(12) UK Patent Application (19) GB (11) 2620584 (13) A

(43) Date of A Publication 17.01.2024

(21) Application No: 2210191.9

(22) Date of Filing: 11.07.2022

(71) Applicant(s):
Canon Kabushiki Kaisha
30-2, Shimomaruko 3-chome, Ohta-ku,
Tokyo 146-8501, Japan

(72) Inventor(s):
Hervé Ruellan
Franck Denoual
Frédéric Maze
Naël Ouedraogo

(74) Agent and/or Address for Service:
Santarelli GB AFS at Canon Europe Limited
4 Roundwood Avenue, Stockley Park, UXBRIDGE,
UB11 1AF, United Kingdom

(51) INT CL:
H04N 21/845 (2011.01) ***G06F 16/41*** (2019.01)
G06F 16/783 (2019.01)

(56) Documents Cited:
GB 2596325 A **US 20090024619 A1**

(58) Field of Search:
INT CL **G06F, G06T, H04N**
Other: **WPI, EPODOC**

(54) Title of the Invention: **Method, Device, and computer program for extrapolation regions in region tracks**
Abstract Title: **Encapsulating geometry of regions in video using extrapolation**

(57) A method of encapsulating video data in a media file, the video data comprising samples and regions being identified across successive samples, a video track 100 is generated comprising successive samples, and a region track 750 is generated comprising samples describing the identified regions. In a region track sample which is synchronised with a video sample where a region is starting geometry of the region is provided, either in the same sample or a different sample information on the movement of the region and indication that the region is extrapolated is provided. A method of reading a media file encapsulated in this way is also provided. There can be an identifier which identifies same regions in the region track samples, and there may be a region track sample synchronised with the first video sample where the region no longer appears or the information on the movement has changed.

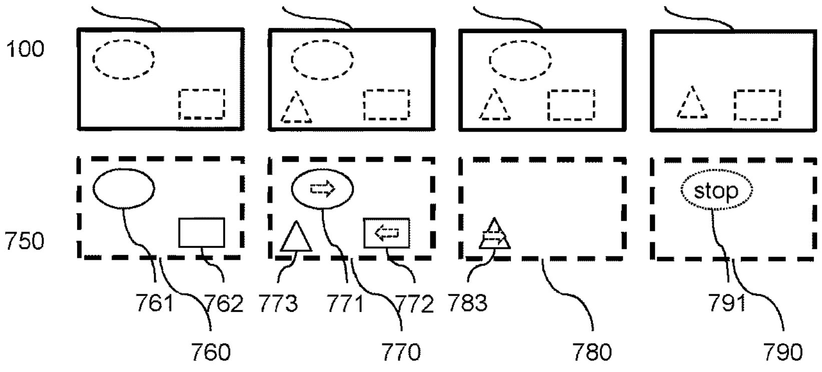

Fig. 7

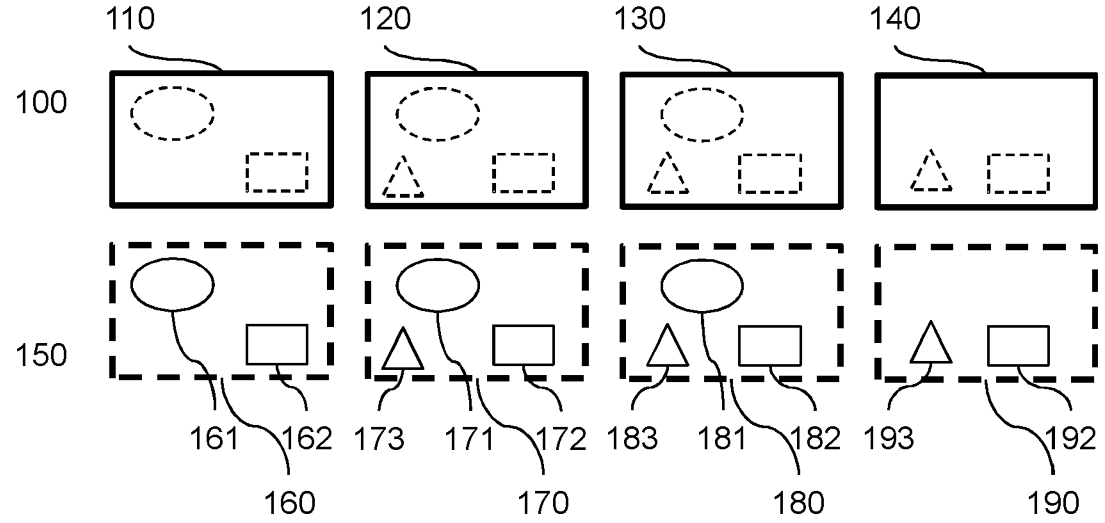
110
120
130
140
100
150
161
162
173
171
172
183
181
182
193
192
160
170
180
190

Fig. 1

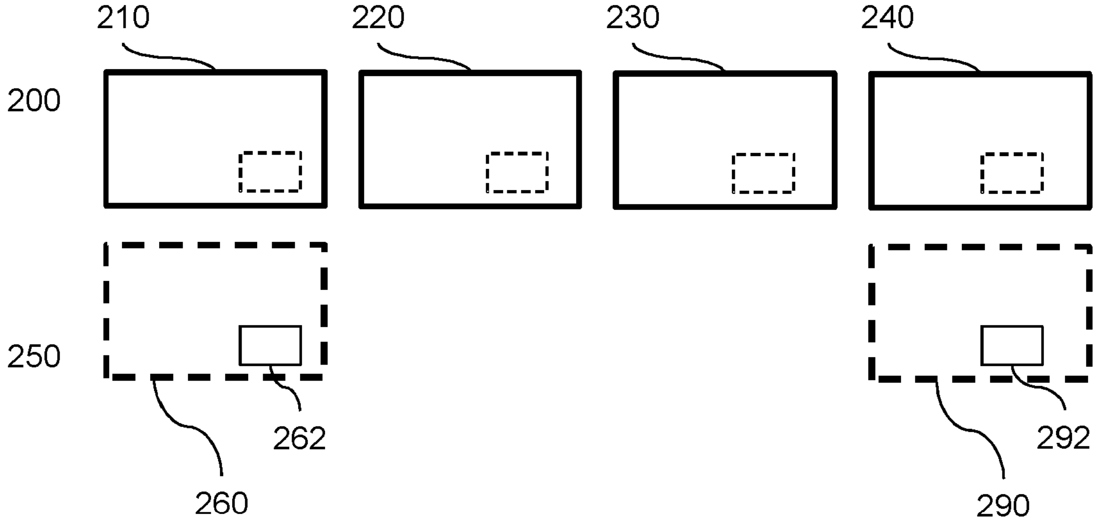
210
220
230
240
200
250
262
292
260
290

Fig. 2

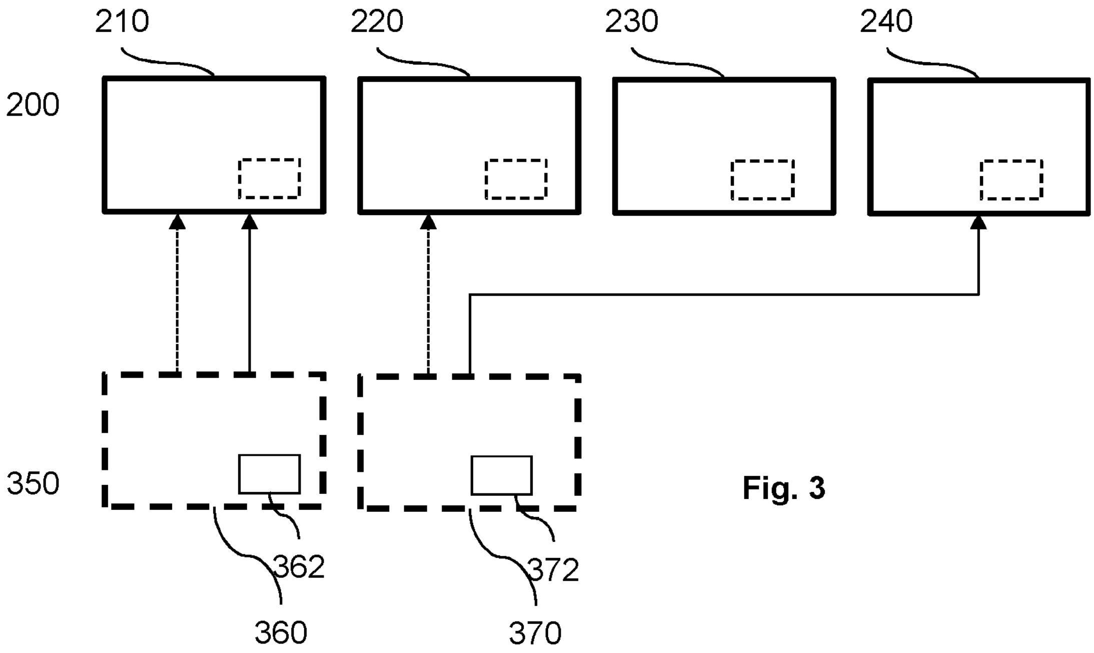
210
220
230
240
200
350
362
372
360
370

Fig. 3

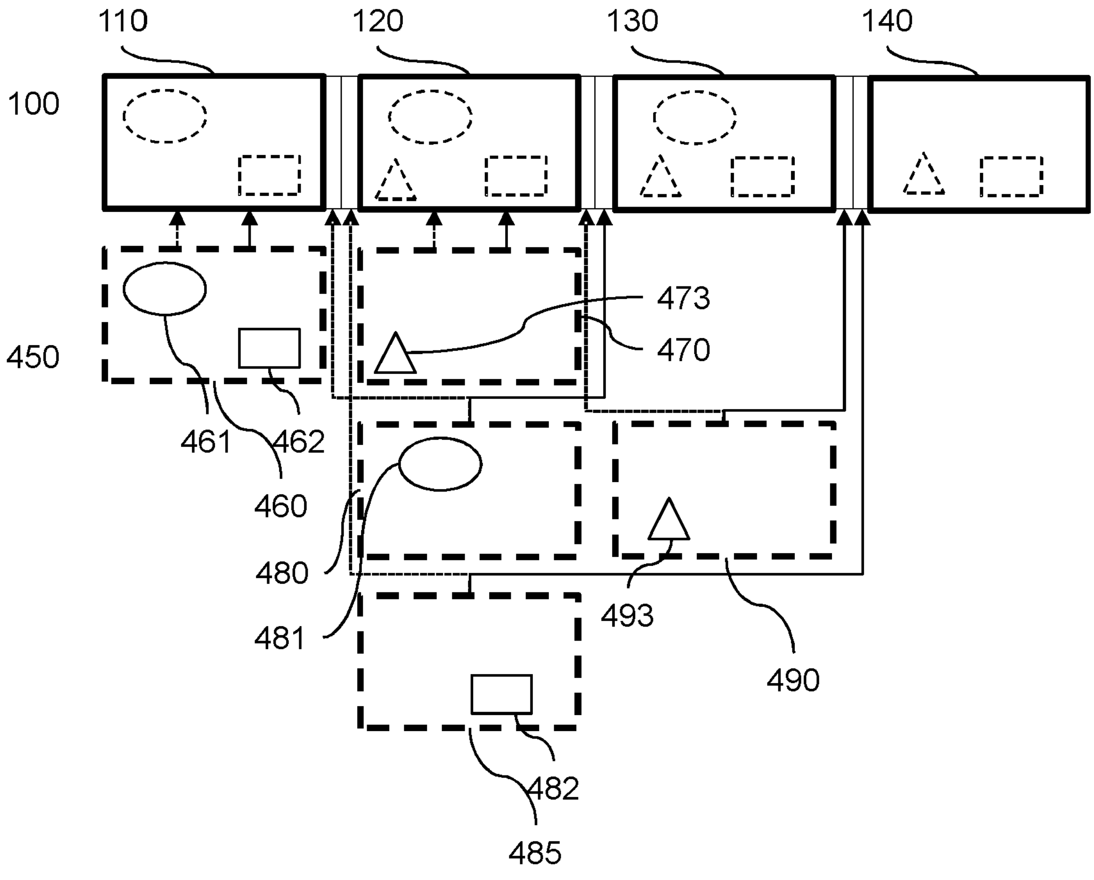
110
120
130
140
100
450
473
470
461
462
460
480
481
493
490
482
485

Fig. 4

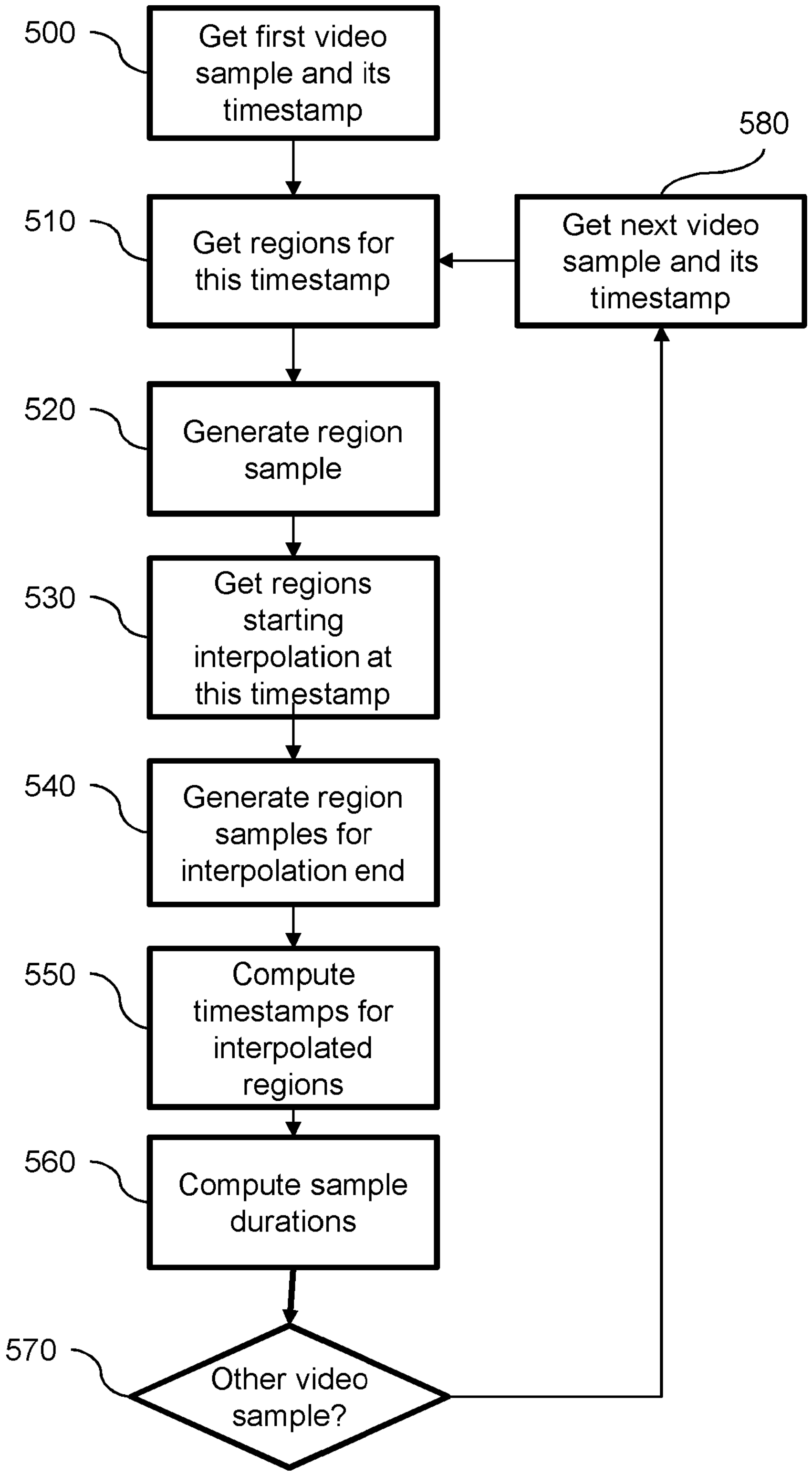
500
Get first video sample and its timestamp
510
Get regions for this timestamp
580
Get next video sample and its timestamp
520
Generate region sample
530
Get regions starting interpolation at this timestamp
540
Generate region samples for interpolation end
550
Compute timestamps for interpolated regions
560
Compute sample durations
570
Other video sample?

