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

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
29 November 2018 (29.11.2018)

(10) International Publication Number
WO 2018/218047 A1

(51) International Patent Classification:
H04N 21/235 (2011.01) *H04N 21/236* (2011.01) *H04N 21/81* (2011.01) *H04N 21/845* (2011.01) *H04N 21/854* (2011.01)

(21) International Application Number: PCT/US2018/034435

(22) International Filing Date: 24 May 2018 (24.05.2018)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:

62/511,189	25 May 2017 (25.05.2017)	US
15/987,231	23 May 2018 (23.05.2018)	US

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

(72) Inventors: **WANG, Ye-Kui**; 6264 Sunrose Crest Way, San Diego, California 92130 (US). **BI, Ning**; 5775 Morehouse Drive, San Diego, California 92121-1714 (US). **FORUTANPOUR, Bijan**; 5775 Morehouse Drive, San Diego, California 92121-1714 (US).

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

(81) Designated States *(unless otherwise indicated, for every kind of national protection available)*: AE, AG, AL, AM, AO, AT, AU, AZ, BA, BB, BG, BH, BN, BR, BW, BY, BZ, CA, CH, CL, CN, CO, CR, CU, CZ, DE, DJ, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IR, IS, JO, JP, KE, KG, KH, KN, KP, KR, KW, KZ, LA, LC, LK, LR, LS, LU, LY, MA, MD, ME, MG, MK, MN, MW, MX, MY, MZ, NA, NG, NI, NO, NZ, OM, PA, PE, PG, PH, PL, PT, QA, RO, RS, RU, RW, SA, SC, SD, SE, SG, SK, SL, SM, ST, SV, SY, TH, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, ZA, ZM, ZW.

(84) Designated States *(unless otherwise indicated, for every kind of regional protection available)*: ARIPO (BW, GH, GM, KE, LR, LS, MW, MZ, NA, RW, SD, SL, ST, SZ, TZ,

(54) Title: HIGH-LEVEL SIGNALLING FOR FISHEYE VIDEO DATA

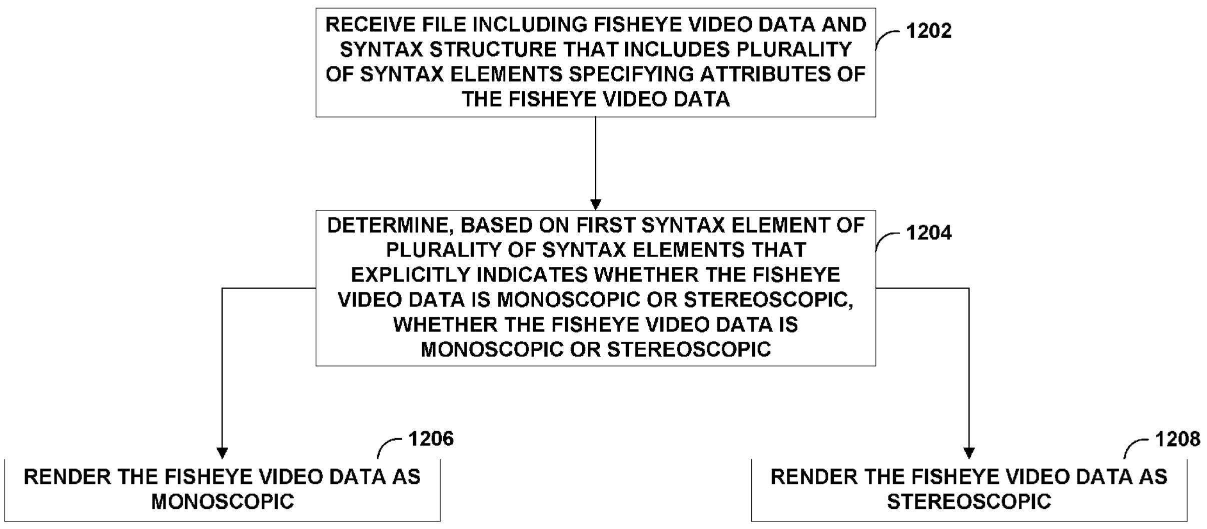

RECEIVE FILE INCLUDING FISHEYE VIDEO DATA AND SYNTAX STRUCTURE THAT INCLUDES PLURALITY OF SYNTAX ELEMENTS SPECIFYING ATTRIBUTES OF THE FISHEYE VIDEO DATA
1202
DETERMINE, BASED ON FIRST SYNTAX ELEMENT OF PLURALITY OF SYNTAX ELEMENTS THAT EXPLICITLY INDICATES WHETHER THE FISHEYE VIDEO DATA IS MONOSCOPIC OR STEREOSCOPIC, WHETHER THE FISHEYE VIDEO DATA IS MONOSCOPIC OR STEREOSCOPIC
1204
1206
RENDER THE FISHEYE VIDEO DATA AS MONOSCOPIC
1208
RENDER THE FISHEYE VIDEO DATA AS STEREOSCOPIC

FIG. 12

(57) Abstract: An example method includes processing a file including fisheye video data, the file including a syntax structure including a plurality of syntax elements that specify attributes of the fisheye video data, wherein the plurality of syntax elements includes: a first syntax element that explicitly indicates whether the fisheye video data is monoscopic or stereoscopic, and one or more syntax elements that implicitly indicate whether the fisheye video data is monoscopic or stereoscopic; determining, based on the first syntax element, whether the fisheye video data is monoscopic or stereoscopic; and rendering, based on the determination, the fisheye video data as monoscopic or stereoscopic.

WO 2018/218047 A1

[Continued on next page]

UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, RU, TJ, TM), European (AL, AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HR, HU, IE, IS, IT, LT, LU, LV, MC, MK, MT, NL, NO, PL, PT, RO, RS, SE, SI, SK, SM, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, KM, ML, MR, NE, SN, TD, TG).

Published:

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

HIGH-LEVEL SIGNALLING FOR FISHEYE VIDEO DATA

[0001] This application claims the benefit of U.S. Provisional Application No. **62/511,189**, filed May 25, 2017, the entire content of which is hereby incorporated by reference.

TECHNICAL FIELD

[0002] This disclosure relates to storage and transport of encoded video data.

BACKGROUND

[0003] Digital video capabilities can be incorporated into a wide range of devices, including digital televisions, digital direct broadcast systems, wireless broadcast systems, personal digital assistants (PDAs), laptop or desktop computers, digital cameras, digital recording devices, digital media players, video gaming devices, video game consoles, cellular or satellite radio telephones, video teleconferencing devices, and the like. Digital video devices implement video compression techniques, such as those described in the standards defined by MPEG-2, MPEG-4, ITU-T H.263 or ITU-T H.264/MPEG-4, Part 10, Advanced Video Coding (AVC), ITU-T H.265 (also referred to High Efficiency Video Coding (HEVC)), and extensions of such standards, to transmit and receive digital video information more efficiently.

[0004] Video compression techniques perform spatial prediction and/or temporal prediction to reduce or remove redundancy inherent in video sequences. For block-based video coding, a video frame or slice may be partitioned into macroblocks. Each macroblock can be further partitioned. Macroblocks in an intra-coded (I) frame or slice are encoded using spatial prediction with respect to neighboring macroblocks. Macroblocks in an inter-coded (P or B) frame or slice may use spatial prediction with respect to neighboring macroblocks in the same frame or slice or temporal prediction with respect to other reference frames.

[0005] After video data has been encoded, the video data may be packetized for transmission or storage. The video data may be assembled into a video file conforming to any of a variety of standards, such as the International Organization for Standardization (ISO) base media file format and extensions thereof, such as AVC.

SUMMARY

[0006] In one example, a method includes processing a file including fisheye video data, the file including a syntax structure including a plurality of syntax elements that specify attributes of the fisheye video data, wherein the plurality of syntax elements includes: a first syntax element that explicitly indicates whether the fisheye video data is monoscopic or stereoscopic, and one or more syntax elements that implicitly indicate whether the fisheye video data is monoscopic or stereoscopic; determining, based on the first syntax element, whether the fisheye video data is monoscopic or stereoscopic; and rendering, based on the determination, the fisheye video data as monoscopic or stereoscopic.

[0007] In another example, a device includes a memory configured to store at least a portion of a file including fisheye video data, the file including a syntax structure including a plurality of syntax elements that specify attributes of the fisheye video data, wherein the plurality of syntax elements includes: a first syntax element that explicitly indicates whether the fisheye video data is monoscopic or stereoscopic, and one or more syntax elements that implicitly indicate whether the fisheye video data is monoscopic or stereoscopic; and one or more processors configured to: determine, based on the first syntax element, whether the fisheye video data is monoscopic or stereoscopic; and render, based on the determination, the fisheye video data as monoscopic or stereoscopic

[0008] In another example, a method includes obtaining fisheye video data and extrinsic parameters of cameras used to capture the fisheye video data; determining, based on the extrinsic parameters, whether the fisheye video data is monoscopic or stereoscopic; and encoding, in a file, the fisheye video data and a syntax structure including a plurality of syntax elements that specify attributes of the fisheye video data, wherein the plurality of syntax elements includes: a first syntax element that explicitly indicates whether the fisheye video data is monoscopic or stereoscopic, and one or more syntax elements that explicitly indicate the extrinsic parameters of the cameras used to capture the fisheye video data.

[0009] In another example, a device includes a memory configured to store at fisheye video data; and one or more processors configured to: obtain extrinsic parameters of cameras used to capture the fisheye video data; determine, based on the extrinsic parameters, whether the fisheye video data is monoscopic or stereoscopic; and encode, in a file, the fisheye video data and a syntax structure including a plurality of syntax elements that specify attributes of the fisheye video data, wherein the plurality of syntax elements includes: a first syntax element that explicitly indicates whether the fisheye video data is monoscopic or stereoscopic, and one or more syntax elements that explicitly indicate the extrinsic parameters of the cameras used to capture the fisheye video data.

[0010] The details of one or more examples are set forth in the accompanying drawings and the description below. Other features, objects, and advantages will be apparent from the description, drawings, and claims.

BRIEF DESCRIPTION OF DRAWINGS

[0011] FIGS. 1A and 1B are block diagrams illustrating example devices for capturing omnidirectional image content, in accordance with one or more example techniques described in this disclosure.

[0012] FIG. 2 is an example of a picture that includes multiple fisheye images.

[0013] FIGS. 3–8 are conceptual diagrams that illustrate various extrinsic parameters and fields of view of omnidirectional images, in accordance with one or more example techniques described in this disclosure.

[0014] FIG. 9 is a block diagram illustrating an example system that implements techniques for streaming media data over a network.

[0015] FIG. 10 is a conceptual diagram illustrating elements of example multimedia content.

[0016] FIG. 11 is a block diagram illustrating elements of an example video file, which may correspond to a segment of a representation.

[0017] FIG. 12 is a flowchart illustrating an example technique for processing a file that includes fisheye video data, in accordance with one or more techniques of this disclosure.

[0018] FIG. 13 is a flowchart illustrating an example technique for generating a file that includes fisheye video data, in accordance with one or more techniques of this disclosure.

DETAILED DESCRIPTION

[0019] The example techniques described in this disclosure are related to processing files representing omnidirectional video or image data. When omnidirectional media content is consumed with certain devices (e.g., a head-mounted display and headphones), only the parts of the media that correspond to the user's viewing orientation are rendered, as if the user were in the spot where and when the media was captured (e.g., where the cameras(s) were). One of the most popular forms of omnidirectional media applications is omnidirectional video, also known as 360-degree video. Omnidirectional video is typically captured by multiple cameras that cover up to 360-degrees of the scene.

[0020] In general, omnidirectional video is formed from a sequence of omnidirectional images. Accordingly, the example techniques described in this disclosure are described with respect to generating omnidirectional image content. Then, for omnidirectional video content, these omnidirectional images can be displayed sequentially. In some examples, a user may desire to take only an omnidirectional image (e.g., as a snapshot of the entire 360-degree surrounding of the user), and the techniques described in this disclosure are applicable to such example cases as well.

[0021] Omnidirectional video may be stereoscopic or monoscopic. When the video is stereoscopic, a different image is shown to each eye such that the viewer may perceive depth. As such, stereoscopic video is typically captured using two cameras facing each direction. When the video is monoscopic, the same image is shown to both eyes.

[0022] Video data may be considered to be fisheye video data where it is captured using one or more fisheye lenses (or generated to appear as if captured using one or more fisheye lenses). A fisheye lens may be an ultra wide-angle lens that produces strong visual distortion intended to create a wide panoramic or hemispherical image.

[0023] The techniques may be applicable to captured video content, virtual reality, and generally to video and image displaying. The techniques may be used in mobile devices, but the techniques should not be considered limited to mobile applications. In general, the techniques may be for virtual reality applications, video game applications, or other applications where a 360-degree spherical video/image environment is desired.

[0024] In some examples, the omnidirectional image content may be captured with a camera device that includes two fisheye lenses. Where the two fisheye lenses are positioned on opposing sides of the camera device to capture opposite portions of the sphere of image content, the image content may be monoscopic and cover the full sphere of the 360-degree video. Similarly, where the two fisheye lenses are positioned on the same side of the camera device to capture the same portion of the sphere of image content, the image content may be stereoscopic and cover half of the sphere of the 360-degree video. The images generated by the cameras are circular images (e.g., one image frame includes two circular images).

[0025] FIGS. 1A and 1B are block diagrams illustrating example devices for capturing omnidirectional image content in accordance with one or more example techniques described in this disclosure. As illustrated in FIG. 1A, computing device 10A is a video capture device that includes fisheye lens 12A and fisheye lens 12B located on opposite sides of computing device 10A to capture monoscopic image content that covers the entire sphere (e.g., full 360-degree video content). As illustrated in FIG. 1B, computing device 10B is a video capture device that includes fisheye lens 12C and fisheye lens 12D located on the same side of computing device 10B to stereoscopic image content that covers about half of the sphere.

[0026] As described above, a camera device includes a plurality of fisheye lenses. Some example camera devices include two fisheye lenses, but the example techniques are not limited to two fisheye lenses. One example camera device may include 16 lenses (e.g., 16-camera array for filming 3D VR content). Another example camera device may include eight lenses, each with 195-degree angle of view (e.g., each lens captures 195 degrees of the 360 degrees of image content). Other example camera devices include three or four lenses. Some examples may include a 360-degree lens that captures 360-degrees of image content.

[0027] The example techniques described in this disclosure are generally described with respect to two fisheye lenses capturing omnidirectional image/video. However, the example techniques are no so limited. The example techniques may be applicable to example camera devices that include a plurality of lenses (e.g., two or more) even if the lenses are not fisheye lenses, and a plurality of fisheye lenses. For instance, the example techniques describe ways to stitch captured images, and the techniques may be applicable to examples where there are a plurality of captured images from a plurality of lenses (which may be fisheye lenses, as an example). While the example techniques are described with respect to two fisheye lenses, the example techniques are not so limited, and are applicable to the various camera types used for capturing omnidirectional images/videos.

[0028] The techniques of this disclosure may be applied to video files conforming to video data encapsulated according to any of ISO base media file format (e.g., ISOBMFF, ISO/IEC 14496-12), and other file formats derived from the ISOBMFF, including MPEG-4 file format (ISO/IEC 14496-15), Third Generation Partnership

Project (3GPP) file format (3GPP TS 26.244) and file formats for AVC and HEVC families of video codecs (ISO/IEC 14496-15), or other similar video file formats.

[0029] The ISOBMFF is used as the basis for many codec encapsulation formats, such as the AVC file format, as well as for many multimedia container formats, such as the MPEG-4 file format, the 3GPP file format (3GP), and the DVB file format. In addition to continuous media, such as audio and video, static media, such as images, as well as metadata can be stored in a file conforming to ISOBMFF. Files structured according to the ISOBMFF may be used for many purposes, including local media file playback, progressive downloading of a remote file, segments for Dynamic Adaptive Streaming over HTTP (DASH), containers for content to be streamed and its packetization instructions, and recording of received real-time media streams.

[0030] A box is the elementary syntax structure in the ISOBMFF, including a four-character coded box type, the byte count of the box, and the payload. An ISOBMFF file consists of a sequence of boxes, and boxes may contain other boxes. A Movie box (“moov”) contains the metadata for the continuous media streams present in the file, each one represented in the file as a track. The metadata for a track is enclosed in a Track box (“trak”), while the media content of a track is either enclosed in a Media Data box (“mdat”) or directly in a separate file. The media content for tracks consists of a sequence of samples, such as audio or video access units.

[0031] When rendering fisheye video data, it may be desirable for a video decoder to determine whether the fisheye video data is monoscopic or stereoscopic. For instance, the manner in which the video decoder displays and/or stitches the circular images included in a picture of fisheye video data is directly dependent on whether the fisheye video data is monoscopic or stereoscopic.

[0032] In some examples, a video decoder may determine whether fisheye video data is monoscopic or stereoscopic based on one or more syntax elements included in a file that implicitly indicate whether the fisheye video data described by the file is monoscopic or stereoscopic. For instance, a video encoder may include syntax elements in the file that describe extrinsic parameters (e.g., physical/locational attributes such as yaw angle, a pitch angle, a roll angle, and one or more spatial offsets) of each of the cameras used to capture the fisheye video data, and the video decoder may process the extrinsic parameters to calculate whether the video data described by the file is monoscopic or stereoscopic. As one example, if processing of the extrinsic parameters leads to an indication that the cameras are facing the same direction and have a spatial offset, the video decoder may determine that the video data is stereoscopic. As another example, if the processing of the extrinsic parameters indicates that the cameras are facing opposite directions, the video decoder may determine that the video data is monoscopic. However, in some examples, it may be desirable for the video decoder to be able to determine whether the fisheye video data is monoscopic or stereoscopic without having to perform such additional, resource intensive, calculations. Additionally, in some examples, the processing of the extrinsic parameters may not yield the correct (e.g., as intended) classification of video data as monoscopic or stereoscopic. As one example, the values of the extrinsic parameters may become corrupted during encoding/transit/decoding. As another example, the extrinsic parameters may not have been accurately provided at the encoder.

[0033] In accordance with one or more techniques of this disclosure, a video encoder may include a first syntax element in a file that explicitly indicates whether the fisheye video data described by the file is monoscopic or stereoscopic. The file may include the first syntax element in addition to the syntax elements that the extrinsic parameters of the fisheye video data. As such, the file may include both an explicit indication (i.e., the first syntax element) and an implicit indication (i.e., the syntax elements that the extrinsic parameters) of whether the fisheye video data described by the file is monoscopic or stereoscopic. In this way, a video decoder may accurately determine whether video data is monoscopic or stereoscopic (e.g., without having to perform additional calculations based on the extrinsic parameters).

[0034] The ISOBMFF specifies the following types of tracks: a media track, which contains an elementary media stream, a hint track, which either includes media transmission instructions or represents a received packet stream, and a timed metadata track, which comprises time-synchronized metadata.

