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

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
09 January 2025 (09.01.2025)

WIPO | PCT

(10) International Publication Number
WO 2025/010337 A1

(51) International Patent Classification:
H04N 19/46 (2014.01) *H04N 19/85* (2014.01)
H04N 19/70 (2014.01)

(21) International Application Number:
PCT/US2024/036717

(22) International Filing Date:
03 July 2024 (03.07.2024)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
63/511,800 03 July 2023 (03.07.2023) US
63/588,188 05 October 2023 (05.10.2023) US

(71) Applicant: **DOLBY LABORATORIES LICENSING CORPORATION** [US/US]; 1275 Market Street, San Francisco, California 94103 (US).

(72) Inventor: **SULLIVAN, Gary J.**; c/o Dolby Laboratories, Inc., 1275 Market Street, San Francisco, California 94103 (US).

(74) Agent: **KONSTANTINIDES, Konstantinos** et al.; Intellectual Property Group, 1275 Market Street, San Francisco, California 94103 (US).

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

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

(54) Title: SIGNALING OF PROCESSING ORDER FOR METADATA MESSAGING IN VIDEO CODING

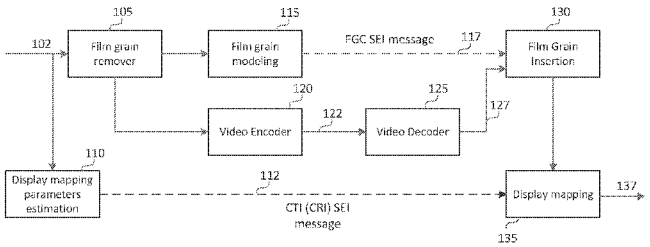


FIG. 1A

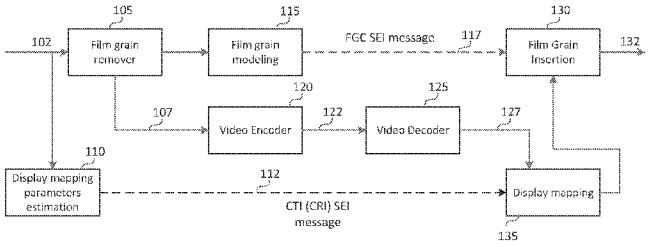


FIG. 1B

(57) Abstract: Methods, systems, and bitstream syntax are described for determining a preferred processing order of metadata messaging, such as supplemental enhancement information (SEI) messaging in MPEG video coding. Examples are provided to address issues related to backwards compatibility with legacy systems. For example, proposed messaging allows encoders to isolate messages critical to a decoder's implementation and assign importance to each SEI message so that backwards compatibility is preserved.

WO 2025/010337 A1

[Continued on next page]

LU, LV, MC, ME, MK, MT, NL, NO, PL, PT, RO, RS, SE, SI, SK, SM, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, KM, ML, MR, NE, SN, TD, TG).

Published:

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

— *before the expiration of the time limit for amending the claims and to be republished in the event of receipt of amendments (Rule 48.2(h))*

SIGNALING OF PROCESSING ORDER FOR METADATA MESSAGING IN VIDEO CODING

CROSS-REFERENCE TO RELATED APPLICATIONS

[0001] This patent application claims the benefit of priority to U.S. Provisional patent application Ser. No. 63/588,188, filed on 05 October 2023, and U.S. Provisional patent application Ser. No. 63/511,800, filed on 03 July 2023, each of which is incorporated by reference in its entirety.

TECHNOLOGY

[0002] The present document relates generally to images and video. More particularly, an embodiment of the present invention relates to signaling a processing order for metadata messaging in images and video sequences.

BACKGROUND

[0003] As described in Annex D of the AVC and HEVC standards, or in H.274 (also referred to as VSEI) (Ref. [1-4]), Supplemental Enhancement Information (SEI) messages in a coded video bitstream assist in processes related to decoding, display, or other purposes in a video processing pipeline. Despite their extensive use, at least up to now, conforming decoders (e.g., an AVC, HEVC, or VVC decoder) are not required to process any SEI messaging to comply with the picture decoding process specifications of any of the MPEG video coding standards (such as AVC, HEVC, and VVC), although such messages may be used for system control functionality such as indicating the buffering and timing information for the video bitstream.

[0004] In a typical video bitstream, multiple SEI messages may co-exist; however, none of the existing video coding standards define the processing order of such messaging. For some SEI messages, outside of information embedded in syntax elements, there is no specific processing defined in the standards. Examples of such messages include SEI messaging defining a mastering display color volume or content light-level information. For some other SEI messages, such as those describing film grain characteristics (FGC) or color remapping information, in addition to the syntax elements, additional post-processing may also be defined. For the later, the final video output may vary depending on the processing order of these SEI messages. As appreciated by the inventors, improved techniques for signaling the processing order of SEI messaging are described herein.

[0005] The term "metadata" herein relates to any auxiliary information transmitted either as part of the coded bitstream or along with it that assists a decoder to render or interpret one or more decoded images. Such metadata may include, but are not limited to, color space or gamut information, reference display parameters, and film grain modeling parameters, as those described herein. While examples presented herein may refer to SEI messaging as it relates to MPEG-based video coding standards, a person of ordinary skill would appreciate that the techniques discussed herein are applicable to any such metadata messaging and any audio or video coding format (e.g., AV1, AVS, VC-1 and the like).

[0006] The approaches described in this section are approaches that could be pursued, but not necessarily approaches that have been previously conceived or pursued. Therefore, unless otherwise indicated, it should not be assumed that any of the approaches described in this section qualify as prior art merely by virtue of their inclusion in this section. Similarly, issues identified with respect to one or more approaches should not assume to have been recognized in any prior art on the basis of this section, unless otherwise indicated.

BRIEF DESCRIPTION OF THE DRAWINGS

[0007] An embodiment of the present invention is illustrated by way of example, and not in way by limitation, in the figures of the accompanying drawings and in which like reference numerals refer to similar elements and in which:

[0008] FIG. 1A and FIG 1B depict examples of video processing pipelines when metadata includes multiple SEI messages and their post-processing order may affect video output;

[0009] FIG. 2 depicts an example processing pipeline, according to prior art, when an SEI Priority (or Processing) Order of Messaging (POM) message is available in a decoder;

[00010] FIG. 3A depicts an example processing pipeline with an SEI processing order message using an additional wrapper loop, according to an embodiment of this invention; and

[00011] FIG. 3B depicts an example processing pipeline with an SEI processing order message using a wrapper loop and importance flags, according to an embodiment of this invention.