Fig. 5

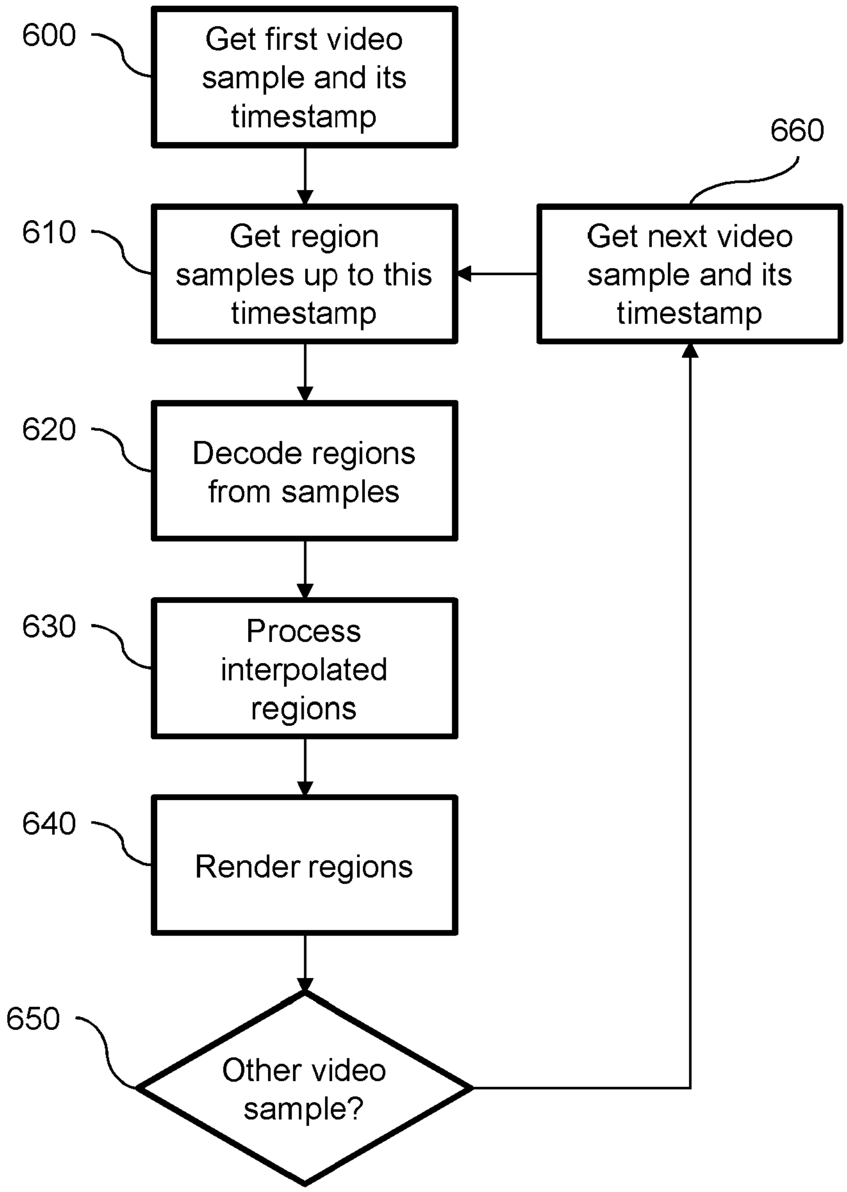
600
Get first video sample and its timestamp
610
Get region samples up to this timestamp
660
Get next video sample and its timestamp
620
Decode regions from samples
630
Process interpolated regions
640
Render regions
650
Other video sample?

Fig. 6

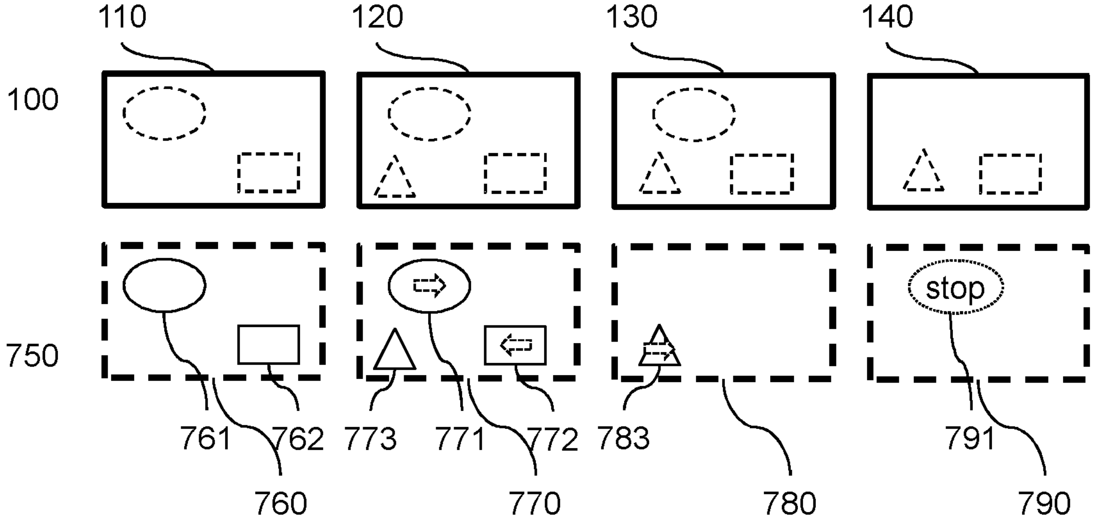
110
120
130
140
100
750
stop
761
762
773
771
772
783
791
760
770
780
790

Fig. 7

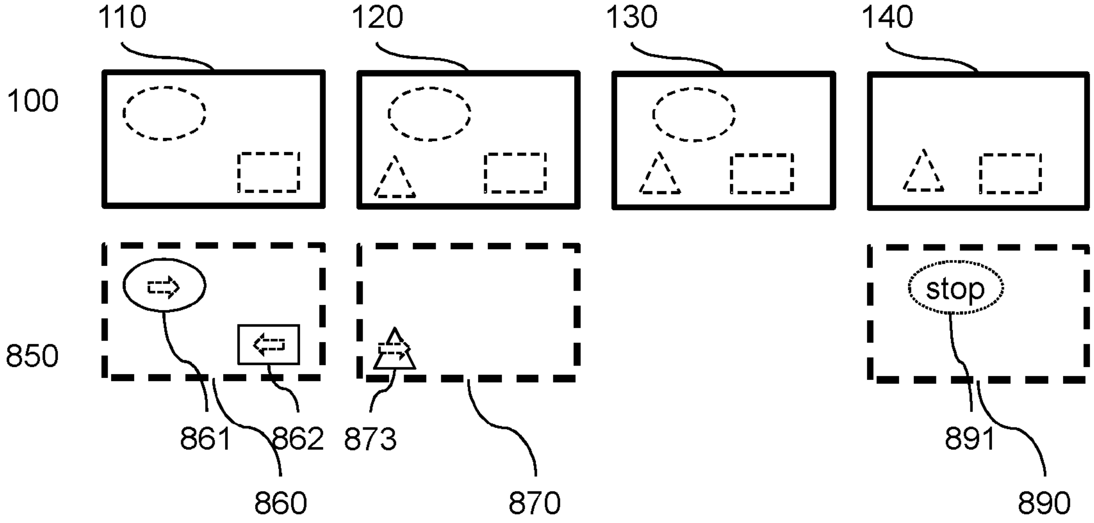
110
120
130
140
100
850
stop
861
862
873
891
860
870
890

Fig. 8

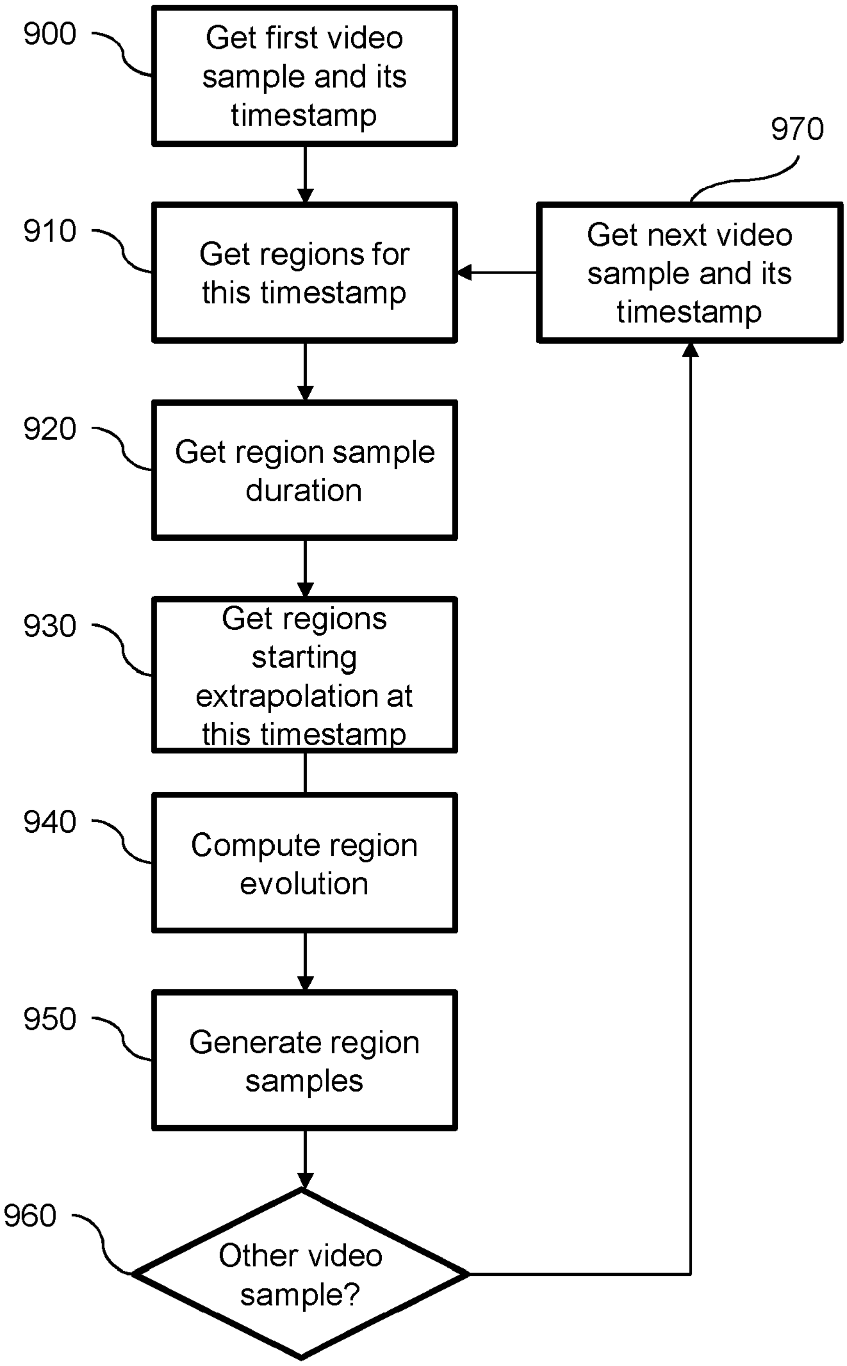
900
Get first video sample and its timestamp
910
Get regions for this timestamp
920
Get region sample duration
930
Get regions starting extrapolation at this timestamp
940
Compute region evolution
950
Generate region samples
960
Other video sample?
970
Get next video sample and its timestamp

Fig. 9

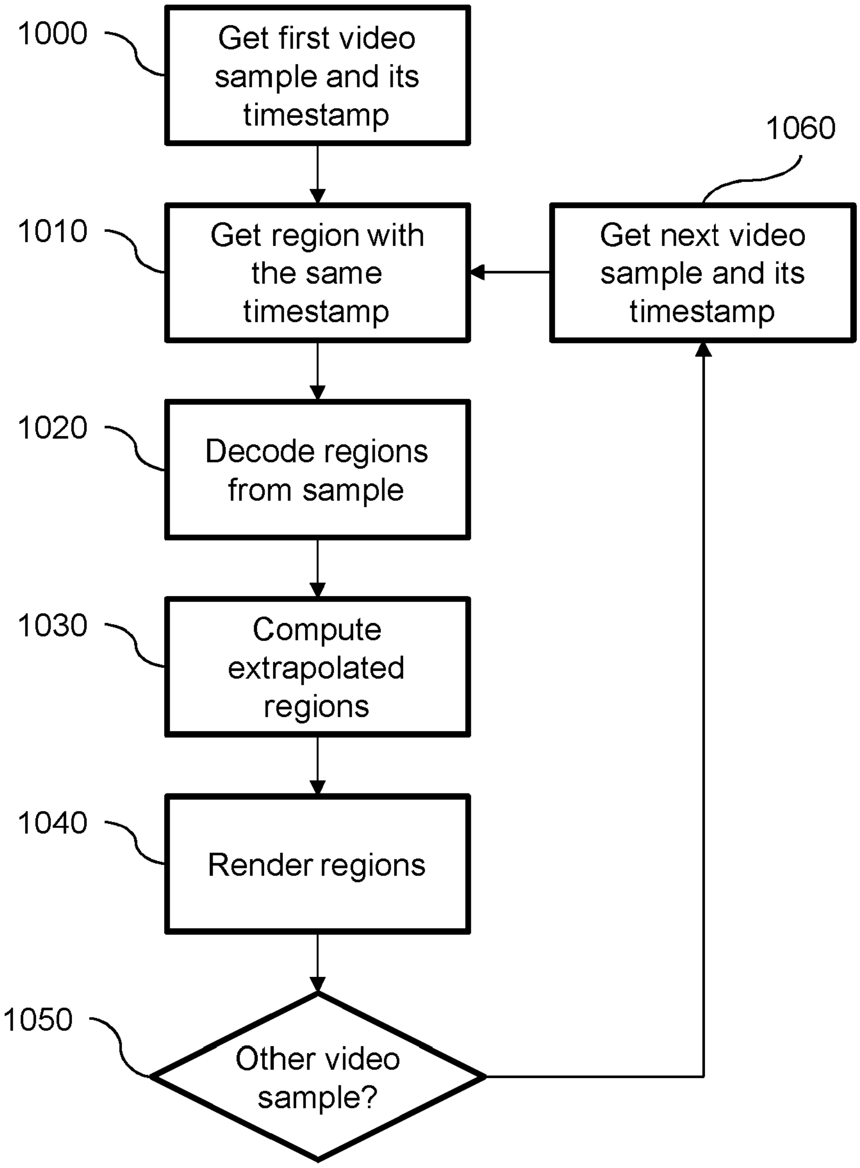
1000
Get first video sample and its timestamp
1010
Get region with the same timestamp
1020
Decode regions from sample
1030
Compute extrapolated regions
1040
Render regions
1050
Other video sample?
1060
Get next video sample and its timestamp

Fig. 10

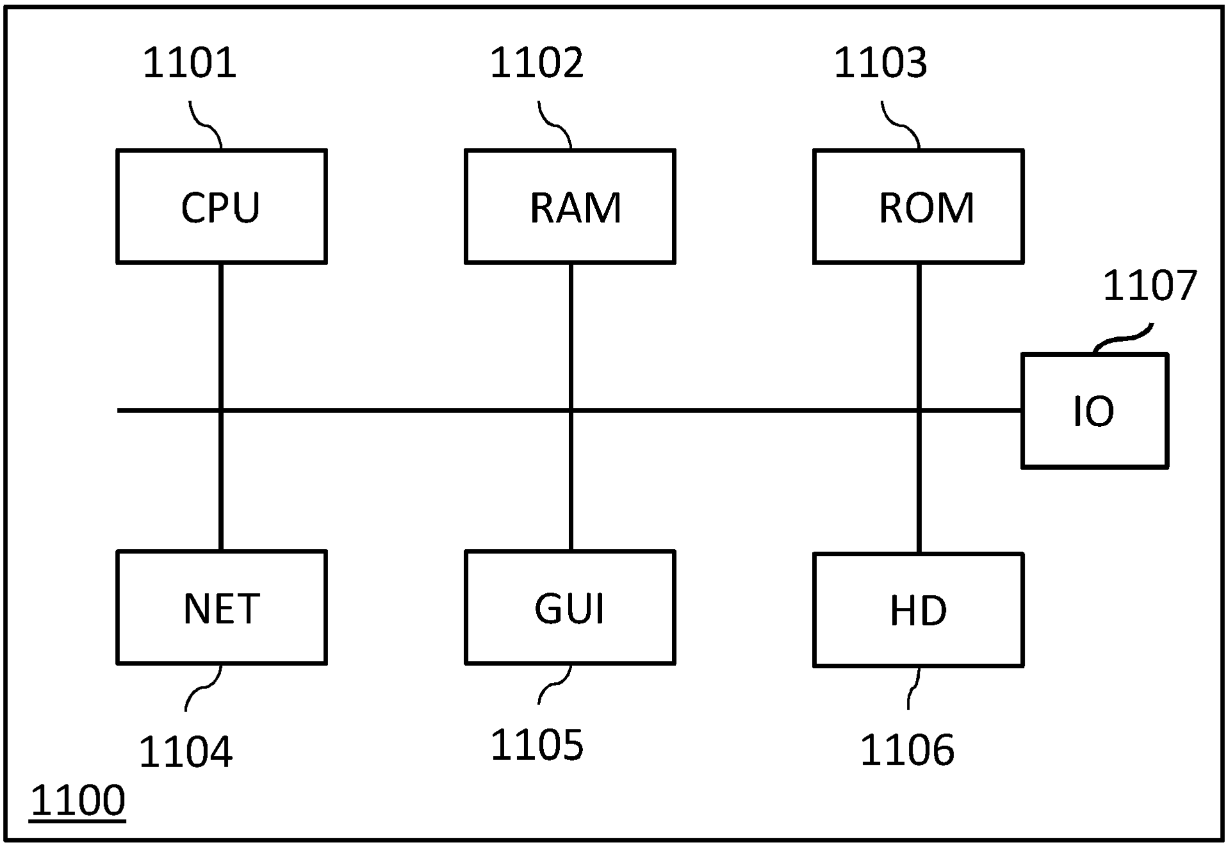
1101
1102
1103
CPU
RAM
ROM
1107
IO
NET
GUI
HD
1104
1105
1106
1100

Fig. 11

METHOD, DEVICE, AND COMPUTER PROGRAM FOR EXTRAPOLATION REGIONS IN REGION TRACKS

FIELD OF THE INVENTION

The present invention relates to a method, a device, and a computer program for extrapolating or interpolating region geometries in region tracks, making it possible to efficiently encode the geometry of regions.