[0035] Although originally designed for storage, the ISOBMFF has proven to be very valuable for streaming, e.g. for progressive download or DASH. For streaming purposes, the movie fragments defined in ISOBMFF can be used.

[0036] The metadata for each track includes a list of sample description entries, each providing the coding or encapsulation format used in the track and the initialization data needed for processing that format. Each sample is associated with one of the sample description entries of the track

[0037] The ISOBMFF enables specifying sample-specific metadata with various mechanisms. Specific boxes within the Sample Table box (“stbl”) have been standardized to respond to common needs. For example, a Sync Sample box ("stss") is used to list the random access samples of the track. The sample grouping mechanism enables mapping of samples according to a four-character grouping type into groups of samples sharing the same property specified as a sample group description entry in the file. Several grouping types have been specified in the ISOBMFF.

[0038] Virtual reality (VR) is the ability to be virtually present in a non-physical world created by the rendering of natural and/or synthetic image and sound correlated by the movements of the immersed user allowing to interact with that world. With the recent progress made in rendering devices, such as head mounted displays (HMD), and VR video (often also referred to as 360 degree video) creation, a significant quality of experience can be offered. VR applications including gaming, training, education, sports video, online shopping, adult entrainment, and so on.

[0039] A typical VR system may include one or more of the following components, which may perform one or more of the following steps:

1) A camera set, which typically consists of multiple individual cameras pointing to different directions and ideally collectively covering all viewpoints around the camera set.
2) Image stitching, where video pictures taken by the multiple individual cameras are synchronized in the time domain and stitched in the space domain, to be a spherical video, but mapped to a rectangular format, such as equi-rectangular (like a world map) or cube map.
3) The video in the mapped rectangular format is encoded/compressed using a video codec, e.g., H.265/HEVC or H.264/AVC.
4) The compressed video bitstream(s) may be stored and/or encapsulated in a media format and transmitted (possibly only the subset covering only the area being seen by a user) through a network to a receiver.
5) The receiver receives the video bitstream(s) or part thereof, possibly encapsulated in a format, and sends the decoded video signal or part thereof to a rendering device.
6) The rendering device can be e.g., an HMD, which can track head movement and even eye move moment and rendering the corresponding part of the video such that an immersive experience is delivered to the user.

[0040] The Omnidirectional Media Application Format (OMAF) is being developed by MPEG to define a media application format that enables omnidirectional media applications, focusing on VR applications with 360° video and associated audio. OMAF specifies projection and region-wise packing that can be used for conversion of a spherical or 360° video into a two-dimensional rectangular video, followed by how to store omnidirectional media and the associated metadata using the ISO base media file format (ISOBMFF) and how to encapsulate, signal, and stream omnidirectional media using dynamic adaptive streaming over HTTP (DASH), and finally which video and audio codecs as well as media coding configurations can be used for compression and playback of the omnidirectional media signal.

[0041] Projection and region-wise packing are the processes used at the content production side to generate 2D video pictures from the sphere signal for projected omnidirectional video. Projection usually goes together with stitching, which may generate the sphere signal from the multiple camera captured images for each video picture. Projection is a may be a fundamental processing step in VR video. Typical projection types include equirectangular and cubemap. The OMAF Draft International Standard (DIS) only supports the equirectangular projection type. Region-wise packing is an optional step after projection (from the viewport of the content production side). Region-wise packing enables manipulations (resize, reposition, rotation, and mirroring) of any rectangular region of the packed picture before encoding.

[0042] FIG. 2 is an example of a picture that includes multiple fisheye images. The OMAF DIS supports a fisheye VR/360 video format, wherein instead of applying a projection and optionally a region-wise packing to generate the 2D video before encoding, for each access unit the circular images from the capturing cameras are directly embedded in a 2D picture. For example, as shown in FIG. 2, first fisheye image 202 and second fisheye image 204 and embedded in 2D picture 200.

[0043] Such fisheye video may then be encoded and the bitstream may be encapsulated in an ISOBMFF file and may be further encapsulated as a DASH representation. In addition, the property of the fisheye video, including parameters indicating the characteristics of the fisheye video, may be signalled and used to correctly rendering the 360 video at the client side. One advantage of the fisheye VR/360 video approach is that it supports low-cost user generated VR content by mobile terminals.

[0044] In OMAF DIS, the use of the fisheye omnidirectional video scheme for the restricted video sample entry type 'resv' indicates that the decoded pictures are fisheye video pictures. The use of the fisheye omnidirectional video scheme is indicated by scheme_type equal to 'fodv' (fisheye omnidirectional video) within the

SchemeTypeBox. The format of fisheye video is indicated with the FisheyeOmnidirectionalVideoBox contained within the SchemeInformationBox, which is included in the RestrictedSchemeInfoBox that is included in the sample entry. In the current draft of OMAF DIS (Information technology — Coded representation of immersive media (MPEG-I) — Part 2: Omnidirectional media format, ISO/IEC FDIS 14496-15:2014(E), ISO/IEC JTC 1/SC 29/WG 11, w16824, 2014-01-13, hereinafter "current draft of OMAF DIS"), one and only one FisheyeOmnidirectionalVideoBox shall be present in the SchemeInformationBox when the scheme type is 'fodv'. When FisheyeOmnidirectionalVideoBox is present in the SchemeInformationBox, StereoVideoBox and RegionWisePackingBox shall not be present in the same SchemeInformationBox. The FisheyeOmnidirectionalVideoBox, as specified in clause 6 of the OMAF DIS, contains the FisheyeOmnidirectionalVideoInfo( ) syntax structure that contains the fisheye video property parameters.

[0045] The syntax of the FisheyeOmnidirectionalVideoInfo( ) syntax structure in the current draft of OMAF DIS are as follows.

```
aligned(8) class FisheyeOmnidirectionalVideoInfo( ) {
    bit(24) reserved = 0;
    unsigned int(8) num_circular_images;
    for(i=0; i< num_circular_images; i++) {
        unsigned int(32) image_center_x;
        unsigned int(32) image_center_y;
        unsigned int(32) full_radius;
        unsigned int(32) picture_radius;
        unsigned int(32) scene_radius;
        unsigned int(32) image_rotation;
        bit(30) reserved = 0;
        unsigned int(2) image_flip;
        unsigned int(32) image_scale_axis_angle;
        unsigned int(32) image_scale_x;
        unsigned int(32) image_scale_y;
        unsigned int(32) field_of_view;
        bit(16) reserved = 0;
        unsigned int (16) num_angle_for_displaying_fov;
        for(j=0; j< num_angle_for_displaying_fov; j++) {
```

```
        unsigned int(32) displayed_fov;
        unsigned int(32) overlapped_fov;
    }
    signed int(32) camera_center_yaw;
    signed int(32) camera_center_pitch;
    signed int(32) camera_center_roll;
    unsigned int(32) camera_center_offset_x;
    unsigned int(32) camera_center_offset_y;
    unsigned int(32) camera_center_offset_z;
    bit(16) reserved = 0;
    unsigned int(16) num_polynomial_coefficeients;
    for(j=0; j< num_polynomial_coefficients; j++) {
        unsigned int(32) polynomial_coefficient_K;
    }
    bit(16) reserved = 0;
    unsigned int (16) num_local_fov_region;
    for(j=0; j<num_local_fov_region; j++) {
        unsigned int(32) start_radius;
        unsigned int(32) end_radius;
        signed int(32) start_angle;
        signed int(32) end_angle;
        unsigned int(32) radius_delta;
        signed int(32) angle_delta;
        for(rad=start_radius; rad<= end_radius; rad+=radius_delta) {
            for(ang=start_angle; ang<= ang_radius; ang+=angle_delta) {
                unsigned int(32) local_fov_weight;
            }
        }
    }
    bit(16) reserved = 0;
    unsigned int(16) num_polynomial_coefficients_lsc;
    for(j=0; j< num_polynomial_coefficients_lsc; j++) {
        unsigned int (32) polynomial_coefficient_K_lsc_R;
        unsigned int (32) polynomial_coefficient_K_lsc_G;
```

```
            unsigned int (32) polynomial_coefficient_K_lsc_B;
        }
    }
    bit(24) reserved = 0;
    unsigned int(8) num_deadzones;
    for(i=0; i< num_deadzones; i++) {
        unsigned int(16) deadzone_left_horizontal_offset;
        unsigned int(16) deadzone_top_vertical_offset;
        unsigned int(16) deadzone_width;
        unsigned int(16) deadzone_height;
    }
}
```

[0046] The semantics of the FisheyeOmnidirectionalVideoInfo( ) syntax structure in the current draft of OMAF DIS is as follows.

[0047] num_circular_images specifies the number of circular images in the coded picture of each sample this box applies to. Typically, the value is equal to 2, but other non-zero values are also possible.

[0048] image_center_x is a fixed-point 16.16 value that specifies the horizontal coordinate, in luma samples, of the center of the circular image in the coded picture of each sample this box applies to.

[0049] image_center_y is a fixed-point 16.16 value that specifies the vertical coordinate, in luma samples, of the center of the circular image in the coded picture of each sample this box applies to.

[0050] full_radius is a fixed-point 16.16 value that specifies the radius, in luma samples, from the center of the circular image to the edge of the full round image.

[0051] picture_radius is a fixed-point 16.16 value that specifies the radius, in luma samples, from the center of the circular image to the closest edge of the image border. The circular fisheye image may be cropped by the camera picture. Therefore, this value indicates the radius of a circle wherein pixels are usable.

[0052] scene_radius is a fixed-point 16.16 value that specifies the radius, in luma samples, from the center of the circular image to the closest edge of the area in the image where it is guaranteed that there are no obstructions from the camera body itself and that within the enclosed area there is no lens distortion being too large for stitching.

[0053] image_rotation is a fixed-point 16.16 value that specifies the amount of rotation, in degrees, of the circular image. The image may be rotated by images +/− 90 degrees, or +/− 180 degrees, or any other value.

[0054] image_flip specifies whether and how the image has been flipped and thus a reverse flipping operation needs to be applied. The value 0 indicates that the image has not been flipped. The value 1 indicates that the image has been vertically flipped. The value 2 indicates that the image has been horizontally flipped. The value 3 indicates that the image has been both vertically and horizontally flipped.

[0055] image_scale_axis_angle, image_scale_x, and image_scale_y are three fixed-point 16.16 values that specify whether and how the image has been scaled along an axis. The axis is defined by a single angle as indicated by the value of image_scale_axis_angle, in degrees. An angle of 0 degrees means a horizontal vector is perfectly horizontal and a vertical vector is perfectly vertical. The values of image_scale_x and image_scale_y indicate the scaling ratios in the directions that are parallel and orthogonal, respectively, to the axis.

[0056] field_of_view is a fixed-point 16.16 value that specifies the field of view of the fisheye lens, in degrees. A typical value for a hemispherical fisheye lens is 180.0 degrees.

[0057] num_angle_for_displaying_fov specifies the number of angles. According to the value of num_angle_for_displaying_fov, multiple values of displayed_fov and overlapped_fov are defined with equal intervals, which start at 12 o'clock and go clockwise.

[0058] displayed_fov specifies the displayed field of view and the corresponding image area of each fisheye camera image. overlapped_fov specifies the region which includes overlapped regions, which are usually used for blending, in terms of the field of view between multiple circular images. The values of displayed_fov and overlapped_fov are smaller than or equal to the value of field_of_view.

[0059] NOTE: The value of field_of_view is determined by the physical property of each fisheye lens, while the values of displayed_fov and overlapped_fov are determined by the configuration of multiple fisheye lenses. For example, when the value of num_circular_images is equal to 2 and two lenses are symmetrically located, the value of displayed_fov and overlapped_fov can be set as 180 and 190 respectively, by default. However, the value can be changed depending on the configuration of the lens and the characteristics of the contents. For example, if the stitching quality with the displayed_fov values (left camera = 170 and right camera = 190) and the overlapped_fov values (left camera = 185 and right camera = 190) is better than the quality with the default values (180 and 190) or if the physical configuration of cameras is asymmetric, then the unequal displayed_fov and overlapped_fov values can be taken. In addition, when it comes to multiple (N>2) fisheye images, a single displayed_fov value cannot specify the exact area of each fisheye image. As shown in FIG. 6, displayed_fov (602) varies according to the direction. In order to manipulate multiple (N>2) fisheye images, num_angle_for_displaying_fov is introduced. For example, if this value is equal to 12, then the fisheye image is divided into 12 sectors where each sector angle is 30 degrees.

[0060] camera_center_yaw specifies the yaw angle, in units of 2^{-16} degrees, of the point that the center pixel of the circular image in the coded picture of each sample is projected to a spherical surface. This is the first of 3 angles that specify the camera extrinsic parameters relative to the global coordinate axes. camera_center_yaw shall be in the range of $-180 * 2^{16}$ to $180 * 2^{16} - 1$, inclusive.

[0061] camera_center_pitch specifies the pitch angle, in units of 2^{-16} degrees, of the point that the center pixel of the circular image in the coded picture of each sample is projected to a spherical surface. camera_center_pitch shall be in the range of $-90 * 2^{16}$ to $90 * 2^{16}$, inclusive.

[0062] camera_center_roll specifies the roll angle, in units of 2^{-16} degrees, of the point that the center pixel of the circular image in the coded picture of each sample is projected to a spherical surface. camera_center_roll shall be in the range of $-180 * 2^{16}$ to $180 * 2^{16} - 1$, inclusive.

[0063] camera_center_offset_x, camera_center_offset_y and camera_center_offset_z are fixed-point 8.24 values that indicate the XYZ offset values from the origin of unit sphere where pixels in the circular image in the coded picture are projected onto. camera_center_offset_x, camera_center_offset_y and camera_center_offset_z shall be in the range of −1.0 to 1.0, inclusive.

[0064] num_polynomial_coefficients is an integer that specifies the number of polynomial coefficients present. The list of polynomial coefficients polynomial_coefficient_K are fixed-point 8.24 values that represent the coefficents in the polynomial that specify the transformation from fisheye space to undistored planar image.

[0065] num_local_fov_region specifies the number of local fitting regions having different field of view.

[0066] start_radius, end_radius, start_angle, and end_angle specify the region for local fitting/warping to change the actual field of view for displaying locally. start_radius and end_radius are fixed-point 16.16 values that specify the minimum and maximum radius values. start_angle and end_angle specify the minimum and maximum angle values that start at 12 o'clock and increase clockwise, in units of 2^{-16} degrees. start_angle, and end_angle shall be in the range of $-180 * 2^{16}$ to $180 * 2^{16} - 1$, inclusive.

[0067] radius_delta is a fixed-point 16.16 value that specifies the delta radius value for representing a different field of view for each radius.

[0068] angle_delta specifies the delta angle value, in units of 2^{-16} degrees, for representing a different field of view for each angle.

[0069] local_fov_weight is a 8.24 fixed point format which specifies the weighting value for the field of view of the position specified by start_radius, end_radius, start_angle, end_angle, the angle index i and the radius index j. The positive value of local_fov_weight specifies the expansion of field of view, while the negative value specifies the contraction of field of view.

[0070] num_polynomial_coefficeients_lsc shall be the order of the polynomial approximation of the lens shading curve.

[0071] polynomial_coefficient_K_lsc_R, polynomial_coefficient_K_lsc_G, and polynomial_coefficient_K_lsc_B are 8.24 fixed point formats that specify the LSC parameters to compensate the shading artifact which reduces colour along the radial direction. The compensating weight (w) to be multiplied to the original colour is approximated as a curve function of the radius from the image center using a polynomial expression. It is formulated as $w = \sum_{i=1}^{N} p_{i-1} \cdot r^{i-1}$, where p indicates the coefficient value equal to polynomial_coefficient_K_lsc_R, polynomial_coefficient_K_lsc_G or polynomial_coefficient_K_lsc_B, and r indicates the radius value after normalization by full_radius. N is equal to the value of num_polynomial_coefficeients_lsc.

[0072] num_deadzones is an integer that specifies the number of dead zones in the coded picture of each sample this box applies to.

[0073] deadzone_left_horizontal_offset, deadzone_top_vertical_offset, deadzone_width, and deadzone_height are integer values that specify the position and size of the deadzone rectangular area in which the pixels are not usable.

deadzone_left_horizontal_offset and deadzone_top_vertical_offset specify the horizontal and vertical coordinates, respectively, in luma samples, of the upper left corner of the deadzone in the coded picture. deadzone_width and deadzone_height specify the width and height, respectively, in luma samples, of the deadzone. To save bits for representing the video, all pixels within a deadzone should be set to the same pixel value, e.g., all black.

[0074] FIGS. 3–8 are conceptual diagrams that illustrate various aspects of the above syntax and semantics. FIG. 3 illustrates the image_center_x and image_center_y syntax as center 302, full_radius syntax as full radius 304, picture_radius as frame radius 306 of frame 300, scene_radius as scene radius 308 (e.g., the area without obstructions from camera body 510). FIG. 4 illustrates the displayed_fov (i.e., displayed field of view (FOV)) for two fisheye images. FOV 402 represents a 170-degree field of view and FOV 404 represents a 190-degree field of view. FIG. 5 illustrates the displayed_fov (i.e., displayed field of view (FOV)) and overlapped_fov for multiple (e.g., N>2) fisheye images. FOV 502 represents a first field of view and FOV 504 represents a second field of view. FIG. 6 is an illustration of camera_center_offset_x (o_x), camera_center_offset_y (o_y), and camera_center_offset_z (o_z) syntax. FIG. 7 is a conceptual diagram illustrating parameters regarding local field of view. FIG. 8 is a conceptual diagram illustrating an example of a local field of view.

[0075] The signalling of fisheye video in the current draft of OMAF DIS may present one or more disadvantages.