DESCRIPTION OF EXAMPLE EMBODIMENTS

[00012] Example embodiments that relate to a processing order of metadata are described herein. In the following description, for the purposes of explanation, numerous specific details are set forth in order to provide a thorough understanding of the various embodiments

of present invention. It will be apparent, however, that the various embodiments of the present invention may be practiced without these specific details. In other instances, well-known structures and devices are not described in exhaustive detail, in order to avoid unnecessarily occluding, obscuring, or obfuscating embodiments of the present invention.

SUMMARY

[00013] Example embodiments described herein relate to signaling of a processing order of metadata (such as SEI messaging) in video coding. To address the issue of backward compatibility with legacy systems, where newly established metadata signaling may have unpredictable results, example embodiments allows encoders to isolate messages critical to a decoder's implementation and assign an importance value to each metadata message, so that backwards compatibility is preserved.

EXAMPLES OF SEI MESSAGING FLOW PROCESSING

[00014] In reference to existing video coding standards, as in AVC, HEVC and VVC (Refs. [1-4]) (collectively to be referred to as MPEG or MPEG video standards), FIG. 1A and FIG. 1B depict example processing pipelines when the SEI messaging includes messaging related to both film grain characteristics (FGC) and color remapping information (CRI) or color transform information (CTI).

[00015] As depicted in FIG. 1A and FIG. 1B, given input video 102, an encoder may generate two sets of SEI messages: a) CTI/CRI SEI metadata (112) related to preferred display parameters (e.g., as generated by block 110), and b) film grain characteristics (117) to be added to a decoder, to emulate film grain that was removed by the encoder (e.g., by block 105), to improve coding efficiency, and modeled by a film grain modeling block (115). Thus, video encoder (120) compresses the output of the film-grain remover (107) to generate a compressed bitstream 122. The compressed bitstream 122 and the metadata (112 and 117) are transmitted to a decoder.

[00016] As depicted in FIG. 1A, after decoding (125), the decoded output (127) is passed to a film-grain insertion unit 130, which generates and adds film grain noise using the information embedded in the FGC metadata (117). The output of the film grain insertion unit (130) is then passed to a display mapping unit 135, which best maps the dynamic range and color gamut of the input stream to a target display by using the information embedded in the CRI/CTI metadata (112). The output is signal 137.

[00017] As depicted in FIG. 1B, after decoding (125), compared to FIG. 1A, the order of processing the two sets of metadata is reversed. Now, the decoded output (127) is processed first by the display mapping unit 135, which maps the dynamic range and color gamut of the decoded stream to the target display by using the information embedded in the CRI/CTI metadata (112). Next, the output of display mapping unit is passed to the film-grain insertion unit 130, which generates and adds film grain noise using the information embedded in the FGC metadata (117), and generates output video 132.

[00018] Because of the different ordering of display mapping and film-grain addition, it is expected that video outputs 132 and 137 will be different, especially if the target display differs significantly from the intended display, therefore it is deemed important to specify in the bitstream the proper order of processing metadata so that a decoder output matches as close as possible the creative intent of the content creators.

Examples for signaling processing order in SEI messaging

[00019] Example embodiments presented herein relate to specifying the processing order among multiple metadata sets when multiple metadata-related messages co-exists in the video coding standards, such as in AVC, HEVC and VVC. In particular, high level syntax (HLS) is proposed to specify such an order. Such high-level syntax can be inserted at a variety of levels of the coded bitstream, say, without limitation, in the video parameter set (VPS), the picture parameter set (PPS), the sequence parameter set (SPS), an adaptation parameter set (APS), a picture header (PH), as separate SEI messaging, and the like.

[00020] In an embodiment, such high level syntax should include the following information: 1) what SEI messages might exist in the coded video stream (CVS); 2) whether processing order is relevant to a particular SEI message; and, 3) if relevant, what is the processing order of the particular SEI messages.

[00021] The processing (or priority) order of an SEI message may be specified in a variety of ways, such as: 1) an absolute order; for example, in one embodiment, a message with a smaller processing/priority order should be processed earlier than one with a larger processing/priority order, 2) a relative order: in one embodiment, the processing order may be specified as a dependency on the processing order of another SEI message. When a coding standard defines processing certain metadata as mandatory in the decoding process, the processing order of such message should always be specified. If a conformance point of certain metadata message is defined, then the processing order of such metadata message should also be specified.

[00022] Table 1 depicts an example to specify the processing order among SEI messages (Ref.[9]). In this example, an "SEI processing order" SEI message, a proposed new SEI message, may specify which SEI messages are part of the CVS where processing order matters, and their processing order. The SEI messages of interest are identified through their unique payload type (see Refs [1-4]).

Table 1. SEI processing order SEI message

sei_processing_order(payloadSize) {	**Descriptor**
po_num_sei_msg_types	u(16)
for(i = 0; i < num_sei_msg_types; i++) {	
po_sei_payload_type[i]	u(16)
po_sei_order[i]	u(8)
}	
}	

po_num_sei_msg_types specifies the number of types of SEI messages for which information is provided in the SEI processing order SEI message.

po_sei_payload_type[i] indicates the payload type value of the i-th type of SEI message for which information is provided in the SEI processing order SEI message. The values of po_sei_payload_type[m] and po_sei_payload_type[n] shall not be identical when m is not equal to n.

po_sei_order[i] provides the absolute order of SEI messages with payload Type equal to po_sei_payload_type[i]. The values of po_sei_order[m] and po_sei_order[n] shall not be identical when m is not equal to n. If the values of po_sei_order[m] is less than the values of po_sei_order[n], the SEI message with po_sei_payload_type[m] is processed before the SEI message with po_sei_payload_type[n]

[00023] In Table 1, instead of the absolute **po_sei_order**[i] parameter, one could also describe a relative order, as in po_sei_rel_order[i][j], where the order of SEI message *i* is defined relative to the order of SEI message *j*. For example, po_sei_rel_order[3][1] = 2, may define that the priority of SEI message "3" is the priority order of SEI message "1" plus (or, alternatively, minus) 2. In such a scenario, one can assume that all SEI messages with priority processing start at a default priority value (say, 1), and then they are adjusted accordingly.

[00024] Table 1 can also be simplified by removing the **po_sei_order**[i] parameter. In such a scenario, the encoder already lists the **po_sei_payload_type**[i] values in an implied and predetermined priority order (e.g., lowest to highest or highest to lowest). Such an

approach forces a priority even if there is no such requirement (for example, SEI messages "2" and "4" could have the same priority but would be listed one after the other), thus limiting possible parallelization of SEI messaging-related tasks. The implicit order should be made when a new CVS starts, where all SEI messages that are present are listed.

[00025] The SEI manifest SEI message in HEVC and VVC conveys information on SEI messages that are indicated as expected (i.e., likely) to be present or not present in the bitstream; however, no priority among these SEI messages is indicated. Table 2 (Ref. [9]) shows another example to extend the SEI manifest SEI and include processing order information. The additional syntax, over the existing SEI manifest message, is shown in an *italic* font.