BACKGROUND OF THE INVENTION

Modern cameras and image analysis services enable to generate annotations (e.g., metadata such as human-readable or computer-readable metadata, images, image sequences, video or audio) associated with a portion of, a subpart of, or even an object within the images, image sequences or video. For example, a camera may generate the focusing region for a video or detect faces while recording a video. As another example, a deep-learning system may detect objects inside a sequence of moving images.

Images or videos captured by a camera and/or processed by an image analysis service are stored on a storage device like a memory card for example or transmitted over a communication network. The recorded videos are typically encoded to reduce the size of data on the storage device or over the network and encapsulated in a file format describing the encoded images or videos. After being stored or transmitted through a communication network, the encapsulated images or videos can be processed and possibly rendered by media players. For example, videos may be encapsulated using the ISO Base Media File Format (ISOBMFF) to produce an ISO Base Media file or a set of ISOBMFF segments. For example, images or burst of images may also be stored as items or image sequences using the Image File Format (also denoted HEIF for High Efficiency Image File Format), which is an extension based on the ISO Base Media File Format.

The International Standard Organization Base Media File Format (ISO BMFF, ISO/IEC 14496-12) is a well-known flexible and extensible file format that encapsulates and describes encoded timed or non-timed media data or bitstreams either for local storage or for transmission via a network or via another bitstream delivery mechanism. This file format has several extensions, e.g., Part-15, ISO/IEC 14496-15 that describes encapsulation tools for various NAL (Network Abstraction Layer) unit-based video encoding formats. Examples of such encoding formats are AVC (Advanced Video Coding), SVC (Scalable Video Coding), HEVC (High Efficiency Video Coding), L-HEVC (Layered HEVC), and VVC (Versatile Video Coding). Another example of file format extension is ISO/IEC 23008-12 that describes encapsulation tools for still images or for sequence of still images such as HEVC Still Image. Still another example of file format extension is ISO/IEC 23090-2 that defines the omnidirectional media application format (OMAF). Still other examples of file format extension are ISO/IEC 23090-10 and ISO/IEC 23090-18 that define the carriage of Visual Volumetric Video-based Coding (V3C) media data and Geometry-based Point Cloud Compression (G-PCC) media data.

This file format is object-oriented. It is composed of building blocks called boxes (or data structures, each of which being identified by a four characters code, also denoted FourCC or 4CC). Full boxes are data structures similar to boxes, further comprising a version and flag value attributes. In the following, the term box may designate both full boxes or boxes. These boxes or full boxes are sequentially or hierarchically organized. They define parameters describing the encoded timed or non-timed media data or bitstream, their structure and the associated timing, if any. In the following, it is considered that encapsulated media data designate encapsulated data comprising metadata and media data (the latter designating the bitstream that is encapsulated). All data in an encapsulated media file (media data and metadata describing the media data) is contained in boxes. There is no other data within the file. File-level boxes are boxes that are not contained in another box.

Lately, some proposal for HEIF (ISO/IEC 23008-12) defines a region track for describing regions inside a sequence of images or inside a video track. It also defines tools for associating these regions with annotations. A limitation of the current proposal for HEIF is that a region has to be repeated in each sample of the region track even if it is static or has a uniform linear movement for example.

SUMMARY OF THE INVENTION

The present invention has been devised to address one or more of the foregoing concerns. The proposed encapsulating methods in this document allows the signaling of regions in a region track associated with a video track based on interpolation or extrapolation of the geometry of the region.

According to a first aspect of the invention, it is proposed a method of encapsulating video data in a media file, the video data comprising video samples, at least one region being identified on several samples, the method comprising:

- generating a video track comprising successive video samples;
- generating a region track comprising region track samples for describing regions identified in video samples;

  wherein for describing a region, the method comprises:
- providing in a region track sample, synchronized with a video sample where the region is starting, a geometry of the region in the video sample where the region is starting;
- providing in a region track sample information on the movement of the region, and an indication indicating that the region is extrapolated.

In an embodiment, the geometry of the region in the video sample where the region is starting, the information on the movement of the region and the indication indicating that the region is extrapolated are provided in a same region track sample synchronized with the video sample where the region is starting.

In an embodiment:

- the geometry of the region in the video sample where the region is starting is provided in a first region track sample synchronized with the video sample where the region is starting; and
- the information on the movement of the region and the indication indicating that the region is extrapolated are provided in a second region track sample.

In an embodiment, a same region is identified in the region track samples by a same region identifier.

In an embodiment, the method further comprises:

- generating a region track sample synchronized with the first video sample where the region no longer appears or where the information on the movement of the region has changed.

In an embodiment, the information on the movement describes the movement of the region for a period of time corresponding to the duration of the sample.

According to another aspect of the invention, it is proposed a method of encapsulating video data in a media file, the video data comprising video samples, at least one region being identified on several samples, the method comprising:

- generating a video track comprising successive video samples;
- generating a region track comprising region track samples for describing regions identified in video samples;

  wherein for describing a region, the method comprises:
- providing in a region track sample a starting geometry of the region corresponding to the geometry of the region in a video sample where the region is present;
- generating a second region track sample comprising an ending geometry of the region corresponding to the geometry of the region in a video sample where movement of the region is ending, and an indication indicating that the region is interpolated; and wherein:
- a decoding timestamp of the second region track sample being set lower than or equal to the presentation timestamp of the video sample following the video sample where the region is present; and
- a presentation timestamp of the second region track sample being set to a value that when rounded is equal to the presentation timestamp of the video sample where the region is ending.

In an embodiment, a same region is identified in the region track samples by a same region identifier;

In an embodiment, the value corresponding to the presentation timestamp of the last video sample where the region is ending has any time value between the presentation timestamp of the last video sample where the region is ending, and the presentation timestamp of the previous video sample.

In an embodiment, the starting geometry is provided in a first region track sample synchronized with the first video sample where the region is starting.

In an embodiment, the starting geometry is provided in the second region track sample.

In an embodiment, the second region track sample comprises a duration of the interpolation.

According to another aspect of the invention, it is proposed a method of reading encapsulating video data from a media file, the video data comprising video samples, at least one region being identified on several samples, the method comprising:

- reading a video track comprising successive video samples;
- reading a region track comprising region track samples for describing regions identified in video samples;

  wherein for a region, the method comprises:
- reading in a region track sample, synchronized with a video sample where the region is starting, a geometry of the region in the video sample where the region is starting;
- reading in a region track sample information on the movement of the region, and an indication indicating that the region is extrapolated; and
- extrapolating the region based on the geometry and the information on the movement.

According to another aspect of the invention, it is proposed a method of reading video data from a media file, the video data comprising video samples, at least one region being identified on several samples, the method comprising:

- reading a video track comprising successive video samples;
- reading a region track comprising region track samples for describing regions identified in video samples;

  wherein for a region, the method comprises:
- reading in a region track sample a starting geometry of the region corresponding to the geometry of the region in a video sample where the region is present;
- reading a second region track sample comprising an ending geometry of the region corresponding to the geometry of the region in a video sample where movement of the region is ending, and an indication indicating that the region is interpolated;

- decoding the second region track sample according to a decoding timestamp of the second region track sample being set lower than the presentation timestamp of the video sample following the video sample where the region is present; and
- interpolating the region between the video sample where the region is present and the video sample identified by the presentation timestamp of the second region track sample.

According to another aspect of the invention, it is proposed a computer program product for a programmable apparatus, the computer program product comprising a sequence of instructions for implementing a method according to the invention, when loaded into and executed by the programmable apparatus.

According to another aspect of the invention, it is proposed a computer-readable storage medium storing instructions of a computer program for implementing a method according to the invention.

According to another aspect of the invention, it is proposed a computer program which upon execution causes the method of the invention to be performed.

According to another aspect of the invention, it is proposed a device for encapsulating video data in a media file, the video data comprising video samples, at least one region being identified on several samples, the device comprising a processor configured for:

- generating a video track comprising successive video samples;
- generating a region track comprising region track samples for describing regions identified in video samples;

  wherein for describing a region, the processor is configured for:
- providing in a region track sample, synchronized with a video sample where the region is starting, a geometry of the region in the first video sample where the region is starting;
- providing in a region track sample information on the movement of the region, and an indication indicating that the region is extrapolated.

According to another aspect of the invention, it is proposed a device for encapsulating video data in a media file, the video data comprising video samples, at least one region being identified on several samples, the device comprising a processor configured for:

- generating a video track comprising successive video samples;
- generating a region track comprising region track samples for describing regions identified in video samples;

  wherein for describing a region, the processor is configured for:
- providing in a region track sample a starting geometry of the region corresponding to the geometry of the region in a video sample where the region is present;
- generating a second region track sample comprising an ending geometry of the region corresponding to the geometry of the region in a video sample where movement of the region is ending, and an indication indicating that the region is interpolated; and wherein:
- a decoding timestamp of the second region track sample being set lower than the presentation timestamp of the video sample following the video sample where the region is present; and
- a presentation timestamp of the second region track sample being set to a value corresponding to the presentation timestamp of the video sample where the region is ending.

According to another aspect of the invention, it is proposed a device for reading encapsulating video data from a media file, the video data comprising video samples, at least one region being identified on several samples, the device comprising a processor configured for:

- reading a video track comprising successive video samples;
- reading a region track comprising region track samples for describing regions identified in video samples;

  wherein for a region, the processor is configured for:
- reading in a region track sample, synchronized with a video sample where the region is starting, a geometry of the region in the first video sample where the region is starting;
- reading in a region track sample information on the movement of the region, and an indication indicating that the region is extrapolated; and

- extrapolating the region based on the geometry and the information on the movement.

According to another aspect of the invention, it is proposed a device for reading video data from a media file, the video data comprising video samples, at least one region being identified on several samples, the device comprising a processor configured for:

- reading a video track comprising successive video samples;
- reading a region track comprising region track samples for describing regions identified in video samples;

  wherein for a region, the processor is configured for:
- reading in a region track sample a starting geometry of the region corresponding to the geometry of the region in a video sample where the region is present;
- reading a second region track sample comprising an ending geometry of the region corresponding to the geometry of the region in a video sample where movement of the region is ending, and an indication indicating that the region is interpolated;
- decoding the second region track sample according to a decoding timestamp of the second region track sample being set lower than the presentation timestamp of the video sample following the video sample where the region is present; and
- interpolating the region between the video sample where the region is present and the video sample identified by the presentation timestamp of the second region track sample.

At least parts of the methods according to the invention may be computer implemented. Accordingly, the present invention may take the form of an entirely hardware embodiment, an entirely software embodiment (including firmware, resident software, micro-code, etc.) or an embodiment combining software and hardware aspects that may all generally be referred to herein as a "circuit", "module" or "system". Furthermore, the present invention may take the form of a computer program product embodied in any tangible medium of expression having computer usable program code embodied in the medium.

Since the present invention can be implemented in software, the present invention can be embodied as computer readable code for provision to a programmable apparatus on any suitable carrier medium. A tangible, non-transitory carrier medium may comprise a storage medium such as a floppy disk, a CD-ROM, a hard disk drive, a magnetic tape device or a solid state memory device and the like. A transient carrier medium may include a signal such as an electrical signal, an electronic signal, an optical signal, an acoustic signal, a magnetic signal or an electromagnetic signal, e.g. a microwave or RF signal.

BRIEF DESCRIPTION OF THE DRAWINGS

Embodiments of the invention will now be described, by way of example only, and with reference to the following drawings in which:

Figure 1 illustrates an example of a video track and an associated region track describing regions of this video track;

Figure 2 illustrates an example of describing a moving region using interpolation as specified in OMAF;

Figure 3 illustrates an example of describing a moving region using interpolation;

Figure 4 illustrates an example of describing several moving regions using interpolation;

Figure 5 illustrates steps for generating a region track with interpolated regions according to the invention;

Figure 6 illustrates steps for parsing and processing a region track with interpolated regions according to the invention;

Figure 7 illustrates an example of describing several regions using extrapolation;

Figure 8 illustrates another example of describing several regions using extrapolation;

Figure 9 illustrates steps for generating a region track with extrapolated regions according to the invention;

Figure 10 illustrates steps for parsing and processing a region track with extrapolated regions according to the invention;

Figure 11 is a schematic block diagram of a computing device for implementation of one or more embodiments of the invention.

DETAILED DESCRIPTION OF THE INVENTION

In all the description the term image, sample, or frame is used as generic terms encompassing both 2D, 3D or volumetric images or frames. Similarly, the term video is used as a generic term encompassing both 2D videos and 3D videos or volumetric media. Volumetric media may encompass media encoded with G-PCC, V3C or V-PCC.

According to the ISO Base Media file format and other ISOBMFF-based specifications, a presentation of timed media data is described in a File-level box called a movie box (with the four-character code `'moov'`). This movie box represents an initialization information container comprising a set of various boxes describing the presentation and its timing. It is logically divided into tracks represented by track boxes (with the four-character code `'trak'`). Each track, uniquely identified by a track identifier `track_ID`, represents a timed sequence of media data belonging to the presentation (for example, a sequence of video frames, a sequence of subparts of video frames, a sequence of audio samples or a sequence of timed metadata samples). Unlike frames of a video, a sequence of images designates a timed sequence of images for which the timing is only advisory; it may be the timing at collection (e.g., of an image burst) or the suggested display timing (e.g., for a slideshow).

Within each track, each timed unit of data is called a sample; this might be a frame or a subpart of a frame of video, audio or timed metadata, or an image of a sequence of images. A sample is defined as all the media data associated with a same presentation time in a track. Samples are implicitly numbered in decoding order. Each track box comprises a hierarchy of boxes describing the samples of a track, e.g., a sample table box (`'stbl'`) includes all the time and data indexing of the media samples in a track. In addition, the sample table box comprises a sample description box, identified with the four characters code `'stsd'`, including a set of sample entries, each sample entry giving the required information about the coding configuration, including a coding type identifying the coding format and various coding parameters characterizing the coding format, of media data in a sample, and any initialization information needed for decoding the sample. The actual sample data is stored in boxes called media data boxes, with the four-character code `'mdat'`, or identified media data boxes, with the four-character code `'imda'`, similar to the media data box but containing an additional identifier, at the same level as the movie box.

The movie may also be fragmented, i.e., organized temporally as a movie box containing information for the whole presentation followed by a list of movie fragments, i.e., a list of couples comprising a movie fragment box, identified with the four-character code `'moof'`, and a media data box (`'mdat'`) or a list of couples comprising a movie fragment box (`'moof'`) and an identified media data box (`'imda'`).