[0076] As one example disadvantage of the signalling of fisheye video in the current draft of OMAF DIS, when there are two circular images in each fisheye video picture (e.g., where num_circular_images in the FisheyeOmnidirectionalVideoInfo( ) syntax structure is equal to 2), depending on the values of the camera extrinsic parameters (e.g., camera_center_yaw, camera_center_pitch, camera_center_roll, camera_center_offset_x, camera_center_offset_y, and camera_center_offset_z), the fisheye video can be either monoscopic or stereoscopic. In particular, when the two cameras capturing the two circular images are on the same side, the fisheye video is stereoscopic covering about half (i.e., 180 degrees horizontally) of the sphere, and when the two cameras are at opposite sides, the fisheye video is monoscopic but covering about the entire (i.e., 360 degrees horizontally of the) sphere. For example, when the two sets of camera extrinsic parameter values are as follows, the fisheye video is monoscopic:

1st set:

camera_center_yaw = 0 degrees (+/- 5 degrees)

camera center_pitch = 0 degrees (+/- 5 degrees)

camera center roll = 0 degrees (+/- 5 degrees)

camera_center_offset_x = 0 mm (+/- 3 mm)

camera_center_offset_y = 0 mm (+/- 3 mm)

camera_center_offset_z = 0 mm (+/- 3 mm)

2nd set:

camera_center_yaw = 180 degrees (+/- 5 degrees)

camera center_pitch = 0 degrees (+/- 5 degrees)

camera center roll = 0 degrees (+/- 5 degrees)

camera_center_offset_x = 0 mm (+/- 3 mm)

camera_center_offset_y = 0 mm (+/- 3 mm)

camera_center_offset_z = 0 mm (+/- 3 mm)

[0077] The above parameter values may correspond to camera extrinsic parameter values of computing device 10B of FIG. 1B. As another example, when the two sets of camera extrinsic parameter values are as follows, the fisheye video is stereoscopic:

1st set:

camera_center_yaw = 0 degrees (+/- 5 degrees)

camera center_pitch = 0 degrees (+/- 5 degrees)

camera center roll = 0 degrees (+/- 5 degrees)

camera_center_offset_x = 0 mm (+/- 3 mm)

camera_center_offset_y = 0 mm (+/- 3 mm)

camera_center_offset_z = 0 mm (+/- 3 mm)

2nd set:

camera_center_yaw = 0 degrees (+/- 5 degrees)

camera center_pitch = 0 degrees (+/- 5 degrees)

camera center roll = 0 degrees (+/- 5 degrees)

camera_center_offset_x = 64 mm (+/- 3 mm)

camera_center_offset_y = 0 mm (+/- 3 mm)

camera_center_offset_z = 0 mm (+/- 3 mm)

[0078] The above parameter values may correspond to camera extrinsic parameter values of computing device 10A of FIG. 1A. Note that the stereo offset X distance of 64 mm is equivalent to 2.5 inches, which is the average distance between human eyes.

[0079] In other words, the information on whether the fisheye video is monoscopic or stereoscopic is hidden (i.e., implicitly but not explicitly coded). However, for high-level system purposes like content selection, it may be desirable for this information to be easily accessible in both file format level and DASH level (e.g., such that the entity that performs the content selection function does not need to parse much information in the FisheyeOmnidirectionalVideoInfo( ) syntax structure to determine whether the corresponding video data is monoscopic or stereoscopic).

[0080] As another example disadvantage of the signalling of fisheye video in the current draft of OMAF DIS, when there are two circular images in each fisheye video picture (e.g., where num_circular_images in the FisheyeOmnidirectionalVideoInfo( ) syntax structure is equal to 2), and the fisheye video is stereoscopic, a client device may determine which of the two circular images is the left-eye view and which is the right-eye view from the camera extrinsic parameters. However, it may not be desirable for the client device to have to determine which image is the left-eye view and which is the right-eye view from the camera extrinsic parameters.

[0081] As another example disadvantage of the signalling of fisheye video in the current draft of OMAF DIS, when there are four fisheye cameras, two on each side, for capturing the fisheye video, the fisheye video would be stereoscopic covering the entire sphere. In this case, there are four circular images in each fisheye video picture (e.g., num_circular_images in the FisheyeOmnidirectionalVideoInfo( ) syntax structure is equal to 4). In such cases (when there are more than two circular images in each fisheye video picture), a client device may determine the pairing of certain two circular images belonging to the same view from the camera extrinsic parameters. However, it may not be desirable for the client device to have to determine the pairing of certain two circular images belonging to the same view from the camera extrinsic parameters.

[0082] As another example disadvantage of the signalling of fisheye video in the current draft of OMAF DIS, for projected omnidirectional video, the coverage information (e.g., which area on the sphere is covered by the video), is either explicitly signalled using the CoverageInformationBox or, when not present, inferred by the client device to be the full sphere. Though the client device may determine coverage from the camera extrinsic parameters, again, explicit signalling of this information to be easily accessible may be desirable.

[0083] As another example disadvantage of the signalling of fisheye video in the current draft of OMAF DIS, the use of region-wise packing is allowed for projected omnidirectional video but disallowed for fisheye video. However, some benefits of region-wise packing applicable to projected omnidirectional video may also applicable to fisheye video.

[0084] As another example disadvantage of the signalling of fisheye video in the current draft of OMAF DIS, carriage of projected omnidirectional video in sub-picture tracks is specified in the OMAF DIS by using the composition track grouping. However, carriage of fisheye omnidirectional video in in sub-picture tracks is not supported.

[0085] This disclosure presents solutions to the above problems. Some of these techniques may be applied independently and some of them may be applied in combination.

[0086] In accordance with one or more techniques of this disclosure, a file including fisheye video data may include an explicit indication of whether the fisheye video data is monoscopic or stereoscopic. In other words, an indication of whether fisheye video is monoscopic or stereoscopic may be explicitly signaled. In this way, a client device may avoid having to infer whether fisheye video data is monoscopic or stereoscopic from other parameters.

[0087] As one example, one or more of the initial 24 bits in the FisheyeOmnidirectionalVideoInfo( ) syntax structure may be used to form a field (e.g., a one-bit flag) to signal the indication of monoscopic or stereoscopic. The field being equal to a first particular value (e.g., 0) may indicate that the fisheye video is monoscopic, and the field equal to a second particular value (e.g., 1) may indicate that the fisheye video is stereoscopic. For example, when generating a file including fisheye video data, a content preparation device (e.g., content preparation device 20 of FIG. 9, and in a particular example, encapsulation unit 30 of content preparation device 20) may encode a box (e.g., a FisheyeOmnidirectionalVideoBox) within the file, the box including a syntax structure (e.g., a FisheyeOmnidirectionalVideoInfo( )) that contains parameters for the fisheye video data, and the syntax structure including an explicit indication of whether the fisheye video data is monoscopic or stereoscopic. A client device (e.g., client device 40 of FIG. 9, and in a particular example, retrieval unit 52 of client device 40) may similarly process the file to obtain the explicit indication of monoscopic or stereoscopic.

[0088] As another example, a new box containing a field, e.g., a one-bit flag, some reserved bits and possibly some other information may be added into the

FisheyeOmnidirectionalVideoBox to signal the indication of monoscopic or stereoscopic. The field being equal to a first particular value (e.g., 0) may indicate that the fisheye video is monoscopic, and the field equal to a second particular value (e.g., 1) may indicate that the fisheye video is stereoscopic. For example, when generating a file including fisheye video data, a content preparation device (e.g., content preparation device 20 of FIG. 9) may encode a first box (e.g., a FisheyeOmnidirectionalVideoBox) within the file, the first box including a second box that includes a field that explicitly indicates whether the fisheye video data is monoscopic or stereoscopic. A client device (e.g., client device 40 of FIG. 9) may similarly process the file to obtain the explicit indication of monoscopic or stereoscopic.

[0089] As another example, a field (e.g., a one-bit flag) may be directly added into the FisheyeOmnidirectionalVideoBox to signal this indication. For example, when generating a file including fisheye video data, a content preparation device (e.g., content preparation device 20 of FIG. 9) may encode a box (e.g., a FisheyeOmnidirectionalVideoBox) within the file, the box including a field that explicitly indicates whether the fisheye video data is monoscopic or stereoscopic. A client device (e.g., client device 40 of FIG. 9) may similarly process the file to obtain the explicit indication of monoscopic or stereoscopic.

[0090] In accordance with one or more techniques of this disclosure, a file including fisheye video data as a plurality of circular images may include, for each respective circular image of the plurality of circular images, an explicit indication of a respective view identification (view ID). In other words, for each circular image of fisheye video data, a field may be added that indicates the view ID of each circular image. When there are only two view ID values 0 and 1, then the view with view ID equal to 0 may be the left view, and the view with view ID equal to 1 may be the right view. For instance, where the plurality of circular images includes only two circular images, a circular image of the plurality of circular images having a first pre-determined video identification is a left-view and a circular image of the plurality of circular images having a second pre-determined video identification is a right-view. In this way, a client device may avoid having to infer which circular image is the left-view and which is the right-view from other parameters.

[0091] The view ID signalled can also be used to indicate the pairing relationship of any two circular images belonging to the same view (e.g., as they both may have the same value of the signalled view ID).

[0092] To enable the view IDs to be easily accessed without the need of parsing all the fisheye video parameters for the circular images, a loop may be added after the num_circular_images field and before the existing loop. For instance, processing the file may include parsing, in a first loop, the explicit indications of the view identifications; and parsing, in a second loop that is after the first loop, other parameters of the circular images. For instance, the view IDs and the other parameters may be parsed as follows:

```
aligned(8) class FisheyeOmnidirectionalVideoInfo( ) {
    bit(24) reserved = 0;
    unsigned int(8) num_circular_images;
    for(i=0; i< num_circular_images; i++) {
        unsigned int(8) view_id;
        bit(24) reserved = 0;
    }
    for(i=0; i< num_circular_images; i++) {
        unsigned int(32) image_center_x;
        unsigned int(32) image_center_y;
    ...
    }
}
```

[0093] view_id indicates the view identifier of the view the circular image belongs to. When there are only two values of view_id 0 and 1 for all the circular images, the circular images with view_id equal to 0 may belong to the left view, and the circular images with view_id equal to 1 may belong to the right view. In some examples, the syntax element(s) that indicates the view identifier (i.e., the syntax element that indicates which circular image is the left view and which is the right view) may be the same as the syntax element that explicitly indicates whether the fisheye video data is stereoscopic or monoscopic.

[0094] In accordance with one or more techniques of this disclosure, a file including fisheye video data may include a first box that include a syntax structure that contains parameters for the fisheye video data, and may optionally include a second box that indicates coverage information for the fisheye video data. For instance, signalling of the coverage information for the fisheye omnidirectional video may be added to the

FisheyeOmnidirectionalVideoBox. For example, the CoverageInformationBox as defined in the OMAF DIS may be allowed to be optionally contained in the FisheyeOmnidirectionalVideoBox, and when it is not present in the FisheyeOmnidirectionalVideoBox, full sphere coverage may be inferred for the fisheye video. When CoverageInformationBox is present in the FisheyeOmnidirectionalVideoBox, the spherical region represented by the fisheye video pictures may be a region specified by two yaw circles and two pitch circles. In this way, a client device may avoid having to infer the coverage information from other parameters.

[0095] In accordance with one or more techniques of this disclosure, a file including fisheye video data may include a first box that includes scheme information, the first box may include a second box that includes a syntax structure that contains parameters for the fisheye video data, and the first box may optionally include a third box that indicates whether pictures of the fisheye video data are packed region-wise. For instance, the RegionWisePackingBox may be optionally included in the SchemeInformationBox for fisheye omnidirectional video (e.g., when FisheyeOmnidirectionalVideoBox is present in the SchemeInformationBox, RegionWisePackingBox can be present in the same SchemeInformationBox). In this case, the presence of the RegionWisePackingBox indicates that fisheye video pictures are packed region-wise and require unpacking prior to rendering. In this way, one or more benefits of region-wise packing may be gained for fisheye video.

[0096] In accordance with one or more techniques of this disclosure, sub-picture composition may be used grouping for fisheye omnidirectional video, and a specification on the application of the sub-picture composition grouping may be added to fisheye omnidirectional video. That specification may apply when any of the tracks mapped to the sub-picture composition track group has a sample entry type equal to 'resv' and scheme_type equal to 'fodv' in the SchemeTypeBox included in the sample entry. In this case, each composed picture is a packed picture that has the fisheye format indicated by any FisheyeOmnidirectionalVideoBox and the region-wise packing format indicated by any RegionWisePackingBox included in the sample entries of the tracks mapped to the sub-picture composition track group. In addition, the following may apply:

[0097] 1) Each track mapped to this grouping shall have a sample entry type equal to 'resv'. The scheme_type shall be equal to 'fodv' in the SchemeTypeBox included in the sample entry.

[0098] 2) The content of all the instances of the FisheyeOmnidirectionalVideoBox included in the sample entries of the tracks mapped to the same sub-picture composition track group shall be identical.

[0099] 3) The content of all the instances of the RegionWisePackingBox included in the sample entries of the tracks mapped to the same sub-picture composition track group shall be identical.

[0100] In HTTP streaming, frequently used operations include HEAD, GET, and partial GET. The HEAD operation retrieves a header of a file associated with a given uniform resource locator (URL) or uniform resource name (URN), without retrieving a payload associated with the URL or URN. The GET operation retrieves a whole file associated with a given URL or URN. The partial GET operation receives a byte range as an input parameter and retrieves a continuous number of bytes of a file, where the number of bytes correspond to the received byte range. Thus, movie fragments may be provided for HTTP streaming, because a partial GET operation can get one or more individual movie fragments. In a movie fragment, there can be several track fragments of different tracks. In HTTP streaming, a media presentation may be a structured collection of data that is accessible to the client. The client may request and download media data information to present a streaming service to a user.

[0101] In the example of streaming 3GPP data using HTTP streaming, there may be multiple representations for video and/or audio data of multimedia content. As explained below, different representations may correspond to different coding characteristics (e.g., different profiles or levels of a video coding standard), different coding standards or extensions of coding standards (such as multiview and/or scalable extensions), or different bitrates. The manifest of such representations may be defined in a Media Presentation Description (MPD) data structure. A media presentation may correspond to a structured collection of data that is accessible to an HTTP streaming client device. The HTTP streaming client device may request and download media data information to present a streaming service to a user of the client device. A media presentation may be described in the MPD data structure, which may include updates of the MPD.

[0102] A media presentation may contain a sequence of one or more Periods. Each period may extend until the start of the next Period, or until the end of the media presentation, in the case of the last period. Each period may contain one or more representations for the same media content. A representation may be one of a number of alternative encoded versions of audio, video, timed text, or other such data. The representations may differ by encoding types, e.g., by bitrate, resolution, and/or codec for video data and bitrate, language, and/or codec for audio data. The term representation may be used to refer to a section of encoded audio or video data corresponding to a particular period of the multimedia content and encoded in a particular way.

[0103] Representations of a particular period may be assigned to a group indicated by an attribute in the MPD indicative of an adaptation set to which the representations belong. Representations in the same adaptation set are generally considered alternatives to each other, in that a client device can dynamically and seamlessly switch between these representations, e.g., to perform bandwidth adaptation. For example, each representation of video data for a particular period may be assigned to the same adaptation set, such that any of the representations may be selected for decoding to present media data, such as video data or audio data, of the multimedia content for the corresponding period. The media content within one period may be represented by either one representation from group 0, if present, or the combination of at most one representation from each non-zero group, in some examples. Timing data for each representation of a period may be expressed relative to the start time of the period.

[0104] A representation may include one or more segments. Each representation may include an initialization segment, or each segment of a representation may be self-initializing. When present, the initialization segment may contain initialization information for accessing the representation. In general, the initialization segment does not contain media data. A segment may be uniquely referenced by an identifier, such as a uniform resource locator (URL), uniform resource name (URN), or uniform resource identifier (URI). The MPD may provide the identifiers for each segment. In some examples, the MPD may also provide byte ranges in the form of a *range* attribute, which may correspond to the data for a segment within a file accessible by the URL, URN, or URI.

[0105] Different representations may be selected for substantially simultaneous retrieval for different types of media data. For example, a client device may select an audio representation, a video representation, and a timed text representation from which to retrieve segments. In some examples, the client device may select particular adaptation sets for performing bandwidth adaptation. That is, the client device may select an adaptation set including video representations, an adaptation set including audio representations, and/or an adaptation set including timed text. Alternatively, the client device may select adaptation sets for certain types of media (e.g., video), and directly select representations for other types of media (e.g., audio and/or timed text).

[0106] FIG. 9 is a block diagram illustrating an example system 10 that implements techniques for streaming media data over a network. In this example, system 10 includes content preparation device 20, server device 60, and client device 40. Client device 40 and server device 60 are communicatively coupled by network 74, which may comprise the Internet. In some examples, content preparation device 20 and server device 60 may also be coupled by network 74 or another network, or may be directly communicatively coupled. In some examples, content preparation device 20 and server device 60 may comprise the same device.

[0107] Content preparation device 20, in the example of FIG. 9, comprises audio source 22 and video source 24. Audio source 22 may comprise, for example, a microphone that produces electrical signals representative of captured audio data to be encoded by audio encoder 26. Alternatively, audio source 22 may comprise a storage medium storing previously recorded audio data, an audio data generator such as a computerized synthesizer, or any other source of audio data. Video source 24 may comprise a video camera that produces video data to be encoded by video encoder 28, a storage medium encoded with previously recorded video data, a video data generation unit such as a computer graphics source, or any other source of video data. Content preparation device 20 is not necessarily communicatively coupled to server device 60 in all examples, but may store multimedia content to a separate medium that is read by server device 60.

[0108] Raw audio and video data may comprise analog or digital data. Analog data may be digitized before being encoded by audio encoder 26 and/or video encoder 28. Audio source 22 may obtain audio data from a speaking participant while the speaking participant is speaking, and video source 24 may simultaneously obtain video data of the speaking participant. In other examples, audio source 22 may comprise a computer-readable storage medium comprising stored audio data, and video source 24 may comprise a computer-readable storage medium comprising stored video data. In this manner, the techniques described in this disclosure may be applied to live, streaming, real-time audio and video data or to archived, pre-recorded audio and video data.

[0109] Audio frames that correspond to video frames are generally audio frames containing audio data that was captured (or generated) by audio source 22 contemporaneously with video data captured (or generated) by video source 24 that is contained within the video frames. For example, while a speaking participant generally produces audio data by speaking, audio source 22 captures the audio data, and video source 24 captures video data of the speaking participant at the same time, that is, while audio source 22 is capturing the audio data. Hence, an audio frame may temporally correspond to one or more particular video frames. Accordingly, an audio frame corresponding to a video frame generally corresponds to a situation in which audio data and video data were captured at the same time and for which an audio frame and a video frame comprise, respectively, the audio data and the video data that was captured at the same time.

[0110] In some examples, audio encoder 26 may encode a timestamp in each encoded audio frame that represents a time at which the audio data for the encoded audio frame was recorded, and similarly, video encoder 28 may encode a timestamp in each encoded video frame that represents a time at which the video data for encoded video frame was recorded. In such examples, an audio frame corresponding to a video frame may comprise an audio frame comprising a timestamp and a video frame comprising the same timestamp. Content preparation device 20 may include an internal clock from which audio encoder 26 and/or video encoder 28 may generate the timestamps, or that audio source 22 and video source 24 may use to associate audio and video data, respectively, with a timestamp.

[0111] In some examples, audio source 22 may send data to audio encoder 26 corresponding to a time at which audio data was recorded, and video source 24 may send data to video encoder 28 corresponding to a time at which video data was recorded. In some examples, audio encoder 26 may encode a sequence identifier in encoded audio data to indicate a relative temporal ordering of encoded audio data but without necessarily indicating an absolute time at which the audio data was recorded, and similarly, video encoder 28 may also use sequence identifiers to indicate a relative temporal ordering of encoded video data. Similarly, in some examples, a sequence identifier may be mapped or otherwise correlated with a timestamp.