Table 2. Example 1 of extending the SEI manifest SEI message

sei_manifest(payloadSize) {	**Descriptor**
manifest_num_sei_msg_types	u(16)
for(i = 0; i < manifest_num_sei_msg_types; i++) {	
manifest_sei_payload_type[i]	u(16)
manifest_sei_description[i]	u(8)
if (manifiest_sei_description[i] = = 1) {	
manifest_sei_po_flag*[i]*	*u(1)*
if (manifest_sei_po_flag[i])	
manifest_sei_order*[i]*	*u(8)*
}	
}	
}	

manifest_sei_po_flag[i] equals to 1 specifies the SEI message with payloadType equal to manifest_sei_payload_type[i] requires processing order information.

manifest_sei_po_flag[i] equals to 0 specifies the SEI message with payloadType equal to manifest_sei_payload_type[i] does not require processing order information.

manifest_sei_order[i] provides the absolute order on SEI messages with payloadType equal to manifest_sei_payload_type[i]. The values of manifest_sei_order[m] and manifest_sei_order[n] shall not be identical when m is not equal to n. If the values of manifest_sei_order[m] is less than the values of manifest_sei_order[n], the SEI message with manifest_sei_payload_type[m] is processed before the SEI message with manifest_sei_payload_type[n].

NOTE: as described earlier, if the processing order is defined implicitly (e.g., by the order SEI messages are listed in **manifest_sei_po_flag**[i]) , then the syntax element manifest_sei_order[i] is not required and can be deleted.

[00026] Table 3 depicts an alternative approach to extend SEI manifest SEI supporting backward compatibility with legacy decoders already supporting older versions of the SEI manifest message.

Table 3. Example 2 of extending the SEI manifest SEI message

sei_manifest(payloadSize) {	**Descriptor**
manifest_num_sei_msg_types	u(16)
for(i = 0; i < manifest_num_sei_msg_types; i++) {	
manifest_sei_payload_type[i]	u(16)
manifest_sei_description[i]	u(8)
}	
if (more_data_in_payload ()) {	
if (payload_extension_present( )) {	
for(i = 0; i < manifest_num_sei_msg_types; i++) {	
if (manifiest_sei_description[i] = = 1) {	
***manifest_sei_po_flag**[i]*	*u(1)*
if (manifest_sei_po_flag[i])	
***manifest_sei_order**[i]*	*u(8)*
}	
}	
}	
}	
}	

[00027] From HEVC, more_data_in_payload( ) is specified as follows:

– If byte_aligned( ) is equal to TRUE and the current position in the sei_payload( ) syntax structure is 8 * payloadSize bits from the beginning of the sei_payload( ) syntax structure, the return value of more_data_in_payload( ) is equal to FALSE.

– Otherwise, the return value of more_data_in_payload( ) is equal to TRUE.

[00028] From HEVC, payload_extension_present( ) is specified as follows:

– If the current position in the sei_payload( ) syntax structure is not the position of the last (least significant, right-most) bit that is equal to 1 that is less than 8 * payloadSize bits from the beginning of the syntax structure (i.e., the position of the payload_bit_equal_to_one syntax element), the return value of payload_extension_present( ) is equal to TRUE.

– Otherwise, the return value of payload_extension_present( ) is equal to FALSE.

[00029] Table 3 allows existing decoders that implement the SEI manifest messaging to continue implementing it as in the past by ignoring the SEI processing order syntax. On the other hand, decoders implemented after the adoption of the new syntax may take advantage of the updated syntax and related processing ordering information.

[00030] As discussed in relation to Table 1, the processing order of SEI messages in Tables 2 and 3 may also be implied or can be defined relative to the processing order of other SEI messages.

[00031] In an embodiment, Table 1 and its semantics may be modified as depicted for the description of Table 4. In this embodiment the semantics depend on the syntax element MaxNumPayloadTypes to address the issue that the number of possible payload types varies across coding standards (e.g., as of this draft, there are 77 SEI messages for AVC, 67 for HEVC, and 39 for VVC), plus the actual number may also increase in future versions of a standard, and to discourage signaling more payloadTypes than could possibly exist.

[00032] Alternatively (or additionally), the actual valid payloadTypes for each standard could be specified as well in a specific list (say, list SeiAssociatedSeiList). For example, one could add this paragraph in the VVC or VSEI standards (Ref. [3-4]):

"Use of the SEI message processing order SEI message syntax

For purposes of interpretation of the SEI message processing order SEI message, the list SeiAssociatedSeiList is set to consist of the payloadType values 3, 4, 5, 19, 137, 142, 144, 147, 148, 149, 165, 177, 210, and 211, inclusive."

In another embodiment, this list could be part of the SEI Processing order SEI message or other header information, modified by the encoder as needed.

Table 4. SEI processing order SEI message

sei_processing_order(payloadSize) {	Descriptor
po_num_sei_payload_types	u(16)
for(i = 0; i < num_sei_payload_types; i++) {	
po_sei_payload_type[i **]**	u(16)
po_sei_processing_order[i **]**	u(8)
}	
}	

po_num_sei_payload_types specifies the maximum number payloadType values for which syntax elements po_sei_payload_type[i] and po_sei_processing_order[i] are present. The

value of po_num_sei_payload_types shall be less than or equal to the MaxNumPayloadTypes, which defines the maximum number of payloadType values.

Note: if the syntax element SeiAssociatedSeiList is added, then the above definition can be replaced with:

po_num_sei_payload_types specifies the maximum number payloadType values for which syntax elements po_sei_payload_type[i] and po_sei_processing_order[i] are present, as specified by an encoder (e.g., from among the values in the list SeiAssociatedSeiList).

po_sei_payload_type[i] specifies the value of payloadType for the i-th SEI message. The value of po_sei_payload_type[m] shall not equal the value of po_sei_payload_type[n] when m is not equal to n.

Note: if a list of allowed payload types SeiAssociatedSeiList is specified, then add: The value of po_sei_payload_type[i] shall be equal to a payloadType value in SeiAssociatedSeiList.

po_sei_processing_order[i] indicates the preferred order of processing an SEI message with payloadType equal to po_sei_payload_type[i]. The value of po_sei_processing_order[m] shall not equal the value of po_sei_processing_order[n] when m is not equal to n. po_sei_processing_order[m] greater than 0 and less than po_sei_processing_order[n] indicates that any SEI message with payloadType equal to po_sei_payload_type[m] should be processed before any SEI message with payloadType equal to po_sei_payload_type[n]. po_sei_processing_order[i] equal to 0 specifies that the preferred order of processing an SEI message with payloadType equal to po_sei_payload_type[i] is determined by external means not specified in the metadata specification standard.

