weighted) average of the two neighboring decoded vertices (the dashed points 1802 in Fig. 11 left). Having more than 1 middle point between two already decoded vertices can also be done in a similar way. The actual vertices to be coded can therefore be reconstructed by adding the displacement vectors to the predictors (Fig. 7 middle). After decoding these additional vertices, the connection among the newly decoded vertices and the existing base vertices are still maintained. In addition, connection among the newly decoded vertices can be further established. Together with the base vertices, more intermediate vertices predictors can be generated along the new edges (Fig. 7 right) by connecting these newly decoded vertices 1803 and base vertices together. Therefore, more actual vertices to be decoded are present with associated displacement vectors.

[112] According to exemplary embodiments, mesh vertices of a mesh frame 1902 can also be predicted from decoded vertices of a previously coded mesh frame 1901. This prediction mechanism is referred to as inter prediction. Examples of mesh geometry inter prediction are illustrated in the example 1200 of Fig. 12 showing vertex geometry prediction using inter prediction (previous mesh frame's vertices become predictors of current frame's vertices). In

some cases, the displacement vector or prediction error, from the to-be-coded vertex to the vertex predictor, is to be further coded.

[113] According to exemplary embodiments, a number of methods are implemented for dynamic mesh compression and are part of the above-mentioned edge-based vertex prediction framework, where a base mesh is coded first and then more additional vertices are predicted based on the connectivity information from the edges of the base mesh. Note that they can be applied individually or by any form of combinations.

[114] For example, consider the vertex grouping for prediction mode example flowchart 1300 of Fig. 13. At S201, vertices inside a mesh may be obtained and can be divided at S202 into different groups for prediction purposes, for example see Fig. 10. In one example, the division is done using the patch/chart partitioning at S204 as previously discussed. In another example, the division is done under each patch/chart S205. The decision S203 whether to proceed to S204 or S205 may be signaled by a flag or the like. In the case of S205, several vertices of the same patch/chart form a prediction group and will share the same prediction mode, while several other vertices of the same patch/chart can use another prediction mode. Such grouping at S206 can be assigned at different levels by determining respective number of vertices involved per group. For example, every 64, 32 or 16 vertices following a scan order inside a patch/chart will be assigned the same prediction mode according to exemplary embodiments and other vertices may be differently assigned. For each group, a prediction mode can be intra prediction mode or inter prediction mode. This can be signaled or assigned. According to the example flowchart 1300, if a mesh frame or mesh slice is determined to be in intra type at S207, such as by checking whether a flag of that mesh frame or mesh slice indicates an intra type, then all groups of vertices inside that mesh frame

or mesh slice shall use intra prediction mode; otherwise, at S208 either intra prediction or inter prediction mode may be chosen per group for all vertices therein.

[115] Further, for a group of mesh vertices using intra prediction mode, its vertices can only be predicted by using previously coded vertices inside the same sub-partition of the current mesh. Sometimes the sub-partition can be the current mesh itself according to exemplary embodiments, and for a group of mesh vertices using inter prediction mode, its vertices can only be predicted by using previously coded vertices from another mesh frame according to exemplary embodiments. Each of the above-noted information may be determined and signaled by a flag or the like. Said prediction features may occur at S210 and results of said prediction and signaling may occur at S211

[116] According to exemplary embodiments, for each vertex in a group of vertices in the example flowchart 1300 and in the flowchart 1400 described below, after prediction, the residue will be a 3D displacement vector, indicating the shift from the current vertex to its predictor. The residues of a group of vertices need to be further compressed. In one example, transformation at S211, along with the signaling thereof, can be applied to the residues of a vertex group, before entropy coding. The following methods may be implemented to handle the coding of a group of displacement vectors. For example, in one method, to properly signal the case where a group of displacement vectors, some displacement vectors, or its components have only zero values. In another embodiment, a flag is signaled for each displacement vectors whether this vector has any non-zero component, and if no, the coding of all components for this displacement vector can be skipped. Further, in another embodiment, a flag is signaled for each group of displacement vectors whether this group has any non-zero vectors, and if no, the coding of all displacement vectors of this group can be skipped. Further, in another embodiment, a flag is signaled for each component

of a group of displacement vectors whether this component of the group has any non-zero vectors, and if no, the coding of this component of all displacement vectors s of this group can be skipped. Further, in another embodiment, there may be a signaling of the case where a group of displacement vectors, or a component of the group of displacement vectors, needs a transformation, and if not, the transformation can be skipped, and quantization/entropy coding can be directly applied to the group or the group components. Further, in another embodiment, a flag may be signaled for each group of displacement vectors whether this group needs to go through transformation, and if no, the transform coding of all displacement vectors of this group can be skipped. Further, in another embodiment, a flag is signaled for each component of a group of displacement vectors whether this component of the group needs to go through transformation, and if no, the transform coding of this component of all displacement vectors of this group can be skipped. The above-described embodiments in this paragraph, which regard handling of vertex prediction residues, may also be combined and implemented in parallel on different patches respectively.

[117] Fig. 14 shows the example flowchart 1400 where, at S221 a mesh frame can be obtained coded as an entire data unit, meaning all vertices or attributes of the mesh frame may have correlation among them. Alternatively, depending on a determination at S222, a mesh frame can be divided at S223 into smaller independent sub-partitions, similar in concept to slices or tiles in 2D videos or images. A coded mesh frame or a coded mesh sub-partition can be assigned with a prediction type at S224. Possible prediction types include intra coded type and inter coded type. For intra coded type, only predictions from the reconstructed parts of the same frame or slice are allowed at S225. On the other hand, an inter prediction type will allow at S225 predictions from a previously coded mesh frame, in addition to intra mesh frame predictions. Further, inter prediction

type may be classified with more sub-types such as P type or B type. In P type, only one predictor can be used for prediction purposes, while in B type, two predictors, from two previously coded mesh frames, may be used to generate the predictor. Weighted average of the two predictors can be one example. When the mesh frame is coded as a whole, the frame can be regarded as an intra or inter coded mesh frame. In case of inter mesh frame, P or B type may be further identified via signaling. Or, if a mesh frame is coded with further splitting inside a frame, assign prediction type for each of the sub-partitions occurs at S224. Each of the above-noted information may be determined and signaled by a flag or the like, and like with S210 and S211 of Fig. 13, said prediction features may occur at S226 and results of said prediction and signaling may occur at S227.

[118] As such, although dynamic mesh sequence may require a large amount of data since it may consist of a significant amount of information changing over time, efficient compression technologies are required to store and transmit such contents, and the above described features for Figs. 20 and 21 represent such improved efficiencies by allowing at least for improved mesh vertex 3D location prediction by either using previously decoded vertices in the same mesh frame (intra prediction) or from a previous coded mesh frame (inter prediction).

[119] Further, exemplary embodiments may generate the displacement vectors of a third layer 2303 of a mesh, based on one or more the reconstructed vertices of its previous layer(s) such as a second layer 2302 and a first layer 2301. Assuming the index of the second layer 2302 is T, the predictors for vertices in third layer 2303 T+1 are generated based on the reconstructed vertices of at least the current layer or second layer 2302. An example of such layer based prediction structure is shown example 1600 in Fig. 16 which illustrates reconstruction based vertex prediction: progressive vertex prediction using edge-based interpolation, where predictors are generated based

on previously decoded vertices, not predictor vertices. The first layer 2301 may be a mesh bounded by a first polygon 2340 having, as vertices thereof, decoded vertices, at boundaries thereof, and interpolated vertices, along ones of lines between ones of those decoded vertices. As the progressive coding proceeds from the first layer 2301 to the second layer 2302, an additional polygon 2341 may be formed by displacement vectors from ones of the interpolated vertices of the first layer to additional vertices of the second layer 2302, and as such, a total number of vertices of the second layer 2302 may be greater than that of the first layer 2301. Likewise, proceeding to the third layer 2303, the additional vertices of the second layer 2302, along with the decoded vertices from the first layer 2301, may then serve in the coding in a similar manner as did the decoded vertices served in proceeding from the first layer 2301 to the second layer 2303; that is, multiple additional polygons may be formed. As note, see the example 1900 in Fig. 19 illustrating such progressive coding where, unlike in Fig. 16, the example 1900 illustrates that, in proceeding from the first layer 2601 to the second layer 2603 and then to the third layer 2603, each of the additionally formed polygons may be entirely within a polygon formed by bounds of the first layer 2601.

[120] For such example 1600, see the example flowchart 1500 of Fig. 15 where since the interpolated vertices on the current layer are predicted values, such values need to be reconstructed, before being used to generate predictors of vertices on the next layer. This is done by coding a base mesh at S231, implementing vertices prediction as such at S232, then at S233 adding the decoded displacement vectors of the current layer to the vertex's predictors, such as of layer 2302. Then the reconstructed vertices of this layer 2303, together with all decoded vertices of previous layer(s), such as checking for addition vertices values of such layers at S234, can be used to generate and signal the predictor vertices of next layer 2303 at S235. This process can also be

summarized as follows: Let P[t](Vi) represent the predictor of vertex Vi on a layer t; let R[t](Vi) represent the reconstructed vertex Vi on layer t; let D[t](Vi) represent the displacement vector of vertex Vi on layer t; let f(*) represent the predictor generator, which, in particular, can be the average of the two existing vertices. Then for each layer t, there is the following according to exemplary embodiments:

$$P[t](Vi) = f(R[s|s<t](Vj), R[m|m<t](Vk)), \text{ where}$$

Vj and Vk are reconstructed vertices of previous layers

$$R[t](Vi) = P[t](Vi) + D[t](Vi) \qquad \text{- Eq. (1)}$$

[121] Then, for all vertices in one mesh frame, divide them into layer 0 (the base mesh), layer 1, layer 2, …. Etc. Then the reconstruction of vertices on one layer relies on the reconstruction of those on previous layer(s). In the above, each of P, R and D represents a 3D vector under the context of 3D mesh representation. D is the decoded displacement vector, and quantization may or may not apply to this vector.

[122] According to exemplary embodiments, the vertex prediction using reconstructed vertices may only apply to certain layers. For example, layer 0 and layer 1. For other layers, the vertex prediction can still use neighboring predictor vertices without adding displacement vectors to them for reconstruction. So that these other layers can be processed at the same time without waiting one previous layer to reconstruct. According to exemplary embodiments, for each layer, whether to choose reconstruction based vertex prediction or predictor based vertex prediction, can be signaled, or the layer (and its subsequent layers) that does not use reconstruction based vertex prediction, can be signaled.

[123] For the displacement vectors whose vertex predictors are generated by reconstructed vertices, quantization can be applied to them, without further performing transformation, such as

wavelet transform, etc. For the displacement vectors whose vertex predictors are generated by other predictor vertices, transformation may be needed and quantization can be applied to the transform coefficients of those displacement vectors.

[124] As such, since a dynamic mesh sequence may require a large amount of data since it may consist of a significant amount of information changing over time. Therefore, efficient compression technologies are required to store and transmit such contents. In the framework of interpolation-based vertex prediction method described above, one important procedure is to compress the displacement vectors, and this takes up a major part in the coded bitstream, and the focus of this disclosure, and the features of Fig. 15 for example alleviate such problem by providing for such compression.

[125] Further, similar to the other examples described above, even with those embodiments, a dynamic mesh sequence may nonetheless require a large amount of data since it may consist of a significant amount of information changing over time, and as such, efficient compression technologies are required to store and transmit such contents. In the framework of 2D atlas sampling based methods indicated above, an important advantage may be achieved by inferring the connectivity information from the sampled vertices plus boundary vertices on decoder side. This is a major part in decoding process, and a focus of further examples described below.

[126] According to exemplary embodiments, the connectivity information of the base mesh can be inferred (derived) from the decoded boundary vertices and the sampled vertices for each chart on both encoder and decoder sides.

[127] As similarly described above, any triangulation method can be applied to create connectivity among vertices (including boundary vertices and sampled vertices). For charts without any sampling of internal vertices, such as with the internal vertices shown in example 1000

of Fig. 10 and example 1800 of Fig. 18 described further below, similar methods of creating connectivity still apply although, according to exemplary embodiments, it may be signaled to use different triangulation methods for boundary vertices and sampled vertices.

[128] For example, according to exemplary embodiments, for every four neighboring points in any sampled positions, it may be determined whether a number of occupied points is larger than or equal to 3 (examples of occupied or unoccupied points are highlighted in Fig. 18 which shows an occupancy map example 1800 which each circle representing an integer pixel), and the connectivity of triangles among the 4 points can be inferred by certain rules. For example, as illustrated in the example 1700 of Fig. 17, if 3 out of 4 points are determined to be occupied, the shown examples (2), (3), (4), and (4), then those points can be interconnected directly to form a triangle as those examples; on the other hand, if 4 points are all determined to be occupied, then those points are used to form two triangles as shown in example (1) of Fig. 17 Note that different rules can be applied to different number of neighboring points. This process may be implemented across many points such as further illustrated in Fig. 18. In this embodiment, the reconstructed mesh is a triangle mesh as in Fig. 17 and as in at least as a regular triangle mesh in the internal portions of Fig. 18, which may not be determined to not be signaled according to such regularity but instead may be coded and decoded by inference rather than by individual signaling, and as irregular triangles at the perimeter which are to be signaled individually.

[129] And in attempt to further reduce the complexity and data processing, a quad mesh of such regular internal triangles shown in Fig. 18 may be inferred as such quad mesh of the example (1) of Fig. 17 thereby reducing even the amount of complexity from inferring the internal regular triangles as instead inferring a reduced number of internal regular quad meshes.

[130] According to exemplary embodiments, a quad mesh may be reconstructed when the 4 neighboring points are determined to be all occupied such as in the example (1) of Fig. 17.

[131] Extrapolating from the above-descriptions, as shown in Fig. 18, it is illustrated that reconstructed mesh in the example 1800 can be a hybrid type, that is, some regions in the mesh frame generate triangle meshes while other regions generate quad meshes, and some of said triangle meshes may be regular as compared to other triangle meshes therein and some may be irregular, such as ones of the boundary though not necessary all of such meshes on the boundary.

[132] According to exemplary embodiments, such connectivity types can be signaled in high-level syntax, such as sequence header, slice header.

[133] As mentioned above, connectivity information can be also reconstructed by explicitly signaling, such as for the irregularly shaped triangle meshes. That is, if it is determined that a polygon cannot be recovered by implicit rules, the encoder can signal the connectivity information in the bitstream. And according to exemplary embodiments, the overhead of such explicit signaling may be reduced depending on the boundaries of polygons. For example, as shown with the example 1800 in Fig. 18, the connectivity information of triangles would be signaled to be reconstructed by both implicit rules, such as according to the regular examples 2400 in Fig. 17 which may be inferred, and explicit signaling for ones of the irregular shaped polygons shown at least on the mesh boundaries in Fig. 18.

[134] According to embodiments, only the connectivity information between boundary vertices and sampled positions is determined to be signaled, while the connectivity information among the sampled positions themselves is inferred.

[135] Also, in any of the embodiments, the connectivity information may be signaled by prediction, such that only the difference from the inferred connectivity (as prediction) from one mesh to another may be signaled in bitstream.

[136] As a note, the orientation of inferred triangles (such as to be inferred in a clockwise manner or in a counterclockwise manner per triangle) can be either signaled for all charts in high-level syntax, such as sequence header, slice header, etc., or fixed (assumed) by encoder and decoder according to exemplary embodiments. The orientation of inferred triangles can be also signaled differently for each chart.

[137] As a further note, any reconstructed mesh may have different connectivity from the original mesh. For example, the original mesh may be a triangle mesh, while the reconstructed mesh may be a polygonal mesh (*e.g.*, quad mesh).