[0112] Audio encoder 26 generally produces a stream of encoded audio data, while video encoder 28 produces a stream of encoded video data. Each individual stream of data (whether audio or video) may be referred to as an elementary stream. An elementary stream is a single, digitally coded (possibly compressed) component of a representation. For example, the coded video or audio part of the representation can be an elementary stream. An elementary stream may be converted into a packetized elementary stream (PES) before being encapsulated within a video file. Within the same representation, a stream ID may be used to distinguish the PES-packets belonging to one elementary stream from the other. The basic unit of data of an elementary stream is a packetized elementary stream (PES) packet. Thus, coded video data generally corresponds to elementary video streams. Similarly, audio data corresponds to one or more respective elementary streams.

[0113] Many video coding standards, such as ITU-T H.264/AVC and the High Efficiency Video Coding (HEVC) standard, define the syntax, semantics, and decoding process for error-free bitstreams, any of which conform to a certain profile or level. Video coding standards typically do not specify the encoder, but the encoder is tasked with guaranteeing that the generated bitstreams are standard-compliant for a decoder. In the context of video coding standards, a "profile" corresponds to a subset of algorithms, features, or tools and constraints that apply to them. As defined by the H.264 standard, for example, a "profile" is a subset of the entire bitstream syntax that is specified by the H.264 standard. A "level" corresponds to the limitations of the decoder resource consumption, such as, for example, decoder memory and computation, which are related to the resolution of the pictures, bit rate, and block processing rate. A profile may be signaled with a profile_idc (profile indicator) value, while a level may be signaled with a level_idc (level indicator) value.

[0114] The H.264 standard, for example, recognizes that, within the bounds imposed by the syntax of a given profile, it is still possible to require a large variation in the performance of encoders and decoders depending upon the values taken by syntax elements in the bitstream such as the specified size of the decoded pictures. The H.264 standard further recognizes that, in many applications, it is neither practical nor economical to implement a decoder capable of dealing with all hypothetical uses of the syntax within a particular profile. Accordingly, the H.264 standard defines a "level" as a specified set of constraints imposed on values of the syntax elements in the bitstream. These constraints may be simple limits on values. Alternatively, these constraints may take the form of constraints on arithmetic combinations of values (e.g., picture width multiplied by picture height multiplied by number of pictures decoded per second). The H.264 standard further provides that individual implementations may support a different level for each supported profile.

[0115] A decoder conforming to a profile ordinarily supports all the features defined in the profile. For example, as a coding feature, B-picture coding is not supported in the baseline profile of H.264/AVC but is supported in other profiles of H.264/AVC. A decoder conforming to a level should be capable of decoding any bitstream that does not require resources beyond the limitations defined in the level. Definitions of profiles and levels may be helpful for interpretability. For example, during video transmission, a pair of profile and level definitions may be negotiated and agreed for a whole transmission session. More specifically, in H.264/AVC, a level may define limitations on the number of macroblocks that need to be processed, decoded picture buffer (DPB) size, coded picture buffer (CPB) size, vertical motion vector range, maximum number of motion vectors per two consecutive MBs, and whether a B-block can have sub-macroblock partitions less than 8x8 pixels. In this manner, a decoder may determine whether the decoder is capable of properly decoding the bitstream.

[0116] In the example of FIG. 9, encapsulation unit 30 of content preparation device 20 receives elementary streams comprising coded video data from video encoder 28 and elementary streams comprising coded audio data from audio encoder 26. In some examples, video encoder 28 and audio encoder 26 may each include packetizers for forming PES packets from encoded data. In other examples, video encoder 28 and audio encoder 26 may each interface with respective packetizers for forming PES packets from encoded data. In still other examples, encapsulation unit 30 may include packetizers for forming PES packets from encoded audio and video data.

[0117] Video encoder 28 may encode video data of multimedia content in a variety of ways, to produce different representations of the multimedia content at various bitrates and with various characteristics, such as pixel resolutions, frame rates, conformance to various coding standards, conformance to various profiles and/or levels of profiles for various coding standards, representations having one or multiple views (e.g., for two-dimensional or three-dimensional playback), or other such characteristics. A representation, as used in this disclosure, may comprise one of audio data, video data, text data (e.g., for closed captions), or other such data. The representation may include an elementary stream, such as an audio elementary stream or a video elementary stream.

Each PES packet may include a stream_id that identifies the elementary stream to which the PES packet belongs. Encapsulation unit 30 is responsible for assembling elementary streams into video files (e.g., segments) of various representations.

[0118] Encapsulation unit 30 receives PES packets for elementary streams of a representation from audio encoder 26 and video encoder 28 and forms corresponding network abstraction layer (NAL) units from the PES packets. Coded video segments may be organized into NAL units, which provide a “network-friendly” video representation addressing applications such as video telephony, storage, broadcast, or streaming. NAL units can be categorized to Video Coding Layer (VCL) NAL units and non-VCL NAL units. VCL units may contain the core compression engine and may include block, macroblock, and/or slice level data. Other NAL units may be non-VCL NAL units. In some examples, a coded picture in one time instance, normally presented as a primary coded picture, may be contained in an access unit, which may include one or more NAL units.

[0119] Non-VCL NAL units may include parameter set NAL units and SEI NAL units, among others. Parameter sets may contain sequence-level header information (in sequence parameter sets (SPS)) and the infrequently changing picture-level header information (in picture parameter sets (PPS)). With parameter sets (e.g., PPS and SPS), infrequently changing information need not to be repeated for each sequence or picture, hence coding efficiency may be improved. Furthermore, the use of parameter sets may enable out-of-band transmission of the important header information, avoiding the need for redundant transmissions for error resilience. In out-of-band transmission examples, parameter set NAL units may be transmitted on a different channel than other NAL units, such as SEI NAL units.

[0120] Supplemental Enhancement Information (SEI) may contain information that is not necessary for decoding the coded pictures samples from VCL NAL units, but may assist in processes related to decoding, display, error resilience, and other purposes. SEI messages may be contained in non-VCL NAL units. SEI messages are the normative part of some standard specifications, and thus are not always mandatory for standard compliant decoder implementation. SEI messages may be sequence level SEI messages or picture level SEI messages. Some sequence level information may be contained in SEI messages, such as scalability information SEI messages in the example of SVC and view scalability information SEI messages in MVC. These example SEI messages may convey information on, e.g., extraction of operation points and characteristics of the operation points. In addition, encapsulation unit 30 may form a manifest file, such as a media presentation descriptor (MPD) that describes characteristics of the representations. Encapsulation unit 30 may format the MPD according to extensible markup language (XML).

[0121] Encapsulation unit 30 may provide data for one or more representations of multimedia content, along with the manifest file (e.g., the MPD) to output interface 32. Output interface 32 may comprise a network interface or an interface for writing to a storage medium, such as a universal serial bus (USB) interface, a CD or DVD writer or burner, an interface to magnetic or flash storage media, or other interfaces for storing or transmitting media data. Encapsulation unit 30 may provide data of each of the representations of multimedia content to output interface 32, which may send the data to server device 60 via network transmission or storage media. In the example of FIG. 9, server device 60 includes storage medium 62 that stores various multimedia contents 64, each including a respective manifest file 66 and one or more representations 68A–68N (representations 68). In some examples, output interface 32 may also send data directly to network 74.

[0122] In some examples, representations 68 may be separated into adaptation sets. That is, various subsets of representations 68 may include respective common sets of characteristics, such as codec, profile and level, resolution, number of views, file format for segments, text type information that may identify a language or other characteristics of text to be displayed with the representation and/or audio data to be decoded and presented, e.g., by speakers, camera angle information that may describe a camera angle or real-world camera perspective of a scene for representations in the adaptation set, rating information that describes content suitability for particular audiences, or the like.

[0123] Manifest file 66 may include data indicative of the subsets of representations 68 corresponding to particular adaptation sets, as well as common characteristics for the adaptation sets. Manifest file 66 may also include data representative of individual characteristics, such as bitrates, for individual representations of adaptation sets. In this manner, an adaptation set may provide for simplified network bandwidth adaptation. Representations in an adaptation set may be indicated using child elements of an adaptation set element of manifest file 66. In some examples, manifest file 66 may include some or all of the data of FisheyeOmnidirectionalVideoInfo( ) discussed herein, or similar data. Additionally or alternatively, segments of representations 68 may include some or all of the data of FisheyeOmnidirectionalVideoInfo( ) discussed herein, or similar data.

[0124] Server device 60 includes request processing unit 70 and network interface 72. In some examples, server device 60 may include a plurality of network interfaces. Furthermore, any or all of the features of server device 60 may be implemented on other devices of a content delivery network, such as routers, bridges, proxy devices, switches, or other devices. In some examples, intermediate devices of a content delivery network may cache data of multimedia content 64, and include components that conform substantially to those of server device 60. In general, network interface 72 is configured to send and receive data via network 74.

[0125] Request processing unit 70 is configured to receive network requests from client devices, such as client device 40, for data of storage medium 62. For example, request processing unit 70 may implement hypertext transfer protocol (HTTP) version 1.1, as described in RFC 2616, “Hypertext Transfer Protocol – HTTP/1.1,” by R. Fielding et al, Network Working Group, IETF, June 1999. That is, request processing unit 70 may be configured to receive HTTP GET or partial GET requests and provide data of multimedia content 64 in response to the requests. The requests may specify a segment of one of representations 68, e.g., using a URL of the segment. In some examples, the requests may also specify one or more byte ranges of the segment, thus comprising partial GET requests. Request processing unit 70 may further be configured to service HTTP HEAD requests to provide header data of a segment of one of representations 68. In any case, request processing unit 70 may be configured to process the requests to provide requested data to a requesting device, such as client device 40.

[0126] Additionally or alternatively, request processing unit 70 may be configured to deliver media data via a broadcast or multicast protocol, such as eMBMS. Content preparation device 20 may create DASH segments and/or sub-segments in substantially the same way as described, but server device 60 may deliver these segments or sub-segments using eMBMS or another broadcast or multicast network transport protocol. For example, request processing unit 70 may be configured to receive a multicast group join request from client device 40. That is, server device 60 may advertise an Internet protocol (IP) address associated with a multicast group to client devices, including client device 40, associated with particular media content (e.g., a broadcast of a live event). Client device 40, in turn, may submit a request to join the multicast group. This request may be propagated throughout network 74, e.g., routers making up network 74, such that the routers are caused to direct traffic destined for the IP address associated with the multicast group to subscribing client devices, such as client device 40.

[0127] As illustrated in the example of FIG. 9, multimedia content 64 includes manifest file 66, which may correspond to a media presentation description (MPD). Manifest file 66 may contain descriptions of different alternative representations 68 (e.g., video services with different qualities) and the description may include, e.g., codec information, a profile value, a level value, a bitrate, and other descriptive characteristics of representations 68. Client device 40 may retrieve the MPD of a media presentation to determine how to access segments of representations 68.

[0128] In particular, retrieval unit 52 may retrieve configuration data (not shown) of client device 40 to determine decoding capabilities of video decoder 48 and rendering capabilities of video output 44. The configuration data may also include any or all of a language preference selected by a user of client device 40, one or more camera perspectives corresponding to depth preferences set by the user of client device 40, and/or a rating preference selected by the user of client device 40. Retrieval unit 52 may comprise, for example, a web browser or a media client configured to submit HTTP GET and partial GET requests. Retrieval unit 52 may correspond to software instructions executed by one or more processors or processing units (not shown) of client device 40. In some examples, all or portions of the functionality described with respect to retrieval unit 52 may be implemented in hardware, or a combination of hardware, software, and/or firmware, where requisite hardware may be provided to execute instructions for software or firmware.

[0129] Retrieval unit 52 may compare the decoding and rendering capabilities of client device 40 to characteristics of representations 68 indicated by information of manifest file 66. For example, retrieval unit 52 may determine whether client device 40 (such as video output 44) is capable of rendering stereoscopic data or only monoscopic data. Retrieval unit 52 may initially retrieve at least a portion of manifest file 66 to determine characteristics of representations 68. For example, retrieval unit 52 may request a portion of manifest file 66 that describes characteristics of one or more adaptation sets. Retrieval unit 52 may select a subset of representations 68 (e.g., an adaptation set) having characteristics that can be satisfied by the coding and rendering capabilities of client device 40. Retrieval unit 52 may then determine bitrates for representations in the adaptation set, determine a currently available amount of network bandwidth, and retrieve segments from one of the representations having a bitrate that can be satisfied by the network bandwidth. In accordance with the techniques of this disclosure, retrieval unit 52 may retrieve data indicating whether corresponding fisheye video data is monoscopic or stereoscopic. For example, retrieval unit 52 may retrieve an initial portion of a file (e.g., a segment) of one of representations 68 to determine whether the file includes monoscopic or stereoscopic fisheye video data, and determine whether to retrieve video data of the file, or a different file of a different one of representations 68, according to whether or not client device 40 is capable of rendering the fisheye video data of the file.

[0130] In general, higher bitrate representations may yield higher quality video playback, while lower bitrate representations may provide sufficient quality video playback when available network bandwidth decreases. Accordingly, when available network bandwidth is relatively high, retrieval unit 52 may retrieve data from relatively high bitrate representations, whereas when available network bandwidth is low, retrieval unit 52 may retrieve data from relatively low bitrate representations. In this manner, client device 40 may stream multimedia data over network 74 while also adapting to changing network bandwidth availability of network 74.

[0131] Additionally or alternatively, retrieval unit 52 may be configured to receive data in accordance with a broadcast or multicast network protocol, such as eMBMS or IP multicast. In such examples, retrieval unit 52 may submit a request to join a multicast network group associated with particular media content. After joining the multicast group, retrieval unit 52 may receive data of the multicast group without further requests issued to server device 60 or content preparation device 20. Retrieval unit 52 may submit a request to leave the multicast group when data of the multicast group is no longer needed, e.g., to stop playback or to change channels to a different multicast group.

[0132] Network interface 54 may receive and provide data of segments of a selected representation to retrieval unit 52, which may in turn provide the segments to decapsulation unit 50. Decapsulation unit 50 may decapsulate elements of a video file into constituent PES streams, depacketize the PES streams to retrieve encoded data, and send the encoded data to either audio decoder 46 or video decoder 48, depending on whether the encoded data is part of an audio or video stream, e.g., as indicated by PES packet headers of the stream. Audio decoder 46 decodes encoded audio data and sends the decoded audio data to audio output 42, while video decoder 48 decodes encoded video data and sends the decoded video data, which may include a plurality of views of a stream, to video output 44.

[0133] Video encoder 28, video decoder 48, audio encoder 26, audio decoder 46, encapsulation unit 30, retrieval unit 52, and decapsulation unit 50 each may be implemented as any of a variety of suitable processing circuitry, as applicable, such as one or more microprocessors, digital signal processors (DSPs), application specific integrated circuits (ASICs), field programmable gate arrays (FPGAs), discrete logic circuitry, software, hardware, firmware or any combinations thereof. Each of video encoder 28 and video decoder 48 may be included in one or more encoders or decoders, either of which may be integrated as part of a combined video encoder/decoder (CODEC). Likewise, each of audio encoder 26 and audio decoder 46 may be included in one or more encoders or decoders, either of which may be integrated as part of a combined CODEC. An apparatus including video encoder 28, video decoder 48, audio encoder 26, audio decoder 46, encapsulation unit 30, retrieval unit 52, and/or decapsulation unit 50 may comprise an integrated circuit, a microprocessor, and/or a wireless communication device, such as a cellular telephone.

[0134] Client device 40, server device 60, and/or content preparation device 20 may be configured to operate in accordance with the techniques of this disclosure. For purposes of example, this disclosure describes these techniques with respect to client device 40 and server device 60. However, it should be understood that content preparation device 20 may be configured to perform these techniques, instead of (or in addition to) server device 60.

[0135] Encapsulation unit 30 may form NAL units comprising a header that identifies a program to which the NAL unit belongs, as well as a payload, e.g., audio data, video data, or data that describes the transport or program stream to which the NAL unit corresponds. For example, in H.264/AVC, a NAL unit includes a 1-byte header and a payload of varying size. A NAL unit including video data in its payload may comprise various granularity levels of video data. For example, a NAL unit may comprise a block of video data, a plurality of blocks, a slice of video data, or an entire picture of video data. Encapsulation unit 30 may receive encoded video data from video encoder 28 in the form of PES packets of elementary streams. Encapsulation unit 30 may associate each elementary stream with a corresponding program.

[0136] Encapsulation unit 30 may also assemble access units from a plurality of NAL units. In general, an access unit may comprise one or more NAL units for representing a frame of video data, as well audio data corresponding to the frame when such audio data is available. An access unit generally includes all NAL units for one output time instance, e.g., all audio and video data for one time instance. For example, if each view has a frame rate of 20 frames per second (fps), then each time instance may correspond to a time interval of 0.05 seconds. During this time interval, the specific frames for all views of the same access unit (the same time instance) may be rendered simultaneously. In one example, an access unit may comprise a coded picture in one time instance, which may be presented as a primary coded picture.

[0137] Accordingly, an access unit may comprise all audio and video frames of a common temporal instance, e.g., all views corresponding to time *X*. This disclosure also refers to an encoded picture of a particular view as a "view component." That is, a view component may comprise an encoded picture (or frame) for a particular view at a particular time. Accordingly, an access unit may be defined as comprising all view components of a common temporal instance. The decoding order of access units need not necessarily be the same as the output or display order.

[0138] A media presentation may include a media presentation description (MPD), which may contain descriptions of different alternative representations (e.g., video services with different qualities) and the description may include, e.g., codec information, a profile value, and a level value. An MPD is one example of a manifest file, such as manifest file 66. Client device 40 may retrieve the MPD of a media presentation to determine how to access movie fragments of various presentations. Movie fragments may be located in movie fragment boxes (moof boxes) of video files.

[0139] Manifest file 66 (which may comprise, for example, an MPD) may advertise availability of segments of representations 68. That is, the MPD may include information indicating the wall-clock time at which a first segment of one of representations 68 becomes available, as well as information indicating the durations of segments within representations 68. In this manner, retrieval unit 52 of client device 40 may determine when each segment is available, based on the starting time as well as the durations of the segments preceding a particular segment.

[0140] After encapsulation unit 30 has assembled NAL units and/or access units into a video file based on received data, encapsulation unit 30 passes the video file to output interface 32 for output. In some examples, encapsulation unit 30 may store the video file locally or send the video file to a remote server via output interface 32, rather than sending the video file directly to client device 40. Output interface 32 may comprise, for example, a transmitter, a transceiver, a device for writing data to a computer-readable medium such as, for example, an optical drive, a magnetic media drive (e.g., floppy drive), a universal serial bus (USB) port, a network interface, or other output interface. Output interface 32 outputs the video file to a computer-readable medium, such as, for example, a transmission signal, a magnetic medium, an optical medium, a memory, a flash drive, or other computer-readable medium.

[0141] Network interface 54 may receive a NAL unit or access unit via network 74 and provide the NAL unit or access unit to decapsulation unit 50, via retrieval unit 52. Decapsulation unit 50 may decapsulate elements of a video file into constituent PES streams, depacketize the PES streams to retrieve encoded data, and send the encoded data to either audio decoder 46 or video decoder 48, depending on whether the encoded data is part of an audio or video stream, e.g., as indicated by PES packet headers of the stream. Audio decoder 46 decodes encoded audio data and sends the decoded audio data to audio output 42, while video decoder 48 decodes encoded video data and sends the decoded video data, which may include a plurality of views of a stream, to video output 44.