Non-timed media data is described in a meta box (with the four-character code `'meta'`). A unit of non-timed media data under this box and its hierarchy relates to “information item” or “item” instead of related samples. It is to be noted that the wording ‘box’ and the wording ‘container’ may be both used with the same meaning to refer to data structures that contain metadata describing the organization or/and properties of the image data in the file.

An ISOBMFF-compliant file may comprise at file-level both a movie box and a meta box or any one of them only.

Figure 1 illustrates an example of a video track and an associated region track describing regions of this video track as it may be represented according to a current proposal for HEIF.

In this figure, like in similar figures 2 – 4, and 7 – 8, a video track with some video samples is illustrated using plain line boxes, one box illustrating a sample, while the associated region track comprising region track samples for describing regions in the video sample of the associated video track is represented with dotted line boxes representing the samples of the region track. Regions in the video samples are represented as dotted lines shapes, for example and ellipse, a triangle, or a rectangle. The corresponding descriptions in the associated region track are represented by plain line shapes.

The video track 100 contains four samples 110, 120, 130 and 140 encoding the content of the video at different time instants. This video contains several regions of interest (RoI) depicted with dashed shapes. These RoIs appear and disappear at different time instants and move from one video frame to another.

The region track 150 is associated with the video track 100 and describes regions inside samples of this video track by signalling their shapes, positions and sizes inside the samples of the region track. Here, the region track 150 contains four samples 160, 170, 180, 190. Sample 160 contains two regions, an elliptic region 161 and a rectangular region 162. These two regions apply to the image encoded by sample 110. Sample 170 contains three regions, the elliptic region 171, the rectangular region 172 and a new triangular region 173. The positions of the elliptic region 171 is different from the position of the elliptic region 161: it has moved toward the right side of the image. Similarly, the rectangular region 172 has moved toward the left side of the image compared to the rectangular region 162. The three regions apply to the image encoded by sample 120. Similarly, sample 180 contains three regions 181, 182 and 183 applying to the image encoded by sample 130. All three regions have moved compared to the regions with the same shape in sample 170. Sample 190 contains only two regions, a rectangular region 192 and a triangular region 193, applying to the image encoded by sample 140.

It should be noted here that current version of HEIF and known proposals do not provide tools for following a region through different samples. The regions described in the region track are defined on a sample basis. The scope of the region identifier is the sample, meaning that the same region has no reason to get the same region identifier in two different samples. A full description of the region is provided in a region track sample for each sample comprising this region in the video track.

Figure 2 illustrates an example of describing a moving region using interpolation as it may be represented according to OMAF. OMAF is an extension of ISOBMFF for the encapsulation of omnidirectional media data, which provides an interpolation mechanism for describing regions. For aligning the example of Figure 2 with the other examples, the images and the regions in the example are represented as rectangles, whereas OMAF defines omnidirectional media applications and allows to define regions inside a sphere.

In this example, the video track 200 containing four samples 210, 220, 230 and 240. This video track contains one RoI. The metadata track 250 describes the position of this RoI as a sphere region. The metadata track 250 contains two samples 260 and 290, corresponding respectively to the samples 210 and 240 of the video track. The correspondence between the samples of the metadata track and the samples of the video track is defined by their respective composition timestamp. A sample of the metadata track corresponds to the sample of the video track with the same composition timestamp. Sample 260 defines the position of the RoI inside the image encoded by sample 210 of the video track by signalling a sphere region 262. Sample 290 defines the position of the RoI inside the image encoded by sample 240 of the video track by signalling a sphere region 292. In addition, the sphere region 292 contains an `interpolate` flag set to 1, indicating that the position and size of the sphere region may be computed for the images encoded by sample 220 and 230 by interpolating it between its position and size as defined in sample 260 and its position and size as defined in sample 290.

OMAF allows to define a single sphere regions in a sphere region sample. The interpolate flag set to 1 indicates that the sphere region can be interpolated between its position and size in the previous sample and its position and size in the sample containing the interpolate flag.

OMAF uses the composition timestamp for associating samples of the region track to samples of the video track. When using interpolation, there may be a large gap between the sample describing the start of the interpolation and the sample describing the end of the interpolation. The OMAF file doesn't provide any hint that the sample describing the end of the interpolation should be decoded and parsed in advance for obtaining the information about the interpolation end and computing the regions using interpolation.

Figure 3 illustrates an example of describing a moving region using interpolation according to embodiments of the invention.

This example uses the same video track 200 as the example of Figure 2.

The region track 350 contains two samples, 360 and 370. The first sample 360 contains the description of the rectangular RoI as a rectangular region 362. This rectangular region 362 is identified by the ID 2 in the sample 360. The sample 360 has a decoding timestamp (shown using a dashed arrow) and a composition timestamp (shown using a plain arrow) equal to the composition timestamp of sample 210 from the video track. The decoding timestamps indicate the decoding order of the samples. The composition timestamps indicate when each sample should be displayed. Composition timestamps allow re-ordering samples when their display order is different from their decoding order. Decoding timestamps assume that the decoding process is instantaneous, which means that the decoding timestamp and the composition timestamp of a sample may be equal.

Sample 370 contains the description of the rectangular RoI as a rectangular region 372 at a time corresponding to the video sample 240. This is indicated by setting the composition timestamp of sample 370 to be equal to the composition timestamp of sample 240. This rectangular region 372 is identified by the ID 2 in the sample 370. In addition, the region 372 has an interpolate flag set to 1, indicating that the position and size of the region are expected to be interpolated between the position and size indicated by sample 360 for the region identified by the same ID, i.e. region 362, and the position and size indicated by sample 370 for region 372.

Moreover, the decoding timestamp of sample 370 is set to the composition timestamp of sample 220 for indicating to the decoder or renderer that sample 370 should be decoded in time for rendering sample 220. This enables the decoder or renderer to know, at the time of rendering sample 220, that the region identified by the ID 2 is expected to be interpolated between its position and size indicated by sample 360 and those indicated by sample 370. In particular, the decoder or renderer knows that for sample 220, the position and size of the region identified by the ID 2 can be computed using interpolation. If the decoding timestamp of sample 370 is set to the same value as the composition timestamp of the sample, then the decoder or renderer may decode sample 370 only at a time corresponding to the time of rendering sample 240. Therefore, it would not know when rendering sample 220 and sample 230 that the region identified by the ID 2 is expected to be interpolated.

Figure 4 illustrates an example of describing several moving regions using interpolation according to embodiments of the invention. This figure expands Figure 3 by using interpolation to describe the regions illustrated by Figure 1.

The region track 450 contains five samples, 460, 470, 480, 485 and 490. The first sample 460 contains the description of two regions, an elliptic region 461 and a rectangular region 462. This sample 460 has a decoding timestamp and a composition timestamp equal to the composition timestamp of sample 110. The second sample 470 contains the description of another region, the triangular region 473. This sample 470 has a decoding timestamp and a composition timestamp equal to the composition timestamp of sample 120.

Samples 480, 485 and 490 describe the ending position for respectively the elliptic region 481, the rectangular region 485 and the triangular region 493.

Following the example illustrated by Figure 3, samples 480 and 485 may have a decoding timestamp equal to the composition timestamp of the sample 120 of the video track and sample 490 may have a decoding timestamp equal to the composition timestamp of sample 130. Sample 480 may have a composition timestamp equal to the composition timestamp of sample 130. Sample 485 and sample 490 may have a composition timestamp equal to the composition timestamp of sample 140.

However, ISOBMFF does not allow two different samples from the same track to have the same decoding timestamp or to have the same composition timestamp. Therefore the example illustrated by Figure 3 cannot be directly applied to this example when using ISOBMFF to store the video track and the region track.

As illustrated by Figure 4, the different decoding timestamps and composition timestamps may be slightly shifted compared to the composition timestamps of samples from the video track.

For instance, the decoding timestamp of sample 485 may be slightly before the decoding timestamp of sample 470. The decoding timestamp of sample 480 may also be slightly before the decoding timestamp of sample 485. The decoding timestamp of sample 490 may be the composition timestamp of sample 130 or may be slightly before it. Similarly, the composition timestamp of sample 480 may be the composition timestamp of sample 130 or may be slightly before it. The composition timestamp of sample 485 may be the composition timestamp of sample 140 or may be slightly before it. The composition timestamp of sample 490 may be slightly before the composition timestamp of sample 485.

Region track samples using interpolation as illustrated by Figure 3 or Figure 4 may be represented using the following structure:

```
    aligned (8) class RegionSample {
     unsigned int field_size =
((RegionTrackConfigBox.field_length_size & 1) + 1) * 16;
     // this is a temporary, non-parsable variable
     unsigned int(32) region_count;
     for (r=0; r < region_count; r++) {
      unsigned int(32) region_identifier;
      unsigned int(8) geometry_type;
      unsigned int(1) interpolate;
      unsigned int(7) reserved;
      }
      if (geometry_type == 0) {
       // point
       signed int(field_size) x;
       signed int(field_size) y;
      }
      else if (geometry_type == 1) {
```

```
      // rectangle
      signed int(field_size) x;
      signed int(field_size) y;
      unsigned int(field_size) width;
      unsigned int(field_size) height;
    }
    else if (geometry_type == 2) {
      // ellipse
      signed int(field_size) x;
      signed int(field_size) y;
      unsigned int(field_size) radius_x;
      unsigned int(field_size) radius_y;
    }
    else if (geometry_type == 3 || geometry_type == 6) {
      // polygon or polyline
      unsigned int(field size) point_count;
      for (i=0; i < point_count; i++) {
        signed int(field_size) px;
        signed int(field_size) py;
      }
    }
    else if (geometry_type == 4) {
      // referenced mask
      signed int(field_size) x;
      signed int(field_size) y;
      unsigned int(field_size) width;
      unsigned int(field_size) height;
      unsigned int(field_size) track_mask_idx;
    }
    else if (geometry_type == 5) {
      // inline mask
      signed int(field_size) x;
      signed int(field_size) y;
      unsigned int(field_size) width;
      unsigned int(field_size) height;
      unsigned int(8) mask_coding_method;
      if (mask_coding_method != 0)
        unsigned int(32) mask_coding_parameters;
      bit(8) data[];
    }
  }
}
```

In this structure, the `interpolate` flag when set to 1 indicates that the region may be interpolated with latest region defined in a previous sample with the same value for the `region_identifier` field, if the two regions have the same `geometry_type`. The interpolation may use the composition time of the previous sample and of the current sample for computing intermediate values for the region.

In a variant, a single `interpolate` flag may be signalled for all the regions described in a sample.

For a point, signalled by the value 0 for the `geometry_type` field, interpolated values may be `x` and `y`. For a rectangle, signalled by the value 1 for the `geometry_type` field, interpolated values may be `x`, `y`, `width` and `height`. For an ellipse, signalled by the value 2 for the `geometry_type` field, interpolated values may be `x`, `y`, `radius_x` and `radius_y`. For a polygon or a polyline, signalled respectively by the values 3 and 6 for the `geometry_type` field, interpolated values are `px` and `py` for each point. Preferably, the number of points in the polygon or polyline doesn't change. Possibly, the interpolation may add or remove points to the polygon or polyline. For a mask, signalled by the value 4 or 5 for the `geometry_type` field, interpolated values may be `x` and `y`. Possibly, the `width` and `height` values may also be interpolated resulting in a scaling of the mask.

Possibly, interpolation may transform one shape into another. For example, a point may become a rectangle, or an ellipse may become a rectangle. This may be realized for example by approximating the initial shape and the final shape with Bezier curves and by interpolating between these curves.

The value V of a characteristic of the geometry of a region at a time T may be computed from the value V_S of this characteristic as defined at the time T_S, corresponding to the interpolation start. and from the value V_E of this characteristic as defined at the time T_E, corresponding to the interpolation end, as follows:

$$V = V_S + (V_E - V_S)\frac{T - T_S}{T_E - T_S}$$

In these embodiments, a region may be described in the region track using only two region track samples, a beginning sample and an ending sample, each comprising a description of the region in, respectively, a first and a last video sample where the region is present. The descriptions of the region in the beginning sample and in the ending sample are associated by using the same region identifier in both descriptions. A flag is set in the region description in the ending sample for indicating that the region can be interpolated between the beginning sample and the ending sample. The decoding timestamp of the ending sample is set to a value lower than or equal to the composition timestamp of the video sample following the video sample corresponding to the beginning region sample. As such, the decoded region description is available for interpolation and can be taken into account at the time of presentation of this video sample, and in addition at the time of presentation of the following video samples up to and including the last video sample. The composition timestamp of the ending sample is set to a value as close as possible to the value of the composition timestamp of the last video sample. Preferably, the composition timestamp of this ending sample is strictly greater than the composition timestamp of the video sample preceding the last video sample and strictly lower than the composition timestamp of the video sample following the last video sample. Preferably, the composition timestamp of this ending sample is closer to the composition timestamp of the last video sample than to the composition timestamp of the video sample preceding the last video sample and closer to the composition timestamp of the last video sample than to the composition timestamp of the video sample following the last video sample. In other words, the composition timestamp of this ending sample is closer to the composition timestamp of the last video sample than to the composition timestamp of any other video sample. This means that the composition timestamp of this ending sample is contained in an interval corresponding to the duration of one frame of the video track centered around the composition timestamp of the last video sample.

In a preferred variant, the composition timestamp of the ending sample is lower than or equal to the composition timestamp of the last video sample.

In another variant, the composition timestamp of the ending sample is greater than or equal to the composition timestamp of the last video sample.

In yet another variant, the composition timestamp of the ending sample is selected among the closest available composition timestamps from the composition timestamp of the last video sample.

Figure 5 illustrates the main steps of a method for generating a region track with interpolated regions according to embodiments of the invention.

These steps may be used for generating a region track associated with a whole video track or for generating a part of the region track associated with a part of a video track. A part of the video track may for example be the portion of the video track included in a media fragment. In the following, the terms video track are used for representing either a whole video track or a part of a video track in a media fragment.

These steps take as input the video track and a list of regions associated with this video track. This list may contain regions occurring at a specific timestamp in the video track. This list may contain regions that may be interpolated between two specific timestamps in the video track.

In a first step 500, the processed video sample is set to the first video sample of the video track. In addition, the composition timestamp of this first video sample is retrieved. Last, the composition timestamp of the next video sample is also retrieved. Note that in these steps, all the video samples are considered ordered according to their composition timestamps.

At this step, a list of used identifiers is initialized to the empty list.

Then at step 510, the regions corresponding to the composition timestamp of the processed video sample are retrieved with their characteristics. At this step, only the regions associated to the processed video sample and that are described without using interpolation or that are described using interpolation starting at the composition timestamp of the processed video sample are retrieved. This means more especially that regions that are described using interpolation with an interpolation starting before the composition timestamp of the processed video sample and ending after the composition timestamp of the processed video sample are not retrieved at this step. The characteristics of the retrieved regions may include the type of the geometry of a region, its position, its size, and any further information that may be included in the region description.

If no region is retrieved at this step, then the next step is step 560.

Possibly, at this step 510, the interpolated regions whose interpolation stopped at a timestamp strictly before the composition timestamp of the processed video sample and that have not been retrieved during this step are obtained. The identifiers associated to these regions are removed from the used identifiers list. They are therefore available to be associated with new regions if needed.

Possibly, at step 510, regions for which a first interpolation ends at the composition timestamp of the processed video sample and a second interpolation begins at the composition timestamp of the processed video sample are obtained. The identifiers for these regions are not removed from the used identifiers list. These regions are not processed at step 520. They are added to the regions retrieved at step 530. Their identifier is memorized for being used at step 540.

At step 520, a sample for describing the regions retrieved at step 510 is generated. The content of this sample may be encoded, for example, according to the structure described here-above. An identifier is associated with each of these retrieved regions. This identifier is selected such as not being present in the list of used identifiers. This identifier is added to the list of used identifiers. This identifier is also associated to the retrieved region to be reused at step 540 if necessary. This identifier is encoded in the `region_identifier` field corresponding to the representation of the region inside the encoding of the sample. For each region, the `interpolate` field corresponding to the region inside the encoding of the sample is set to 0. The composition timestamp of this sample is set to the value of the composition timestamp of the processed video sample retrieved at step 500. The decoding timestamp of this sample is set to the same value.

At step 530, the regions that are described using interpolation with an interpolation starting at the composition timestamp of the processed video sample are retrieved. These regions may be retrieved from the regions retrieved at step 510.

If no region is retrieved at this step, the processing may continue directly at step 560.