To constrain the ordering of the SEI messages in the SEI processing order SEI message, it may be specified that po_sei_processing_order[0] shall be equal to 0 and that for i > 0, po_sei_processing_order[i] shall be equal to po_sei_processing_order[i − 1] or to po_sei_processing_order[i − 1] + 1.

[00033] FIG. 2 depicts an example processing pipeline according to prior art (Ref. [9]) when an SEI processing order of messaging (POM) message is available. Starting from step 210, a decoder may search SEI payloadTypes of incoming SEI messages to detect whether an SEI POM message is present. If there is no such message, processing (215) of SEI messaging continues without taking into consideration their order. If an SEI POM message is present,

then in step 220, the decoder reads the number of selected SEI messages for which priority of processing is defined. Next, in step 225, for each such message, the decoder reads its identity (say, its payloadType) and its priority, typically an integer value with predetermined ordering significance (e.g., whether smaller numbers or larger numbers have higher priority). In an embodiment, priority may also be implied by the order of how the select SEI messages are listed. Finally, in step 230, the decoder processes these select SEI messages with their proper priority.

Embodiments Supporting Backward Compatibility

[00034] During the development of coding standards, backward compatibility with existing deployed video decoders and decoding systems is an important consideration. In various system environments such as those using the above-described video coding standards, there is a large number of legacy deployed decoders and decoding systems that would not understand any newly defined metadata signaling sent by a new encoding system (unless they are updated, e.g., by installation of a software or firmware update). It is thus important to consider how these legacy decoders and systems would react if such new signaling is encountered.

[00035] SEI messages are typically prefixed with (or otherwise associated with) a payload type code that identifies the purpose of the SEI message and a payload length value that identifies the amount of data (e.g., in units of bytes) that is sent for the SEI message. In some cases, there also is an extension mechanism that provides extensibility for individual SEI messages, either by indications sent within the syntax of the SEI message or by appending additional data after the specified syntax, the presence of which can be detected using some mechanism. For example, there are special provisions for detecting appended additional data in the HEVC and VVC standards that can enable a decoder to detect the precise number of bits of such appended data, although the payload length value for conveying SEI messages in the context of these standards is sent in units of bytes.

[00036] An SEI message that indicates a post-processing operation, one that has been defined in a legacy version of the core decoding specification or related system specification, ordinarily, has a meaning that has been defined by the previously existing specification documents and implementations: its input has been defined to be the decoded picture that is output from the core video decoding process (i.e. not having its input be the output of some other post-processing operation) and, typically, its output is to be used for display. Similarly, SEI messages that indicate properties of the decoded video often have legacy definitions that

indicate that they are for indicating properties of the decoded picture that is output from the core video decoding process (i.e., not for indicating the properties of the pictures produced by some post-processing operation).

[00037] When metadata is present for multiple post-processing operations or for some post-processing operation(s) and also some properties of decoded video content, it may become confusing to interpret these multiple metadata indicators. If some new SEI message is defined that attempts to redefine what is the input and output of a post-processing operation or what is the video data that has the described properties, such a new SEI message may be ignored by legacy decoders and decoding systems, so some indicated post-processing operation may be applied with a different input than what the transmitting system intended to be used, or the decoding system may interpret the video that is directly produced by the core decoding process as having properties that are different from what the encoding system intended (since an intended post-processing operation may not be applied by the legacy decoding system).

[00038] In recent years, the Joint Video Experts Team (JVET), a collaboration between ITU-T SG 16 WP3 and ISO/IEC JTC 1/SC 29 (also known as MPEG), has been working to define a new SEI message for signaling processing order among SEI messages (Refs.[5-8, 10]).

[00039] In Ref. [5], the proposal included a sequence of **po_sei_payload_type**[i] coded as u(16) and **po_sei_processing_order**[i] coded as u(8), where u(n) denotes unsigned integer n-bit coding. There was a special treatment if po_sei_processing_order[i] = 0. When multiple SEI messages have the same payload type code, this approach can't distinguish between them.

[00040] In Ref. [6], there is again a sequence of **po_sei_payload_type**[i] and **po_sei_processing_order**[i], both coded as u(16). Equal processing order indicates that there is no preferred order between them. There is no special treatment of processing order value 0.

[00041] In Ref. [7], a loop with payloadType using **po_sei_payload_type**[i] coded as u(16) was added, and there is special prefix handling for T.35. For the processing order indicator, equal order indication values are allowed and gaps are allowed.

[00042] In Ref. [8], **po_sei_payload_type**[i] is coded as u(15). Here, SEI messages may be identified by prefix information that contains a sufficient number of bytes to uniquely identify which SEI message is indicated. Because the SEI messages are present along with the processing order SEI message, they could be detected and possibly interpreted by legacy

decoders. It is possible, that this interpretation could produce different results from what the encoding system originally intended.

A "Wrapping" Approach

[00043] In a first embodiment, operations and property indicators that are intended to be applied with a different input or different output from what has previously been specified in a legacy specification are to be "wrapped" inside of a new syntax structure. This is conceptually similar to the "nesting" concept previously used for applying SEI messages to different "layers" or "output layer sets" in a layered scalable representation. The advantage of this is that when using this approach, the SEI messages that are "wrapped" as part of an indicated multi-stage operation will not be interpreted by a legacy decoding system. Thus, they will just be ignored rather than misinterpreted. This behaviour is preferable to having the SEI messages applied selectively and in an unknown order, since it is ordinarily expected that SEI messages can be ignored if they cannot be properly interpreted.

[00044] Table 5 below depicts an example syntax of such an embodiment. Here the wrapping is "implicit." In other words, the indicated SEI messages are always present within the new syntax structure. A legacy decoder can decide to process SEI messages only if they are specified outside of this processing order SEI. As another example, a legacy receiver that was designed before the processing order message is deployed won't see anything that is wrapped inside it.

Table 5. Example processing order SEI message

sei_processing_order(payloadSize) {	**Descriptor**
for(i = 0, b = 0; b < payloadSize; i++) {	
sei_message( ) /* read SEI message */	
b += size of the above sei_message( ) syntax structure	
po_sei_processing_order[i]	u(8)
}	
}	

Note: An example of the syntax of the function sei_message() can be found in Ref. [1], section 7.3.2.3.1 "Supplemental enhancement information message syntax." Reading sei_message() allows a decoder to determine all related information for the SEI message, including the payload type, the payload size, and any other syntax parameters of the specified SEI message in the bitstream.

[00045] In another embodiment, wrapping of legacy SEI messaging could be explicit. In this approach, the wrapper SEI message would contain a loop, and each iteration of the loop would be for one stage of a sequence of stages, where the output of one stage is considered the input of the next stage. Any property-indication messages that are present would be considered to apply at the corresponding stage of the sequence of stages.