[0142] In this manner, client device 40 represents an example of a device for retrieving media data, the device including a device for retrieving fisheye video data files as described above and/or as claimed below. For instance, client device 40 may retrieve fisheye video data files and/or render fisheye video based on a determination of whether the fisheye video is monoscopic or stereoscopic and this determination may be based on a syntax element that explicitly specifies whether the fisheye video is monoscopic or stereoscopic.

[0143] Similarly, content preparation device 20 represents a device for generating fisheye video data files as described above and/or as claimed below. For instance, content preparation device 20 may include, in a fisheye video data, a syntax element that explicitly specifies whether the fisheye video is monoscopic or stereoscopic.

[0144] FIG. 10 is a conceptual diagram illustrating elements of example multimedia content 120. Multimedia content 120 may correspond to multimedia content 64 (FIG. 9), or another multimedia content stored in storage medium 62. In the example of FIG. 10, multimedia content 120 includes media presentation description (MPD) 122 and a plurality of representations 124A–124N (representations 124). Representation 124A includes optional header data 126 and segments 128A–128N (segments 128), while representation 124N includes optional header data 130 and segments 132A–132N (segments 132). The letter N is used to designate the last movie fragment in each of representations 124 as a matter of convenience. In some examples, there may be different numbers of movie fragments between representations 124.

[0145] MPD 122 may comprise a data structure separate from representations 124. MPD 122 may correspond to manifest file 66 of FIG. 9. In general, MPD 122 may include data that generally describes characteristics of representations 124, such as coding and rendering characteristics, adaptation sets, a profile to which MPD 122 corresponds, text type information, camera angle information, rating information, trick mode information (e.g., information indicative of representations that include temporal sub-sequences), and/or information for retrieving remote periods (e.g., for targeted advertisement insertion into media content during playback).

[0146] Header data 126, when present, may describe characteristics of segments 128, e.g., temporal locations of random access points (RAPs, also referred to as stream access points (SAPs)), which of segments 128 includes random access points, byte offsets to random access points within segments 128, uniform resource locators (URLs) of segments 128, or other aspects of segments 128. Header data 130, when present, may describe similar characteristics for segments 132. Additionally or alternatively, such characteristics may be fully included within MPD 122.

[0147] Segments 128, 132 include one or more coded video samples, each of which may include frames or slices of video data. Each of the coded video samples of segments 128 may have similar characteristics, e.g., height, width, and bandwidth requirements. Such characteristics may be described by data of MPD 122, though such data is not illustrated in the example of FIG. 10. MPD 122 may include characteristics as described by the 3GPP Specification, with the addition of any or all of the signaled information described in this disclosure.

[0148] Each of segments 128, 132 may be associated with a unique uniform resource locator (URL). Thus, each of segments 128, 132 may be independently retrievable using a streaming network protocol, such as DASH. In this manner, a destination device, such as client device 40, may use an HTTP GET request to retrieve segments 128 or 132. In some examples, client device 40 may use HTTP partial GET requests to retrieve specific byte ranges of segments 128 or 132.

[0149] FIG. 11 is a block diagram illustrating elements of an example video file 150, which may correspond to a segment of a representation, such as one of segments 114, 124 of FIG. 10. Each of segments 128, 132 may include data that conforms substantially to the arrangement of data illustrated in the example of FIG. 11. Video file 150 may be said to encapsulate a segment. As described above, video files in accordance with the ISO base media file format and extensions thereof store data in a series of objects, referred to as "boxes." In the example of FIG. 11, video file 150 includes file type (FTYP) box 152, movie (MOOV) box 154, segment index (sidx) boxes 162, movie fragment (MOOF) boxes 164, and movie fragment random access (MFRA) box 166. Although FIG. 11 represents an example of a video file, it should be understood that other media files may include other types of media data (e.g., audio data, timed text data, or the like) that is structured similarly to the data of video file 150, in accordance with the ISO base media file format and its extensions.

[0150] File type (FTYP) box 152 generally describes a file type for video file 150. File type box 152 may include data that identifies a specification that describes a best use for video file 150. File type box 152 may alternatively be placed before MOOV box 154, movie fragment boxes 164, and/or MFRA box 166.

[0151] In some examples, a Segment, such as video file 150, may include an MPD update box (not shown) before FTYP box 152. The MPD update box may include information indicating that an MPD corresponding to a representation including video file 150 is to be updated, along with information for updating the MPD. For example, the MPD update box may provide a URI or URL for a resource to be used to update the MPD. As another example, the MPD update box may include data for updating the MPD. In some examples, the MPD update box may immediately follow a segment type (STYP) box (not shown) of video file 150, where the STYP box may define a segment type for video file 150.

[0152] MOOV box 154, in the example of FIG. 11, includes movie header (MVHD) box 156, track (TRAK) box 158, and one or more movie extends (MVEX) boxes 160. In general, MVHD box 156 may describe general characteristics of video file 150. For example, MVHD box 156 may include data that describes when video file 150 was originally created, when video file 150 was last modified, a timescale for video file 150, a duration of playback for video file 150, or other data that generally describes video file 150.

[0153] TRAK box 158 may include data for a track of video file 150. TRAK box 158 may include a track header (TKHD) box that describes characteristics of the track corresponding to TRAK box 158. In some examples, TRAK box 158 may include coded video pictures, while in other examples, the coded video pictures of the track may be included in movie fragments 164, which may be referenced by data of TRAK box 158 and/or sidx boxes 162.

[0154] In some examples, video file 150 may include more than one track. Accordingly, MOOV box 154 may include a number of TRAK boxes equal to the number of tracks in video file 150. TRAK box 158 may describe characteristics of a corresponding track of video file 150. For example, TRAK box 158 may describe temporal and/or spatial information for the corresponding track. A TRAK box similar to TRAK box 158 of MOOV box 154 may describe characteristics of a parameter set track, when encapsulation unit 30 (FIG. 9) includes a parameter set track in a video file, such as video file 150. Encapsulation unit 30 may signal the presence of sequence level SEI messages in the parameter set track within the TRAK box describing the parameter set track.

[0155] MVEX boxes 160 may describe characteristics of corresponding movie fragments 164, e.g., to signal that video file 150 includes movie fragments 164, in addition to video data included within MOOV box 154, if any. In the context of streaming video data, coded video pictures may be included in movie fragments 164 rather than in MOOV box 154. Accordingly, all coded video samples may be included in movie fragments 164, rather than in MOOV box 154.

[0156] MOOV box 154 may include a number of MVEX boxes 160 equal to the number of movie fragments 164 in video file 150. Each of MVEX boxes 160 may describe characteristics of a corresponding one of movie fragments 164. For example, each MVEX box may include a movie extends header box (MEHD) box that describes a temporal duration for the corresponding one of movie fragments 164.

[0157] Furthermore, in accordance with the techniques of this disclosure, video file 150 may include a FisheyeOmnidirectionalVideoBox in a SchemeInformationBox, which may be included in MOOV box 154. In some examples, the FisheyeOmnidirectionalVieoBox may be included in TRAK box 158, if different tracks of video file 150 may include monoscopic or stereoscopic fisheye video data. In some examples, the FisheyeOmnidirectionalVieoBox may be included in FOV box 157.

[0158] As noted above, encapsulation unit 30 may store a sequence data set in a video sample that does not include actual coded video data. A video sample may generally correspond to an access unit, which is a representation of a coded picture at a specific time instance. In the context of AVC, the coded picture may include one or more VCL NAL units which contains the information to construct all the pixels of the access unit and other associated non-VCL NAL units, such as SEI messages. Accordingly, encapsulation unit 30 may include a sequence data set, which may include sequence level SEI messages, in one of movie fragments 164. Encapsulation unit 30 may further signal the presence of a sequence data set and/or sequence level SEI messages as being present in one of movie fragments 164 within the one of MVEX boxes 160 corresponding to the one of movie fragments 164.

[0159] SIDX boxes 162 are optional elements of video file 150. That is, video files conforming to the 3GPP file format, or other such file formats, do not necessarily include SIDX boxes 162. In accordance with the example of the 3GPP file format, a SIDX box may be used to identify a sub-segment of a segment (e.g., a segment contained within video file 150). The 3GPP file format defines a sub-segment as "a self-contained set of one or more consecutive movie fragment boxes with corresponding Media Data box(es) and a Media Data Box containing data referenced by a Movie Fragment Box must follow that Movie Fragment box and precede the next Movie Fragment box containing information about the same track." The 3GPP file format also indicates that a SIDX box "contains a sequence of references to subsegments of the (sub)segment documented by the box. The referenced subsegments are contiguous in presentation time. Similarly, the bytes referred to by a Segment Index box are always contiguous within the segment. The referenced size gives the count of the number of bytes in the material referenced."

[0160] SIDX boxes 162 generally provide information representative of one or more sub-segments of a segment included in video file 150. For instance, such information may include playback times at which sub-segments begin and/or end, byte offsets for the sub-segments, whether the sub-segments include (e.g., start with) a stream access point (SAP), a type for the SAP (e.g., whether the SAP is an instantaneous decoder refresh (IDR) picture, a clean random access (CRA) picture, a broken link access (BLA) picture, or the like), a position of the SAP (in terms of playback time and/or byte offset) in the sub-segment, and the like.

[0161] Movie fragments 164 may include one or more coded video pictures. In some examples, movie fragments 164 may include one or more groups of pictures (GOPs), each of which may include a number of coded video pictures, e.g., frames or pictures. In addition, as described above, movie fragments 164 may include sequence data sets in some examples. Each of movie fragments 164 may include a movie fragment header box (MFHD, not shown in FIG. 11). The MFHD box may describe characteristics of the corresponding movie fragment, such as a sequence number for the movie fragment. Movie fragments 164 may be included in order of sequence number in video file 150.

[0162] MFRA box 166 may describe random access points within movie fragments 164 of video file 150. This may assist with performing trick modes, such as performing seeks to particular temporal locations (i.e., playback times) within a segment encapsulated by video file 150. MFRA box 166 is generally optional and need not be included in video files, in some examples. Likewise, a client device, such as client device 40, does not necessarily need to reference MFRA box 166 to correctly decode and display video data of video file 150. MFRA box 166 may include a number of track fragment random access (TFRA) boxes (not shown) equal to the number of tracks of video file 150, or in some examples, equal to the number of media tracks (e.g., non-hint tracks) of video file 150.

[0163] In some examples, movie fragments 164 may include one or more stream access points (SAPs), such as IDR pictures. Likewise, MFRA box 166 may provide indications of locations within video file 150 of the SAPs. Accordingly, a temporal sub-sequence of video file 150 may be formed from SAPs of video file 150. The temporal sub-sequence may also include other pictures, such as P-frames and/or B-frames that depend from SAPs. Frames and/or slices of the temporal sub-sequence may be arranged within the segments such that frames/slices of the temporal sub-sequence that depend on other frames/slices of the sub-sequence can be properly decoded. For example, in the hierarchical arrangement of data, data used for prediction for other data may also be included in the temporal sub-sequence.

[0164] FIG. 12 is a flowchart illustrating an example technique for processing a file that includes fisheye video data, in accordance with one or more techniques of this disclosure. The techniques of FIG. 12 are described as being performed by client device 40 of FIG. 9, although devices having configurations other than client device 40 may perform the technique of FIG. 12.

[0165] Client device 40 may receive a file including fisheye video data and a syntax structure that includes a plurality of syntax elements specifying attributes of the fisheye video data (1202). As discussed above, the plurality of syntax elements may include a first syntax element that explicitly indicates whether the fisheye video data is monoscopic or stereoscopic, and one or more syntax elements that implicitly indicate whether the fisheye video data is monoscopic or stereoscopic. The one or more syntax elements that implicitly indicate whether the fisheye video data is monoscopic or stereoscopic may be syntax elements that explicitly indicate extrinsic parameters of the cameras used to capture the fisheye video data. In some examples, the one or more syntax elements may be, or may be similar to, syntax elements included in the FisheyeOmnidirectionalVideoInfo( ) syntax structure in the current draft of OMAF DIS. In some examples, the first syntax element may be included in a set of initial bits of the syntax structure (e.g., an initial 24 bits of the FisheyeOmnidirectionalVideoInfo( ) syntax structure).

[0166] Client device 40 may determine, based on the first syntax element, whether the fisheye video data is monoscopic or stereoscopic (1204). A value of the first syntax element may explicitly indicate whether the fisheye video data is monoscopic or stereoscopic. As one example, where the first syntax element has a first value, client device 40 may determine that the fisheye video data is monoscopic (i.e., that the circular images included in pictures of the fisheye video data have aligned optical axes and face opposite directions). As another example, where the first syntax element has a second value, client device 40 may determine that the fisheye video data is stereoscopic (i.e., that the circular images included in pictures of the fisheye video data have parallel optical axes and face the same direction).

[0167] As discussed above, while it may be possible for client device 40 to determine whether the fisheye video data is monoscopic or stereoscopic based on the syntax elements that explicitly indicate extrinsic parameters of the cameras used to capture the fisheye video data, such a calculation may increase the computational load on client device 40. As such, in order to reduce the calculations performed (and thus the computational resources used), client device 40 may determine whether the fisheye video data is monoscopic or stereoscopic based on the first syntax element.

[0168] Client device 40 may render the fisheye video data based on the determination. For instance, responsive to determining that the fisheye video data is monoscopic, client device 40 may render the fisheye data as monoscopic (1206). For instance, client device 40 may determine a viewport (i.e., a portion of the sphere in which a user is currently “looking”), identify a portion of the circular images of the fisheye video data that correspond to the viewport, and display the same portion of the circular images to each of a viewer’s eyes. Similarly, responsive to determining that the fisheye video data is stereoscopic, client device 40 may render the fisheye data as stereoscopic (1208). For instance, client device 40 may determine a viewport (i.e., a portion of the sphere in which a user is currently “looking”), identify a respective portion of each of the circular images of the fisheye video data that correspond to the viewport, and display the respective portions of the circular images to the viewer's eyes.

[0169] FIG. 13 is a flowchart illustrating an example technique for generating a file that includes fisheye video data, in accordance with one or more techniques of this disclosure. The techniques of FIG. 13 are described as being performed by content preparation device 20 of FIG. 9, although devices having configurations other than content preparation device 20 may perform the technique of FIG. 13.

[0170] Content preparation device 20 may obtain fisheye video data and extrinsic parameters of cameras used to capture the fisheye video data (1302). For instance, content preparation device 20 may obtain pictures of the fisheye video data that are encoded using a video codec, such as HEVC. Each of the picture of fisheye video data may include a plurality of circular images that each correspond to an image captured by a different camera with a fisheye lens (e.g., a picture may include a first circular image captured via fisheye lens 12A of FIG. 1A and a second circular image captured via fisheye lens 12B of FIG. 1A). The extrinsic parameters may specify various attributes of the cameras. For instance, the extrinsic parameters may specify a yaw angle, a pitch angle, a roll angle, and one or more spatial offsets of each of the cameras used to capture the fisheye video data.

[0171] Content preparation device 20 may determine, based on the extrinsic parameters, whether the fisheye video data is monoscopic or stereoscopic (1304). As one example, when the two sets of camera extrinsic parameter values are as follows, content preparation device 20 may determine that the fisheye video is stereoscopic:

1st set:

camera_center_yaw = 0 degrees (+/- 5 degrees)
camera center_pitch = 0 degrees (+/- 5 degrees)
camera center roll = 0 degrees (+/- 5 degrees)
camera_center_offset_x = 0 mm (+/- 3 mm)
camera_center_offset_y = 0 mm (+/- 3 mm)
camera_center_offset_z = 0 mm (+/- 3 mm)

2nd set:

camera_center_yaw = 0 degrees (+/- 5 degrees)
camera center_pitch = 0 degrees (+/- 5 degrees)
camera center roll = 0 degrees (+/- 5 degrees)
camera_center_offset_x = 64 mm (+/- 3 mm)

camera_center_offset_y = 0 mm (+/- 3 mm)

camera_center_offset_z = 0 mm (+/- 3 mm)

[0172] As another example, when the two sets of camera extrinsic parameter values are as follows, content preparation device 20 may determine that the fisheye video is monoscopic:

1st set:

camera_center_yaw = 0 degrees (+/- 5 degrees)

camera center_pitch = 0 degrees (+/- 5 degrees)

camera center roll = 0 degrees (+/- 5 degrees)

camera_center_offset_x = 0 mm (+/- 3 mm)

camera_center_offset_y = 0 mm (+/- 3 mm)

camera_center_offset_z = 0 mm (+/- 3 mm)

2nd set:

camera_center_yaw = 180 degrees (+/- 5 degrees)

camera center_pitch = 0 degrees (+/- 5 degrees)

camera center roll = 0 degrees (+/- 5 degrees)

camera_center_offset_x = 0 mm (+/- 3 mm)

camera_center_offset_y = 0 mm (+/- 3 mm)

camera_center_offset_z = 0 mm (+/- 3 mm)

[0173] Content preparation device 20 may encode, in a file, the fisheye video data and a syntax structure including a plurality of syntax elements that specify attributes of the fisheye video data (1306). The plurality of syntax elements includes: a first syntax element that explicitly indicates whether the fisheye video data is monoscopic or stereoscopic, and one or more syntax elements that explicitly indicate the extrinsic parameters of the cameras used to capture the fisheye video data.

[0174] In one or more examples, the functions described may be implemented in hardware, software, firmware, or any combination thereof. If implemented in software, the functions may be stored on or transmitted over as one or more instructions or code on a computer-readable medium and executed by a hardware-based processing unit. Computer-readable media may include computer-readable storage media, which corresponds to a tangible medium such as data storage media, or communication media including any medium that facilitates transfer of a computer program from one place to another, e.g., according to a communication protocol. In this manner, computer-readable media generally may correspond to (1) tangible computer-readable storage media which is non-transitory or (2) a communication medium such as a signal or carrier wave. Data storage media may be any available media that can be accessed by one or more computers or one or more processors to retrieve instructions, code, and/or data structures for implementation of the techniques described in this disclosure. A computer program product may include a computer-readable medium.

[0175] By way of example, and not limitation, such computer-readable storage media can comprise RAM, ROM, EEPROM, CD-ROM or other optical disk storage, magnetic disk storage, or other magnetic storage devices, flash memory, or any other medium that can be used to store desired program code in the form of instructions or data structures and that can be accessed by a computer. Also, any connection is properly termed a computer-readable medium. For example, if instructions are transmitted from a website, server, or other remote source using a coaxial cable, fiber optic cable, twisted pair, digital subscriber line (DSL), or wireless technologies such as infrared, radio, and microwave, then the coaxial cable, fiber optic cable, twisted pair, DSL, or wireless technologies such as infrared, radio, and microwave are included in the definition of medium. It should be understood, however, that computer-readable storage media and data storage media do not include connections, carrier waves, signals, or other transitory media, but are instead directed to non-transitory, tangible storage media. Disk and disc, as used herein, includes compact disc (CD), laser disc, optical disc, digital versatile disc (DVD), floppy disk and Blu-ray disc where disks usually reproduce data magnetically, while discs reproduce data optically with lasers. Combinations of the above should also be included within the scope of computer-readable media.