At step 540, for each region retrieved at step 530, a sample describing the interpolation end for this region is generated. The content of this region track sample may be encoded according to the structure described here-above. The same identifier as the one selected at step 520 is used for the interpolation end of the region. This identifier is encoded in the `region_identifier` field corresponding to the representation of the region inside the encoding of the sample. This identifier is added to the list of used identifiers. The `interpolate` field corresponding to the representation of the region inside the encoding of the region track sample is set to 1. Possibly, as an optimization, a single region track sample may be created for representing all the regions whose interpolation ends at the same time.

At step 550, the decoding and composition timestamps associated to the region track samples created at step 540 are computed.

The decoding timestamps are selected as strictly greater than the composition timestamp of the processed video sample and strictly smaller than the composition timestamp of the next video sample and as being all different. For instance, the region track samples are processed one after another. The decoding timestamps of the first region track sample may be computed as the decoding timestamp of the region track sample generated at step 520 increased by a small amount of time. Then, the decoding timestamp of each following region track sample may be computed as the decoding timestamp of the previous region track sample increased by a small amount of time. Preferably, the small amount of time may be constant for all the region track samples. Preferably, the small amount of time may be set to 1 time unit according to the timescale of the region track.

The composition timestamps are selected as strictly smaller than the ending time of the interpolation. This ending time usually is the composition timestamps of a video sample. The composition timestamps are selected to all be different. For instance, all the region track samples are processed one after another. For each timestamp, the ending time of the interpolation is retrieved. The number of region track samples already encoded and associated with this ending time is retrieved. The composition timestamp of the region track sample is computed as the ending time of the interpolation decreased by a small amount of time multiplied by this number of associated region track samples increased by one. Preferably, the small amount of time may be set to a constant value. Preferably, the small amount of time may be set to 1 time unit according to the timescale of the region track. The number of region track samples already encoded and associated with the ending time of the interpolation is increased by one.

In a variant, the composition timestamp of the region track sample is computed as the ending time of the interpolation increased by a small amount of time multiplied by this number of associated region track samples increased by one.

In another variant, the composition timestamp of the region track sample is computed as the ending time of the interpolation decreased by a small amount of time multiplied by half this number of associated region track samples increased by one if this number is even and increased by a small amount of time multiplied by half this number plus one of associated region track samples if this number is odd.

At step 560, the duration of all the region track samples generated at step 520 and step 540 are computed from the decoding timestamps computed previously. This is realized by ordering the region track samples according to their decoding timestamps and by setting the duration of each region track sample to the difference between the decoding timestamp of the next region track sample and the decoding timestamp of the considered region track sample. Setting the duration of the last region track sample generated at step 520 or step 540 is deferred. In addition, if this step has already been processed and setting the duration of the previous last region track sample was deferred during the previous processing of this step 560, then the duration of this previous last region track sample is set as the difference between the decoding timestamp of the first region track sample generated at step 520 and step 540 and the decoding timestamp of this previous last region track sample.

At step 570, it is checked whether there are more video samples to process.

If this is the case, then at step 580, the next video sample in composition timestamp order is set as the processed video sample. In addition, the composition timestamp of this next video sample is retrieved. Last, the composition timestamp of the video sample following this next video sample is also retrieved.

Note that at step 500 or at step 580, if the processed video sample is the last one, the composition timestamp of the video sample following this processed video sample cannot be retrieved. However, this composition timestamp is used for computing timestamps of region track samples corresponding to region whose interpolation starts with the processed video sample. If the processed video sample is the last one, then no such region exist and the composition timestamp of the video sample following the processed video sample is not needed.

After step 580, the next step is step 510.

If at step 570 it is determined that there are no more video samples to process, then the generation of the region track is finished. As a final step, if at step 560 setting the duration of the previous last sample was deferred, then the duration of this last sample is set to the duration of the last processed video sample.

Note that preferably, an encoder may avoid encoding region interpolations that span several media fragments or over random access points. For example, a region that may be interpolated from a first sample in a first media fragment up to a second sample in the next media fragment may be encoded as two region interpolations, the first one starting at the first sample and ending at the last sample of the first media fragment and the second one starting at the initial sample of the second media fragment and ending at the second sample. As another example, a region that may be interpolated from a first sample before a random access point up to a second sample after the random access point may be encoded as two region interpolations, the first one starting at the first sample and ending at the sample before the random access point and the second one starting at the sample corresponding to the random access point and ending at the second sample. In this way, a renderer may start computing interpolated regions at the beginning of any media fragment or at any random access point.

Preferably, the timescale of the region track may be set to a large value to create a large number of time units between two successive video samples. Indeed, the number of region whose interpolation starts at the composition timestamp of the same video sample is limited by the number of time units of the region track between two video samples minus one. Similarly, the number of regions whose interpolation ends at the composition timestamp of the same video sample is also limited by this same number. Therefore, the larger this number of time units, the larger is the maximum number of regions whose interpolation starts or ends at the same timestamp.

At step 550, if the decoding timestamp or the composition timestamp of a sample cannot be computed because one of these limits is reached, several possibilities may be used.

First, the region whose interpolation's end is encoded in this sample may be encoded without interpolation, encoding its complete characteristics in each region track sample generated at step 520 corresponding to a video sample to which the region is associated.

Second, this region may be encoded with an interpolation starting at a time corresponding to the composition timestamp of the next video sample. This means that at the next execution of step 520, the region will be again encoded with its complete characteristics in the region sample generated at this step.

Third, the interpolation duration of this region may be reduced to match the interpolation duration of another region whose interpolation starts at the same time. This enables to encode the interpolation end of the two regions in the same sample. At the end of this reduced duration, the region may be encoded using a new interpolation, or it may be encoded without using interpolation.

Possibly, a check may be realized at the end of step 530, once the regions using interpolation with an interpolation starting at the composition timestamp of the processed video sample are retrieved. This check may verify whether the samples that will be generated at step 540 will reach one of the limits regarding their decoding timestamps or composition timestamps. If this is the case, an additional step may select which possibility or possibilities among the three listed possibilities is or are selected and to which regions it is or they are applied. This additional step may choose for example the solution giving the smallest encoding size for the region track. For instance, if it selects the first solution, that is to encode a region without using interpolation, it may select the region with the shortest interpolation time. As another example, if it selects the third solution, it may select to combine the two regions with the closest interpolation ending time. The additional step may also take into account the encoding size of each region. For example, it may select to encode a polygon region using interpolation and a point region without using interpolation even if the interpolation duration for the point region is longer than the interpolation duration for the polygon region as encoding a polygon is more costly than encoding a point.

Figure 6 illustrates the main steps of a method for processing a region track with interpolated regions according to embodiments of the invention.

Similarly to Figure 5, these steps may be used for processing a region track associated with a whole video track or for processing a part of a region track associated with a part of a video track, typically a fragment.

In a first step 600, the processed video sample is set to the first video sample of the video track. In addition, the composition timestamp of this first video sample is retrieved.

During this step, the list of current regions is initialized to an empty list. This list is used to link the ending point of a region interpolation with its starting point. It is also used to compute an interpolated region from its starting point and ending point.

At step 610, all the region track samples from the region track with a decoding timestamp lower than or equal to the composition timestamp of the first video sample and that have not already been retrieved are retrieved.

At step 620, the retrieved region track samples are decoded to obtain the description of regions in the associated video track. These regions are split into two groups. The first group contains the regions encoded with the interpolate flag set to 0. The second group contains the regions encoded with the interpolate flag set to 1.

At step 630, the regions encoded with the interpolate flag set to 1, namely the regions subject to interpolation, are processed. First, for each region encoded with the interpolate flag, a matching region with the same identifier and the same geometry type is searched among the list of current region. If no matching region is found, then the region is discarded. Possibly an error is raised and the processing of the region track and of the video track stops. If a matching region is found, then the encoded region is added to the list of current region and is linked to the matching region.

Second, for each region encoded with the interpolate flag the composition timestamp of the sample inside which the region is encoded is retrieved. The timestamp associated with this region is computed as the composition timestamp of the first video sample occurring after this composition timestamp. The composition timestamp of this first video sample may be retrieved by looking in advance at the composition timestamps of future video samples. It may also be estimated using the composition timestamp of the processed video sample and the frame rate of the video track. It may also be retrieved from metadata associated with the region track or with the video track, or by any other means.

This operation may be seen as rounding the composition timestamp of the sample inside which the region is encoded to the next greater composition timestamp of a video sample. In other variants, this rounding operation may be to the preceding lower composition timestamp of a video sample or to the closest composition timestamp of a video sample.

Furthermore, at step 630, regions corresponding to past video samples are removed from the list of current regions. A region with a timestamp strictly before the timestamp of the processed video sample and linked to another region for interpolation and such that the region is the starting point of the interpolation and the timestamp of the other region is strictly before the timestamp of the processed video sample is removed from the list. A region with a timestamp strictly before the timestamp of the processed video sample and not linked to another region for interpolation is removed from the list.

Last, regions from the first group determined at step 620 are added to the list of current regions with a timestamp equal to the composition timestamp of the processed video sample.

At step 640, the regions in the list of current regions are rendered. In the case of a region with a timestamp equal to the composition timestamp of the processed video sample, it is used directly. In the case of a pair of regions linked together as the starting and ending points of an interpolation and such that neither of their timestamps is equal to the composition timestamp of the processed video sample, an interpolated region is computed. In the case of a pair of regions linked together as the starting and ending points of an interpolation and such that one of their respective timestamps is equal to the composition timestamp of the processed video sample, the corresponding region is used directly.

The processing associated to the rendering of the regions depends on the application processing the region track. The regions may be displayed as an overlay on the rendering of the video. The regions may be listed as metadata associated with the video. The regions may be used as part of a graphical interface for displaying specific information depending on the area of the video selected by a user, or any other use depending on the application.

At step 650, it is checked whether there are more video samples to process.

If this is the case, then at step 660, the next video sample in composition timestamp order is set as the processed video sample. In addition, the composition timestamp of this next video sample is retrieved. The next step is then step 610.

If at step 650 it is determined that there are no more video samples to process, then the processing of the region track ends.

In a variant, the starting point and the ending point for an interpolated region are encoded in the same sample. In this variant, the timestamp associated with the starting point of the interpolation may be the decoding timestamp of the sample containing the region. This decoding timestamp may be adjusted to the composition timestamp of the first video sample occurring after this decoding timestamp. The timestamp associated with the ending point of the interpolation may be computed as described at step 630.

In another variant, the duration of an interpolation may be encoded in the region sample containing the interpolation end. This duration may be encoded as a duration, for example as a number of time units for the region track. This duration may be encoded by specifying the timestamp of the end of the interpolation. In this variant, the composition timestamp of the region sample containing the interpolation end may have the same value as the decoding timestamp of this region sample.

Figure 7 illustrates an example of describing several regions using extrapolation according to embodiments of the invention. In the present document, the movement describes the evolution of a region comprising an evolution in its location figuring a displacement of the region in the media data frames as well as a possible evolution in size.

This example uses the same video track 100 as illustrated in Figure 1.

The region track 750 contains four samples, 760, 770, 780 and 790.

The first sample 760 contains the description of two regions, an elliptic region 761 and a rectangular region 762.

The second sample 770 contains the description of a new region, a triangular region 773. It also contains the description of the evolution of the elliptic region 771 and of the evolution of the rectangular region 772. The evolution of the elliptic region 771 describes the movement of the elliptic region from one frame to another and also possibly its change in size. This enables to compute the position and size of the elliptic region using its starting position and size 761 modified by its movement 771 multiplied by the elapsed time. Similarly, the evolution of the rectangular region 772 enables to compute the position and size of this rectangular region for the different frames.

Sample 780 contains the description of the evolution of the triangular region 783.

Sample 790 contains an indication 791 that the elliptic region is no more present in the video track and that it should not be computed anymore.

Region track samples using extrapolation as illustrated by Figure 7 may be represented using the following structure.

```
    aligned (8) class RegionSample {
     unsigned      int      field_size      =
((RegionTrackConfigBox.field_length_size & 1) + 1) * 16;
     // this is a temporary, non-parsable variable
     unsigned int(32) region_count;
     for (r=0; r < region_count; r++) {
      unsigned int(32) region_identifier;
      unsigned int(8) geometry_type;
      unsigned int(1) extrapolate;
      unsigned int(7) reserved;
      if (extrapolate == 0) {
       if (geometry_type == 0) {
        // point
        signed int(field_size) x;
        signed int(field_size) y;
       }
       else if (geometry_type == 1) {
        // rectangle
        signed int(field_size) x;
        signed int(field_size) y;
        unsigned int(field_size) width;
        unsigned int(field_size) height;
       }
       else if (geometry_type == 2) {
        // ellipse
        signed int(field_size) x;
        signed int(field_size) y;
        unsigned int(field_size) radius_x;
        unsigned int(field_size) radius_y;
       }
       else if (geometry_type == 3 || geometry_type == 6)
{
        // polygon or polyline
        unsigned int(field size) point_count;
        for (i=0; i < point_count; i++) {
         signed int(field_size) px;
         signed int(field_size) py;
        }
       }
       else if (geometry_type == 4) {
        // referenced mask
        signed int(field_size) x;
        signed int(field_size) y;
        unsigned int(field_size) width;
        unsigned int(field_size) height;
        unsigned int(field_size) track_mask_idx;
       }
       else if (geometry_type == 5) {
        // inline mask
        signed int(field_size) x;
        signed int(field_size) y;
        unsigned int(field_size) width;
        unsigned int(field_size) height;
        unsigned int(8) mask_coding_method;
        if (mask_coding_method != 0)
         unsigned int(32) mask_coding_parameters;
        bit(8) data[];
```

```
               }
             }
             else {
               if (geometry_type == 0) {
                 // point
                 signed int(field_size) delta_x;
                 signed int(field_size) delta_y;
               }
               else if (geometry_type == 1) {
                 // rectangle
                 signed int(field_size) delta_x;
                 signed int(field_size) delta_y;
                 signed int(field_size) delta_width;
                 signed int(field_size) delta_height;
               }
               else if (geometry_type == 2) {
                 // ellipse
                 signed int(field_size) delta_x;
                 signed int(field_size) delta_y;
                 signed int(field_size) delta_radius_x;
                 signed int(field_size) delta_radius_y;
               }
               else if (geometry_type == 3 || geometry_type == 6)
{
                 // polygon or polyline
                 for (i=0; i < point_count; i++) {
                   signed int(field_size) delta_px;
                   signed int(field_size) delta_py;
                 }
               }
               else if (geometry_type == 4) {
                 // referenced mask
                 signed int(field_size) delta_x;
                 signed int(field_size) delta_y;
               }
               else if (geometry_type == 5) {
                // inline mask
                signed int(field_size) delta_x;
                signed int(field_size) delta_y;
            }
        }
    }
}
```

In this structure, the extrapolate flag indicates whether the region is encoded with its full description or if only its evolution is encoded. When the extrapolate flag is set to 0, then the full description of the region is encoded. For example, for a rectangle, its position specified by the x and y fields, its width and its height are encoded. When the extrapolate flag is set to 1 only the evolution of the region is encoded. For example, for a rectangle, the evolution of its position, specified by the delta_x and delta_y fields, of its width, through the delta_width field, and of its height, through the delta_height field, are encoded.

The evolution of a region over time is optional. It may be represented by the evolution speed of some of its parameters inside the reference space of the video track.

The parameters defining the evolution of a region depend on the geometry of the region. When the geometry of a region is represented by a point, the evolution of the region may be defined by the evolution of the position of this point. When the geometry of a region is represented by a rectangle or an ellipse, the evolution of the region may be defined by the evolution of the position and size of the rectangle or ellipse. When the geometry of a region is represented by a polygon or a polyline, the evolution of the region may be defined by the evolution of the position of each point of the polygon or polyline. Preferably, the number of points in the polygon or polyline doesn't change. When the geometry of a region is represented by a mask, the evolution of the region may be defined by the evolution of the position of the mask.

For a point, the extrapolated values may be x and y, and the evolution of these values may be encoded using the delta_x and delta_y fields. For a rectangle, the extrapolated values may be x, y, width and height and the evolution of these values may be encoded using the delta_x, delta_y, delta_width and delta_height fields. For an ellipse, the extrapolated values may be x, y, radius_x and radius_y and the evolution of these values may be encoded using the delta_x, delta_y, delta_radius_x and delta_radius_y fields. For a polygon or a polyline, the extrapolated values may be `px` and `py` for each point, and the evolution of these values may be encoded using the `delta_px` and `delta_py` fields. For a mask, the extrapolated values may be `x` and `y` and the evolution of these values may be encoded using the `delta_x` and `delta_y` fields. Possibly, the `width` and `height` values may also be extrapolated resulting in a scaling of the mask.