[00046] FIG. 3A depicts an example processing pipeline using SEI process order messaging and an additional wrapper loop. As depicted in FIG. 3A, if processing order SEI message is available (305), it is determined (310) if a selected SEI message is inside a "wrapper" within the SEI processing order message. If it is, next (315), the decoder reads the SEI message information (say, payload type, payload size, and associated syntax parameters) within this processing order message, otherwise it will read the SEI message information from the bitstream. Next, the decoder reads the preferred processing order (320). When reading of all SEI messages with processing order is complete, in step 330, a decoder has the option to either process all read SEI messages in the preferred processing order or treat the "wrapped" SEI messages differently. For example, a legacy decoder may decide to ignore all wrapped SEI messages, while another decoder may decide to apply the processing order to all SEI messages identified in the processing order SEI message.

[00047] Table 6 below depicts an example syntax of an explicit "wrapping" approach.

Table 6. Example processing order SEI message with explicit wrapper approach

sei_processing_order(payloadSize) {	**Descriptor**
for(i = 0, b = 0; b < payloadSize; i++) {	
po_sei_wrapping_flag[i]	u(1)
reserved_alignment_7bits	u(7)
if(po_sei_wrapping_flag[i]) {	
sei_message( )	
b += size of the above sei_message( ) syntax structure + 1 byte	
} else {	
po_sei_prefix_flag[i] /*start prefix payload processing	u(1)
po_sei_payload_type[i]	u(14)
… /* other optional prefix processing (per Table 7) */	
}	
po_sei_processing_order[i]	u(8)
}	
}	

po_sei_wrapping_flag[i] equal to 0 indicates that the associated SEI message for the processing order indicated by po_sei_processing_order[i] should be present outside of the SEI processing order SEI message.

If **po_sei_wrapping_flag**[i] is equal to 1, then a decoder reads an SEI message that is "wrapped" within the SEI processing order SEI message.

For each value of i that has po_sei_processing_order[i] not equal to 0, the input for the i-th SEI message is intended to be the output of the SEI message which has the processing order equal to po_sei_processing_order[i] − 1 .

To constrain the ordering of the SEI messages in the SEI processing order SEI message, it may be specified that po_sei_processing_order[0] shall be equal to 0 and that for i > 0, po_sei_processing_order[i] shall be equal to po_sei_processing_order[i − 1] or to po_sei_processing_order[i − 1] + 1.

[00048] In another embodiment, the syntax could contain an importance indication for whether each SEI message in the list is important for proper interpretation of the sequence of SEI messages (e.g., an important post-processing operation that could alter the size of the pictures or the representation of colour in the pictures) or not as important (e.g., a relatively unimportant property-indication message, say a region of interest message). This indication would be useful, as a decoder may be able to ignore some messages that it does not understand or is unable to apply, but should not apply any stages of an indicated sequence if it will not apply one or more of the fundamentally important stages of that sequence. In some

cases, a post-processing operation may be identified as non-fundamental if it does not change the basic characteristics of the decoded video. A property-indication message could also be identified as being important if it would not be appropriate to use the video content without understanding the indicated property (e.g., if it identifies the way colour information should be interpreted after applying a post-processing operation).

[00049] FIG. 3B depicts depicts an example processing pipeline using SEI processing order messaging with the wrapper loop and importance flags. It is a variation of FIG. 3A, so some steps in FIG. 3A are omitted. As depicted in FIG. 3B, if there is an SEI processing order message, for each SEI message within this message, in step 302 the decoder reads the importance flag and the wrapper flag. Processing continues as in FIG. 3A, until step 340. In step 340, if a message is deemed important, but the decoder does not know how to properly handle it, then the whole SEI processing order message may be skipped. Otherwise, processing continues as in FIG. 3A. This embodiment provides the ability for three classes of decoders: legacy decoders with no understanding of any SEI messages classified as "wrapped," which can be ignored, a recent decoder which can handle all messages identified in the processing order SEI message, and a third type of decoder which can handle all messages that are not deemed "important," but would rather process sequentially or in some other order these important messages, instead of risking processing them in the specified order.

[00050] An example of such syntax is presented in Table 7. Note that in Table 7, the syntax after the first "else" statement, as it relates to the prefix payload, is more or less identical with the syntax and semantics proposed in Ref.[8], thus, only the new semantics are described here.

Table 7. Example SEI processing order SEI message with a wrapping approach and an importance flag

sei_processing_order(payloadSize) {	**Descriptor**
for(i = 0, b = 0; b < payloadSize; i++) {	
po_sei_wrapping_flag[i]	u(1)
po_sei_importance_flag[i]	u(1)
if(po_sei_wrapping_flag[i]) {	
reserved_alignment_6bits	u(6)
sei_message( )	
b += size of the above sei_message( ) syntax structure + 1 byte	
} else {	
po_sei_prefix_flag[i] /*start prefix payload processing	u(1)
po_sei_payload_type[i]	u(13)
if(po_sei_prefix_flag[i]) {	
po_num_prefix_bytes[i]	b(8)
for(j = 0; j < po_num_prefix_bytes[i]; j++)	
po_prefix_byte[i][j]	b(8)
b += po_num_prefix_bytes[i] + 3	
} else	
b += 2	
} /*end prefix payload processing	
po_sei_processing_order[i]	u(8)
}	
}	

[00051] **po_sei_importance_flag**[i] indicates the importance value of a SEI messages with a processing order for backward compatibility. po_sei_importance_flag[i] equal to 1 indicates that the i-th SEI message is important for purposes of applying the sequence of SEI message processing operations in the correct order. po_sei_importance_flag[i] equal to 0 indicates that the i-th SEI message is not important.

If the decoding system cannot interpret or does not support any indicated SEI message that has po_sei_importance_flag[i] equal to 1, it should ignore the entire SEI processing order SEI message.

When present, **reserved_alignment_6bits** has no meaning and is included in the syntax only so that the following syntax bits will be at a byte-aligned location. For the sake of enabling checking of the bitstream format or for extensibility to be able use some other value in the future, it may be prescribed that encoders shall set this syntax element to some particular value. For example, it may be required to be equal to 0.

po_sei_processing_order[i] indicates the preferred order of processing of the i-th SEI message for which preferred processing order information is provided in the SEI processing order SEI message.

For each value of i that has po_sei_processing_order[i] not equal to 0, the input for the i-th SEI message is intended to be the output of the SEI message processing stage which has the processing order equal to po_sei_processing_order[i] − 1 .

To constrain the ordering of the SEI messages in the SEI processing order SEI message, it may be specified that po_sei_processing_order[0] shall be equal to 0 and that for i > 0, po_sei_processing_order[i] shall be equal to po_sei_processing_order[i − 1] or to po_sei_processing_order[i − 1] + 1.