[138] According to exemplary embodiments, the connectivity information of any base vertices may not be signaled and instead the edges among base vertices may be derived using the same algorithm at both encoder and decoder side. For example, see how the bottommost vertices in the example 1800 are all occupied, and therefore, the coding may take advantage of such information by therefore determining that such vertices are occupied as a base and thereby later inferring such that the connectivity information of any base vertices may not be signaled and instead the edges among base vertices may be derived using the same algorithm at both encoder and decoder side. And according to exemplary embodiments, interpolation of predicted vertices for the additional mesh vertices may be based on the derived edges of the base mesh.

[139] According to exemplary embodiments, a flag may be used to signal whether the connectivity information of the base vertices is to be signaled or derived, and such flag can be signaled at different level of the bitstream, such as at sequences level, frame level, etc.

[140] According to exemplary embodiments, the edges among the base vertices are first derived using the same algorithm at both encoder and decoder side. Then compared with the original connectivity of the base mesh vertices, the difference between the derived edges and the actual edges will be signaled. Therefore, after decoding the difference, the original connectivity of the base vertices can be restored.

[141] In one example, for a derived edge, if determined to be wrong when compared to the original edge, such information may signaled in the bitstream (by indicating the pair of vertices that form this edge); and for an original edge, if not derived, may be signaled in the bitstream (by indicating the pair of vertices that form this edge). Further, connectivity on boundary edges and vertex interpolation involving boundary edges may be done separately from the internal vertices and edges.

[142] Accordingly, by exemplary embodiments described herein, the technical problems noted above may be advantageously improved upon by one or more of these technical solutions. For example, since a dynamic mesh sequence may require a large amount of data since it may consist of a significant amount of information changing over time, and therefore, the exemplary embodiments described herein represent at least efficient compression technologies to store and transmit such contents.

[143] The above-described embodiments may be further applied to instance-based mesh coding, where an instance may be a mesh of an object or a part of an object. For example, the illustration example 2100 of Fig. 21 illustrates a mesh example 2801 in which various instances 2802 (representing a mesh of a cup), 2803 (representing a mesh of a spoon), and 2804 (representing a mesh of a plate) are present and may be separated and coded respectively. And each of the instances 2801, 2802, 2803, and 2804 are illustrated in respective ones of bounding

boxes which will be described further below, but, as a note, it may be considered that the instance 2801 may be illustrated as a bounded by a "mesh-based bounding box" whereas each of instances 2802, 2803, and 2804 may be considered illustrated as bounding by respective ones of an "instance-based bounding box."

[144] According to exemplary embodiments, the proposed methods may be used separately or combined in any order. The proposed methods may be used for arbitrary polygon mesh, but even though only a triangle mesh may have been used for demonstration of various embodiments. As noted above, it will be assumed that an input mesh may contain one or multiple instances, that a submesh is a part of input mesh with an instance or multiple instance, and that multiple instances can be grouped to form a submesh.

[145] In that light, Fig. 20 illustrates an example 2000 in which it is proposed to separately quantize different objects or parts at a given input bitdepth (where that bitdepth may be referred to as "QP"). For example, at 2701 an one or more input meshes may be obtained and each separated into multiple submeshes. A submesh can be an object, an instance of an object or a segmented region, and will be quantized at S2702 independently according to exemplary embodiments.

[146] According to exemplary embodiments, a mesh $\mathcal{M}$ with m points in (x, y, z) coordinate may be quantized at S2702 by a QP bitdepth. The quantization step size for all three dimensions (x, y, z) may be decided based on a largest length of the bounding box in all dimension - $d_{bbox} > 0$. And same quantization step size may applied at S2704 for all objects, identified at S2703, in the mesh as

$$\Delta_{qp} = \frac{d_{bbox}}{2^{QP}-1}, \qquad \text{- Eq. (1)}$$

and a scalar quantization thereof may applied for the j-th point at i-th coordinate a_{ij} as

$$\tilde{a}_{ij} = \left\lfloor \frac{a_{ij}-\theta_i}{\Delta_{qp}} + \theta_{QP} \right\rfloor, i \in \{x, y, z\}, j \in [1, .., m], \quad \text{-Eq. (2)}$$

where $\theta_{QP} = 0.5$ is an offset parameter for quantization. θ_i is the minimum coordinate of the mesh in $\mathcal{M}$ at i-th dimension. Notation $\lfloor \cdot \rfloor$ stands for the floor rounding operator. And the dequantized coordinate may be calculated with uniform dequantization as follow

$$\hat{a}_{i,j} = \tilde{a}_{i,j} * \Delta_{qp} + \theta_i, i \in \{x, y, z\}, j \in [1, .., m], \quad \text{-Eq. (3)}$$

with the mean square error of quantization as

$$\epsilon_{QP} = \frac{1}{n} \sum_i^{\{x,y,z\}} \sum_j^m \left(a_{ij} - \hat{a}_{ij}\right)^2 \quad \text{- Eq. (4)}$$

[147] However, in complex scenes, a largest object is the background which may relatively often be simple and can tolerate a higher quantization step size. Meanwhile, the main objects are at smaller scale and suffer huge quantization error which may be accounted for by various embodiments described further below.

[148] Therefore, as shown in the example 2200 in Fig. 22, as the maximum length of the bounding box of the input mesh d_{bbox} may always be set equal to or larger than the maximum length of the bounding box of each instance d_{bbox}^j as

$$d_{bbox} \geq \max_{j \in \mathbb{O}} \{d_{bbox}^j\}, \quad \text{- Eq. (5)}$$

where $\mathbb{O}$ is the set of all instances or segmentation in the input mesh.

[149] At a given bitdepth QP, the quantization step size of every instance, each of instances 2802 (representing a mesh of a cup), 2803 (representing a mesh of a spoon), and 2804 (representing a mesh of a plate), may always smaller than or equal to the mesh-based quantization step size that satisfies $\Delta_{qp}^j \leq \Delta qp, \forall j \in \mathbb{O}$.

[150] Therefore, the quantization error for each instance becomes smaller, thus reducing the overall quantization error.

[151] According to various embodiments, the bitdepth may be assigned adaptively for each instance/region, referred to as a "submesh" in S2902, and may be decided based on the face density of that particular instance. Each submesh may be obtained from the volumetric data of the mesh which may itself have signaled each instance within the mesh individually, and each submesh being derived from that mesh on per instance basis at S2902. For example, each of the instances 2802, 2803, and 2804 may be assigned its own respective bitdepth, at S2904, depending on its own particular face density or numbers of vertices, forming one or more of the above-described polygons, therein. In general, the more faces each instance has, which may be determined at S2903 by counting a number of such polygons therein or the like, the less quantization should be applied at S2702 to that instance. For example, given a mesh $\mathcal{M}$, a total number of faces is n, and corresponding faces for submesh k-th is n_k that satisfies

$$n = \sum_{k=1}^{K} n_k , n_k > 0 , \qquad \text{- Eq. (6)}$$

where K is the total number of submeshes. The submesh face density is defined as $\frac{n_k}{V_{bbox}^k}$ with V_{bbox}^k standing for the volume of the bounding box set at S2906 of the k-th submesh. Then in one example, the adaptive quantization for instance k, referred to as QP_k, can be defined in a limited range $[QP_{min}, QP_{max}]$ as

$$QP_k = \text{Clip}\left(QP * \frac{n_k * V_{bbox}}{n * V_{bbox}^k}, QP_{min}, QP_{max} \right). \qquad \text{- Eq. (7)}$$

[152] According to various embodiments, a mesh is represented as a base-mesh B and its corresponding displacement D and quantized at S2702 at different bitdepth. For example, for the

k-th object, the bithdepth base mesh QP_k^b can be calculated from Eq. (3), and the bitdepth of its displacement QP_k^d could be derived as

$$QP_k^d = \lfloor \alpha_k \times QP_k^b + \beta_k \rfloor, \qquad \text{- Eq. (8)}$$

with α_k, β_k is the adaptive scaling factor and offset for the jth object. In one example, $\alpha_k = 1$ and $\beta_k = 2$.

[153] According to various embodiments, adaptive bitdepth parameters based on minimizing distortion may be used. For example, given an input bithdepth QP, the mean squared error (MSE) of a quantization method is ϵ_QP may be as in Eq. (4). The MSE of each submesh is derived as ϵ_QP^k=ω_k*ϵ_QP,∀k∈[1,…,K], where ω_k>0 is a weighting factor. In one example, ω_k=1 ∀k. A linear search is performed for each submesh to find the best bithdepth for base mesh that satisfies

$$QP_k^b = \min_{q \in [QP_{min}, QP_{max}]} \left| \epsilon_q - \omega_k \epsilon_{QP} \right|_2^2, \qquad \text{- Eq. (9)}$$

[154] Additionally, a best bithdepth for displacement may also obtained via

$$QP_k^d = \min_{q \in [QP_{min}, QP_{max}], \alpha, \beta} \left| \epsilon_q - \omega_k \epsilon_{QP} \right|_2^2, \qquad \text{- Eq. (10)}$$

[155] According to exemplary embodiments, there may be signaling of quantization for each object such as by signaling at S2907 signal bithdepth through bitstream. The set of base quantization bitdepth in the increasing order may be $\{QP_k^b\}_{k=0,..,K}$ with corresponding displacement quantization bitdepth $\{QP_k^d\}_{k=0,..,K}$. This information may be signaled as mesh instance parameter syntax. For signaling, b_0 bits may be used to signal a bounding box offset θ_i. To avoid signaling overhead, all instances may share the same bounding box offset. Number $K - 1$ is limited to b_1 bit, the maximum base quantization bithdepth is b_2 bit, the maximum difference in bitdepth between base and displacement is b_3 bit. In one example, $b_1 = 4, b_2 =$

$5, b_3 = 4$. An example syntax table is shown below, where the instances are arranged in the order of ascending quantization values. In this way, the signaled quantization difference for each instance may be always non-negative. In a more general case, the instances may not be arranged by quantization values, for each instance, and in addition to the absolute difference, the sign may also be signaled.

mesh_instance_parameter_set() {	
for (i = 0; i < num_dim; i++) {	
mips_min_bbox [i] /* θ_i */	i(b_0)
}	
mips_num_instances_minus1 /* $K - 1$ */	u(b_1)
mips_base_bithdepth_minus1 /* $QP_0^b - 1$ */	u(b_2)
misp_dist_bitdepth[0] /*$QP_0^d - QP_0^b$*/	
for (k = 1; j <= mips_num_instance_minus1; j++) {	
mips_base_bitdepth [k-1] /* $QP_k^b - QP_{k-1}^b$ */	u(b_1)
mips_dist_bitdepth[k] /*$QP_k^d - QP_k^b$*/	i(b_3)
}	
}	

where

- u(n) is unsigned integer using n bits, i(n) is integer using n bits, and mips_quant() is a series of signaling data,
- mips_min_bbox[k] is the minimum of the bounding box at i-th dimension,
- mips_num_instances_minus1 is the number of instances - 1 in the mesh,
- mips_base_bitdepth_minus1 is the bitdepth of the first instance in the order,

- mips_base_quant[k] is the difference in quantization of the $(k+1)$-th and k-th submesh. As the quantization set is sorted in the increasing order, this number is always non-negative, and
- mips_dist_quant[k] is the k-th quantization data for base mesh bithdepth.

[156] According to various embodiments, multiple instances may be grouped to K groups with a same bitdepth to reduce the signaling overhead. Instances may be clustered based on the maximum distance of the bounding box d_{bbox}^{j} with a simple clustering method like K-mean clustering.

[157] However, since such 3D scenes may often consist of multiple instances with similar mesh structures reusing an asset, further improvements may be achieved by not only exploiting local characteristics but also considering the similarity between instances according to exemplary embodiments.

[158] For example, Fig. 23 illustrates an example 2300 flowchart in which an instance-based matching prediction (IMP) method is used to find a redundant mesh and encode corresponding displacement, which may advantageously normalize instances to maximize their similarity, and may be used with any of the embodiments described above.

[159] For example, at S2801, an input mesh may be obtained and partitioned into to multiple submeshes as described above. A submesh could be an instance of an individual object or a part of an object according to exemplary embodiments.

[160] At S2802, there may be a grouping of instances into similarity group using simple scaling features and a similarity measure. For example, according to embodiments, instances may be aligned and normalised so that only transitional asset may be reused. Scale and orientation

information may be signaled through channel for IMP mode, and given input mesh $\mathcal{M}$ with m instances, an instance i-th may have a corresponding bounding box of d_x^i, d_y^i, d_z^i. And as such, similar instances with a same ratio of the bounding box $d_x/d_y, d_x/d_z$ may be grouped to one asset group. Additionally, pairwise d1PSNR (peak signal to noise ratio) between two instances of the same group is greater than a threshold τ may be applied to verify the similarity and remove outlier instances. For example, threshold $\tau = 150dB$ may used, and in total, $\mathcal{M}$ has K asset group as $\mathcal{S} = \{S_0, \dots, S_{K-1}\}, |S_k| = m_k$, with m_k is the number of instances for the k-th asset group.

[161] According to exemplary embodiments, at S2803, it may be determined at S2803 to implement coding such that depending on an indication of a flag, IMP may be used to encode instances in an asset group $S_k = \{\mathcal{M}_0^k, \dots, \mathcal{M}_{m_K}^k\}$ with large than one in size $m_k > 1$. The first instance is encoded, and its decoded version is used as a base mesh for the remainder instance in the group. For example:

encode $\mathcal{M}_0^k$ via its base mesh B_0^k and displacement D_0^k
decode $\mathcal{M}_0^k$ as $\widehat{\mathcal{M}}_0^k$
for (i = 1; i < m_K; i++) {
use decoded $\widehat{\mathcal{M}}_0^k$ as base mesh B_i^k, derive displacement D_i^k
encode displacement D_i^k
}

Note that, in lossless compression, $\mathcal{M}_i^k$ may be identical to $\widehat{\mathcal{M}}_i^k$.

[162] Depending on the flag S2804, coding at S2805 may be signaled to code the submeshes independently into sub-bitstreams. Each submesh can be coded by mesh codecs with different coding parameters. Note that each submesh can be also coded by different mesh codecs, in which case the codec index indicating which mesh codec is used needs to be signaled such as in the

header of the sub-bitstream. Sub-bitstreams of submeshes can be encoded and decoded in parallel without data dependency issue according to exemplary embodiments.

[163] If the flag at S2804 instead indicates dependent coding, then at S2806, an additional flag may be considered so as to indicate a mode of coding the submeshes dependently as well. For example, according to embodiments, a submesh can be coded by prediction from other submeshes which are already coded. Prediction indices can be coded to indicate which sub-mesh to be used as prediction. The prediction indices can be signaled at different levels.

[164] For example, at S2807, only one prediction index may be coded for the entire submesh, so that all the vertices in current submesh will be predicted from the same submesh as indicated by the index according to exemplary embodiments.

[165] At S2808 the prediction indices may be signaled for each vertex of current submesh, so that each vertex can predict from different submeshes. Note that the prediction indices can be coded by predictive coding as well, where the prediction index of a vertex can be predicted from neighboring coded vertices. The prediction index residual can be then coded by arithmetic coding according to exemplary embodiments.

[166] At S2809, the prediction indices may be signaled at an intermediate level between vertex-level and submesh-level, e.g., at the group-of-vertices level, where a group of vertices share the same prediction index. Note that the prediction index of different groups can be also coded by predictive coding according to exemplary embodiments.

[167] Then, given the prediction indices of each vertex of current submesh, each vertex can be either predicted from the vertices in corresponding submesh, such as described above with any of Figs. 13, 14, and 15. According to exemplary embodiments, a rigid motion may be estimated

from the prediction submesh to current submesh, and the parameters of the rigid motion (e.g., rotation and translation parameters) can be coded. Then, after applying the rigid motion to the prediction submesh, the residual of the current vertex's attribute can be obtained by subtracting the corresponding vertex's attribute in the transformed prediction submesh. The attribute of the vertex can include but not limited to the following: geometry, color, normal, uv coordinates, connectivity etc. The residual information can then be coded by arithmetic coding according to exemplary embodiments.