In a variant, the number of points for a polygon or a polyline may be an extrapolated value, and its evolution may be encoded using a `delta_point_count` field.

In another variant, other region shapes may be described and their evolution may be described by fields corresponding to their characteristics.

In yet another variant, only some types of regions may be extrapolated. For example, only points, rectangle and ellipse may be extrapolated.

In another variant, only some characteristics of the regions may be extrapolated. For example, only the positions of the regions may be extrapolated. Possibly, in this variant, the `extrapolate` field may have different values for signalling which characteristics of a region are extrapolated. For example, for a rectangle, the value 1 for the `extrapolate` field may signal that its position and its size are extrapolated, the value 2 may signal that only its position is extrapolated and the value 3 may signal that only its size is extrapolated. As another example, for a polygon or a polyline, the `extrapolate` field may be used to signal which points of the polygon or polyline are extrapolated.

The value V of a characteristic of the geometry of a region at a time T may be computed from the value V_S of this characteristic as defined at the time T_S corresponding to the extrapolation start and from the value δV of the evolution of this characteristic as follows:

$$V = V_S + \delta V(T - T_S)$$

In other words, in this embodiment, a first region track sample describes the region at a starting position and is timely synchronized with the video sample where the region appears or is present. A second region track sample, timely synchronized with the following video sample comprises a description of the movement of the region. And a last region track sample timely synchronized with the video sample following the last video sample where the region appears or is present, comprises an indication that this region is no longer present.

Figure 8 illustrates another example of describing several regions using extrapolation according to embodiments of the invention.

This example uses the same video track 100 as illustrated in Figure 1.

The region track 850 contains three samples, 860, 870 and 890.

The first sample 860 contains the description of two regions, an elliptic region 861 and a rectangular region 862. These descriptions contain both the initial geometry of the region and the evolution of the region. Using this information, the position and size of the elliptic region can be computed for any timestamp starting from the composition timestamp of video sample 110 as described previously for Figure 7.

The second sample 870 contains the description of a new region, the triangular region 873, describing both its initial geometry and its evolution.

There is no sample for the region track 850 corresponding to the video sample 130 of the video track 100. Indeed, there no need to provide any information about the regions corresponding to this video sample 130 as they can be computed from the previous information carried by the region track 850.

The last sample 890 contains an indication 891 that the elliptic region is no more present in the video track and that it should not be computed anymore.

Region track samples using extrapolation as illustrated by Figure 8 may be represented using the following structure.

```
aligned (8) class RegionSample {
  unsigned int field_size = ((RegionTrackConfigBox.field_length_size & 1) + 1) * 16;
  // this is a temporary, non-parsable variable
  unsigned int(32) region_count;
  for (r=0; r < region_count; r++) {
    unsigned int(32) region_identifier;
    unsigned int(8) geometry_type;
    unsigned int(1) extrapolate;
    unsigned int(7) reserved;
    if (geometry_type == 0) {
      // point
      signed int(field_size) x;
      signed int(field_size) y;
      if (extrapolate) {
        signed int(field_size) delta_x;
      signed int(field_size) delta_y;
    }
  }
  else if (geometry_type == 1) {
    // rectangle
    signed int(field_size) x;
    signed int(field_size) y;
    unsigned int(field_size) width;
    unsigned int(field_size) height;
    if (extrapolate) {
      signed int(field_size) delta_x;
      signed int(field_size) delta_y;
      signed int(field_size) delta_width;
      signed int(field_size) delta_height;
    }
  }
  else if (geometry_type == 2) {
    // ellipse
    signed int(field_size) x;
    signed int(field_size) y;
    unsigned int(field_size) radius_x;
    unsigned int(field_size) radius_y;
    if (extrapolate) {
      signed int(field_size) delta_x;
      signed int(field_size) delta_y;
      signed int(field_size) delta_radius_x;
      signed int(field_size) delta_radius_y;
    }
  }
  else if (geometry_type == 3 || geometry_type == 6) {
    // polygon or polyline
    unsigned int(field size) point_count;
    for (i=0; i < point_count; i++) {
      signed int(field_size) px;
      signed int(field_size) py;
    }
```

```
      if (extrapolate) {
        for (i=0; i < point_count; i++) {
          signed int(field_size) delta_px;
          signed int(field_size) delta_py;
        }
      }
    }
    else if (geometry_type == 4) {
      // referenced mask
      signed int(field_size) x;
      signed int(field_size) y;
      unsigned int(field_size) width;
      unsigned int(field_size) height;
      unsigned int(field_size) track_mask_idx;
      if (extrapolate) {
        signed int(field_size) delta_x;
        signed int(field_size) delta_y;
      }
    }
    else if (geometry_type == 5) {
      // inline mask
      signed int(field_size) x;
      signed int(field_size) y;
      unsigned int(field_size) width;
      unsigned int(field_size) height;
      unsigned int(8) mask_coding_method;
      if (mask_coding_method != 0)
        unsigned int(32) mask_coding_parameters;
      bit(8) data[];
      if (extrapolate) {
        signed int(field_size) delta_x;
        signed int(field_size) delta_y;
      }
    }
  }
}
```

In this structure, the `extrapolate` flag indicates whether the evolution of the region is encoded in addition to its full description. When the `extrapolate` flag is set to 0, then only the full description of the region is encoded. When the `extrapolate` flag is set to 1, then both the full description of the region and its evolution are encoded, i.e., the region is an evolving region, it is a region described using extrapolation and it is defined by an initial geometry and its evolution over time. In other words, `extrapolate` is a flag indicating whether the geometry changes of the region are specified or not. When equal to 0, it indicates that no geometry changes are specified for the region. When equal to 1, it indicates that both the geometry and the geometry changes are specified for the region. The meanings of the different fields are similar to those described in relation with Figure 7.

The embodiment corresponds to the previous one where the initial geometry of the region and movement are gathered in the same region track sample instead of being spread in two successive region track samples.

In this structure as well as in the structure described in relation with Figure 7, the evolution of a characteristic of a region may be encoded using the same unit as the characteristic itself. For example, the evolution of the horizontal position of a region may be encoded as a number of pixels. As another example, the evolution of the width of a region may be encoded as a number of pixels.

However, keeping the same unit as the characteristic itself may lead to approximation errors when computing the extrapolated characteristics of a region. For example, if a region moves horizontal of 1.5 pixel per frame, then there is an approximation error of 0.5 pixel per frame. Therefore, the evolution of a characteristic of a region may be encoded using a more precise unit than the encoding of the characteristic itself. For example, the evolution of the horizontal position of a region may be encoded as a number of tenths of a pixel or even hundredths of a pixel.

In a variant, the evolution of a characteristic of a region may be encoded as a fractional value instead of as an integer, for example using a fixed point notation. For example, half of the encoding size may be used for the integer part of the value and half of the encoding size may be used for the fractional part of the value.

In this variant, the evolution speed of the charateristics of a region may be signaled using a scaling factor for increasing its precision. The scaling factor S for each field representing an evolution of a characteristic of the corresponding region may be computed using the value f of the `field_size` field as:

$$S = 2^{\frac{f}{2}}$$

Using this scaling factor, the value V of a characteristic of the geometry of a region at a time T may be computed from the value V_S of this characteristic as defined at the time T_S corresponding to the extrapolation start and from the value δV of the evolution of this characteristic as follows:

$$V = V_S + \frac{\delta V}{S}(T - T_S)$$

In this variant, the evolution of the characteristics of a region are described using $\frac{1}{S}$ units of the reference space of the associated video track. As yet another possibility, the scaling factor may be indicated for each region in the encoding of its evolution. For example, this scaling factor may be encoded using the following structure.

```
        aligned (8) class RegionSample {
          unsigned int field_size =
((RegionTrackConfigBox.field_length_size & 1) + 1) * 16;
          // this is a temporary, non-parsable variable
          unsigned int(32) region_count;
          for (r=0; r < region_count; r++) {
            unsigned int(32) region_identifier;
            unsigned int(8) geometry_type;
            unsigned int(1) extrapolate;
            unsigned int(6) scaling;
            unsigned int(1) reserved;
            if (geometry_type == 0) {
              // point
              signed int(field_size) x;
              signed int(field_size) y;
              if (extrapolate) {
                signed int(field_size) delta_x;
                signed int(field_size) delta_y;
              }
            }
            else if (geometry_type == 1) {
              // rectangle
              signed int(field_size) x;
              signed int(field_size) y;
              unsigned int(field_size) width;
```

```
  unsigned int(field_size) height;
  if (extrapolate) {
    signed int(field_size) delta_x;
    signed int(field_size) delta_y;
    signed int(field_size) delta_width;
    signed int(field_size) delta_height;
  }
}
else if (geometry_type == 2) {
  // ellipse
  signed int(field_size) x;
  signed int(field_size) y;
  unsigned int(field_size) radius_x;
  unsigned int(field_size) radius_y;
  if (extrapolate) {
    signed int(field_size) delta_x;
    signed int(field_size) delta_y;
    signed int(field_size) delta_radius_x;
    signed int(field_size) delta_radius_y;
  }
}
else if (geometry_type == 3 || geometry_type == 6) {
  // polygon or polyline
  unsigned int(field size) point_count;
  for (i=0; i < point_count; i++) {
    signed int(field_size) px;
    signed int(field_size) py;
  }
  if (extrapolate) {
    for (i=0; i < point_count; i++) {
      signed int(field_size) delta_px;
      signed int(field_size) delta_py;
    }
  }
}
else if (geometry_type == 4) {
      // referenced mask
      signed int(field_size) x;
      signed int(field_size) y;
      unsigned int(field_size) width;
      unsigned int(field_size) height;
      unsigned int(field_size) track_mask_idx;
      if (extrapolate) {
        signed int(field_size) delta_x;
        signed int(field_size) delta_y;
      }
    }
    else if (geometry_type == 5) {
      // inline mask
      signed int(field_size) x;
      signed int(field_size) y;
      unsigned int(field_size) width;
      unsigned int(field_size) height;
      unsigned int(8) mask_coding_method;
      if (mask_coding_method != 0)
        unsigned int(32) mask_coding_parameters;
      bit(8) data[];
      if (extrapolate) {
        signed int(field_size) delta_x;
        signed int(field_size) delta_y;
      }
    }
  }
}
```

In this structure, the value s of the `scaling` field is used to compute the scaling factor S for each field representing an evolution of a characteristic of the corresponding region. The number of bits used for representing the fractional part of a characteristic’s value may be computed as 2^s. Therefore, the scaling factor S may be computed as:

$$S = 2^{2^s}$$

Using this scaling factor, the value V of a characteristic of the geometry of a region at a time T may be computed from the value V_S of this characteristic as defined at the time T_S corresponding to the extrapolation start and from the value δV of the evolution of this characteristic as follows:

$$V = V_S + \frac{\delta V}{S}(T - T_S)$$

The `scaling` field may also be included in other structures defining the evolution of regions to encode the scaling factor for fields representing the evolution of a characteristic of a region.

In a variant, the `scaling` field may be common to a plurality of regions of a sample or to all the regions of a sample. In a variant, the `scaling` field may be common to a plurality of samples.

The size of the `scaling` field may be different, using for example fewer or more bits. The computation of the scaling factor from the `scaling` field may be different, for example, each `scaling` field value may be associated with a scaling factor through a pre-defined table.

The end of the extrapolation of a region may be signaled by adding a specific field to one of the previously described structure. For example, the following structure may be used:

```
aligned (8) class RegionSample {
  unsigned int field_size =
((RegionTrackConfigBox.field_length_size & 1) + 1) * 16;
  // this is a temporary, non-parsable variable
  unsigned int(32) region_count;
  for (r=0; r < region_count; r++) {
    unsigned int(32) region_identifier;
    unsigned int(8) geometry_type;
    unsigned int(1) extrapolate;
    unsigned int(1) extrapolate_end;
    unsigned int(6) reserved;
    if (extrapolate_end == 0) {
      if (geometry_type == 0) {
        // point
        signed int(field_size) x;
        signed int(field_size) y;
        if (extrapolate) {
         signed int(field_size) delta_x;
         signed int(field_size) delta_y;
        }
       }
       else if (geometry_type == 1) {
        // rectangle
        signed int(field_size) x;
        signed int(field_size) y;
        unsigned int(field_size) width;
        unsigned int(field_size) height;
        if (extrapolate) {
         signed int(field_size) delta_x;
         signed int(field_size) delta_y;
         signed int(field_size) delta_width;
         signed int(field_size) delta_height;
        }
       }
       else if (geometry_type == 2) {
        // ellipse
        signed int(field_size) x;
        signed int(field_size) y;
        unsigned int(field_size) radius_x;
        unsigned int(field_size) radius_y;
        if (extrapolate) {
         signed int(field_size) delta_x;
         signed int(field_size) delta_y;
         signed int(field_size) delta_radius_x;
         signed int(field_size) delta_radius_y;
        }
       }
       else if (geometry_type == 3 || geometry_type == 6)
{
        // polygon or polyline
        unsigned int(field size) point_count;
```

```
      for (i=0; i < point_count; i++) {
        signed int(field_size) px;
        signed int(field_size) py;
      }
      if (extrapolate) {
        for (i=0; i < point_count; i++) {
          signed int(field_size) delta_px;
          signed int(field_size) delta_py;
        }
      }
    }
    else if (geometry_type == 4) {
      // referenced mask
      signed int(field_size) x;
      signed int(field_size) y;
      unsigned int(field_size) width;
      unsigned int(field_size) height;
      unsigned int(field_size) track_mask_idx;
      if (extrapolate) {
        signed int(field_size) delta_x;
        signed int(field_size) delta_y;
      }
    }
    else if (geometry_type == 5) {
      // inline mask
      signed int(field_size) x;
      signed int(field_size) y;
      unsigned int(field_size) width;
      unsigned int(field_size) height;
      unsigned int(8) mask_coding_method;
      if (mask_coding_method != 0)
        unsigned int(32) mask_coding_parameters;
      bit(8) data[];
      if (extrapolate) {
        signed int(field_size) delta_x;
        signed int(field_size) delta_y;
        }
      }
    }
  }
}
```

In this structure, the `extrapolate_end` field indicates that the region with the signaled `region_identifier` which was previously described using extrapolation is no more present in the associated video track and that the extrapolation should stop.

As a variant, both this `extrapolate_end` field and the `scaling` field may be signaled in the description of the regions, for example using the following structure:

```
aligned (8) class RegionSample {
  unsigned int field_size =
((RegionTrackConfigBox.field_length_size & 1) + 1) * 16;
  // this is a temporary, non-parsable variable
  unsigned int(32) region_count;
  for (r=0; r < region_count; r++) {
    unsigned int(32) region_identifier;
    unsigned int(8) geometry_type;
    unsigned int(1) extrapolate;
    unsigned int(1) extrapolate_end;
    unsigned int(6) scaling;
    if (extrapolate_end == 0) {
      if (geometry_type == 0) {
        // point
        signed int(field_size) x;
        signed int(field_size) y;
        if (extrapolate) {
          signed int(field_size) delta_x;
          signed int(field_size) delta_y;
        }
      }
      else if (geometry_type == 1) {
        // rectangle
        signed int(field_size) x;
        signed int(field_size) y;
      unsigned int(field_size) width;
      unsigned int(field_size) height;
      if (extrapolate) {
        signed int(field_size) delta_x;
        signed int(field_size) delta_y;
        signed int(field_size) delta_width;
        signed int(field_size) delta_height;
      }
    }
    else if (geometry_type == 2) {
      // ellipse
      signed int(field_size) x;
      signed int(field_size) y;
      unsigned int(field_size) radius_x;
      unsigned int(field_size) radius_y;
      if (extrapolate) {
        signed int(field_size) delta_x;
        signed int(field_size) delta_y;
        signed int(field_size) delta_radius_x;
        signed int(field_size) delta_radius_y;
      }
    }
    else if (geometry_type == 3 || geometry_type == 6)
{
      // polygon or polyline
      unsigned int(field size) point_count;
      for (i=0; i < point_count; i++) {
        signed int(field_size) px;
        signed int(field_size) py;
      }
      if (extrapolate) {
        for (i=0; i < point_count; i++) {
          signed int(field_size) delta_px;
          signed int(field_size) delta_py;
        }
      }
      }
      else if (geometry_type == 4) {
        // referenced mask
        signed int(field_size) x;
        signed int(field_size) y;
        unsigned int(field_size) width;
        unsigned int(field_size) height;
        unsigned int(field_size) track_mask_idx;
        if (extrapolate) {
          signed int(field_size) delta_x;
          signed int(field_size) delta_y;
        }
      }
      else if (geometry_type == 5) {
        // inline mask
        signed int(field_size) x;
        signed int(field_size) y;
        unsigned int(field_size) width;
        unsigned int(field_size) height;
        unsigned int(8) mask_coding_method;
        if (mask_coding_method != 0)
          unsigned int(32) mask_coding_parameters;
        bit(8) data[];
        if (extrapolate) {
          signed int(field_size) delta_x;
          signed int(field_size) delta_y;
        }
      }
    }
  }
}
```

In a variant, the end of the extrapolation for a region may be signaled using a duration field associated with the region. This duration field may be expressed as a number of time units according to the timescale of the region track.