Note: The use of prefix payload processing is described in Ref. [4] as follows: "**po_sei_processing_order**[i] indicates the preferred order of processing of the i-th SEI message type for which preferred processing order information is provided in the SEI processing order SEI message. For any two different integer values of m and n that are greater than or equal to 0, po_sei_processing_order[m] less than po_sei_processing_order[n] indicates any SEI message type with payloadType equal to po_sei_payload_type[m] and, when present, bytes po_prefix_byte[m][p] for p ranging from 0 to po_num_prefix_bytes[m] − 1, inclusive, should be processed before any SEI message type with payloadType equal to po_sei_payload_type[n], and, when present, bytes po_prefix_byte[n][q] for q ranging from 0 to po_num_prefix_bytes[n] − 1, inclusive, and po_sei_processing_order[m] equal to po_sei_processing_order[n] indicates that there is no preferred order of processing between the SEI message types. When there are multiple SEI messages with the same values of po_sei_payload_type[i], po_num_prefix_bytes [i], and bytes po_prefix_ byte[i][j] for j ranging from 0 to po_num_prefix_bytes [i] − 1, inclusive, they shall have the same value of po_sei_processing_order[i]."

po_sei_wrapping_flag[i] equal to 0 indicates that the i-th indicated SEI message should be present outside of the SEI processing order SEI message and has payloadType equal to po_sei_payload_type[i]. However, if po_sei_wrapping_flag[i] is equal to 0 and no SEI message is present with payloadType equal to po_sei_payload_type[i], the following applies.

– If po_sei_importance_flag[i] is equal to 1, the decoder should ignore the entire SEI processing order SEI message.

– Otherwise, the decoder should ignore all data associated with the loop variable value of i.

If **po_sei_wrapping_flag**[i] is equal to 1, then a decoder reads an SEI message that is "wrapped" within the SEI processing order SEI message.

[00052] The importance flag and the wrapping flag are independent of each other. In a variation of the example in Table 7, one can remove the wrapping flag (e.g., po_sei_wrapping_flag[i]) and associated operations. As before, if the decoding system cannot interpret or does not support any indicated SEI message that has po_sei_importance_flag[i] equal to 1, it can ignore the entire SEI processing order SEI message. Alternatively, an encoder may use the wrapping flag only for SEI messages it believes a legacy decoder should not see.

[00053] In another variation of the example in Table 7, in an embodiment, the importance and payload flags of all messages in the processing order message may be read in the beginning, in a separate loop, before reading the whole information related to the wrapped SEI messages. This may provide some advantage in decoding, since a decoder may quit as soon as a particular message has po_sei_importance_flag[i] =1, and the decoder system cannot interpret or does not support the indicated SEI message.

[00054] In another embodiment, one may replace the importance flag, which only takes two values, with a syntax element describing importance type (say, **po_sei_importance_type**[i], coded as u(2)). This allows an encoder to classify SEI messages with additional weights of importance, such as: important, unimportant, very important, undetermined, and the like, and may provide additional flexibility on how messages are handled by legacy and non-legacy decoders.

[00055] The sequential operation wrapping approach (e.g., see Table 5) could be combined with the "nesting" approach in the case of using a combination of the sequential operation indication together with a layered coding approach. In this case, the existing scalable nesting syntax can be used (e.g., see scalable nesting SEI message syntax in Annex D of HEVC (Ref.[2])) with the SEI processing order SEI message payloadType and syntax placed within the location identified as an sei_message( ) at the end of the scalable nesting SEI message syntax.

[00056] In another embodiment, the new "wrapping" approach can be combined with the prior approaches, by having some SEI metadata present outside of the "wrapper" and some other SEI messages sent within the wrapper. This combination approach can cover use cases when the encoder considers it acceptable for the ordinary interpretation to be applied by

legacy decoding systems, while the new interpretation (e.g., with a different input or output than previously specified) is indicated to be preferred for interpretation by the updated decoding systems.

[00057] In such an approach, the wrapper could identify, for each other associated SEI message, *either* of the following:

a) A type code or prefix information for an SEI message that is present outside of the "wrapper" message, or

b) The full syntax of a "wrapped" SEI message.

The wrapper could contain a loop that contains such information for each associated SEI message.

[00058] In another embodiment, multiple property indications may apply to the output of a sequence of post-processing operations.

[00059] It may not be especially useful to have syntax that identifies properties of some intermediate stage if the subsequent stages are also expected to be performed as part of the indicated sequence. For example, both a "content light level" and "alternative transfer characteristics" could be indicated as applying to the output of the sequential processing operations.

[00060] The wrapper (implicit or explicit) could contain a loop for indicating these properties. For example, the wrapper syntax could contain a loop for signaling the ordered sequence of post-processing operations that are to be applied (say, film grain or tone-mapping), followed by another loop for indicating the properties of the resulting video content after the sequence of post-processing operations has been performed (e.g., maximum luma value). An example of such syntax with implicit wrapping is shown in Table 8.

Table 8. Example SEI processing order SEI message according to classification of the SEI message

sei_processing_order(payloadSize) {	**Descriptor**
po_num_sei_minus1 /* number of SEI messages of interest */	u(8)
for(i = 0; i <= po_num_sei_minus1; i++) {	
sei_message( ) /* SEI messages with post-processing requirements*/	
po_sei_processing_order[i]	u(8)
}	
po_sei_num_properties /* SEI messages defining properties only */	u(8)
for(j = 0; j < po_sei_num_properties; j++)	
sei_message() /* SEI messages indicating properties of the video (after all stages of post-processing have been performed) */	
}	

po_sei_num_properties specifies the number of property indications present that apply after completing the indicated sequence of post-processing operations.

[00061] In another embodiment, instead of using the previous approach, for each sequential stage, there can be multiple other SEI messages that apply to the output of that stage – especially in regard to property-indication metadata. Thus, the syntax could contain two loops – an outer loop for a sequence of post-processing operations, and an inner loop that lists the properties of the output of that stage of post-processing (in the case where subsequent stages of post-processing are to be skipped).

Table 9. Second example of SEI processing order SEI message according to classification of the SEI message

sei_processing_order(payloadSize) {	**Descriptor**
po_num_sei_minus1	u(8)
for(i = 0; i <= po_num_sei_minus1; i++) {	
sei_message( ) /* SEI message with post-processing requirements */	
po_sei_processing_order[i]	u(8)
po_sei_num_properties[i]	u(8)
for(j = 0; j < po_sei_num_properties[i]; j++)	
sei_message( ) /* SEI message for property that applies to output of this stage */	
}	
}	

po_sei_num_properties[i] specifies the number of property indications present that apply after completing the corresponding stage of post-processing operation.

(Note: In this example, the syntax explicitly specifies the number of property indication SEI messages which apply to the output of the i-th post-processing SEI message.)