[168] Further, such coding may be applied to code material and texture information for each submesh. This information can include but is not necessarily limited to the following: ambient color, diffuse color, specular color, focus of specular highlights, factor for dissolve, illumination model, texture image id etc.

[169] According to exemplary embodiments, one submesh only allows one set of material and texture information, and in such case, this information can be simply coded at the header of the sub-bitstream.

[170] Or, according to exemplary embodiments, one submesh can have more than one sets of material and texture information, and in this case, those sets can be coded at the header of the sub-bitstream. Note that these parameters in different sets can be coded independently or dependently. If the dependent coding is applied, predictions can be applied, and the prediction residual of material parameters can be coded instead. Then, for each vertex in the submesh, a material id can be coded to indicate which set of material information is used for this vertex. Note that the material id (identifier) can be coded by predicting from neighboring coded vertices to reduce the redundancies, according to exemplary embodiments.

[171] Therefore, since it is realized according to embodiments herein that a complex mesh may often contain information about multiple instances to relate associate texture maps, and that information may be available at the encoding time, each instance may be considered as a 3D asset since, in 3D design according to exemplary embodiments herein, the asset may be frequently used to reduces cost in designing models, especially for composite of complex scenes. For example, a 3D model may be reused with or without modified textures at difference in scale, orientation, etc. This addresses problems involved in matching and retrieving 3D objects since, among other aspects described above, the meshes may be normalized to a position, a size, and orientations based on a PCA (principal component analysis) or bilateral symmetry plane and thereby more efficiently retrieved.

[172] The techniques described above, can be implemented as computer software using computer-readable instructions and physically stored in one or more computer-readable media or by a specifically configured one or more hardware processors. For example, FIG. 24 shows a computer system 2400 suitable for implementing certain embodiments of the disclosed subject matter.

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

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

[175] The components shown in FIG. 24 for computer system 2400 are exemplary in nature and are not intended to suggest any limitation as to the scope of use or functionality of the computer software implementing embodiments of the present disclosure. Neither should the configuration of components be interpreted as having any dependency or requirement relating to any one or combination of components illustrated in the exemplary embodiment of a computer system 2400.

[176] Computer system 2400 may include certain human interface input devices. Such a human interface input device may be responsive to input by one or more human users through, for example, tactile input (such as: keystrokes, swipes, data glove movements), audio input (such as: voice, clapping), visual input (such as: gestures), olfactory input (not depicted). The human interface devices can also be used to capture certain media not necessarily directly related to conscious input by a human, such as audio (such as: speech, music, ambient sound), images (such as: scanned images, photographic images obtain from a still image camera), video (such as two-dimensional video, three-dimensional video including stereoscopic video).

[177] Input human interface devices may include one or more of (only one of each depicted): keyboard 2401, mouse 2402, trackpad 2403, touch screen 2410, joystick 2405, microphone 2406, scanner 2408, camera 2407.

[178] Computer system 2400 may also include certain human interface output devices. Such human interface output devices may be stimulating the senses of one or more human users through, for example, tactile output, sound, light, and smell/taste. Such human interface output devices may include tactile output devices (for example tactile feedback by the touch-screen 2410, or joystick 2405, but there can also be tactile feedback devices that do not serve as input devices), audio output devices (such as: speakers 2409, headphones (not depicted)), visual output

devices (such as screens 2410 to include CRT screens, LCD screens, plasma screens, OLED screens, each with or without touch-screen input capability, each with or without tactile feedback capability—some of which may be capable to output two dimensional visual output or more than three dimensional output through means such as stereographic output; virtual-reality glasses (not depicted), holographic displays and smoke tanks (not depicted)), and printers (not depicted).

[179] Computer system 2400 can also include human accessible storage devices and their associated media such as optical media including CD/DVD ROM/RW 2420 with CD/DVD 2411 or the like media, thumb-drive 2422, removable hard drive or solid state drive 2423, legacy magnetic media such as tape and floppy disc (not depicted), specialized ROM/ASIC/PLD based devices such as security dongles (not depicted), and the like.

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

[181] Computer system 2400 can also include interface 2499 to one or more communication networks 2498. Networks 2498 can for example be wireless, wireline, optical. Networks 2498 can further be local, wide-area, metropolitan, vehicular and industrial, real-time, delay-tolerant, and so on. Examples of networks 2498 include local area networks such as Ethernet, wireless LANs, cellular networks to include GSM, 3G, 4G, 5G, LTE and the like, TV wireline or wireless wide area digital networks to include cable TV, satellite TV, and terrestrial broadcast TV, vehicular and industrial to include CANBus, and so forth. Certain networks 2498 commonly require external network interface adapters that attached to certain general-purpose data ports or peripheral buses (2450 and 2451) (such as, for example USB ports of the computer system 2400; others are commonly integrated into the core of the computer system 2400 by attachment to a

system bus as described below (for example Ethernet interface into a PC computer system or cellular network interface into a smartphone computer system). Using any of these networks 2498, computer system 2400 can communicate with other entities. Such communication can be uni-directional, receive only (for example, broadcast TV), uni-directional send-only (for example CANbusto certain CANbus devices), or bi-directional, for example to other computer systems using local or wide area digital networks. Certain protocols and protocol stacks can be used on each of those networks and network interfaces as described above.

[182] Aforementioned human interface devices, human-accessible storage devices, and network interfaces can be attached to a core 2440 of the computer system 2400.

[183] The core 2440 can include one or more Central Processing Units (CPU) 2441, Graphics Processing Units (GPU) 2442, a graphics adapter 2417, specialized programmable processing units in the form of Field Programmable Gate Areas (FPGA) 2443, hardware accelerators for certain tasks 2444, and so forth. These devices, along with Read-only memory (ROM) 2445, Random-access memory 2446, internal mass storage such as internal non-user accessible hard drives, SSDs, and the like 2447, may be connected through a system bus 2448. In some computer systems, the system bus 2448 can be accessible in the form of one or more physical plugs to enable extensions by additional CPUs, GPU, and the like. The peripheral devices can be attached either directly to the core's system bus 2448, or through a peripheral bus 2449. Architectures for a peripheral bus include PCI, USB, and the like.

[184] CPUs 2441, GPUs 2442, FPGAs 2443, and accelerators 2444 can execute certain instructions that, in combination, can make up the aforementioned computer code. That computer code can be stored in ROM 2445 or RAM 2446. Transitional data can be also be stored in RAM 2446, whereas permanent data can be stored for example, in the internal mass

storage 2447. Fast storage and retrieval to any of the memory devices can be enabled through the use of cache memory, that can be closely associated with one or more CPU 2441, GPU 2442, mass storage 2447, ROM 2445, RAM 2446, and the like.

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

[186] As an example and not by way of limitation, the computer system having architecture 2400, and specifically the core 2440 can provide functionality as a result of processor(s) (including CPUs, GPUs, FPGA, accelerators, and the like) executing software embodied in one or more tangible, computer-readable media. Such computer-readable media can be media associated with user-accessible mass storage as introduced above, as well as certain storage of the core 2440 that are of non-transitory nature, such as core-internal mass storage 2447 or ROM 2445. The software implementing various embodiments of the present disclosure can be stored in such devices and executed by core 2440. A computer-readable medium can include one or more memory devices or chips, according to particular needs. The software can cause the core 2440 and specifically the processors therein (including CPU, GPU, FPGA, and the like) to execute particular processes or particular parts of particular processes described herein, including defining data structures stored in RAM 2446 and modifying such data structures according to the processes defined by the software. In addition or as an alternative, the computer system can provide functionality as a result of logic hardwired or otherwise embodied in a circuit (for example: accelerator 2444), which can operate in place of or together with software to execute particular processes or particular parts of particular processes described herein. Reference to

software can encompass logic, and vice versa, where appropriate. Reference to a computer-readable media can encompass a circuit (such as an integrated circuit (IC)) storing software for execution, a circuit embodying logic for execution, or both, where appropriate. The present disclosure encompasses any suitable combination of hardware and software.

[187] While this disclosure has described several exemplary embodiments, there are alterations, permutations, and various substitute equivalents, which fall within the scope of the disclosure. It will thus be appreciated that those skilled in the art will be able to devise numerous systems and methods which, although not explicitly shown or described herein, embody the principles of the disclosure and are thus within the spirit and scope thereof.

WHAT IS CLAIMED IS:

1. A method for video decoding, the method performed by at least one processor and comprising:

obtaining an input mesh corresponding to volumetric data of at least one three-dimensional (3D) visual content;

partitioning the input mesh corresponding to the volumetric data into a plurality of submeshes, each of the submeshes comprising a respective one of instances of objects;

determining a similarity between at least two of the instances of objects;

grouping the plurality of submeshes based on the determined similarity; and

decoding the volumetric data based on the grouping of the plurality of submeshes.

2. The method for video decoding according to claim 1,

wherein determining the similarity comprises comparing bounding boxes of the at least two of the instances of objects.

3. The method for video decoding according to claim 2,

wherein comparing the bounding boxes comprises comparing ratios of dimensions of the bounding boxes.

4. The method for video decoding according to claim 1,

wherein decoding the volumetric data comprises signaling different ones of codecs respective to ones of the at least two instances of objects.

5. The method for video decoding according to claim 1,

wherein decoding the volumetric data comprises signaling prediction of a first one of the at least two instances of objects based on a prediction of a second one of the at least two instances of objects.

6. The method for video decoding according to claim 5,

wherein decoding the volumetric data comprises signaling only a single one prediction index for an entirety of the first one of the at least two instances of objects.

7. The method for video decoding according to claim 5,

wherein decoding the volumetric data comprises signaling only a plurality of prediction indices for the first one of the at least two instances of objects.

8. The method for video decoding according to claim 7,

wherein a first one of the plurality of prediction indices is based on a first prediction index,

wherein a second one of the plurality of prediction indices is based on a second prediction index,

wherein the first prediction index is based on a first other prediction index which is of the second one of the at least two instances of objects, and

wherein the second one of the plurality of prediction indices is based on a second other prediction index which is of a third one of the at least two instances of objects.

9. The method for video decoding according to claim 5,

wherein signaling prediction of the first one of the at least two instances of objects comprises signaling a first prediction index and a second prediction index,

wherein the first prediction index is a single index that is collectively set to each of a plurality of vertices of the first one of the at least two instances of objects, and

wherein the second prediction index is set to at least one other vertex of the first one of the at least two instances of objects.

10. The method for video decoding according to claim 10,

wherein at least one of the first prediction index and the second prediction index is based on a prediction index set to at least one vertex of the second one of the at least two instances of objects.

11. An apparatus for video decoding, the apparatus comprising:

at least one memory configured to store computer program code;

at least one processor configured to access the computer program code and operate as instructed by the computer program code, the computer program code including:

obtaining code configured to cause the at least one processor to obtain an input mesh comprising volumetric data of at least one three-dimensional (3D) visual content;

partitioning code configured to cause the at least one processor to partition the input mesh corresponding to the volumetric data into a plurality of submeshes, each of the submeshes comprising a respective one of instances of objects;

determining code configured to cause the at least one processor to determine a similarity between at least two of the instances of objects;

grouping code configured to cause the at least one processor to group the plurality of submeshes based on the determined similarity; and

decoding code configured to cause the at least one processor to code the volumetric data based on the grouping of the plurality of submeshes.

12. The apparatus for video decoding according to claim 11,

wherein determining the similarity comprises comparing bounding boxes of the at least two instances of objects.

13. The apparatus for video decoding according to claim 12,

wherein comparing the bounding boxes comprises comparing ratios of dimensions of the bounding boxes.

14. The apparatus for video decoding according to claim 11,

wherein decoding the volumetric data comprises signaling different ones of codecs respective to ones of the at least two instances of objects.

15. The apparatus for video decoding according to claim 11,

wherein decoding the volumetric data comprises signaling prediction of a first one of the at least two instances of objects based on a prediction of a second one of the at least two instances of objects.

16. The apparatus for video decoding according to claim 15,

wherein decoding the volumetric data comprises signaling only a single one prediction index for an entirety of the first one of the at least two instances of objects.

17. The apparatus for video decoding according to claim 15,

wherein decoding the volumetric data comprises signaling only a plurality of prediction indices for the first one of the at least two instances of objects.

18. The apparatus for video decoding according to claim 17,

wherein a first one of the plurality of prediction indices is based on a first prediction index,

wherein a second one of the plurality of prediction indices is based on a second prediction index,

wherein the first prediction index is based on a first other prediction index which is of the second one of the at least two instances of objects, and

wherein the second one of the plurality of prediction indices is based on a second other prediction index which is of a third one of the at least two instances of objects.

19. The apparatus for video decoding according to claim 15,

wherein signaling prediction of the first one of the at least two of the submeshes comprises signaling a first prediction index and a second prediction index,

wherein the first prediction index is a single index that is collectively set to each of a plurality of vertices of the first one of the at least two instances of objects, and

wherein the second prediction index is set to at least one other vertex of the first one of the at least two instances of objects.

20. A non-transitory computer readable medium storing a program causing a computer to:

obtain an input mesh corresponding to volumetric data of at least one three-dimensional (3D) visual content;

partitioning the input mesh corresponding to the volumetric data into a plurality of submeshes, each of the submeshes comprising a respective one of instances of objects;

determine a similarity between at least two of the instances of objects;

grouping the plurality of submeshes based on the determined similarity; and

decode the volumetric data based on the grouping the plurality of submeshes.