In a variant, the end of the extrapolation for a region may be signaled by signaling a region with the same value for the `region_identifier` field. A new region type corresponding to an empty region may be added for signaling the end of the extrapolation for a region. In this variant, region track samples using extrapolation as illustrated by Figure 8 may be represented using the following structure:

```
        aligned (8) class RegionSample {
          unsigned int field_size =
((RegionTrackConfigBox.field_length_size & 1) + 1) * 16;
          // this is a temporary, non-parsable variable
          unsigned int(32) region_count;
          for (r=0; r < region_count; r++) {
            unsigned int(32) region_identifier;
            unsigned int(8) geometry_type;
            unsigned int(1) extrapolate;
            unsigned int(7) reserved;
            if (geometry_type == 0) {
              // point
              signed int(field_size) x;
              signed int(field_size) y;
              if (extrapolate) {
                signed int(field_size) delta_x;
                signed int(field_size) delta_y;
              }
            }
            else if (geometry_type == 1) {
              // rectangle
              signed int(field_size) x;
              signed int(field_size) y;
              unsigned int(field_size) width;
              unsigned int(field_size) height;
              if (extrapolate) {
                signed int(field_size) delta_x;
                signed int(field_size) delta_y;
                signed int(field_size) delta_width;
                signed int(field_size) delta_height;
              }
```

```
}
else if (geometry_type == 2) {
  // ellipse
  signed int(field_size) x;
  signed int(field_size) y;
  unsigned int(field_size) radius_x;
  unsigned int(field_size) radius_y;
  if (extrapolate) {
    signed int(field_size) delta_x;
    signed int(field_size) delta_y;
    signed int(field_size) delta_radius_x;
    signed int(field_size) delta_radius_y;
  }
}
else if (geometry_type == 3 || geometry_type == 6) {
  // polygon or polyline
  unsigned int(field size) point_count;
  for (i=0; i < point_count; i++) {
    signed int(field_size) px;
    signed int(field_size) py;
  }
  if (extrapolate) {
    for (i=0; i < point_count; i++) {
      signed int(field_size) delta_px;
      signed int(field_size) delta_py;
    }
  }
}
else if (geometry_type == 4) {
  // referenced mask
  signed int(field_size) x;
  signed int(field_size) y;
  unsigned int(field_size) width;
  unsigned int(field_size) height;
  unsigned int(field_size) track_mask_idx;
  if (extrapolate) {
        signed int(field_size) delta_x;
        signed int(field_size) delta_y;
      }
    }
    else if (geometry_type == 5) {
      // inline mask
      signed int(field_size) x;
      signed int(field_size) y;
      unsigned int(field_size) width;
      unsigned int(field_size) height;
      unsigned int(8) mask_coding_method;
      if (mask_coding_method != 0)
        unsigned int(32) mask_coding_parameters;
      bit(8) data[];
      if (extrapolate) {
        signed int(field_size) delta_x;
        signed int(field_size) delta_y;
      }
    }
    else if (geometry_type == 7) {
      // empty region
    }
  }
}
```

In this structure, the value 7 for the `geometry_type` field indicates an empty region. This region may be used for signaling the end of the evolution of a previous region with the same identifier.

Using this structure, the geometry of a region may be defined by specifying the shape, position and size of the region in a sample of the region track. The geometry of a region may also be defined as an initial geometry and its evolution over time by specifying the initial geometry of the region and its evolution in a sample of the region track.

Preferably, the end of the extrapolation for a region may correspond to the composition time of the first video sample inside which the region doesn't appear or inside which the movement parameters has changed. Possibly, the end of the extrapolation may correspond to the composition time of the last video sample inside which the region appears or inside which the movement parameters didn't change. In the following, the former choice is used.

In a variant the extrapolation for a region may end at the end of the duration of the region sample in which this extrapolation is described. In other words, the duration of the extrapolation for a region is the duration of the region sample in which it is described. This variant may be signaled using a flag in the structure describing the region.

In a variant, the extrapolation for a region may be extended for the duration of the region sample following the region sample containing the description of the extrapolation of the region. For example, an `extend_extrapolation` field in the structure describing the regions may be set to the value 1 to signal that the extrapolation of a region is extended for the duration of the region sample containing this `extend_extrapolation` field.

In another variant, the extrapolation for a region may be extended for the duration of one or more region samples following the region sample containing the description of the extrapolation of the region. For example, an `extension_count` field in the structure describing the regions may be used to indicate the number of region samples during which the extrapolation for the region is extended.

In yet another variant, a region sample may contain a `continue_extrapolation` field signaling that the extrapolation is extended for the duration of this region sample for all the regions described using extrapolation.

Preferably, the extrapolation for a region ends with each sync sample of the video track, that is a sample that starts a new independent sequence of samples. This means that a sync sample of the video track preferably has a corresponding region sample signaling the end of the extrapolation for all regions. This signaling may be realized by including in the region sample either a definition of the geometry of the region or an indication that the extrapolation of the region has reached its end. In other words, the evolution of a region stops for each sync sample of the video track, i.e. the geometry of a region defined for a sync sample of the video track preferably is not computed using extrapolation. In addition, the geometry of a region defined for a sample of the video track preferably is not computed using extrapolation from an initial geometry defined in a region sample preceding the previous sync sample of the video track.

Possibly, the `RegionSample` structure may contain a flag, for example named `extrapolation_end`, indicating the end of the extrapolation for all the regions.

The value 1 for the `extrapolation_end` field indicates that the extrapolation for all the regions previously described with extrapolation ends, i.e., it indicates that none of the regions previously described with extrapolation are present in the video sample corresponding to the region sample comprising the `extrapolation_end` field. The value 0 for the `extrapolation_end` field indicates that the extrapolation for the regions previously described with extrapolation continues normally.

Figure 9 illustrates the main steps of a method for generating a region track with extrapolated regions according to embodiments of the invention, following the example illustrated by Figure 8. These steps can be adapted for generating a region track with extrapolated regions following the example illustrated by Figure 7.

These steps may be used for generating a region track associated with a whole video track or for generating a part of the region track associated with a part of a video track. A part of the video track may for example be the portion of the video track included in a media fragment. In the following, the terms video track are used for representing either a whole video track or a part of a video track.

These steps take as input the video track and a list of regions associated with this video track. This list may contain regions occurring at a specific timestamp in the video track. This list may contain regions that may be extrapolated starting from a first specific timestamp and ending at a second specific timestamp in the video track.

In a first step 900, the processed video sample is set to the first video sample of the video track. In addition, the composition timestamp of this first video sample is retrieved.

At this step, a list of used identifiers is initialized to the empty list.

At step 910, the regions corresponding to this timestamp are retrieved with their characteristics. At this step, some of the regions associated with the processed video sample are retrieved. These regions are those that are described without using extrapolation, those that are described using an extrapolation starting at the composition timestamp of the processed video sample. In some variants, regions that are described using an extrapolation ending at the composition timestamp of the processed video sample may also be retrieved. This means that regions that are described using extrapolation with an extrapolation starting before the composition timestamp of the processed video sample and ending strictly after the composition timestamp of the processed video sample are not retrieved at this step. The characteristics of the retrieved regions may include the type of the geometry of a region, its position, its size and any further information describing the region according to embodiments.

If no region is retrieved at this step, then the next step is step 960.

Possibly, at this step 910, the identifiers associated to regions whose extrapolation ended at the composition timestamp strictly before the composition timestamp of the processed video sample are removed from the used identifiers list.

At step 920, the duration of the region track sample is obtained. This duration may be obtained by considering the video samples following the processed video sample in composition timestamp order. For each considered video sample the regions corresponding to its composition timestamp are retrieved, similarly to what is described at step 910. This retrieval is realized until a video sample is found with at least one retrieved region. The duration of the region track sample is the difference between the composition timestamp of this found video sample and the composition timestamp of the processed video sample. This duration may be obtained through other means. For example, it may be obtained from a list of the regions associated with the video where the regions are ordered according to their initial occurrence timestamp.

If no video sample is found with at least one retrieved region, then the duration of the region track sample may be computed as the sum of the duration of the processed video sample and of the duration of all the following video samples.

At step 930, the regions that are described using extrapolation with an extrapolation starting at the composition timestamp of the processed video sample are retrieved. These regions may be retrieved from the regions retrieved at step 910.

If no region is retrieved at this step, the processing may continue directly at step 950.

At step 940, the evolution of the regions described using extrapolation is computed. The encoded evolution $\varDelta V$ of the value V of a characteristic of a region described using extrapolation from a value V_S to a value V_E from a timestamp T_S to a timestamp T_E may be defined inside a region track sample of duration D as follows:

$$\varDelta V = (V_E - V_S)\frac{D}{T_E - T_S}$$

At step 950, the regions retrieved at step 910 are encoded inside a sample, using for example one of the structure described in illustration of Figure 8.

A retrieved region described without using extrapolation is encoded by fully describing it. A retrieved region described using extrapolation with an extrapolation starting at the composition timestamp of the processed video sample is encoded by fully describing its starting characteristics and by describing the evolution of these characteristics. In both cases, a region identifier is associated with each retrieved region and encoded as part of the encoding of the region. These region identifiers are added to the list of used identifiers.

In some variants, a retrieved region described using an extrapolation ending at the composition timestamp of the processed video sample, may be encoded by signaling that its extrapolation has ended. The identifier associated with the retrieved region is obtained. It is used as part of the encoding of the region.

As an optimization, in the case a retrieved region is described using extrapolation starting at the composition timestamp of the processed video sample and the same region was also described by a previous extrapolation ending at the composition timestamp of the processed video sample, the full description of its characteristics may be omitted. Indeed, these characteristics may be obtained from the previous extrapolation. In this case, the identifier previously associated with the region is retrieved and is used as part of the encoding of the retrieved region.

As another optimization, in the case a first retrieved region is described using extrapolation ending at the composition timestamp of the processed video sample and another retrieved region is described in the sample with the same region identifier, then the encoding of the first retrieved region may be omitted from the sample.

At step 960, it is checked whether there are more video samples to process.

If this is the case, then at step 970, the next video sample in composition timestamp order is set as the processed video sample. In addition, the composition timestamp of this next video sample is retrieved.

After step 970, the next step is step 910.

If at step 960 it is determined that there are no more video samples to process, then the generation of the region track is finished.

Note that preferably, an encoder may avoid encoding regions using extrapolations that span several media fragments or over random access points. For example, a region that may be extrapolated starting from a first sample in a first media fragment up to a second sample in the next media fragment may be encoded as two region extrapolations, the first one starting at the first sample and ending after the last sample of the first media fragment and the second one starting at the initial sample of the second media fragment and ending at the second sample. As another example, a region that may be extrapolated starting at a first sample before a random access point up to a second sample after the random access point may be encoded as two region extrapolations, the first one starting at the first sample and ending at the sample corresponding to the random access point and the second one starting at this same random access point sample and ending at the second sample. In this way, a renderer may start computing extrapolated regions at the beginning of any media fragment or at any random access point.

Figure 10 illustrates the main steps of a method for parsing and processing a region track with extrapolated regions according to embodiments of the invention, following the example illustrated by Figure 8. These steps could be adapted for parsing and processing a region track with extrapolated regions following the example illustrated by Figure 7.

Similarly to Figure 9, these steps may be used for processing a region track associated with a whole video track or for processing a part of a region track associated with a part of a video track.

In a first step 1000, the processed video sample is set to the first video sample of the video track. In addition, the composition timestamp of this first video sample is retrieved.

During this step, the list of current regions is initialized to an empty list. This list is used to compute an extrapolated region from its full description and its evolution.

At step 1010, the region track sample with the same composition timestamp as the processed video sample is retrieved. The duration of the region track sample is also retrieved. If there is no region track sample with a composition timestamp matching the composition timestamp of the processed video sample, then the next step is step 1030.

At step 1020, the retrieved region track sample is decoded to obtain the description of regions in the associated video track.

At this step, any region in the list of current regions described without using extrapolation is removed from the list of current regions.

The decoded regions described without using extrapolation are added to the list.

The decoded regions described using extrapolation are also added to the list. The duration of the region track sample is associated to these regions.

Possibly, if a decoded region is described using extrapolation and only its evolution is described in the region track sample, then a region with the same region identifier is searched for in the list of current regions. If no such region is found, then the decoded region is ignored. If such a region is found, it is described using extrapolation, as otherwise it would have been removed from the list previously. The characteristics of the found region are computed for the composition timestamp of the processed video sample. These characteristics are used in place of fully described characteristics for the decoded region,

For a decoded region for which the extrapolation end is signaled, a region with the same region identifier is searched for in the list of current regions. If such a region is found, it is removed from the list of current regions.

Possibly, a region in the list with the same region identifier as a decoded region may be removed from the list.

Possibly, a region in the list described using extrapolation for which the extrapolation end is signaled is removed from the list. For example, the extrapolation end may be signaled through a duration comprised in the encoded description of the region.

At step 1030, the regions in the list of current regions described using extrapolation are retrieved and their characteristics for the composition timestamp of the processed video sample are computed.

The evolution δV of a characteristic of a region described using extrapolation may be computed from the encoded evolution ΔV and from the duration D associated to it, which is the duration of the region track sample that contained the description of the region, as follows:

$$\delta V = \frac{\Delta V}{D}$$

The value V of a characteristic of the region may be computed as described previously, for example using the following formula:

$$V = V_S + \delta V(T - T_S)$$

When using a scaling factor S for each field representing an evolution of a characteristic of a region that corresponds to half the encoding size used for the characteristics of a region, the value V of a characteristic of the region may be computed using the following formula:

$$V = V_S + \frac{\Delta V}{S}\frac{(T - T_S)}{D}$$

Where

T_S is the composition time of the sample defining the region.

D is the duration of the sample defining the region.

V_S is the initial value of the characteristic as defined in the initial geometry of the region at time T$_S$.

ΔV is the evolution of the characteristic as defined in the evolution of the region.

S is the scaling factor that may be computed as $S = 2^{\frac{f}{2}}$ where f is the `field_size` and is equal to: `((RegionTrackConfigBox.field_length_size & 1) + 1) * 16.`

According to some embodiments, the value V may be rounded to an integer value, for example the closest integer value.

At step 1040, the regions are rendered. For a region described without using extrapolation, its full characteristics are used directly. For a region described using extrapolation, its characteristics as computed at step 1030 are used.

The processing associated to the rendering of the regions depends on the application processing the region track. The regions may be displayed as an overlay on the rendering of the video. The regions may be listed as metadata associated with the video. The regions may be used as part of a graphical interface for displaying specific information depending on the area of the video selected by a user...

At step 1050, it is checked whether there are more video samples to process.

If this is the case, then at step 1060, the next video sample in composition timestamp order is set as the processed video sample. In addition, the composition timestamp of this next video sample is retrieved. The next step is then step 1010.

If at step 1050 it is determined that there are no more video samples to process, then the processing of the region track ends.

Possibly, a region track may use a single type of sample entry and a single type of sample. The structure of these samples may be one of those described above or a similar one.