[00062] Table 10 depicts another embodiment of an SEI processing order SEI message. Compared to Table 7, Table 10 includes the same wrapping and priority-related syntax elements discussed earlier; however, allows a decoder to read them in a more efficient way. More specifically, it uses now a two-loop approach. The number of affected SEI messages is not extracted based on the payloadSize, but it is defined instead using syntax element **po_num_sei_messages_minus2.** Furthermore**,** syntax elements po_num_prefix_bytes[i] and po_prefix_byte[i,j], are replaced by new syntax elements **po_num_bits_in_prefix_indication_minus1**[i] and **po_sei_prefix_data_bit**[i][j], with similar interpretation as the scalable nesting SEI message (Annex D of HEVC (Ref.[2])).

[00063] The function byte_aligned() is defined as:

– If the current position in the bitstream is on a byte boundary, i.e., the next bit in the bitstream is the first bit in a byte, the return value of byte_aligned( ) is equal to TRUE.

– Otherwise, the return value of byte_aligned( ) is equal to FALSE.

Table 10. Third example of SEI processing order SEI message with a wrapping approach and an importance flag

sei_processing_order(payloadSize) {	**Descriptor**
po_num_sei_messages_minus2	u(8)
for(i = 0; i < po_num_sei_messages_minus2 + 2; i++) {	
po_sei_wrapping_flag[i]	u(1)
po_sei_prefix_flag[i]	u(1)
po_sei_importance_flag[i]	u(1)
po_sei_payload_type[i]	u(13)
po_sei_processing_order[i]	u(8)
}	
for(i = 0; i < po_num_sei_messages_minus2 + 2; i++)	
if(po_sei_wrapping_flag[i])	
sei_message( )	
else if(po_sei_prefix_flag[i]) {	
po_num_bits_in_prefix_indication_minus1[i]	b(8)
for(j = 0; j <= po_num_bits_in_prefix_indication_minus1[i]; j++)	
po_sei_prefix_data_bit[i][j]	u(1)
while(!byte_aligned( ))	
po_byte_alignment_bit_equal_to_one /* equal to 1 */	f(1)
}	
}	

[00064] The new syntax elements may be defined as follows:

po_num_sei_messages_minus2 plus 2 indicates the number of SEI messages that have a processing order indicated in the SEI processing order SEI message.

po_sei_prefix_flag[i] equal to 1 specifies that po_num_bits_in_prefix_indication_minus1[i] and po_sei_prefix_data_bit[i][j] syntax elements are present. po_sei_prefix_flag[i] equal to 0 specifies that these syntax elements are not present.

po_num_bits_in_prefix_indication_minus1[i] and **po_sei_prefix_data_bit**[i][j], when present, have the same semantics as the **num_bits_in_prefix_indication_minus1**[i] and **sei_prefix_data_bit**[i][j] syntax elements of the SEI prefix indication SEI message (Ref.[1], Annex D).

po_num_bits_in_prefix_indication_minus1[i] plus 1 specifies the number of bits in the i-th SEI prefix indication.

po_sei_prefix_data_bit[i][j] specifies the j-th bit of the i-th SEI prefix indication.

The bits po_sei_prefix_data_bit[i][j] for j ranging from 0 to po_num_bits_in_prefix_indication_minus1[i], inclusive, follow the syntax of the SEI payload with payloadType equal to prefix_sei_payload_type, and contain a number of complete syntax elements starting from the first syntax element in the SEI payload syntax, and may or may not contain all the syntax elements in the SEI payload syntax. The last bit of these bits (i.e., the bit po_sei_prefix_data_bit[i][po_num_bits_in_prefix_indication_minus1[i]]) shall be the last bit of a syntax element in the SEI payload syntax, unless it is a bit within an itu_t_t35_payload_byte or user_data_payload_byte.

Note: The value of prefix_sei_payload_type could be replaced by po_sei_payload_type[i]

po_byte_alignment_bit_equal_to_one shall be equal to 1.

[00065] Table 11 provides an alternative embodiment to Table 10. The top loop remains the same, but the second loop is now split into two separate loops to better match existing practices in parsing SEI messages (like the scalable nesting and prefix indication SEI messages).

Table 11. Fourth alternative example SEI processing order SEI message with a wrapping approach and an importance flag

sei_processing_order(payloadSize) {	**Descriptor**
po_num_sei_messages_minus2	u(8)
for(i = 0; i < po_num_sei_messages_minus2 + 2; i++) {	
po_sei_wrapping_flag[i]	u(1)
po_sei_prefix_flag[i]	u(1)
po_sei_importance_flag[i]	u(1)
po_sei_payload_type[i]	u(13)
po_sei_processing_order[i]	u(8)
}	
for(i = 0; i < po_num_sei_messages_minus2 + 2; i++)	
if(po_sei_wrapping_flag[i])	
sei_message( )	
}	
for(i = 0; i < po_num_sei_messages_minus2 + 2; i++)	
if(po_sei_prefix_flag[i]) {	
po_num_bits_in_prefix_indication_minus1[i]	b(8)
for(j = 0; j <= po_num_bits_in_prefix_indication_minus1[i]; j++)	
po_sei_prefix_data_bit[i][j]	u(1)
while(!byte_aligned( ))	
po_byte_alignment_bit_equal_to_one /* equal to 1 */	f(1)
}	
}	

[00066] The semantics for the bottom loop remain the same as before, the semantics for the top loop are copied below for completeness.

If **po_sei_wrapping_flag**[i] is equal to 0, an SEI message should be present outside of the SEI processing order SEI message with payloadType equal to po_sei_payload_type[i]. However, if po_sei_wrapping_flag[i] is equal to 0 and no SEI message is present with payloadType equal to po_sei_payload_type[i], the following applies:

– If po_sei_importance_flag[i] is equal to 1, the decoder should ignore the entire SEI processing order SEI message.

– Otherwise, the decoder should ignore all data associated with the loop variable value of i.

NOTE – po_sei_wrapping_flag[i] equal to 1 enables SEI messages to be carried within the SEI processing order SEI message to prevent such SEI messages from being incorrectly interpreted by decoders that do not process the SEI processing order SEI message. Thus, po_sei_wrapping_flag[i] equal to 1 is intended to be used when

po_sei_wrapping_flag[i] equal to 0 can lead to unintended results being produced by such decoders.

po_sei_prefix_flag[i] equal to 1 specifies that po_num_bits_in_prefix_indication_minus1[i] and po_sei_prefix_data_bit[i][j] syntax elements are present. po_sei_prefix_flag[i] equal to 0 specifies that these syntax elements are not present.

po_sei_importance_flag[i] indicates the degree of importance determined by the encoder for the SEI message with index i.