[0176] Instructions may be executed by one or more processors, such as one or more digital signal processors (DSPs), general purpose microprocessors, application specific integrated circuits (ASICs), field programmable logic arrays (FPGAs), or other equivalent integrated or discrete logic circuitry. Accordingly, the term “processor,” as used herein may refer to any of the foregoing structure or any other structure suitable for implementation of the techniques described herein. In addition, in some aspects, the functionality described herein may be provided within dedicated hardware and/or software modules configured for encoding and decoding, or incorporated in a combined codec. Also, the techniques could be fully implemented in one or more circuits or logic elements.

[0177] The techniques of this disclosure may be implemented in a wide variety of devices or apparatuses, including a wireless handset, an integrated circuit (IC) or a set of

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

[0178] Various examples have been described. These and other examples are within the scope of the following claims.

WHAT IS CLAIMED IS:

1. A method of processing a file including video data, the method comprising:

processing a file including fisheye video data, the file including a syntax structure including a plurality of syntax elements that specify attributes of the fisheye video data, wherein the plurality of syntax elements includes: a first syntax element that explicitly indicates whether the fisheye video data is monoscopic or stereoscopic, and one or more syntax elements that implicitly indicate whether the fisheye video data is monoscopic or stereoscopic;

determining, based on the first syntax element, whether the fisheye video data is monoscopic or stereoscopic; and

outputting, based on the determination, the fisheye video data for rendering as monoscopic or stereoscopic.

2. The method of claim 1, wherein the first syntax element is included in a set of initial bits of the syntax structure.

3. The method of claim 2, wherein the set of initial bits is 24 bits long.

4. The method of claim 1, wherein the file includes a box that includes the syntax structure.

5. The method of claim 4, wherein the box is a first box that is included in a second box that includes scheme information, the method further comprising:

determining whether the first box includes a third box that indicates whether pictures of the fisheye video data are packed region-wise.

6. The method of claim 5, further comprising:

in response to determining that the first box includes the third box, unpacking the pictures of the fisheye video data prior to rendering the pictures of the fisheye video data; or

in response to determining that the first box does not include the third box, rendering the pictures of the fisheye video data without unpacking the pictures of the fisheye video data.

7. The method of claim 5, wherein the first box is a SchemeInformationBox, the second box is a FisheyeOmnidirectionalVideoBox, and the third box is a RegionWisePackingBox.

8. The method of claim 1, wherein the syntax structure further includes a second syntax element that specifies a number of circular images included in each picture of the fisheye video data.

9. The method of claim 8, wherein the syntax structure comprises, for each respective circular image, a respective third syntax element that indicates a view identifier of the respective circular image.

10. The method of claim 1, wherein the syntax structure is external to video coding layer (VCL) data encapsulated by the file.

11. The method of claim 1, wherein determining whether the fisheye video data is monoscopic or stereoscopic comprises:

determining, based on the first syntax element and regardless of the syntax elements that implicitly indicate whether the fisheye video data is monoscopic or stereoscopic, whether the fisheye video data is monoscopic or stereoscopic.

12. A method for generating a file including video data, the method comprising:

obtaining fisheye video data and extrinsic parameters of cameras used to capture the fisheye video data;

determining, based on the extrinsic parameters, whether the fisheye video data is monoscopic or stereoscopic; and

encoding, in a file, the fisheye video data and a syntax structure including a plurality of syntax elements that specify attributes of the fisheye video data, wherein the plurality of syntax elements includes: a first syntax element that explicitly indicates whether the fisheye video data is monoscopic or stereoscopic, and one or more syntax elements that explicitly indicate the extrinsic parameters of the cameras used to capture the fisheye video data.

13. The method of claim 12, wherein encoding the first syntax element comprises encoding the first syntax element in a set of initial bits of the syntax structure.

14. The method of claim 13, wherein the set of initial bits is 24 bits long.

15. The method of claim 12, wherein the file includes a box that includes the syntax structure.

16. The method of claim 15, wherein the box is a first box that is included in a second box that includes scheme information, the method further comprising:

encoding, in the first box, a third box that indicates whether pictures of the fisheye video data are packed region-wise.

17. The method of claim 16, wherein the first box is a SchemeInformationBox, the second box is a FisheyeOmnidirectionalVideoBox, and the third box is a RegionWisePackingBox.

18. The method of claim 12, wherein the syntax structure further includes a second syntax element that specifies a number of circular images included in each picture of the fisheye video data.

19. The method of claim 18, wherein the syntax structure comprises, for each respective circular image, a respective third syntax element that indicates a view identifier of the respective circular image.

20. The method of claim 12, wherein the fisheye video data is encoded in a video coding layer (VCL) and wherein the syntax structure is external to the VCL.

21. A device for processing video data, the device comprising:

a memory configured to store at least a portion of a file including fisheye video data, the file including a syntax structure including a plurality of syntax elements that specify attributes of the fisheye video data, wherein the plurality of syntax elements includes: a first syntax element that explicitly indicates whether the fisheye video data is monoscopic or stereoscopic, and one or more syntax elements that implicitly indicate whether the fisheye video data is monoscopic or stereoscopic; and

one or more processors configured to:

determine, based on the first syntax element, whether the fisheye video data is monoscopic or stereoscopic; and

output, based on the determination, the fisheye video data for rendering as monoscopic or stereoscopic.

22. The device of claim 21, wherein the first syntax element is included in a set of initial bits of the syntax structure.

23. The device of claim 22, wherein the set of initial bits is 24 bits long.

24. The device of claim 21, wherein the file includes a box that includes the syntax structure.

25. The device of claim 24, wherein the box is a first box that is included in a second box that includes scheme information, the one or more processors are further configured to:

determine whether the first box includes a third box that indicates whether pictures of the fisheye video data are packed region-wise.

26. The device of claim 25, wherein:

in response to determining that the first box includes the third box, the one or more processors are further configured to, unpack the pictures of the fisheye video data prior to rendering the pictures of the fisheye video data; or

in response to determining that the first box does not include the third box, the one or more processors are further configured to, render the pictures of the fisheye video data without unpacking the pictures of the fisheye video data.

27. The device of claim 25, wherein the first box is a SchemeInformationBox, the second box is a FisheyeOmnidirectionalVideoBox, and the third box is a RegionWisePackingBox.

28. The device of claim 21, wherein the syntax structure further includes a second syntax element that specifies a number of circular images included in each picture of the fisheye video data.

29. The device of claim 21, wherein the syntax structure is external to video coding layer (VCL) data encapsulated by the file.

30. The device of claim 21, wherein, to determine whether the fisheye video data is monoscopic or stereoscopic, the one or more processors are configured to:

determine, based on the first syntax element and regardless of the syntax elements that implicitly indicate whether the fisheye video data is monoscopic or stereoscopic, whether the fisheye video data is monoscopic or stereoscopic

31. A device for generating a file including video data, the device comprising:

a memory configured to store at fisheye video data; and

one or more processors configured to:

obtain extrinsic parameters of cameras used to capture the fisheye video data;

determine, based on the extrinsic parameters, whether the fisheye video data is monoscopic or stereoscopic; and

encode, in a file, the fisheye video data and a syntax structure including a plurality of syntax elements that specify attributes of the fisheye video data, wherein the plurality of syntax elements includes: a first syntax element that explicitly indicates whether the fisheye video data is monoscopic or stereoscopic, and one or more syntax elements that explicitly indicate the extrinsic parameters of the cameras used to capture the fisheye video data.

32. The device of claim 31, wherein, to encode the first syntax element, the one or more processors are configured to encode the first syntax element in a set of initial bits of the syntax structure.

33. The device of claim 32, wherein the set of initial bits is 24 bits long.

34. The device of claim 31, wherein the file includes a box that includes the syntax structure.

35. The device of claim 34, wherein the box is a first box that is included in a second box that includes scheme information, the one or more processors are further configured to:

encode, in the first box, a third box that indicates whether pictures of the fisheye video data are packed region-wise.

36. The device of claim 35, wherein the first box is a SchemeInformationBox, the second box is a FisheyeOmnidirectionalVideoBox, and the third box is a RegionWisePackingBox.

37. The device of claim 31, wherein the syntax structure further includes a second syntax element that specifies a number of circular images included in each picture of the fisheye video data.

38. The device of claim 31, wherein the fisheye video data is encoded in a video coding layer (VCL) and wherein the syntax structure is external to the VCL.

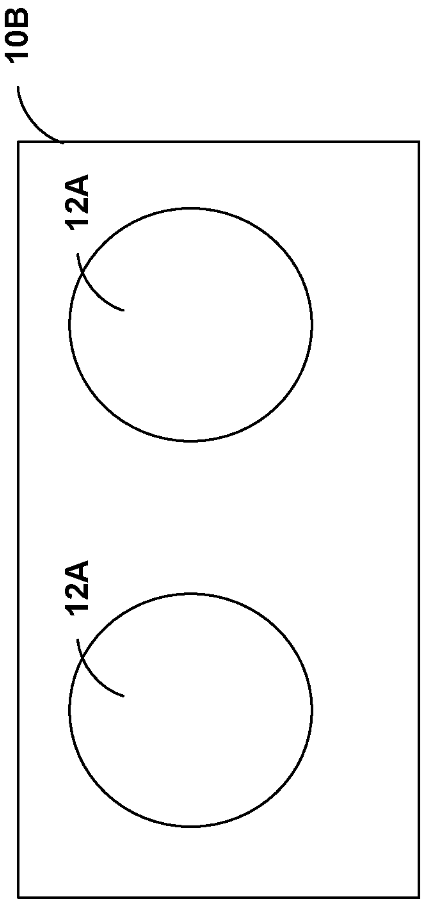


FIG. 1B

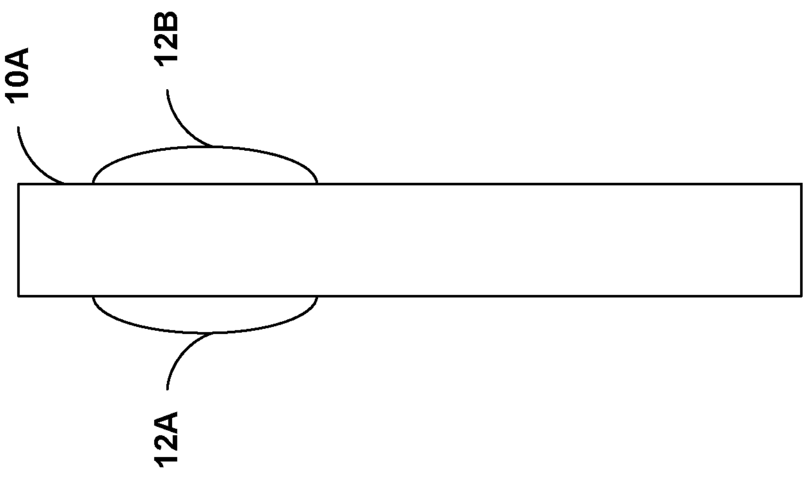


FIG. 1A

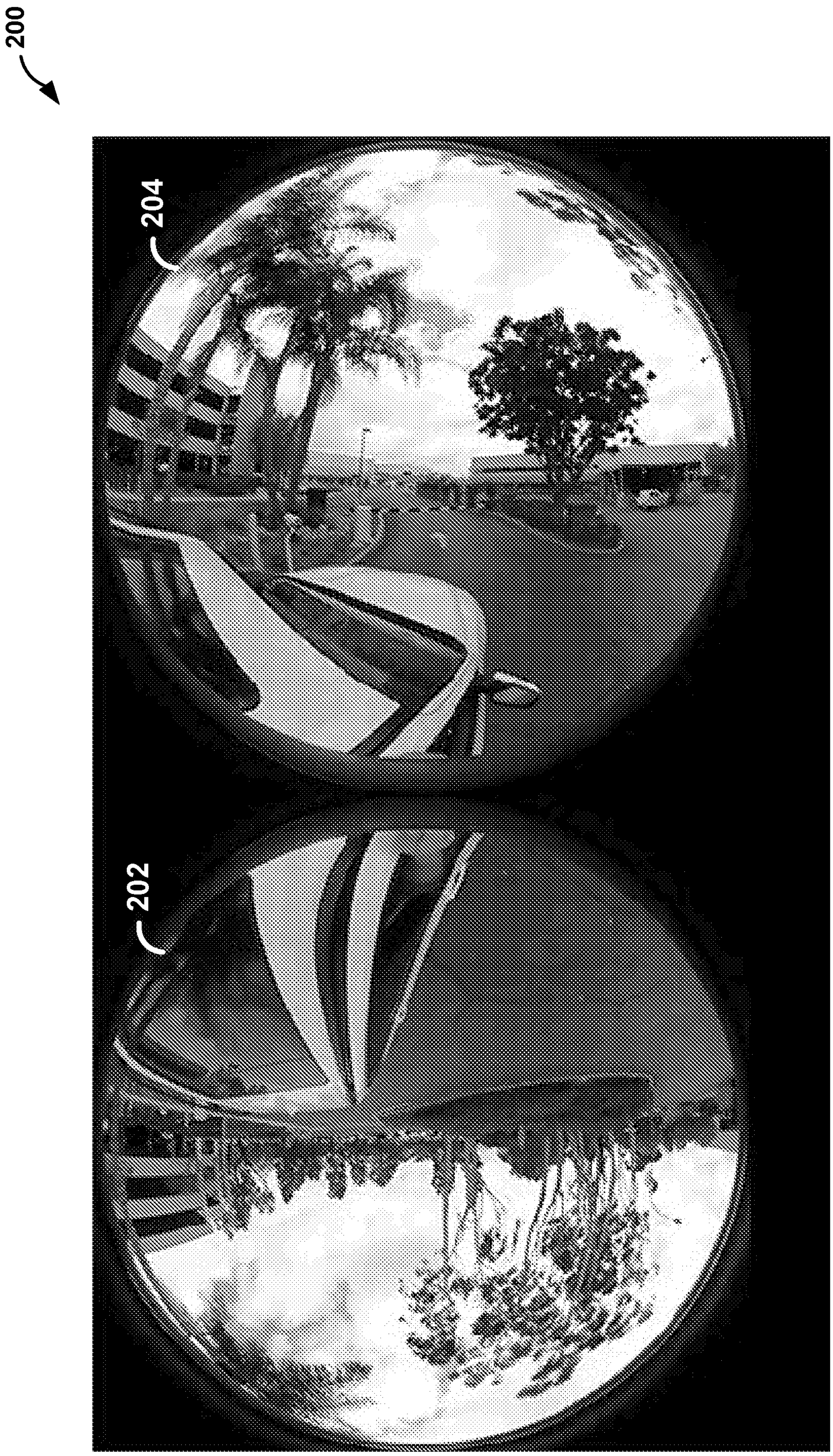


FIG. 2

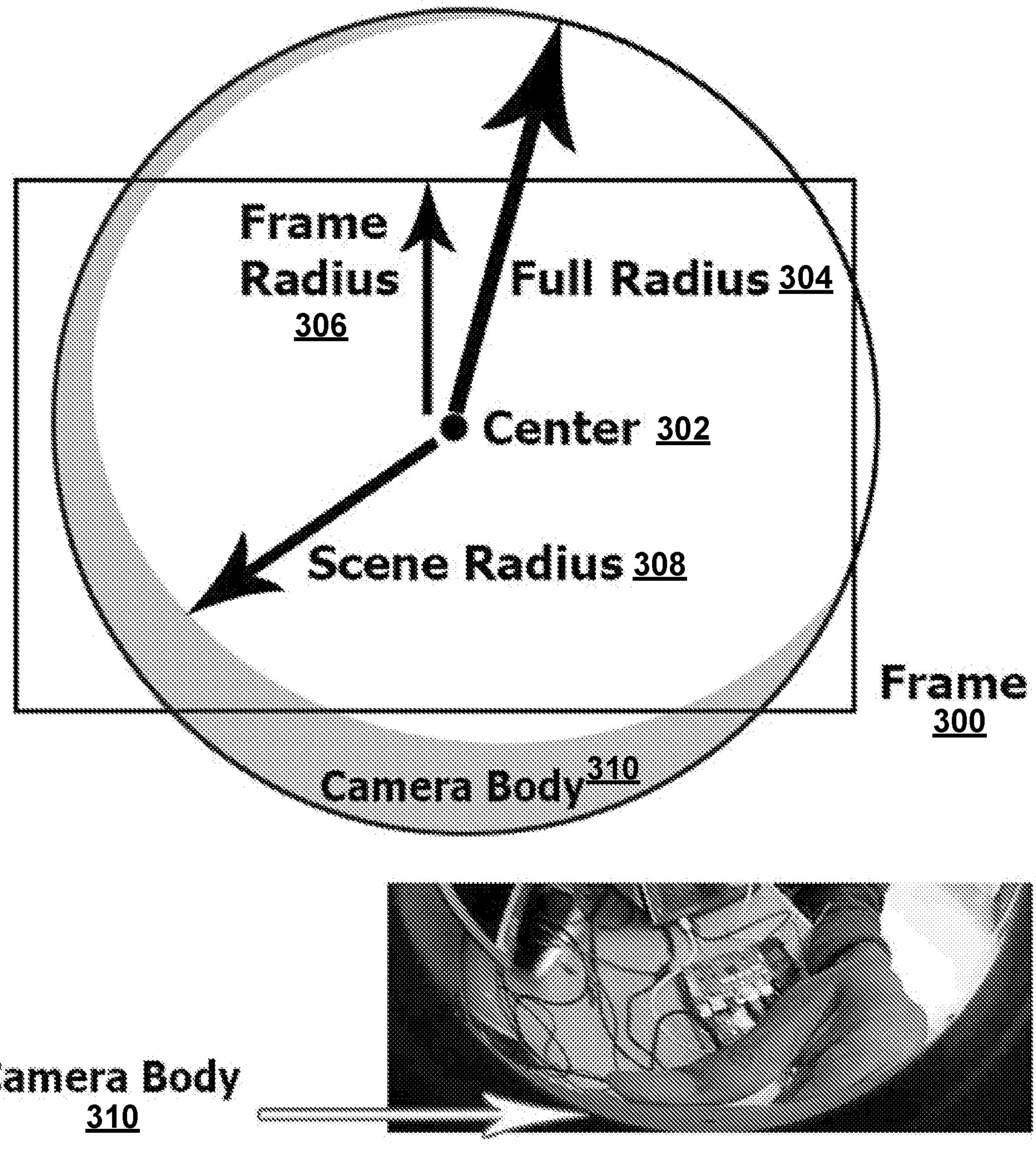

Frame
Radius
306
Full Radius 304
Center 302
Scene Radius 308
Frame
300
Camera Body 310
Camera Body
310