FIG. 1

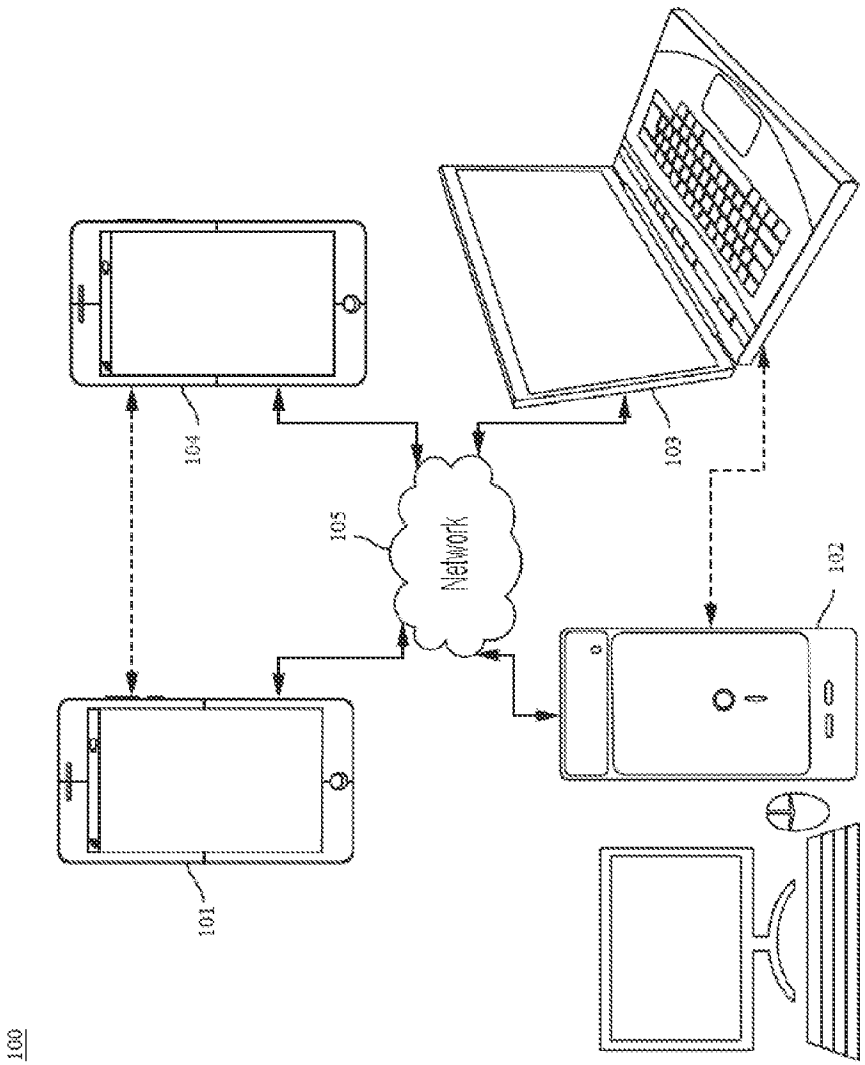


FIG. 2

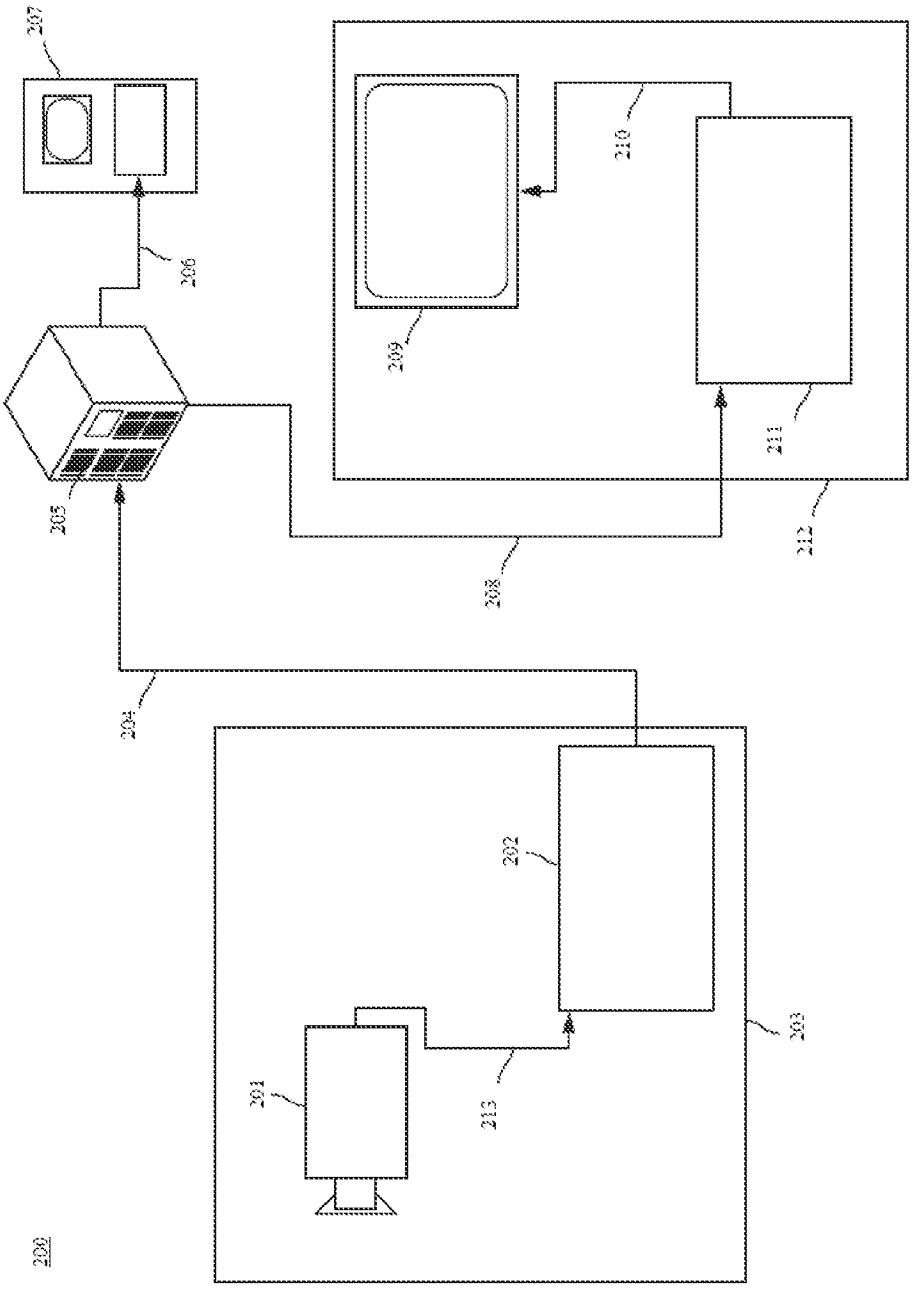


FIG. 3

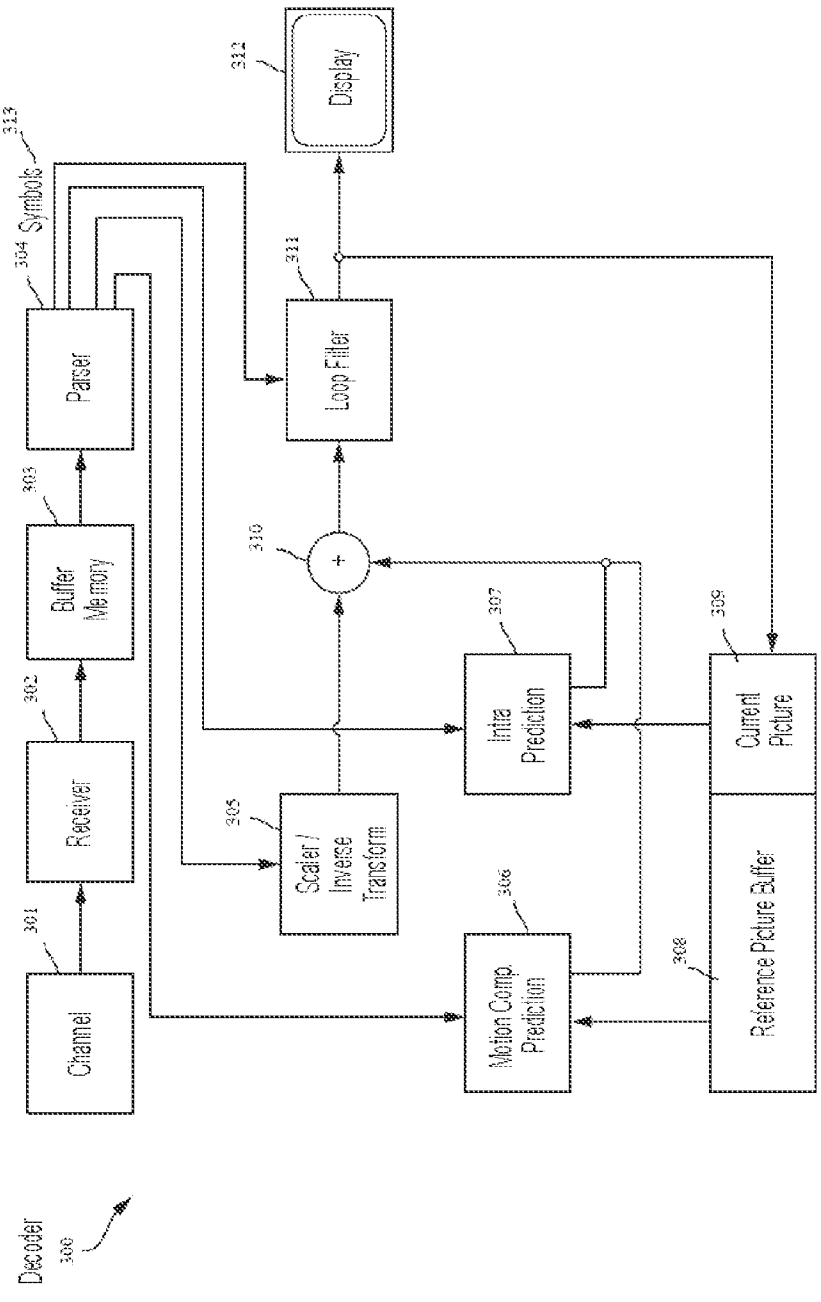

Decoder
300
Channel
301
Receiver
302
Buffer Memory
303
Parser
304
Symbols
313
Scaler / Inverse Transform
305
Motion Comp Prediction
306
Intra Prediction
307
Reference Picture Buffer
308
Current Picture
309
+
310
Loop Filter
311
Display
312

FIG. 4

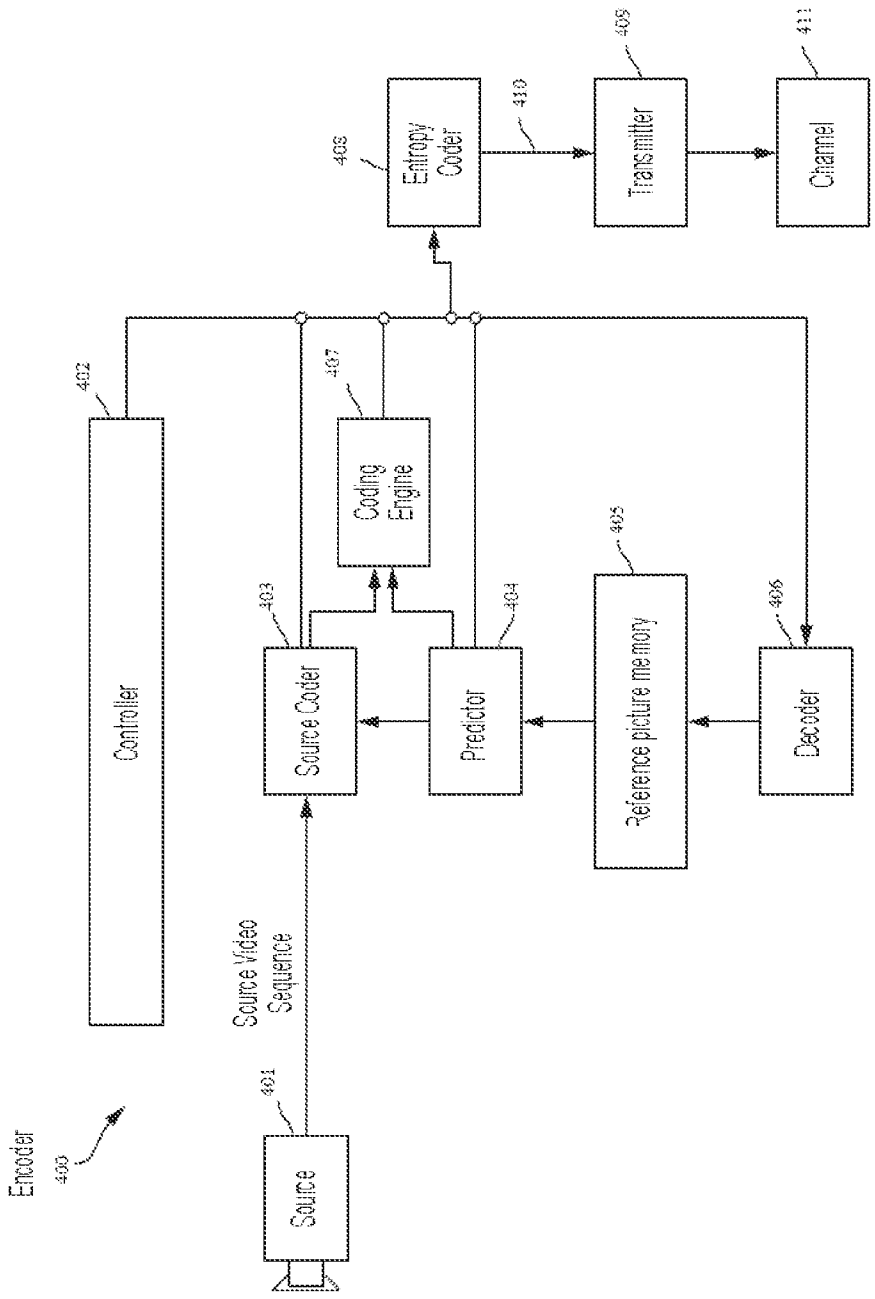

Encoder
400
Source
401
Source Video Sequence
Controller
402
Source Coder
403
Predictor
404
Reference picture memory
405
Decoder
406
Coding Engine
407
Entropy Coder
408
410
Transmitter
409
Channel
411

FIG. 5

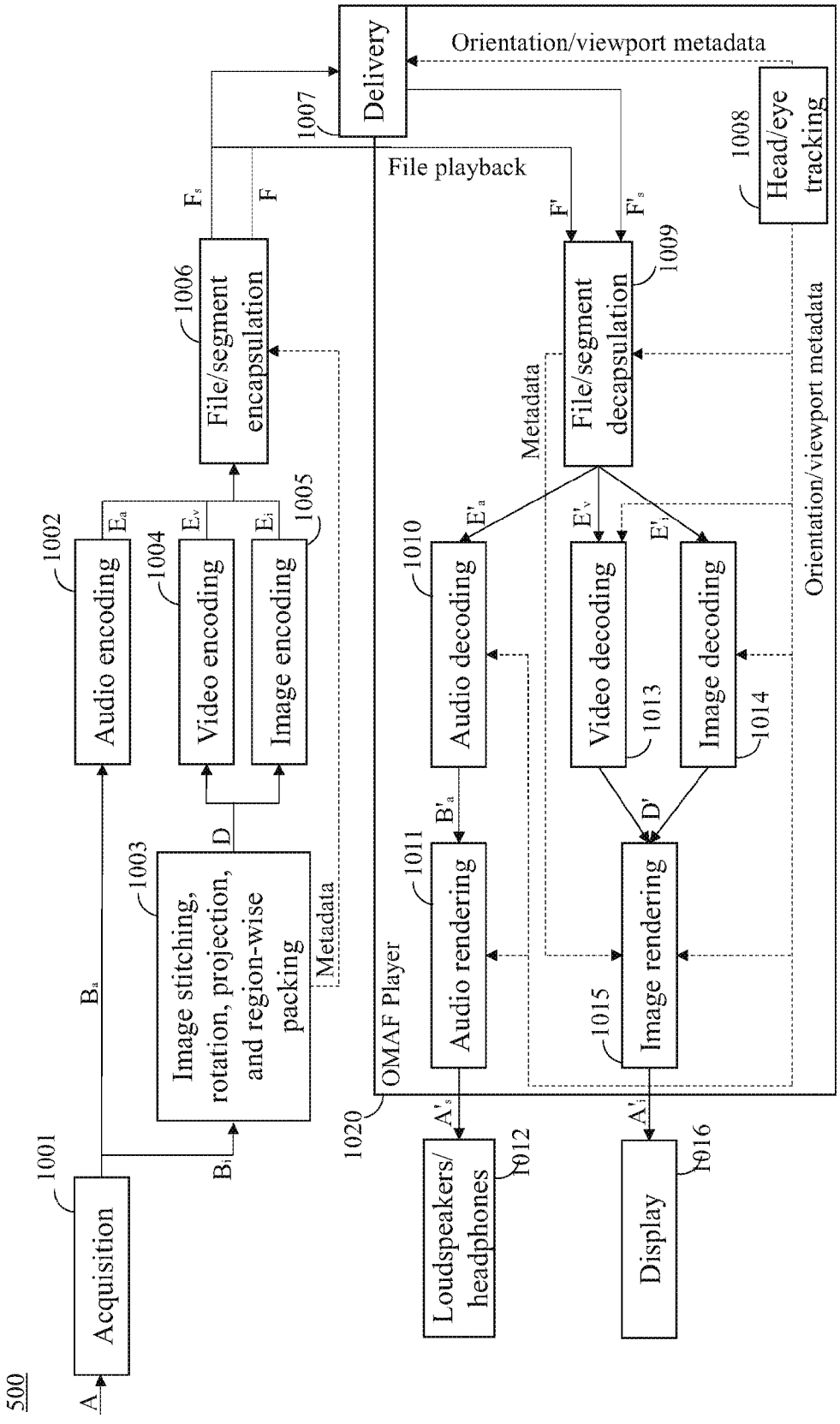

500
A
Acquisition
1001
B_a
B_i
Audio encoding
1002
Image stitching, rotation, projection, and region-wise packing
1003
D
Video encoding
1004
Image encoding
1005
E_a
E_v
E_i
Metadata
File/segment encapsulation
1006
F_s
F
Delivery
1007
File playback
Orientation/viewport metadata
Head/eye tracking
1008
F'
F'_s
File/segment decapsulation
1009
Metadata
E'_a
E'_v
E'_i
Audio decoding
1010
Video decoding
1013
Image decoding
1014
B'_a
D'
Audio rendering
1011
Image rendering
1015
OMAF Player
1020
Orientation/viewport metadata
A'_s
A'_i
Loudspeakers/ headphones
1012
Display
1016