Possibly, a region track may use a single type of sample entry and two types of samples. One of these sample types may be used to describe the full characteristics of one or more regions, for example using the following structure:

```
aligned (8) class RegionSample {
  unsigned int sample_type = 0x00;
  unsigned int field_size =
((RegionTrackConfigBox.field_length_size & 1) + 1) * 16;
  // this is a temporary, non-parsable variable
```

```
unsigned int(32) region_count;
for (r=0; r < region_count; r++) {
  unsigned int(32) region_identifier;
  unsigned int(8) geometry_type;
  if (geometry_type == 0) {
    // point
    signed int(field_size) x;
    signed int(field_size) y;
  }
  else if (geometry_type == 1) {
    // rectangle
    signed int(field_size) x;
    signed int(field_size) y;
    unsigned int(field_size) width;
    unsigned int(field_size) height;
  }
  else if (geometry_type == 2) {
    // ellipse
    signed int(field_size) x;
    signed int(field_size) y;
    unsigned int(field_size) radius_x;
    unsigned int(field_size) radius_y;
  }
  else if (geometry_type == 3 || geometry_type == 6) {
    // polygon or polyline
    unsigned int(field size) point_count;
    for (i=0; i < point_count; i++) {
      signed int(field_size) px;
      signed int(field_size) py;
    }
  }
  else if (geometry_type == 4) {
    // referenced mask
    signed int(field_size) x;
    signed int(field_size) y;
    unsigned int(field_size) width;
```

```
      unsigned int(field_size) height;
      unsigned int(field_size) track_mask_idx;
    }
    else if (geometry_type == 5) {
      // inline mask
      signed int(field_size) x;
      signed int(field_size) y;
      unsigned int(field_size) width;
      unsigned int(field_size) height;
      unsigned int(8) mask_coding_method;
      if (mask_coding_method != 0)
        unsigned int(32) mask_coding_parameters;
      bit(8) data[];
    }
  }
}
```

In this structure, the `sample_type` field is used to identify the type of the sample.

The second type of sample may be used to describe the interpolation or the extrapolation for one or more regions. For example, when using extrapolation, the structure of this second type of sample may be the following:

```
aligned (8) class RegionSample {
  unsigned int sample_type = 0x01;
  unsigned int field_size = ((RegionTrackConfigBox.field_length_size & 1) + 1) * 16;
  // this is a temporary, non-parsable variable
  unsigned int(32) region_count;
  for (r=0; r < region_count; r++) {
    unsigned int(32) region_identifier;
    unsigned int(8) geometry_type;
    unsigned int(1) extrapolate_end;
    unsigned int(7) reserved;
    if (geometry_type == 0) {
      // point
      signed int(field_size) delta_x;
      signed int(field_size) delta_y;
    }
    else if (geometry_type == 1) {
      // rectangle
      signed int(field_size) delta_x;
      signed int(field_size) delta_y;
      unsigned int(field_size) delta_width;
      unsigned int(field_size) delta_height;
    }
    else if (geometry_type == 2) {
      // ellipse
      signed int(field_size) delta_x;
      signed int(field_size) delta_y;
      unsigned int(field_size) delta_radius_x;
      unsigned int(field_size) delta_radius_y;
    }
    else if (geometry_type == 3 || geometry_type == 6) {
      // polygon or polyline
      for (i=0; i < point_count; i++) {
        signed int(field_size) delta_px;
        signed int(field_size) delta_py;
      }
    }
    else if (geometry_type == 4) {
      // referenced mask
      signed int(field_size) delta_x;
      signed int(field_size) delta_y;
    }
    else if (geometry_type == 5) {
      // inline mask
      signed int(field_size) delta_x;
      signed int(field_size) delta_y;
    }
  }
}
```

In this structure, the `sample_type` field is used to identify the type of the sample.

In this structure, the `geometry_type` field may be omitted and its value retrieved from the definition of the region with its full characteristics in a sample of the first type.

Possibly, a region track may use two types of sample entry and two types of sample. One of these sample types may be used to describe the full characteristics of one or more regions. The second type of sample may be used to describe the interpolation or the extrapolation for one or more regions. These two types of sample may have structures similar to those described above. In these structures, the `sample_type` field may be omitted. The type of a sample may be retrieved using the associated sample entry.

In a variant, the evolution of the region described using extrapolation may be signaled by describing the evolving characteristics of the region at time greater than the composition timestamp of the processed video sample. For example, this may be at a time corresponding to the composition timestamp of the processed video sample increased by the duration of the region track sample. Possibly, a scaling factor may be used to encode the value of these characteristics to increase their precision and therefore to increase the precision of the evolution of the region.

In another variant, to increase the precision of the evolution of the region described using extrapolation, the evolution of the region may be signaled for a long duration. For example, the evolution of the region may be signaled for a duration equal to ten times the duration of the region track sample. Possibly, a varying multiplicative factor may be used. This multiplicative factor may depend on the duration of the region track sample and may depend on the evolution of the region. This multiplicative factor may be encoded in the sample. It may be specified for each region, for one or more regions or for one or more samples. For example, it may be specified in a sample group or in a sample entry.

In another variant, the evolution of a region described using extrapolation is signaled by describing the evolving characteristics of the region at the end of the extrapolation duration.

Possibly, the evolution of a region described using extrapolation may be signaled using a transformation such as a translation, a scaling, a rotation or a combination of these.

Possibly, the interpolation or the extrapolation of a region may be signaled using a Bezier curve or a Spline.

In some embodiments, some region may be described using interpolation while other ones may be described using extrapolation, using for each the solutions described above in a mixed way.

Figure 11 is a schematic block diagram of a computing device 1100 for implementation of one or more embodiments of the invention. The computing device 1100 may be a device such as a micro-computer, a workstation or a light portable device. The computing device 1100 comprises a communication bus connected to:

- a central processing unit 1101, such as a microprocessor, denoted CPU;

- a random access memory 1102, denoted RAM, for storing the executable code of the method of embodiments of the invention as well as the registers adapted to record variables and parameters necessary for implementing the method according to embodiments of the invention, the memory capacity thereof can be expanded by an optional RAM connected to an expansion port for example;

- a read only memory 1103, denoted ROM, for storing computer programs for implementing embodiments of the invention;

- a network interface 1104 is typically connected to a communication network over which digital data to be processed are transmitted or received. The network interface 1104 can be a single network interface, or composed of a set of different network interfaces (for instance wired and wireless interfaces, or different kinds of wired or wireless interfaces). Data packets are written to the network interface for transmission or are read from the network interface for reception under the control of the software application running in the CPU 1101;

- a graphical user interface 1105 may be used for receiving inputs from a user or to display information to a user;

- a hard disk 1106 denoted HD may be provided as a mass storage device;

- an I/O module 1107 may be used for receiving/sending data from/to external devices such as a video source or display.

The executable code may be stored either in read only memory 1103, on the hard disk 1106 or on a removable digital medium such as for example a disk. According to a variant, the executable code of the programs can be received by means of a communication network, via the network interface 1104, in order to be stored in one of the storage means of the communication device 1100, such as the hard disk 1106, before being executed.

The central processing unit 1101 is adapted to control and direct the execution of the instructions or portions of software code of the program or programs according to embodiments of the invention, which instructions are stored in one of the aforementioned storage means. After powering on, the CPU 1101 is capable of executing instructions from main RAM memory 1102 relating to a software application after those instructions have been loaded from the program ROM 1103 or the hard-disc (HD) 1106 for example. Such a software application, when executed by the CPU 1101, causes the steps of the flowcharts of the invention to be performed.

Any step of the algorithms of the invention may be implemented in software by execution of a set of instructions or program by a programmable computing machine, such as a PC ("Personal Computer"), a DSP ("Digital Signal Processor") or a microcontroller; or else implemented in hardware by a machine or a dedicated component, such as an FPGA ("Field-Programmable Gate Array") or an ASIC ("Application-Specific Integrated Circuit").

Although the present invention has been described hereinabove with reference to specific embodiments, the present invention is not limited to the specific embodiments, and modifications will be apparent to a skilled person in the art which lie within the scope of the present invention.

Many further modifications and variations will suggest themselves to those versed in the art upon making reference to the foregoing illustrative embodiments, which are given by way of example only and which are not intended to limit the scope of the invention, that being determined solely by the appended claims. In particular the different features from different embodiments may be interchanged, where appropriate.

Each of the embodiments of the invention described above can be implemented solely or as a combination of a plurality of the embodiments. Also, features from different embodiments can be combined where necessary or where the combination of elements or features from individual embodiments in a single embodiment is beneficial.

In the claims, the word “comprising” does not exclude other elements or steps, and the indefinite article “a” or “an” does not exclude a plurality. The mere fact that different features are recited in mutually different dependent claims does not indicate that a combination of these features cannot be advantageously used.

CLAIMS

1. A method of encapsulating video data in a media file, the video data comprising video samples, at least one region being identified on several samples, the method comprising:
   - generating a video track comprising successive video samples;
   - generating a region track comprising region track samples for describing regions identified in video samples;

     wherein for describing a region, the method comprises:
   - providing in a region track sample, synchronized with a video sample where the region is starting, a geometry of the region in the video sample where the region is starting;
   - providing in a region track sample information on the movement of the region, and an indication indicating that the region is extrapolated.

2. The method of claim 1, wherein the geometry of the region in the video sample where the region is starting, the information on the movement of the region and the indication indicating that the region is extrapolated are provided in a same region track sample synchronized with the video sample where the region is starting.

3. The method of claim 1, wherein:
   - the geometry of the region in the video sample where the region is starting is provided in a first region track sample synchronized with the video sample where the region is starting; and
   - the information on the movement of the region and the indication indicating that the region is extrapolated are provided in a second region track sample.

4. The method of any one claim 1 to 3, wherein a same region is identified in the region track samples by a same region identifier.

5. The method of any one claim 1 to 3, wherein the method further comprises:

- generating a region track sample synchronized with the first video sample where the region no longer appears or where the information on the movement of the region has changed.

6. The method of any one claim 1 to 4, wherein the information on the movement describes the movement of the region for a period of time corresponding to the duration of the sample.

7. A method of encapsulating video data in a media file, the video data comprising video samples, at least one region being identified on several samples, the method comprising:
   - generating a video track comprising successive video samples;
   - generating a region track comprising region track samples for describing regions identified in video samples;

     wherein for describing a region, the method comprises:
   - providing in a region track sample a starting geometry of the region corresponding to the geometry of the region in a video sample where the region is present;
   - generating a second region track sample comprising an ending geometry of the region corresponding to the geometry of the region in a video sample where movement of the region is ending, and an indication indicating that the region is interpolated; and wherein:
   - a decoding timestamp of the second region track sample being set lower than or equal to the presentation timestamp of the video sample following the video sample where the region is present; and
   - a presentation timestamp of the second region track sample being set to a value that when rounded is equal to the presentation timestamp of the video sample where the region is ending.

8. The method of claim 7, wherein a same region is identified in the region track samples by a same region identifier;

9. The method of claim 7, wherein the value corresponding to the presentation timestamp of the last video sample where the region is ending has any time value between the presentation timestamp of the last video sample where the

region is ending, and the presentation timestamp of the previous video sample.

10. The method of claim 7, wherein the starting geometry is provided in a first region track sample synchronized with the first video sample where the region is starting.

11. The method of claim 7, wherein the starting geometry is provided in the second region track sample.

12. The method of claim 7, wherein the second region track sample comprises a duration of the interpolation.

13. A method of reading encapsulating video data from a media file, the video data comprising video samples, at least one region being identified on several samples, the method comprising:
    - reading a video track comprising successive video samples;
    - reading a region track comprising region track samples for describing regions identified in video samples;

      wherein for a region, the method comprises:
    - reading in a region track sample, synchronized with a video sample where the region is starting, a geometry of the region in the video sample where the region is starting;
    - reading in a region track sample information on the movement of the region, and an indication indicating that the region is extrapolated; and
    - extrapolating the region based on the geometry and the information on the movement.

14. A method of reading video data from a media file, the video data comprising video samples, at least one region being identified on several samples, the method comprising:
    - reading a video track comprising successive video samples;
    - reading a region track comprising region track samples for describing regions identified in video samples;

      wherein for a region, the method comprises:

- reading in a region track sample a starting geometry of the region corresponding to the geometry of the region in a video sample where the region is present;
- reading a second region track sample comprising an ending geometry of the region corresponding to the geometry of the region in a video sample where movement of the region is ending, and an indication indicating that the region is interpolated;
- decoding the second region track sample according to a decoding timestamp of the second region track sample being set lower than the presentation timestamp of the video sample following the video sample where the region is present; and
- interpolating the region between the video sample where the region is present and the video sample identified by the presentation timestamp of the second region track sample.

15. A computer program product for a programmable apparatus, the computer program product comprising a sequence of instructions for implementing a method according to any one of claims 1 to 14, when loaded into and executed by the programmable apparatus.

16. A computer-readable storage medium storing instructions of a computer program for implementing a method according to any one of claims 1 to 14.

17. A computer program which upon execution causes the method of any one of claims 1 to 14 to be performed.

18. A device for encapsulating video data in a media file, the video data comprising video samples, at least one region being identified on several samples, the device comprising a processor configured for:
    - generating a video track comprising successive video samples;
    - generating a region track comprising region track samples for describing regions identified in video samples;

    wherein for describing a region, the processor is configured for:

- providing in a region track sample, synchronized with a video sample where the region is starting, a geometry of the region in the first video sample where the region is starting;
- providing in a region track sample information on the movement of the region, and an indication indicating that the region is extrapolated.

19. A device for encapsulating video data in a media file, the video data comprising video samples, at least one region being identified on several samples, the device comprising a processor configured for:
    - generating a video track comprising successive video samples;
    - generating a region track comprising region track samples for describing regions identified in video samples;

      wherein for describing a region, the processor is configured for:
    - providing in a region track sample a starting geometry of the region corresponding to the geometry of the region in a video sample where the region is present;
    - generating a second region track sample comprising an ending geometry of the region corresponding to the geometry of the region in a video sample where movement of the region is ending, and an indication indicating that the region is interpolated; and wherein:
    - a decoding timestamp of the second region track sample being set lower than the presentation timestamp of the video sample following the video sample where the region is present; and
    - a presentation timestamp of the second region track sample being set to a value corresponding to the presentation timestamp of the video sample where the region is ending.

20. A device for reading encapsulating video data from a media file, the video data comprising video samples, at least one region being identified on several samples, the device comprising a processor configured for:
    - reading a video track comprising successive video samples;
    - reading a region track comprising region track samples for describing regions identified in video samples;

      wherein for a region, the processor is configured for:

- reading in a region track sample, synchronized with a video sample where the region is starting, a geometry of the region in the first video sample where the region is starting;
- reading in a region track sample information on the movement of the region, and an indication indicating that the region is extrapolated; and
- extrapolating the region based on the geometry and the information on the movement.

21. A device for reading video data from a media file, the video data comprising video samples, at least one region being identified on several samples, the device comprising a processor configured for:
    - reading a video track comprising successive video samples;
    - reading a region track comprising region track samples for describing regions identified in video samples;

    wherein for a region, the processor is configured for:
    - reading in a region track sample a starting geometry of the region corresponding to the geometry of the region in a video sample where the region is present;
    - reading a second region track sample comprising an ending geometry of the region corresponding to the geometry of the region in a video sample where movement of the region is ending, and an indication indicating that the region is interpolated;
    - decoding the second region track sample according to a decoding timestamp of the second region track sample being set lower than the presentation timestamp of the video sample following the video sample where the region is present; and
    - interpolating the region between the video sample where the region is present and the video sample identified by the presentation timestamp of the second region track sample.

Intellectual Property Office

Application No:	GB2210191.9	**Examiner:**	Mr Philip Rogers
Claims searched:	1-6, 13, 15-17 (when appended to 1-6, 13), 18, 20	**Date of search:**	10 February 2023

Patents Act 1977: Search Report under Section 17

Documents considered to be relevant:

Category	Relevant to claims	Identity of document and passage or figure of particular relevance
X	1-6,13, 15-17 (when appended to claims 1-6, 13), 18, 20	US 2009/0024619 A1 (DALLMEIER et al.) See EPODOC abstract, figure 2, paragraphs [0019]-[0023]
A	-	GB 2596325 A (CANON KK) See EPODOC abstract and figure 4

Categories:

X	Document indicating lack of novelty or inventive step	A	Document indicating technological background and/or state of the art.
Y	Document indicating lack of inventive step if combined with one or more other documents of same category.	P	Document published on or after the declared priority date but before the filing date of this invention.
&	Member of the same patent family	E	Patent document published on or after, but with priority date earlier than, the filing date of this application.

Field of Search:

Search of GB, EP, WO & US patent documents classified in the following areas of the UKC^X :

Worldwide search of patent documents classified in the following areas of the IPC

G06F; G06T; H04N

The following online and other databases have been used in the preparation of this search report

WPI, EPODOC

International Classification:

Subclass	Subgroup	Valid From
H04N	0021/845	01/01/2011
G06F	0016/41	01/01/2019
G06F	0016/783	01/01/2019