FIG. 3

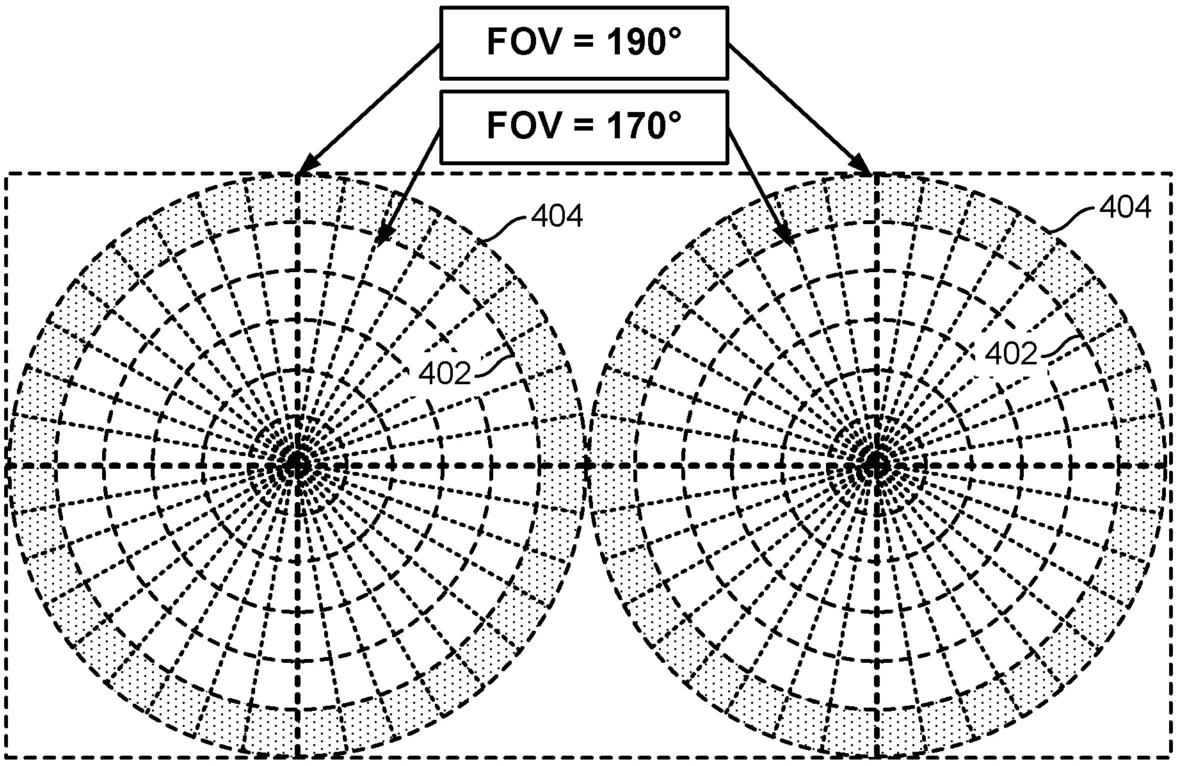


FIG. 4

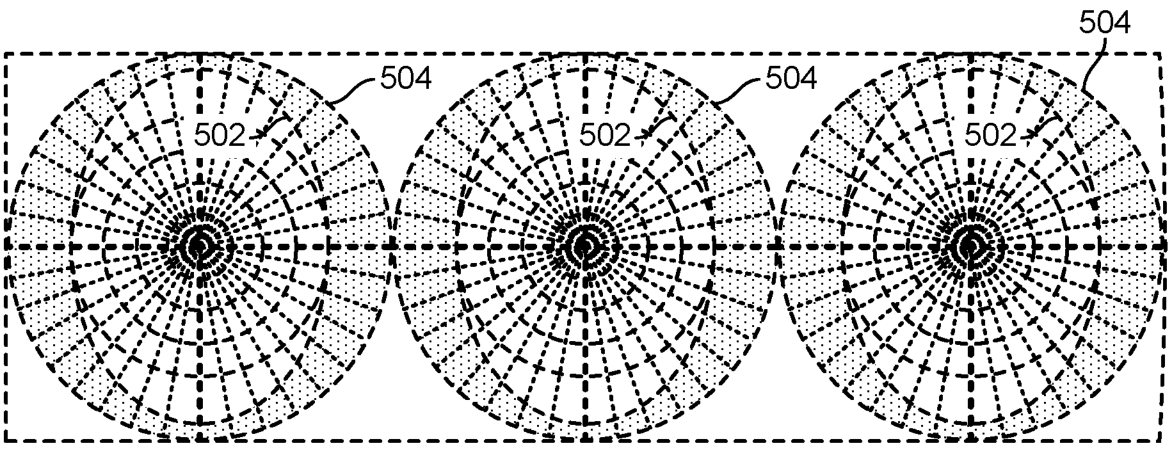


FIG. 5

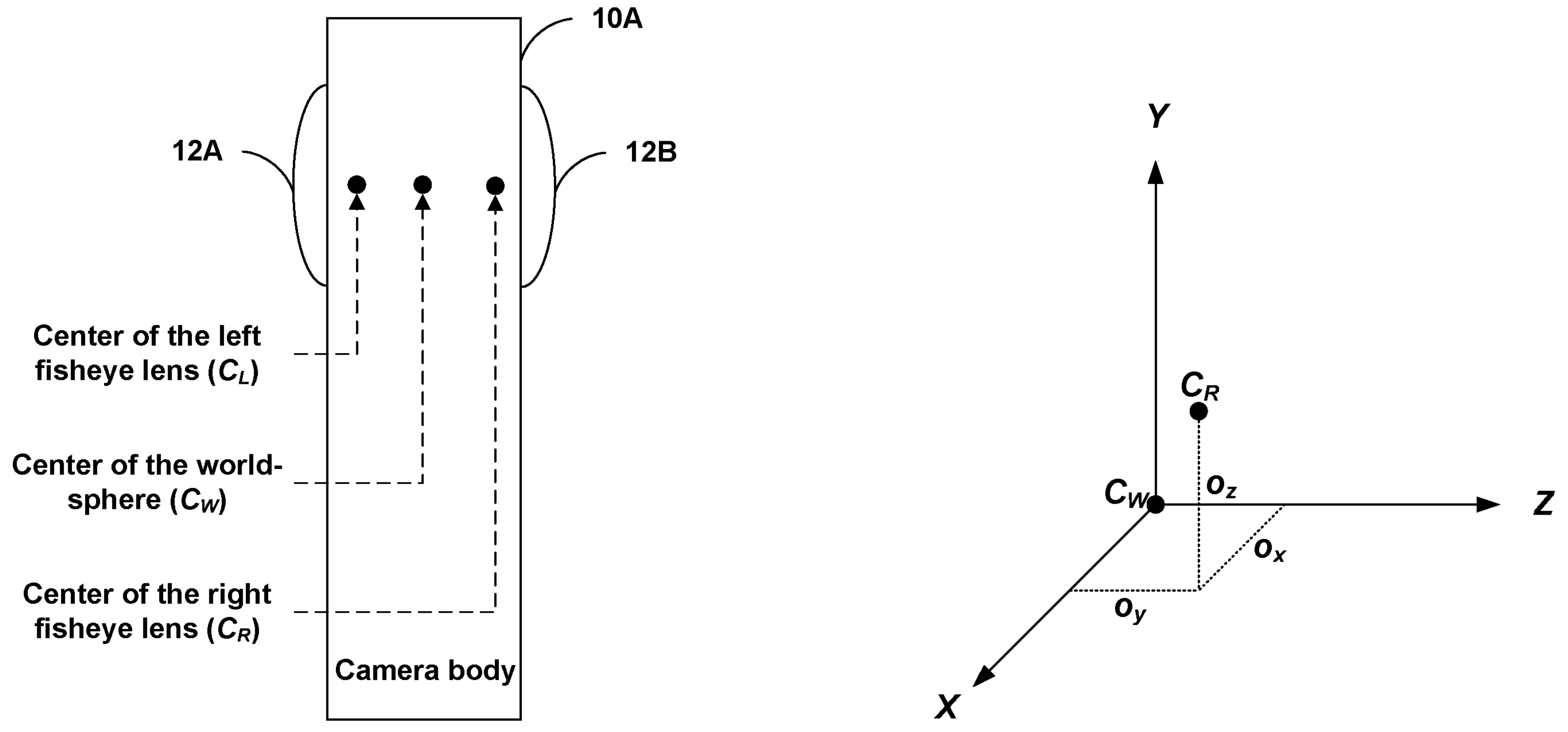
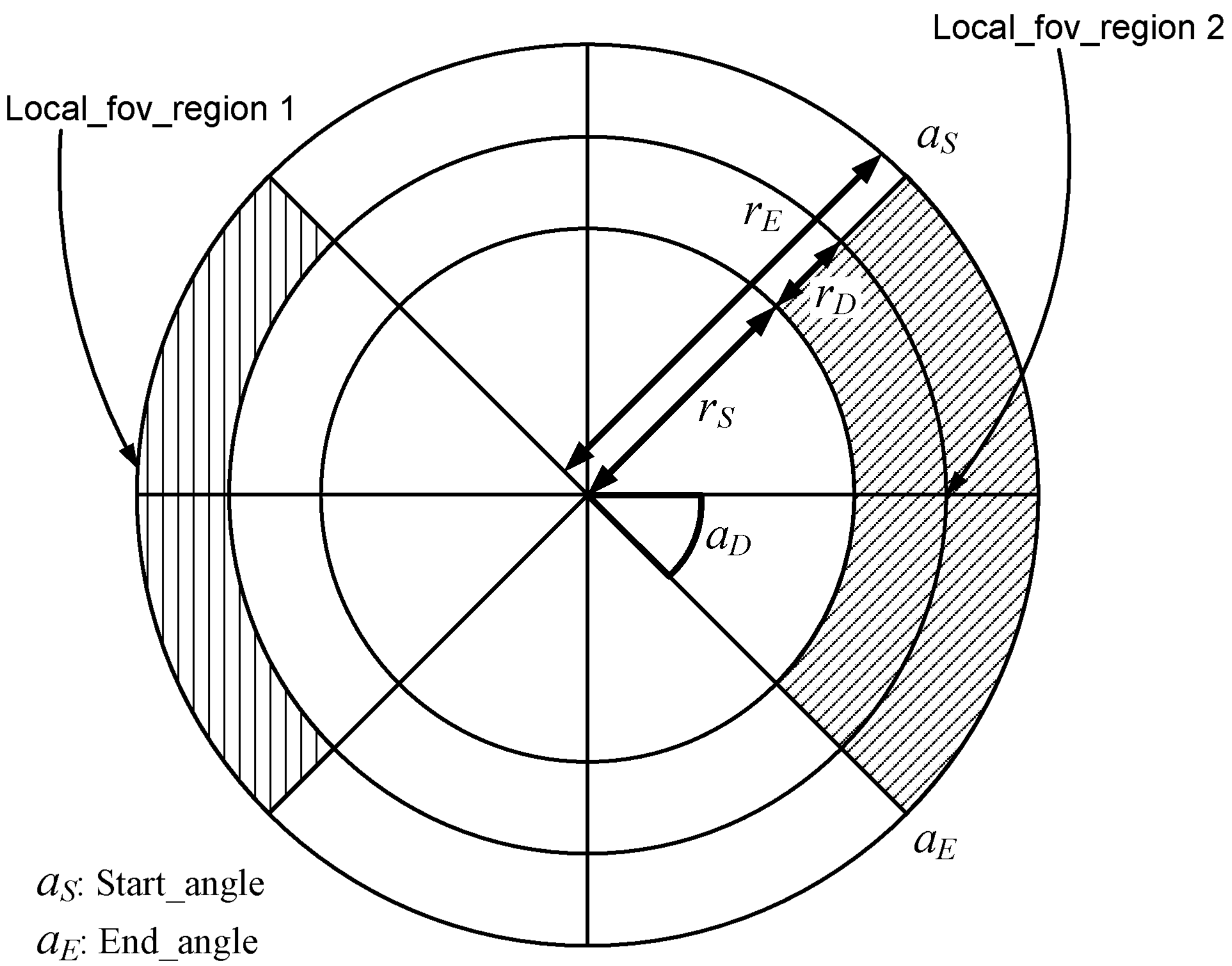

10A
12A
12B
Center of the left fisheye lens (C_L)
Center of the world-sphere (C_W)
Center of the right fisheye lens (C_R)
Camera body
Y
C_R
C_W
o_z
Z
o_x
o_y
X

FIG. 6



a_S: Start_angle

a_E: End_angle

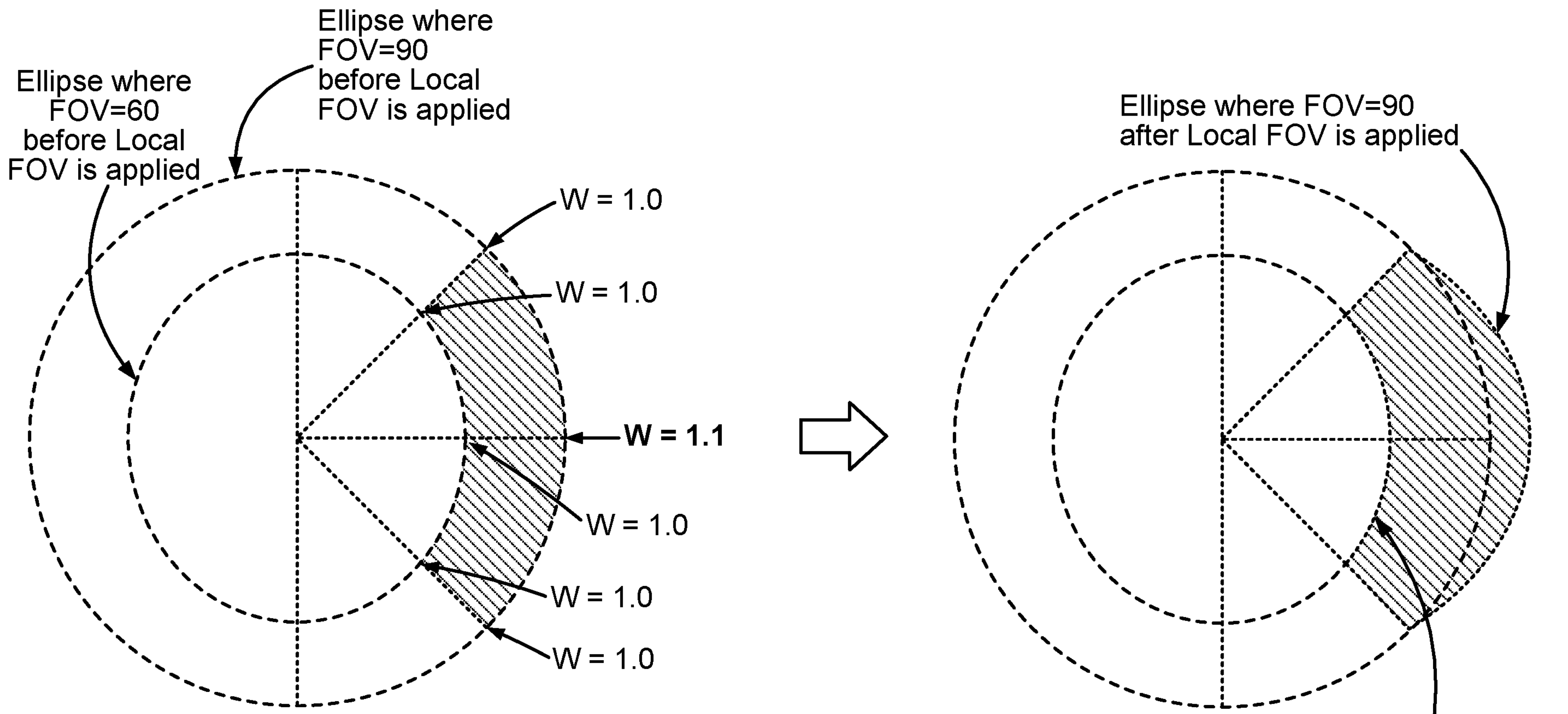
a_D: angle_delta

r_S: Start_radius

r_E: End_radius

r_D: radius_delta

FIG. 7


Ellipse where FOV=60 before Local FOV is applied
Ellipse where FOV=90 before Local FOV is applied
W = 1.0
W = 1.0
W = 1.1
W = 1.0
W = 1.0
W = 1.0
Ellipse where FOV=90 after Local FOV is applied
Ellipse where FOV=60 after Local FOV is applied

a_S: Start_angle = 45

a_E: End_angle = 135

a_D: angle_delta = 45

r_S: Start_radius = $0.5R_{max}$

r_E: End_radius = $1.0R_{max}$

r_D: radius_delta = $0.5R_{max}$

FIG. 8

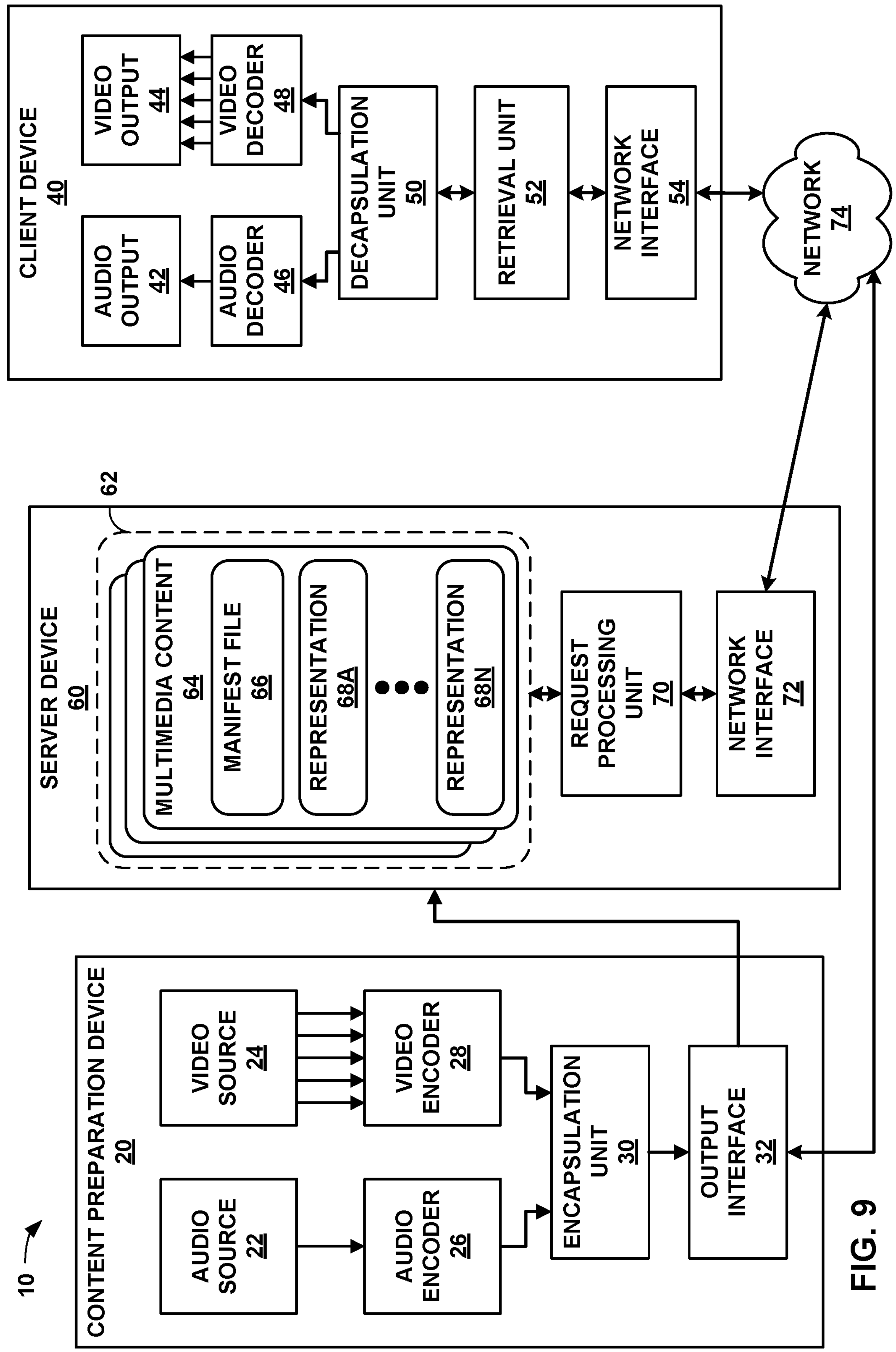

10
CONTENT PREPARATION DEVICE 20
AUDIO SOURCE 22
VIDEO SOURCE 24
AUDIO ENCODER 26
VIDEO ENCODER 28
ENCAPSULATION UNIT 30
OUTPUT INTERFACE 32
SERVER DEVICE 60
62
MULTIMEDIA CONTENT 64
MANIFEST FILE 66
REPRESENTATION 68A
REPRESENTATION 68N
REQUEST PROCESSING UNIT 70
NETWORK INTERFACE 72
CLIENT DEVICE 40
AUDIO OUTPUT 42
VIDEO OUTPUT 44
AUDIO DECODER 46
VIDEO DECODER 48
DECAPSULATION UNIT 50
RETRIEVAL UNIT 52
NETWORK INTERFACE 54
NETWORK 74