FIG. 6

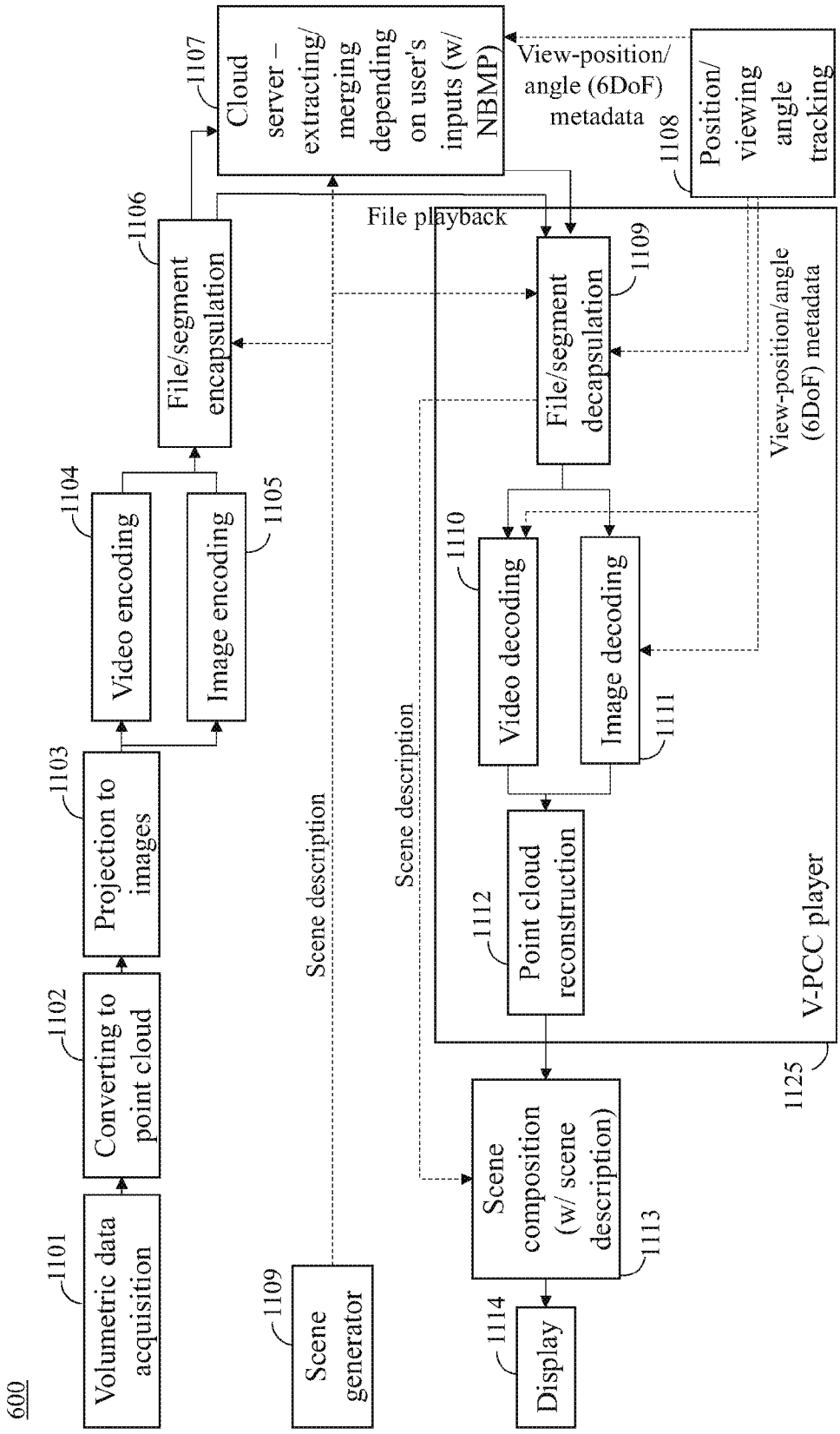

600
Volumetric data acquisition
1101
Converting to point cloud
1102
Projection to images
1103
Video encoding
1104
Image encoding
1105
File/segment encapsulation
1106
Cloud server – extracting/ merging depending on user's inputs (w/ NBMP)
1107
Scene generator
1109
Scene description
Scene description
File playback
View-position/ angle (6DoF) metadata
Position/ viewing angle tracking
1108
File/segment decapsulation
1109
Video decoding
1110
Image decoding
1111
Point cloud reconstruction
1112
Scene composition (w/ scene description)
1113
Display
1114
V-PCC player
1125
View-position/angle (6DoF) metadata

FIG. 7

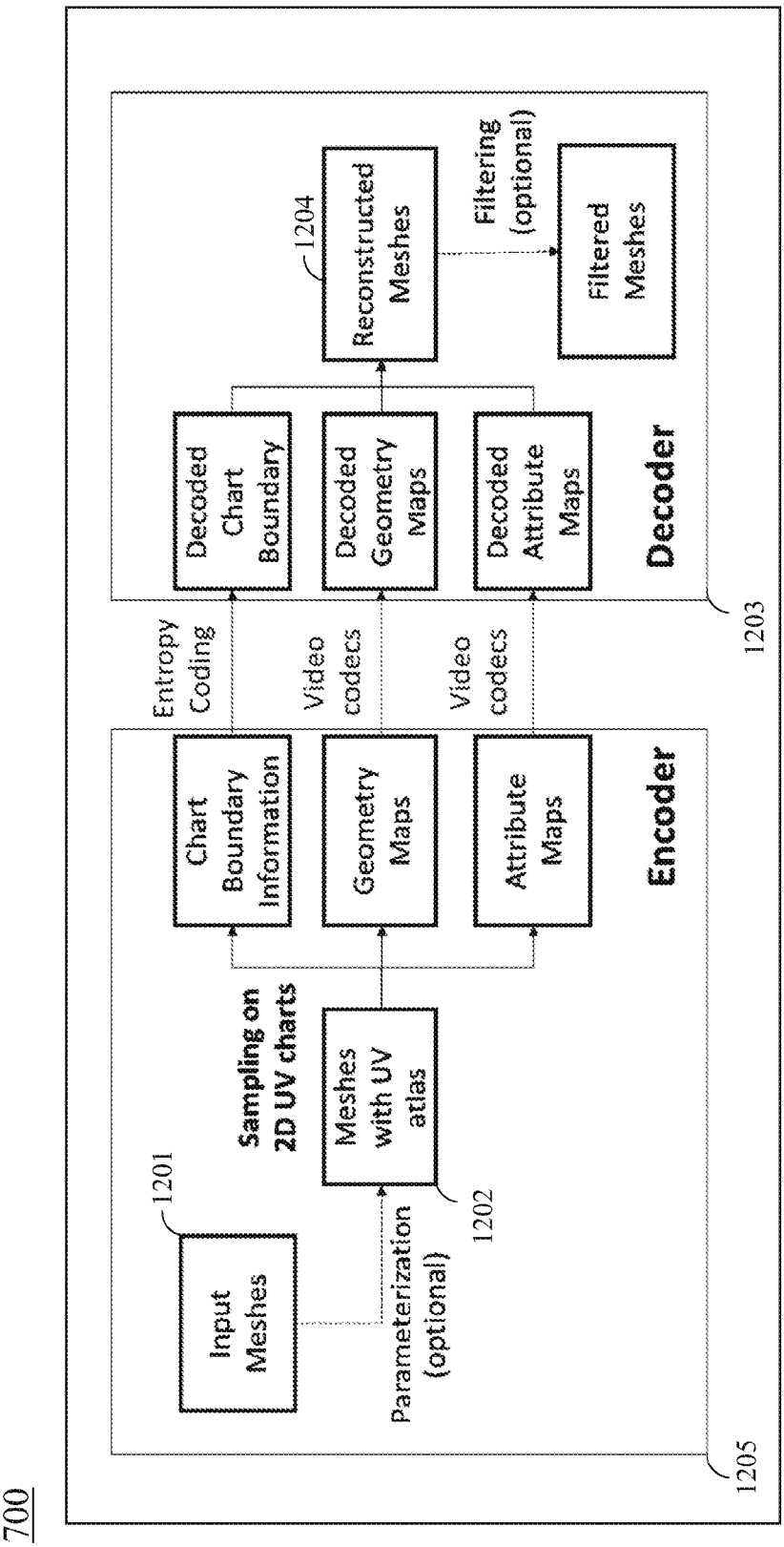

700
Input Meshes
1201
Parameterization (optional)
Sampling on 2D UV charts
Meshes with UV atlas
1202
Chart Boundary Information
Geometry Maps
Attribute Maps
Encoder
1205
Entropy Coding
Video codecs
Video codecs
Decoded Chart Boundary
Decoded Geometry Maps
Decoded Attribute Maps
Decoder
1203
1204
Reconstructed Meshes
Filtering (optional)
Filtered Meshes

FIG. 8

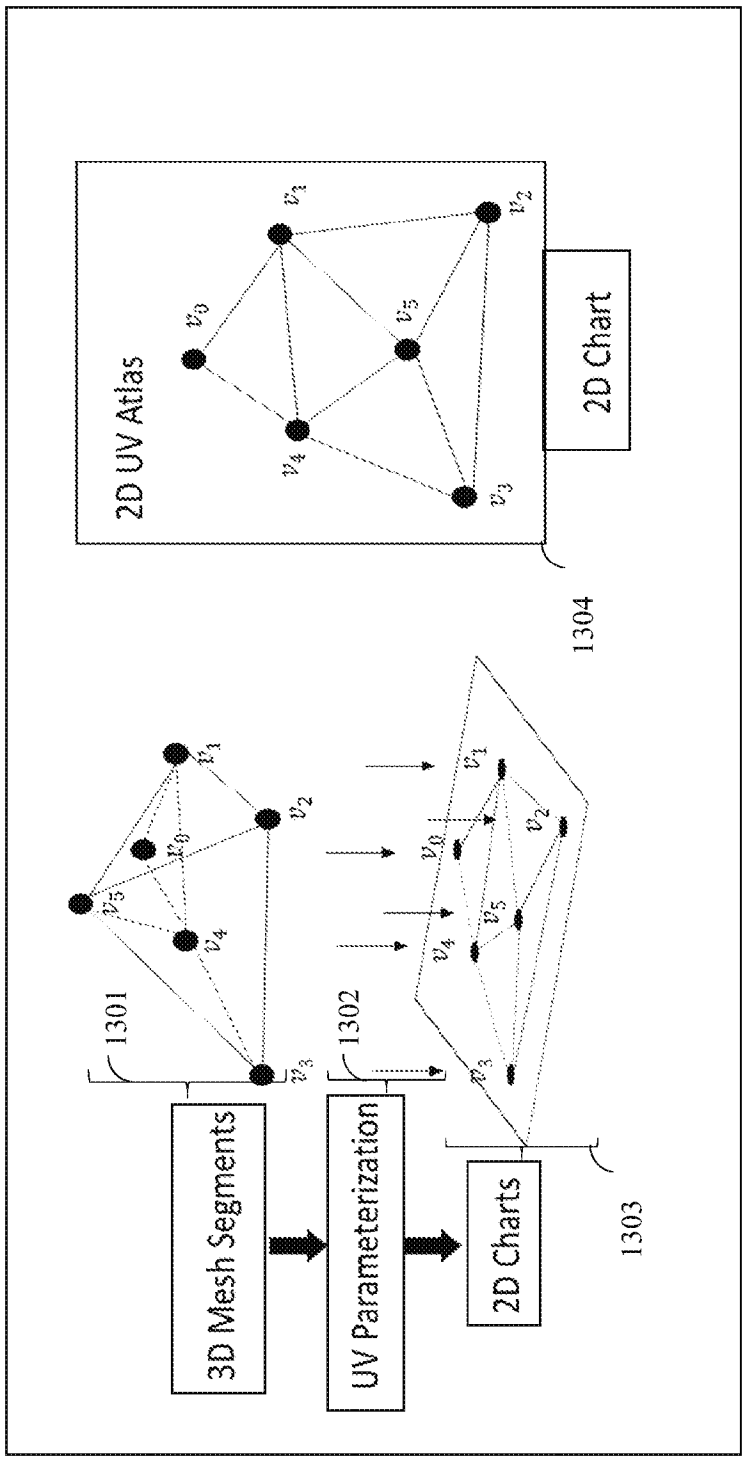

800
3D Mesh Segments
UV Parameterization
2D Charts
1301
1302
1303
1304
2D UV Atlas
2D Chart