FIG. 9

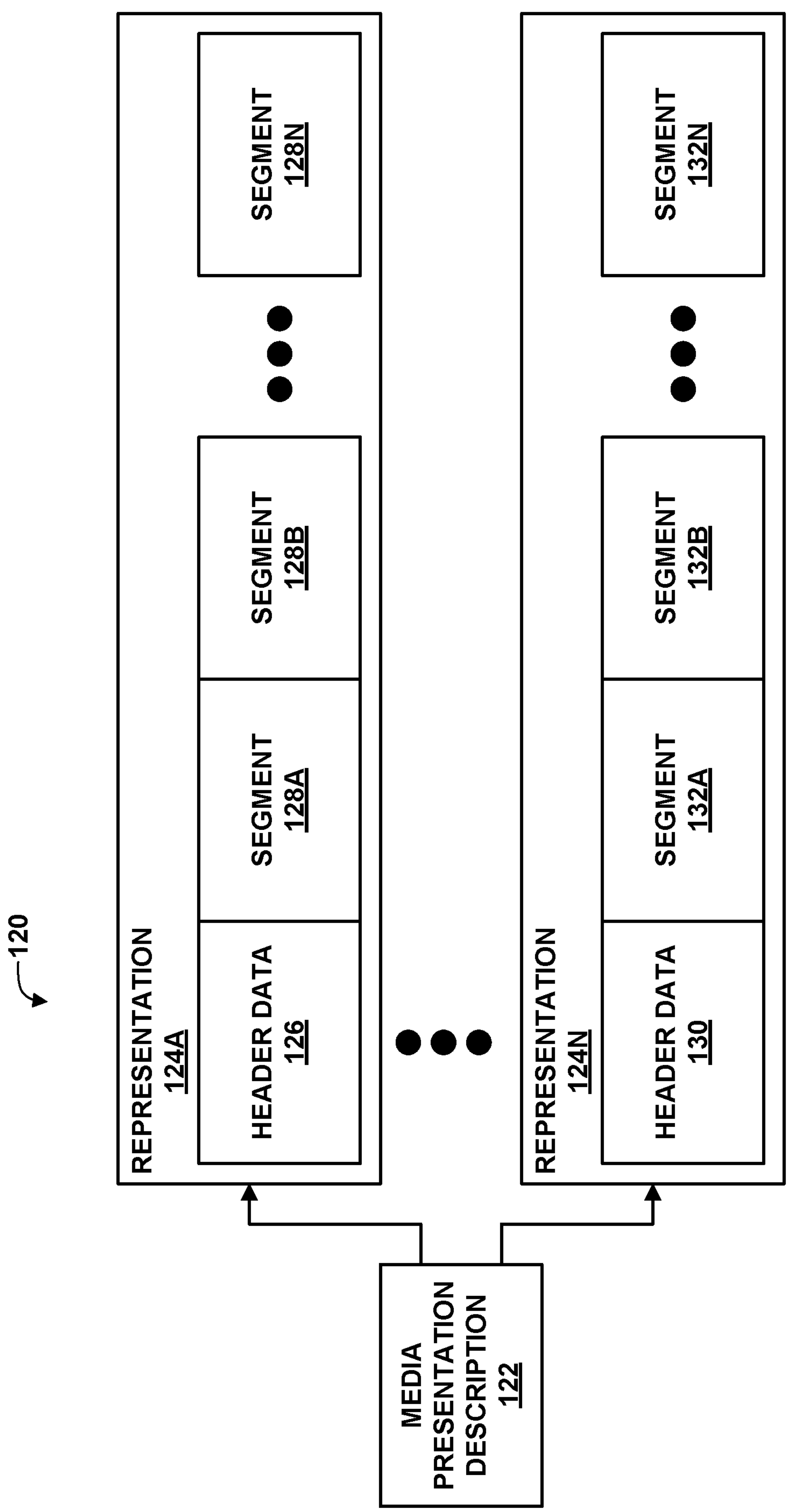

120
MEDIA PRESENTATION DESCRIPTION 122
REPRESENTATION 124A
HEADER DATA 126
SEGMENT 128A
SEGMENT 128B
SEGMENT 128N
REPRESENTATION 124N
HEADER DATA 130
SEGMENT 132A
SEGMENT 132B
SEGMENT 132N

FIG. 10

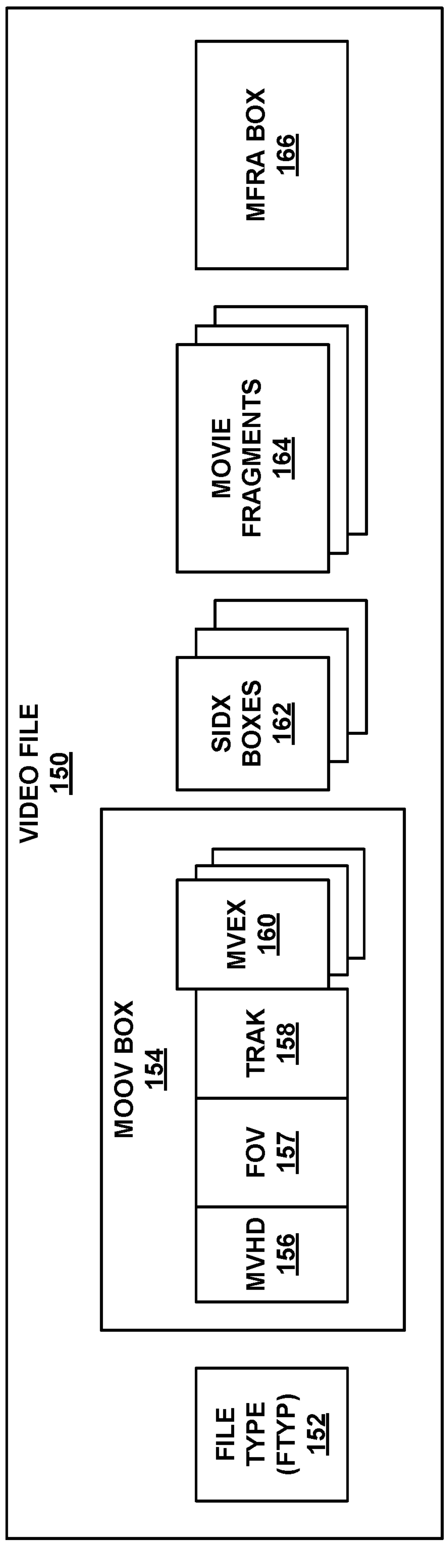

VIDEO FILE 150
FILE TYPE (FTYP) 152
MOOV BOX 154
MVHD 156
FOV 157
TRAK 158
MVEX 160
SIDX BOXES 162
MOVIE FRAGMENTS 164
MFRA BOX 166

FIG. 11

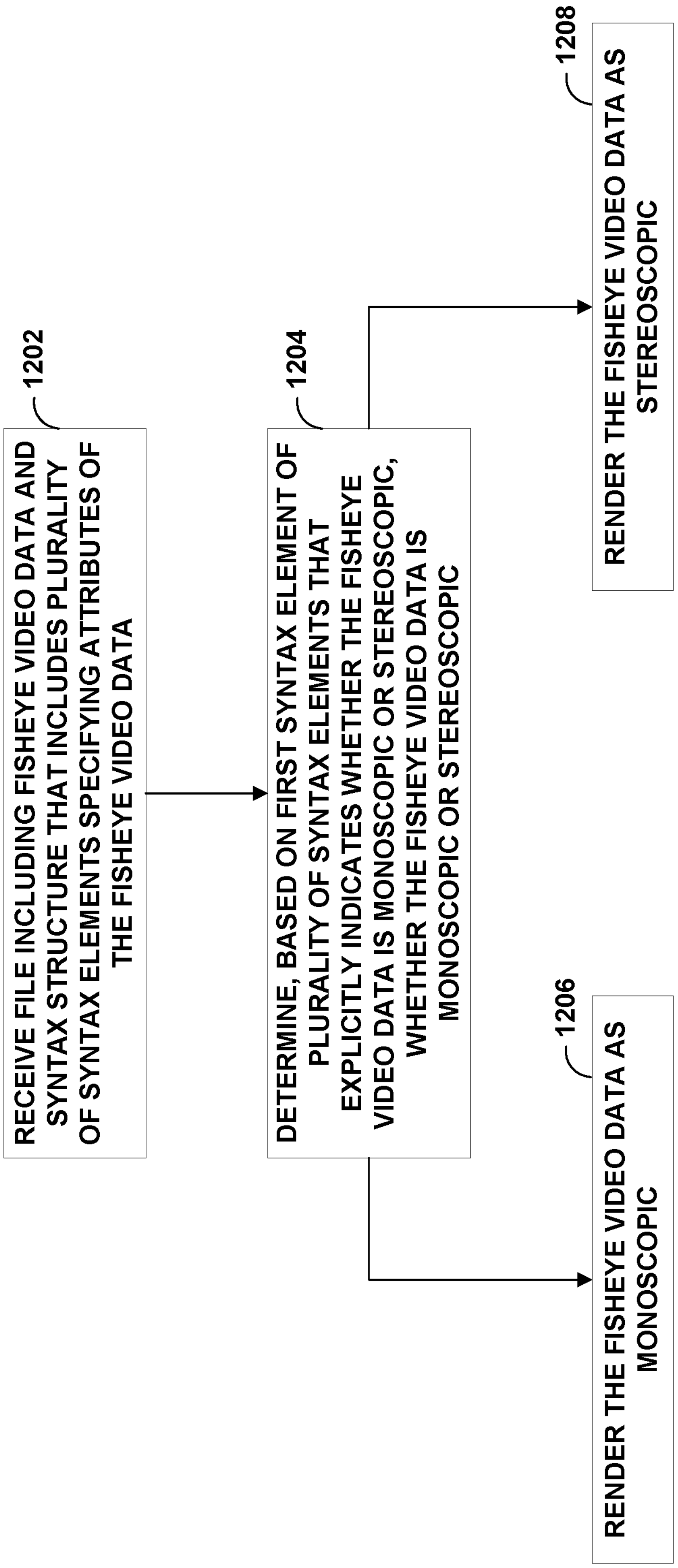

RECEIVE FILE INCLUDING FISHEYE VIDEO DATA AND SYNTAX STRUCTURE THAT INCLUDES PLURALITY OF SYNTAX ELEMENTS SPECIFYING ATTRIBUTES OF THE FISHEYE VIDEO DATA
1202
DETERMINE, BASED ON FIRST SYNTAX ELEMENT OF PLURALITY OF SYNTAX ELEMENTS THAT EXPLICITLY INDICATES WHETHER THE FISHEYE VIDEO DATA IS MONOSCOPIC OR STEREOSCOPIC, WHETHER THE FISHEYE VIDEO DATA IS MONOSCOPIC OR STEREOSCOPIC
1204
RENDER THE FISHEYE VIDEO DATA AS MONOSCOPIC
1206
RENDER THE FISHEYE VIDEO DATA AS STEREOSCOPIC
1208

FIG. 12

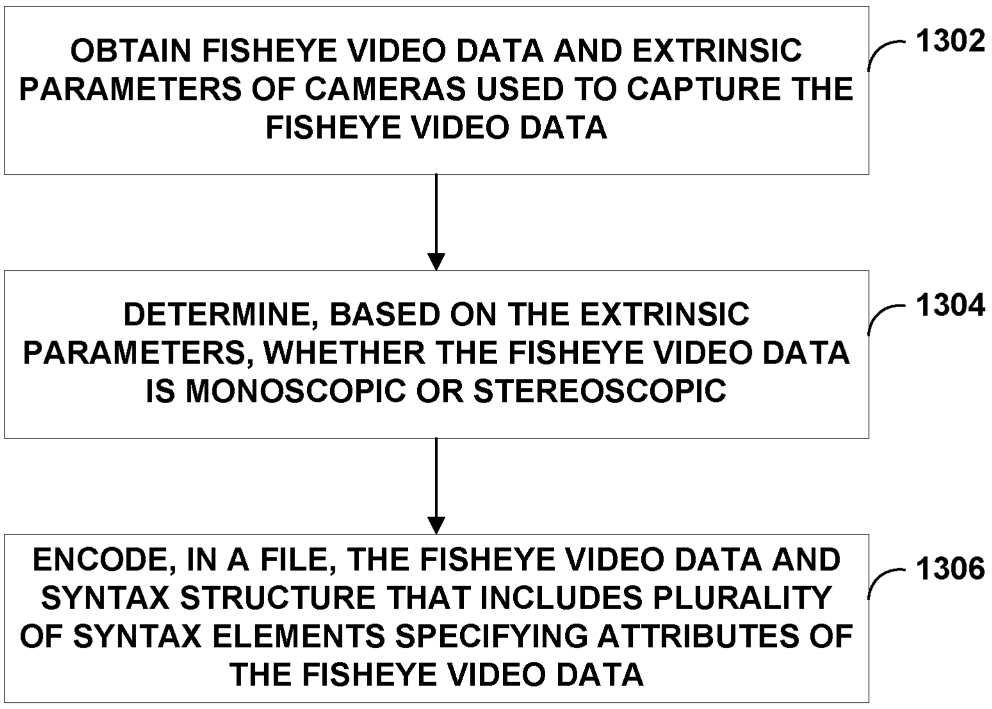

OBTAIN FISHEYE VIDEO DATA AND EXTRINSIC PARAMETERS OF CAMERAS USED TO CAPTURE THE FISHEYE VIDEO DATA
1302
DETERMINE, BASED ON THE EXTRINSIC PARAMETERS, WHETHER THE FISHEYE VIDEO DATA IS MONOSCOPIC OR STEREOSCOPIC
1304
ENCODE, IN A FILE, THE FISHEYE VIDEO DATA AND SYNTAX STRUCTURE THAT INCLUDES PLURALITY OF SYNTAX ELEMENTS SPECIFYING ATTRIBUTES OF THE FISHEYE VIDEO DATA
1306

FIG. 13

INTERNATIONAL SEARCH REPORT

International application No
PCT/US2018/034435

A. CLASSIFICATION OF SUBJECT MATTER
INV. H04N21/235 H04N21/81 H04N21/854
ADD. H04N21/236 H04N21/845

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

B. FIELDS SEARCHED

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

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

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

EPO-Internal, WPI Data

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	"Text of ISO/IEC DIS 23090-2 Omnidirectional MediA Format", 118. MPEG MEETING;3-4-2017 - 7-4-2017; HOBART; (MOTION PICTURE EXPERT GROUP OR ISO/IEC JTC1/SC29/WG11),, no. N16824, 28 April 2017 (2017-04-28), XP030023490, page 9 page 14 page 16 page 20 - page 26 page 28 page 49 page 32 - page 35 -----	1-38

-/--

[X] Further documents are listed in the continuation of Box C. [X] See patent family annex.

* Special categories of cited documents :

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

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

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

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

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

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

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

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

"&" document member of the same patent family

Date of the actual completion of the international search 18 July 2018	Date of mailing of the international search report 25/07/2018
Name and mailing address of the ISA/ European Patent Office, P.B. 5818 Patentlaan 2 NL - 2280 HV Rijswijk Tel. (+31-70) 340-2040, Fax: (+31-70) 340-3016	Authorized officer Lefol, Damien

1

INTERNATIONAL SEARCH REPORT

International application No
PCT/US2018/034435

C(Continuation). DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X,P	Y-K WANG (QUALCOMM): "[OMAF] Comments on OMAF DIS text", 119. MPEG MEETING; 17-7-2017 - 21-7-2017; TORINO; (MOTION PICTURE EXPERT GROUP OR ISO/IEC JTC1/SC29/WG11),, no. m40783, 26 May 2017 (2017-05-26), XP030069127, page 2 -----	1-38
X	HANNUKSELA M M ET AL: "OMAF signaling in DASH", 118. MPEG MEETING; 3-4-2017 - 7-4-2017; HOBART; (MOTION PICTURE EXPERT GROUP OR ISO/IEC JTC1/SC29/WG11),, no. m40468, 28 March 2017 (2017-03-28), XP030068813, section 3; page 2 - page 4 -----	1-38
A	US 2015/358539 A1 (CATT JACOB [US]) 10 December 2015 (2015-12-10) paragraph [0041]; figure 4 -----	1-38

1

INTERNATIONAL SEARCH REPORT

Information on patent family members

International application No

PCT/US2018/034435

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
US 2015358539 A1	10-12-2015	NONE	