FIG. 9

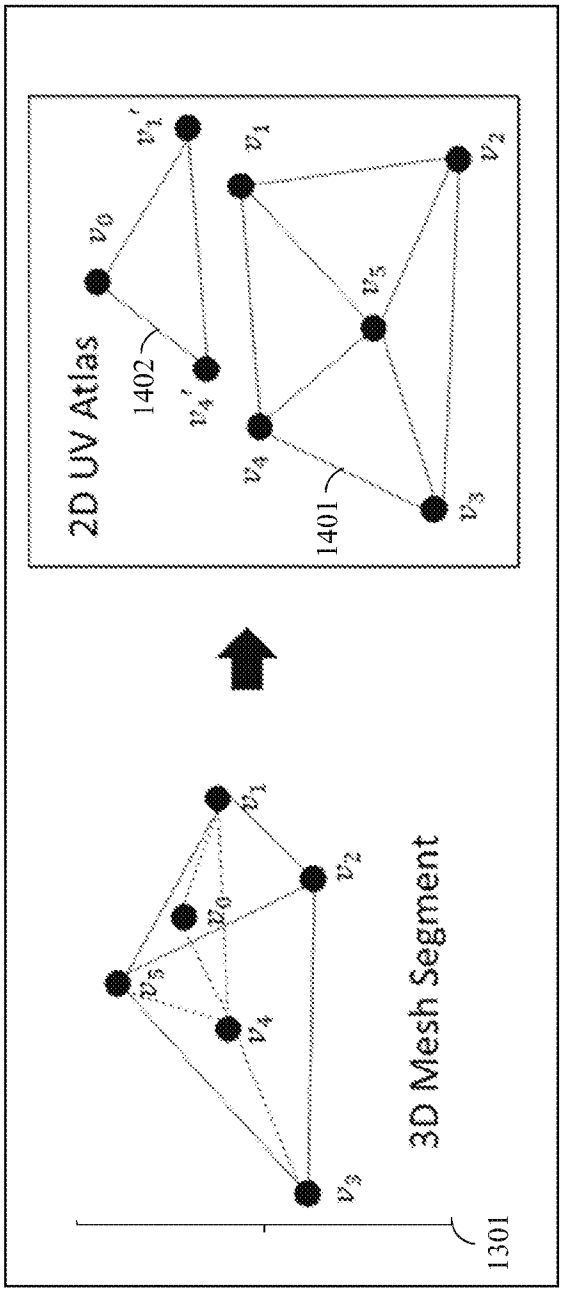


FIG. 10

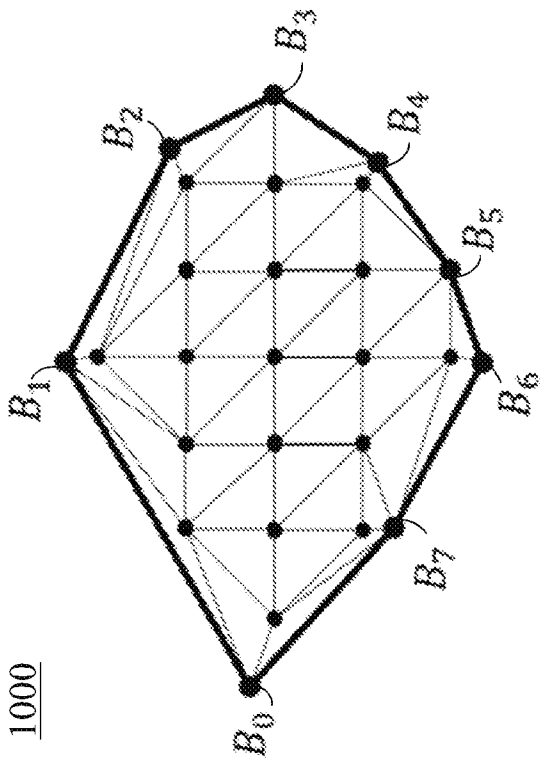


FIG. 11

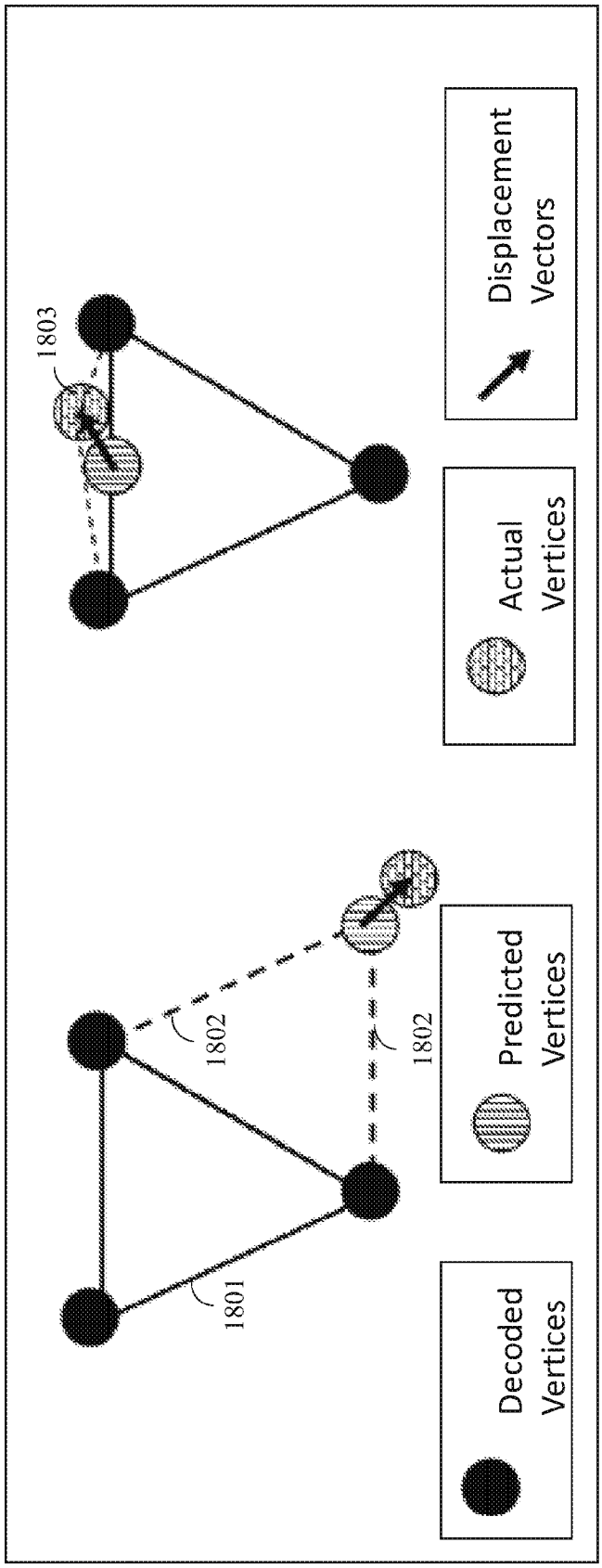

1100
Decoded Vertices
Predicted Vertices
Actual Vertices
Displacement Vectors
1801
1802
1802
1803

FIG. 12

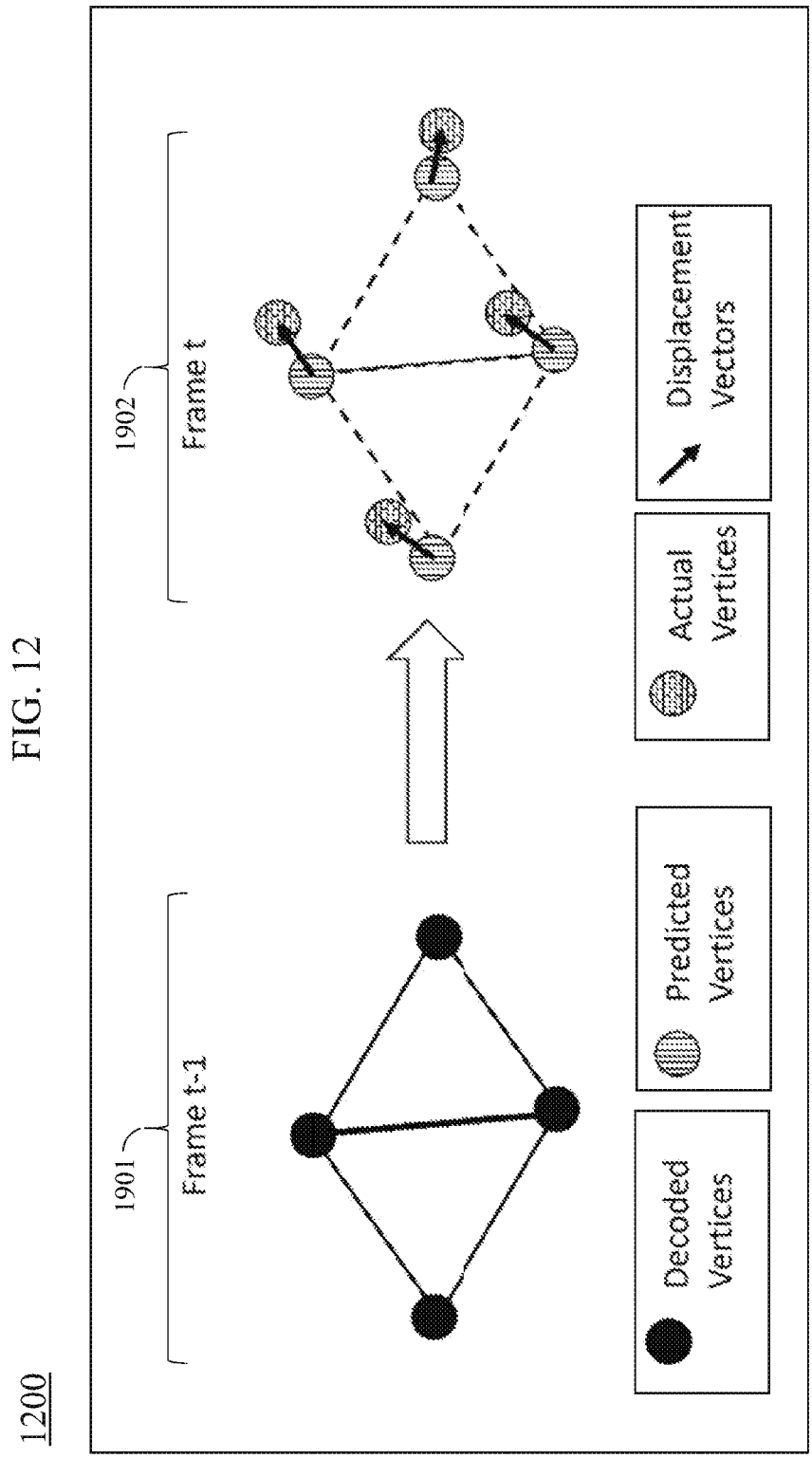

1200
1901
Frame t-1
1902
Frame t
Decoded Vertices
Predicted Vertices
Actual Vertices
Displacement Vectors

FIG. 13

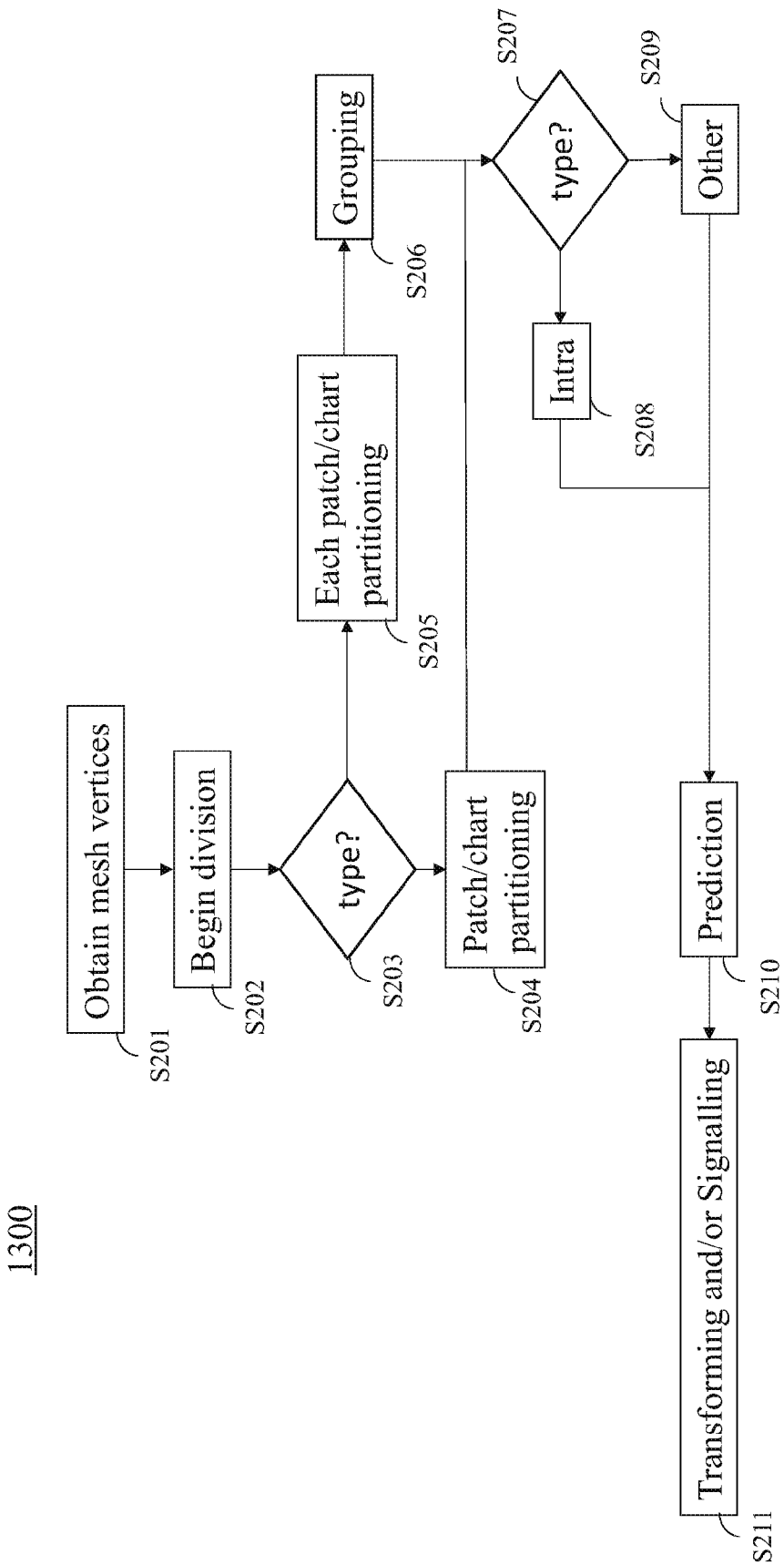

1300
Obtain mesh vertices
S201
Begin division
S202
type?
S203
Each patch/chart partitioning
S205
Grouping
S206
Patch/chart partitioning
S204
type?
S207
Intra
S208
Other
S209
Prediction
S210
Transforming and/or Signalling
S211

FIG. 14

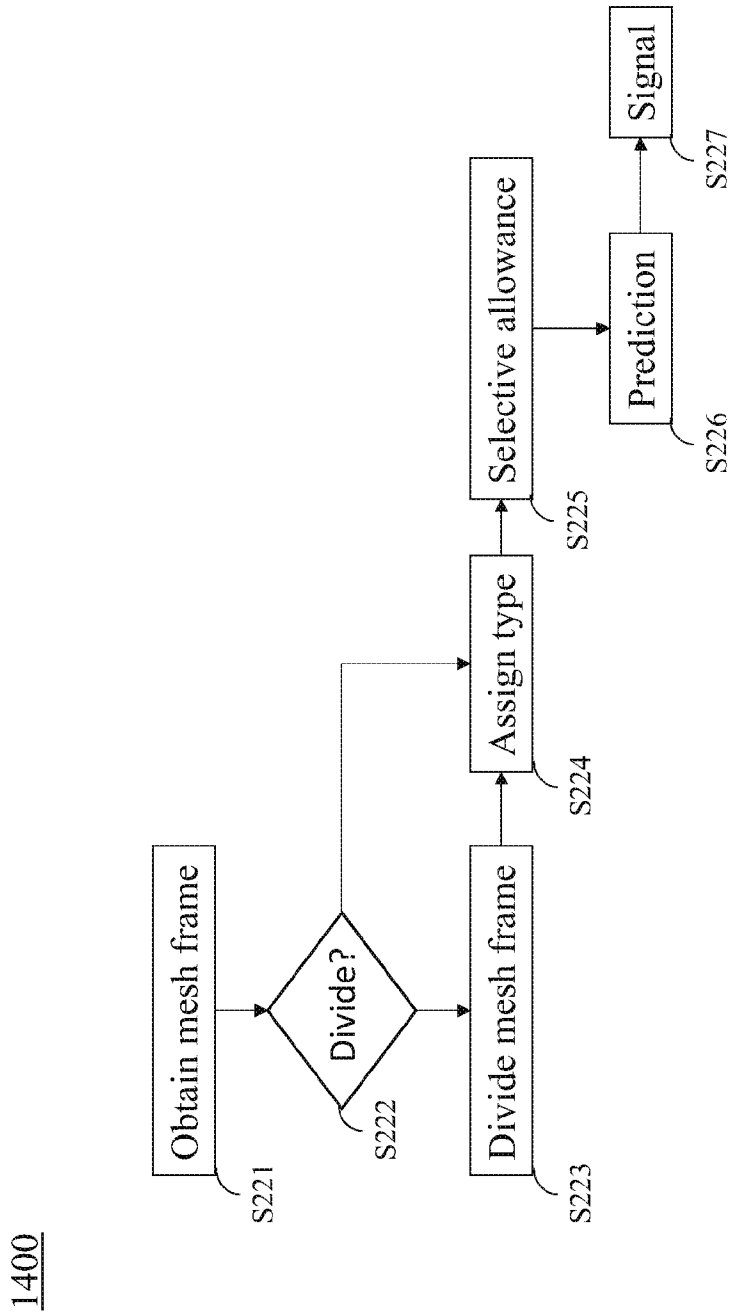

1400
Obtain mesh frame
S221
Divide?
S222
Divide mesh frame
S223
Assign type
S224
Selective allowance
S225
Prediction
S226
Signal
S227

FIG. 15

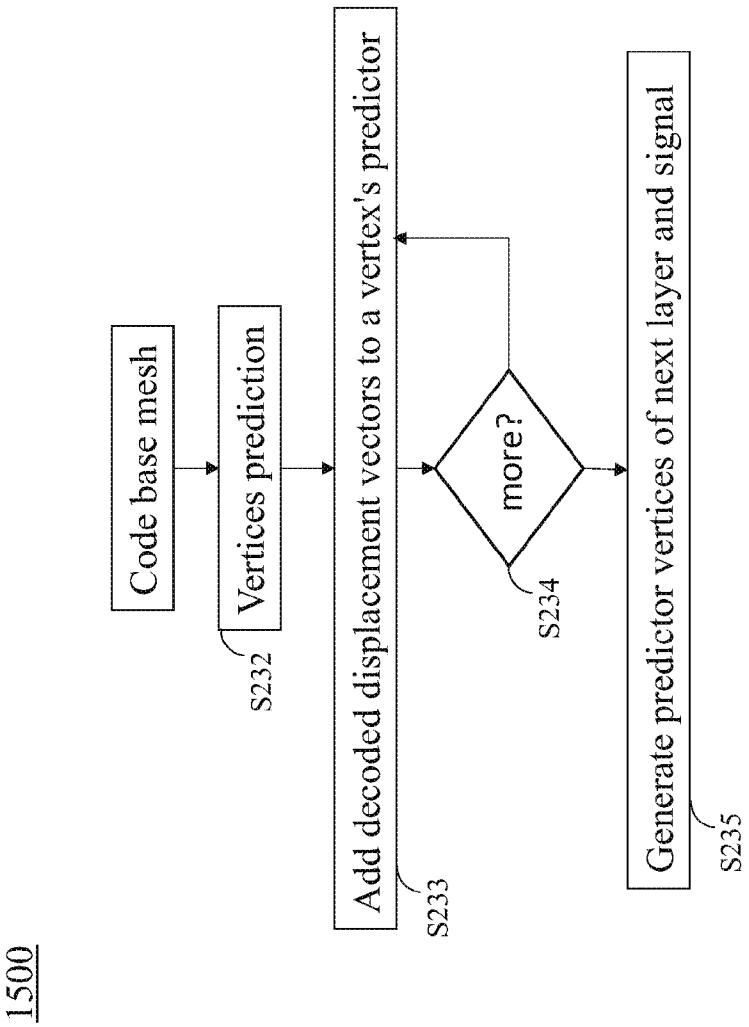

1500
Code base mesh
Vertices prediction
S232
Add decoded displacement vectors to a vertex's predictor
S233
more?
S234
Generate predictor vertices of next layer and signal
S235

FIG. 16

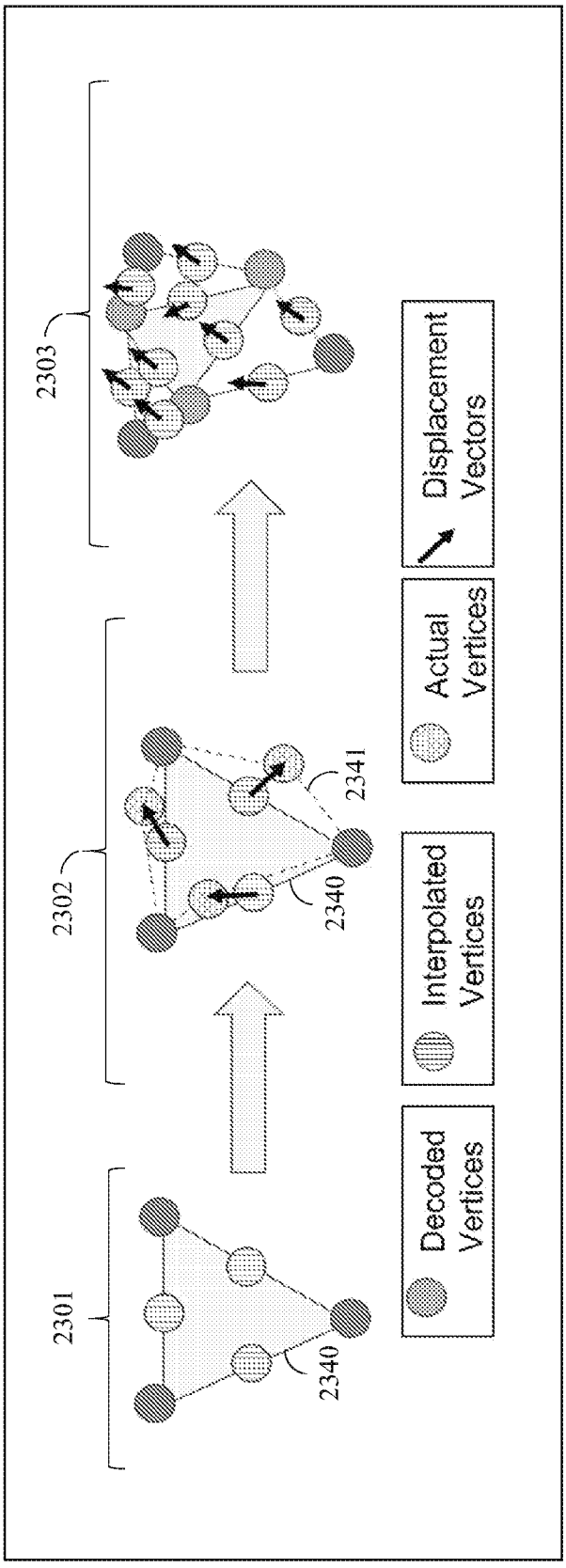

1600
2301
2340
2302
2340
2341
2303
Decoded Vertices
Interpolated Vertices
Actual Vertices
Displacement Vectors

FIG. 17

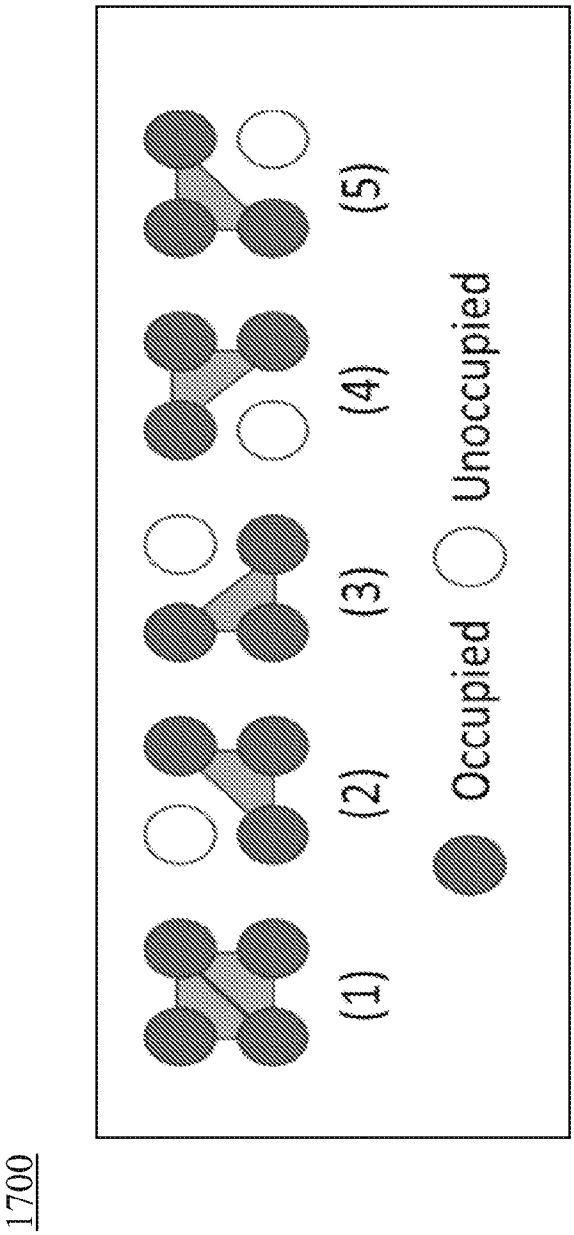


FIG. 18

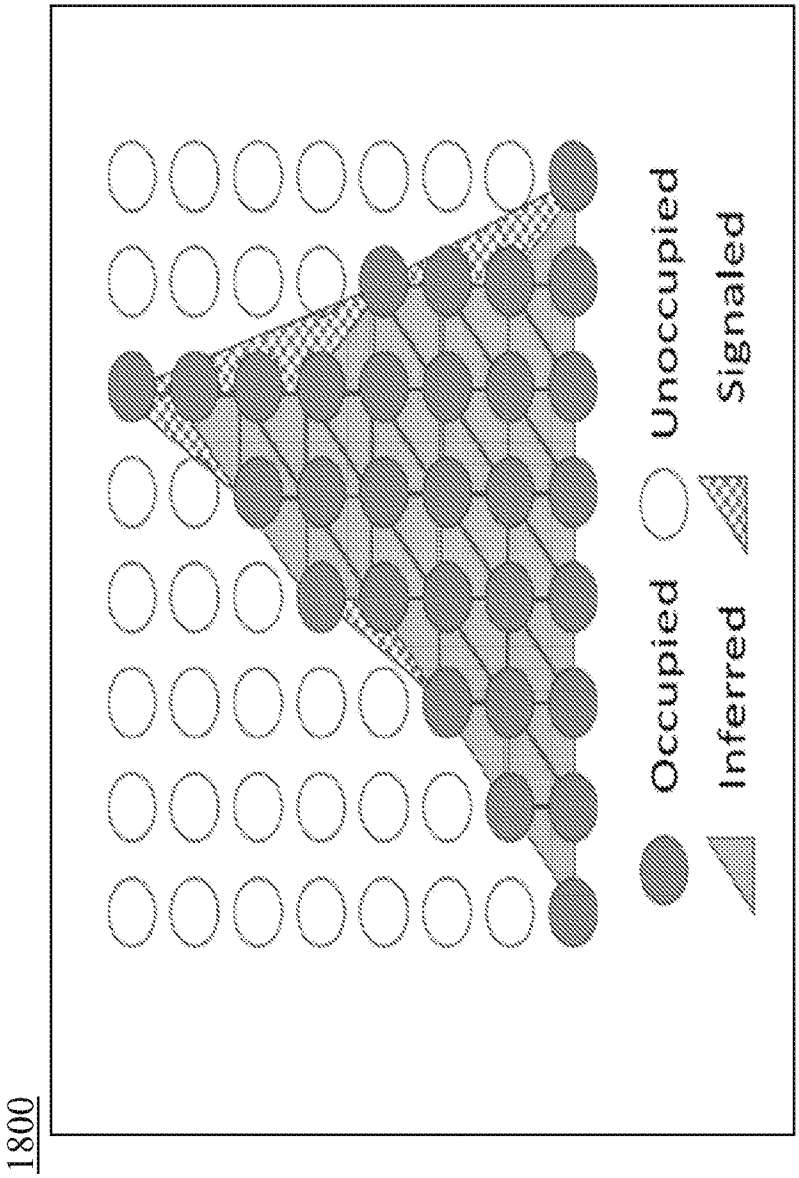


FIG. 19

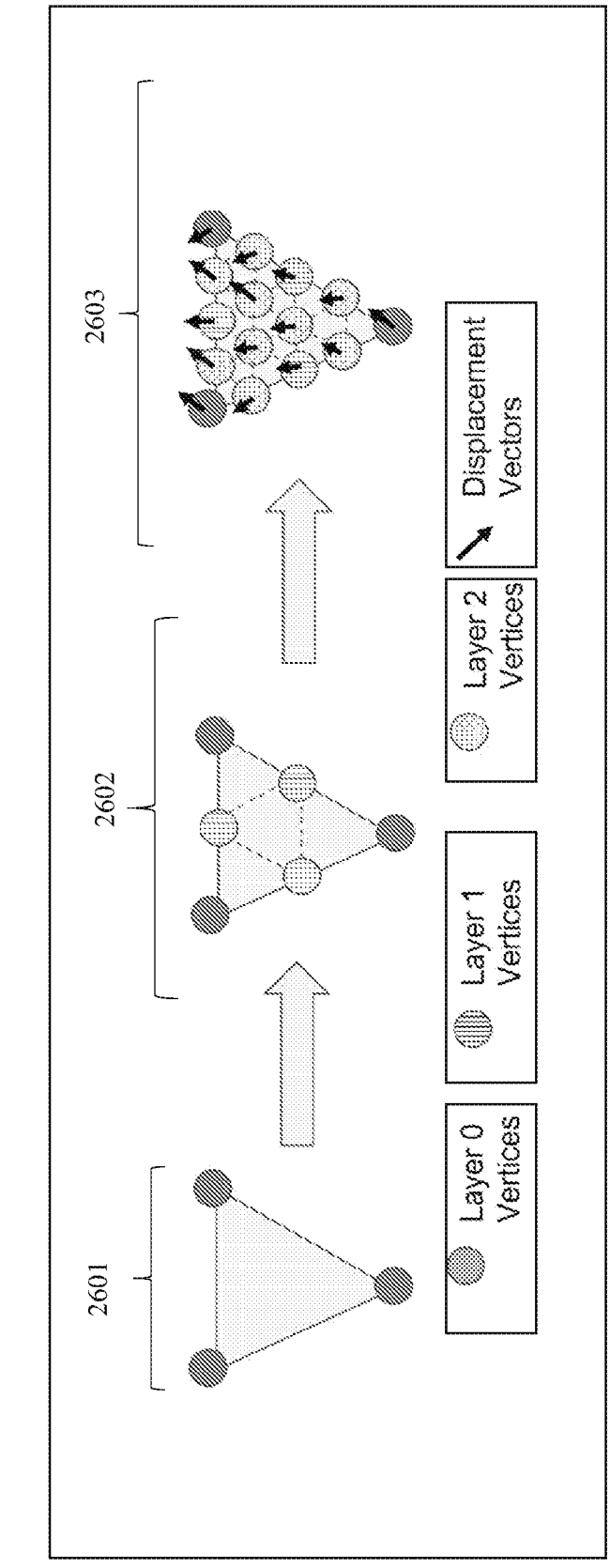

1900
2601
2602
2603
Layer 0 Vertices
Layer 1 Vertices
Layer 2 Vertices
Displacement Vectors

FIG. 20

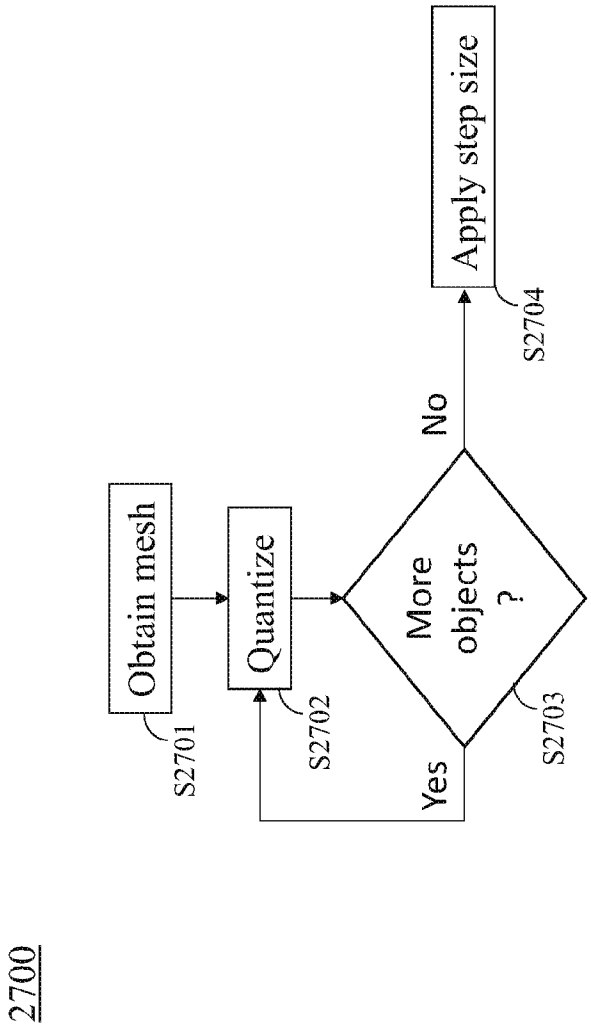

2700
Obtain mesh
S2701
Quantize
S2702
More objects ?
S2703
Yes
No
Apply step size
S2704

FIG. 21

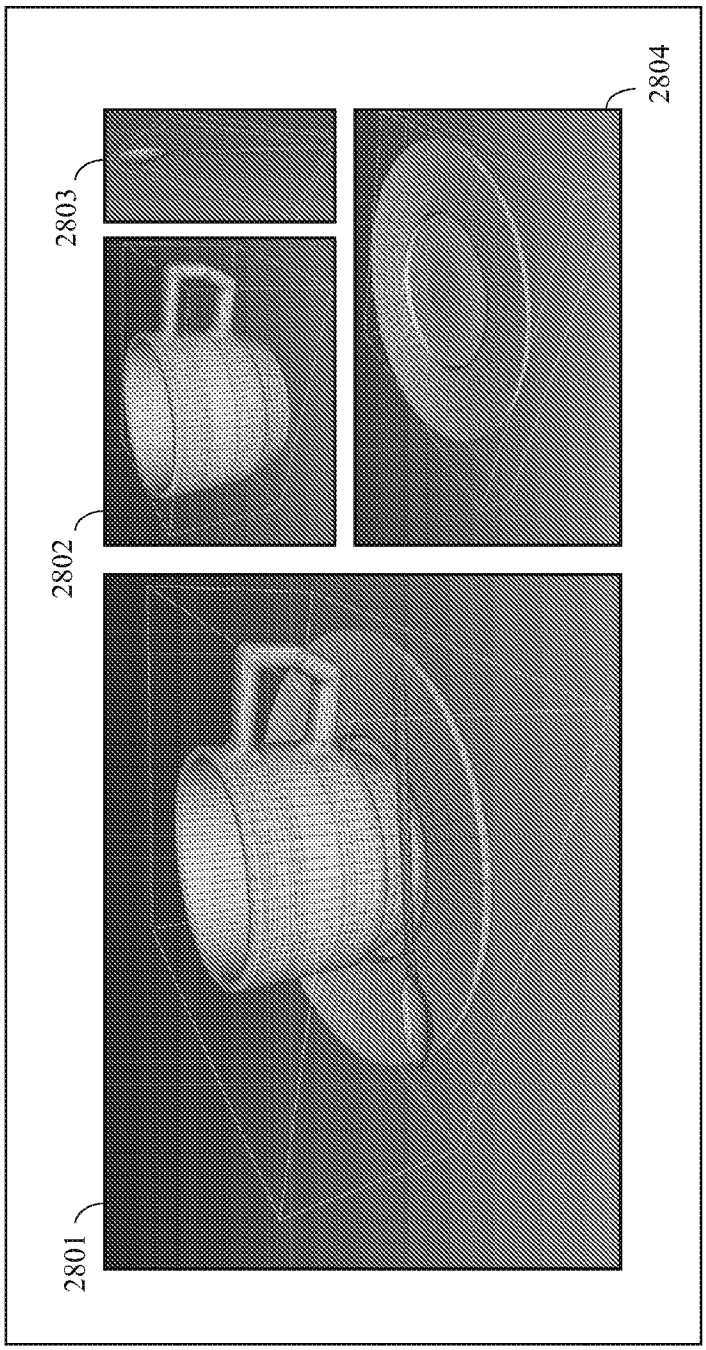


FIG. 22

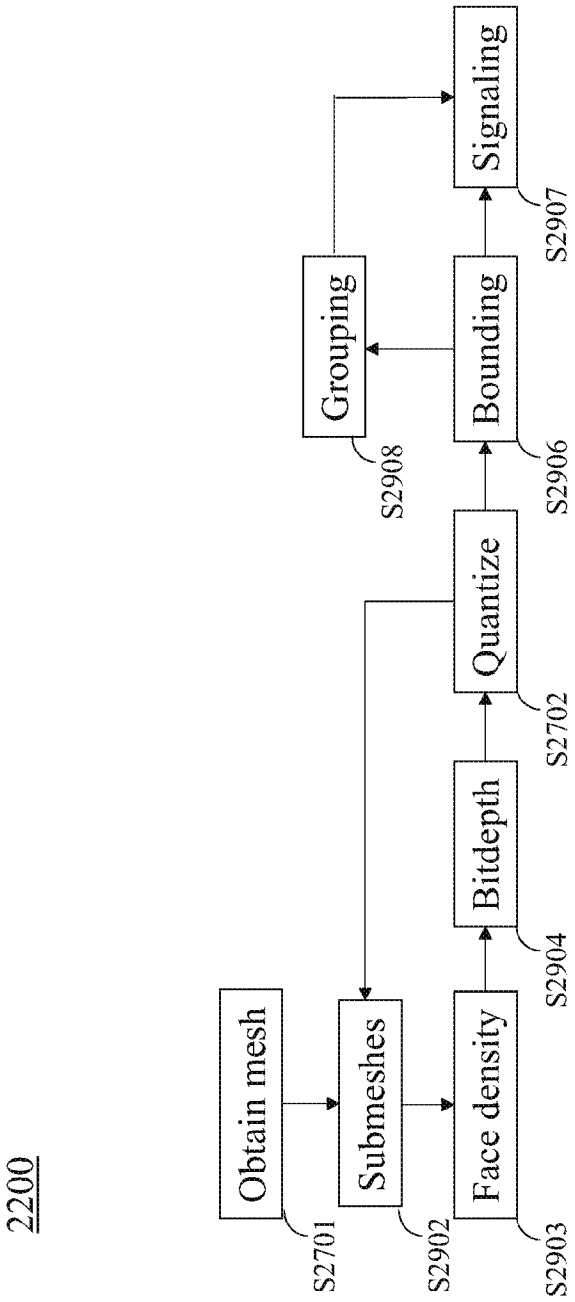

2200
Obtain mesh
S2701
Submeshes
S2902
Face density
S2903
Bitdepth
S2904
Quantize
S2702
Bounding
S2906
Grouping
S2908
Signaling
S2907

FIG. 23

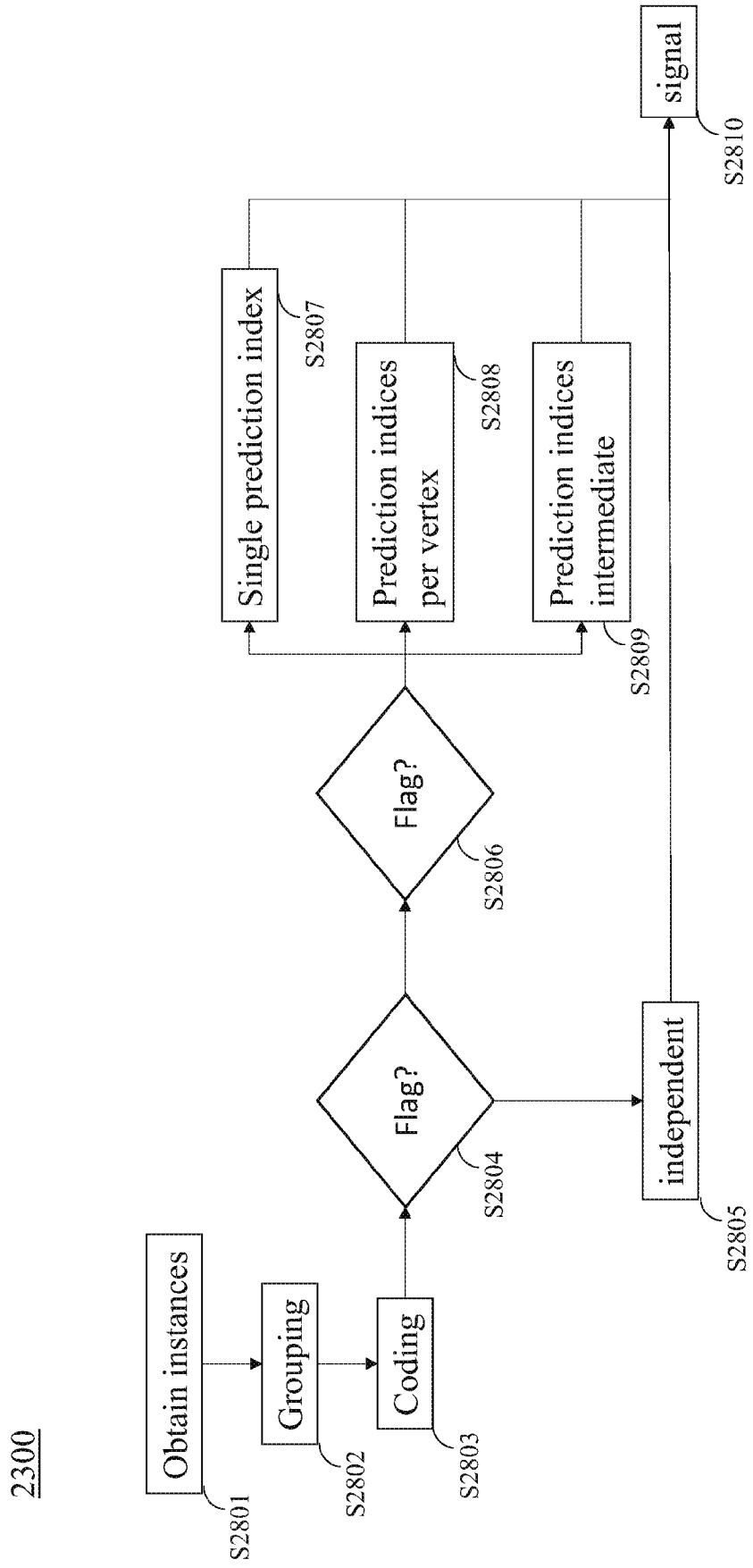
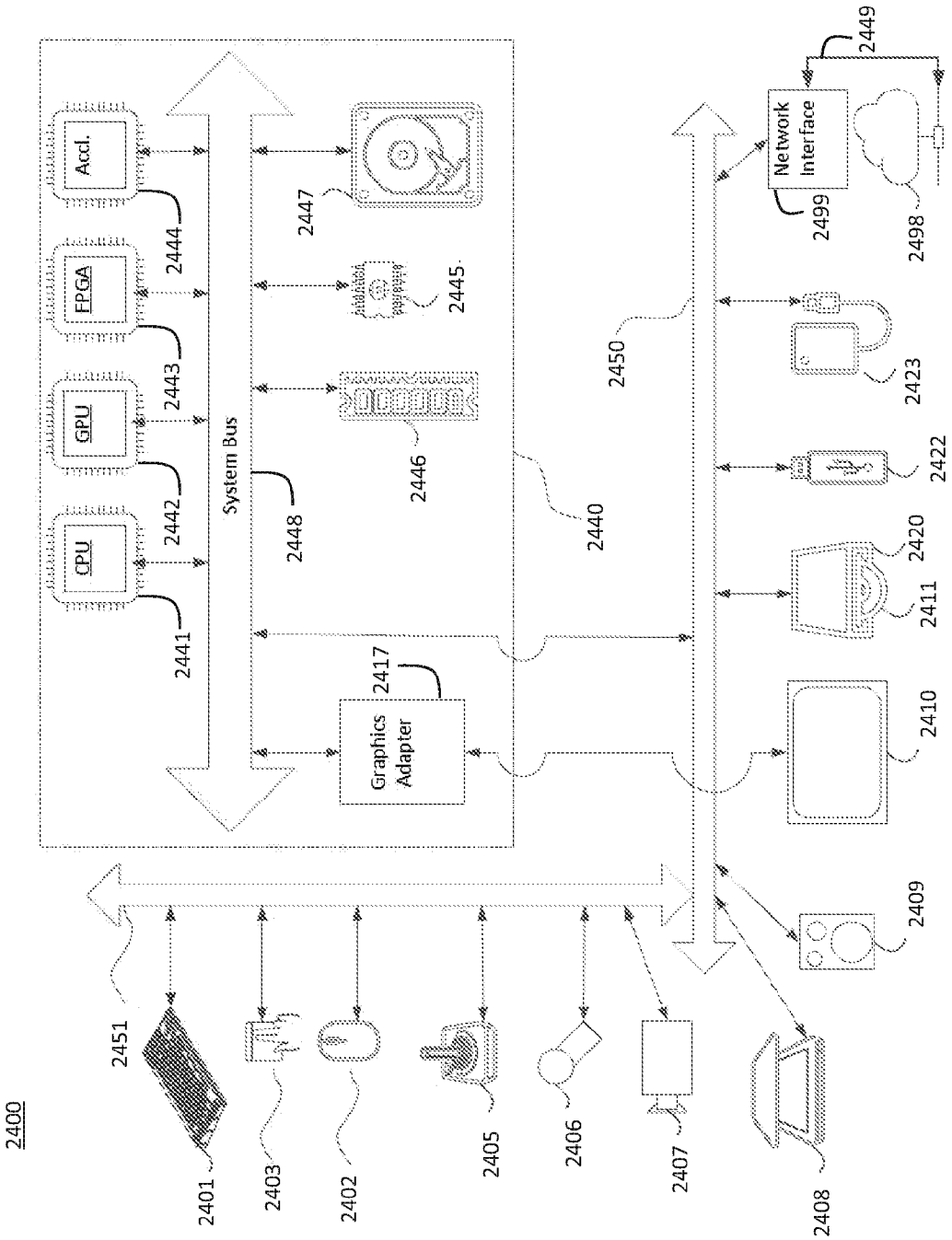

2300
Obtain instances
S2801
Grouping
S2802
Coding
S2803
Flag?
S2804
independent
S2805
Flag?
S2806
Single prediction index
S2807
Prediction indices per vertex
S2808
Prediction indices intermediate
S2809
signal
S2810

FIG. 24


2400
2401
2402
2403
2405
2406
2407
2408
2409
2410
2411
2417
2420
2422
2423
2440
2441
2442
2443
2444
2445
2446
2447
2448
2449
2450
2451
2498
2499
CPU
GPU
FPGA
Accl.
System Bus
Graphics Adapter
Network Interface

INTERNATIONAL SEARCH REPORT

International application No.
PCT/US 23/23296

A. CLASSIFICATION OF SUBJECT MATTER

IPC - INV. G06T 17/20, G06T 17/00, G06T 9/00 (2023.01)
ADD. G06T 19/00 (2023.01)

CPC - INV. G06T 17/20, G06T 17/00, G06T 9/001

ADD. G06T 19/00

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)
See Search History document

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

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)
See Search History document

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
A	US 2020/0410752 A1 (HypeVR) 31 December 2020 (31.12.2020), entire document.	1-20
A	US 10,121,279 B1 (Ansys, Inc.) 06 November 2018 (06.11.2018), entire document.	1-20
A	US 2006/0290695 A1 (Salomie) 28 December 2006 (28.12.2006), entire document.	1-20

☐ 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
"D" document cited by the applicant in the international application
"E" earlier application or patent but published on or after the international filing date
"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)
"O" document referring to an oral disclosure, use, exhibition or other means
"P" document published prior to the international filing date but later than the priority date claimed
"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention
"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone
"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art
"&" document member of the same patent family

Date of the actual completion of the international search	Date of mailing of the international search report
20 July 2023	AUG 22 2023
Name and mailing address of the ISA/US Mail Stop PCT, Attn: ISA/US, Commissioner for Patents P.O. Box 1450, Alexandria, Virginia 22313-1450 Facsimile No. 571-273-8300	Authorized officer Kari Rodriquez Telephone No. PCT Helpdesk: 571-272-4300