If the decoding system cannot interpret or does not support any indicated SEI message that has po_sei_importance_flag[i] equal to 1, it should ignore the entire SEI processing order SEI message.

po_sei_payload_type[i] specifies the payloadType value of the i-th SEI message type for which preferred processing order information is provided in the SEI processing order SEI message.

When po_sei_prefix_flag[i] is equal to 1, the po_num_bits_in_prefix_indication_minus1[i] and po_sei_prefix_data_bit[i][j] syntax elements should provide sufficient information to determine the specific processing order for SEI messages having the same value of payloadType but different preferred processing order.

For any two different non-negative integer values of m and n, the values of po_sei_payload_type[m] and po_sei_payload_type[n] shall not be identical unless both of the following conditions apply:

- po_sei_wrapping_flag[m] is equal to 1 or po_sei_prefix_flag[m] is equal to 1
- po_sei_wrapping_flag[n] is equal to 1 or po_sei_prefix_flag[n] is equal to 1.

When po_sei_wrapping_flag[i] is equal to 1, po_sei_prefix_flag[i] shall be equal to 0, and the value of po_sei_payload_type[i] shall be equal to the payloadType value within the associated sei_message( ) syntax.

References

Each one of the references listed herein is incorporated by reference in its entirety.

[1] *Advanced Video Coding*, Rec. ITU-T H.264, August 2021, ITU.

[2] *High Efficiency Video Coding*, Rec. ITU-T H.265, August 2021, ITU.

[3] *Versatile Video Coding*, Rec. ITU-T H.266, August 2020, ITU.

[4] *Versatile supplemental enhancement information messages for coded video bitstreams*, Rec. ITU-T H.274, Aug. 2020, ITU.

[5] Sean McCarthy, et al., "SEI processing order SEI message in VVC (Draft 1)," JVET-AA2027, 27th Meeting, by teleconference, July 2022.

[6] Sean McCarthy, et al., "SEI processing order SEI message in VVC (Draft 2)," JVET-AB2027, 28th Meeting, Mainz, Germany, October 2022.

[7] Sean McCarthy, et al., "SEI processing order SEI message in VVC (Draft 3)," JVET-AC2027, 29th Meeting, by teleconference, Jan. 2023.

[8] Sean McCarthy, et al., "SEI processing order SEI message in VVC (Draft 4)," JVET-AD2027, 30th Meeting, Antalya, Turkey, Apr. 2023.

[9] P. Yin, et al. "Signaling of priority processing order for metadata messaging in video coding," PCT Patent Application Publication, WO 2023/278302.

[10] Sean McCarthy, et al., "SEI processing order SEI message in VVC (Draft 5)," JVET-AE2027, 31st Meeting, Geneva, Switzerland, July 2023.

EXAMPLE COMPUTER SYSTEM IMPLEMENTATION

[00067] Embodiments of the present invention may be implemented with a computer system, systems configured in electronic circuitry and components, an integrated circuit (IC) device such as a microcontroller, a field programmable gate array (FPGA), or another configurable or programmable logic device (PLD), a discrete time or digital signal processor (DSP), an application specific IC (ASIC), and/or apparatus that includes one or more of such systems, devices or components. The computer and/or IC may perform, control, or execute instructions relating to signaling processing order for metadata messaging in video coding, such as those described herein. The computer and/or IC may compute any of a variety of parameters or values that relate to signaling processing order for metadata messaging in video coding described herein. The image and video embodiments may be implemented in hardware, software, firmware and various combinations thereof.

[00068] Certain implementations of the invention comprise computer processors which execute software instructions which cause the processors to perform a method of the invention. For example, one or more processors in a display, an encoder, a set top box, a transcoder, or the like may implement methods related to signaling processing order for metadata messaging in video coding as described above by executing software instructions in a program memory accessible to the processors. Embodiments of the invention may also be provided in the form of a program product. The program product may comprise any non-

transitory and tangible medium which carries a set of computer-readable signals comprising instructions which, when executed by a data processor, cause the data processor to execute a method of the invention. Program products according to the invention may be in any of a wide variety of non-transitory and tangible forms. The program product may comprise, for example, physical media such as magnetic data storage media including floppy diskettes, hard disk drives, optical data storage media including CD ROMs, DVDs, electronic data storage media including ROMs, flash RAM, or the like. The computer-readable signals on the program product may optionally be compressed or encrypted. Where a component (e.g. a software module, processor, assembly, device, circuit, etc.) is referred to above, unless otherwise indicated, reference to that component (including a reference to a "means") should be interpreted as including as equivalents of that component any component which performs the function of the described component (e.g., that is functionally equivalent), including components which are not structurally equivalent to the disclosed structure which performs the function in the illustrated example embodiments of the invention.

EQUIVALENTS, EXTENSIONS, ALTERNATIVES AND MISCELLANEOUS

[00069] Example embodiments that relate to signaling processing order for metadata messaging in video coding are thus described. In the foregoing specification, embodiments of the present invention have been described with reference to numerous specific details that may vary from implementation to implementation. Thus, the sole and exclusive indicator of what is the invention, and what is intended by the applicants to be the invention, is the set of claims that issue from this application, in the specific form in which such claims issue, including any subsequent correction. Any definitions expressly set forth herein for terms contained in such claims shall govern the meaning of such terms as used in the claims. Hence, no limitation, element, property, feature, advantage or attribute that is not expressly recited in a claim should limit the scope of such claim in any way. The specification and drawings are, accordingly, to be regarded in an illustrative rather than a restrictive sense.

CLAIMS

What is claimed is:

1. A method to determine processing order among multiple metadata messages, the method comprising:

receiving an input video bitstream and corresponding input metadata messages defining supplemental processing to be applied to decoded video pictures of the input video bitstream;

parsing the input metadata messages to identify a processing order of metadata messaging (POM) message defining a preferred processing order of selected input metadata messages; and upon detecting the POM message (305):

for an i-th input metadata message among the selected input metadata messages:

detecting (310) whether a wrapping flag is set to 1 for the i-th input metadata message, and

if the wrapping flag is set to 1, then reading messaging information for the i-th input metadata message; and

determining a preferred processing order for the i-th input metadata message (320).

2. A method to determine processing order among multiple metadata messages, the method comprising:

receiving an input video bitstream and corresponding input metadata messages defining supplemental processing to be applied to decoded video pictures of the input video bitstream;

parsing the input metadata messages to identify a processing order of metadata messaging (POM) message defining a preferred processing order of selected input metadata messages; and upon detecting the POM message (305):

for an i-th input metadata message among the selected input metadata messages:

reading an importance flag for the i-th input metadata message, and if the importance flag classifies the i-th input metadata message as important and a decoder does not recognize or support it, then the decoder skips processing the POM message and ignores any identified processing order in the POM message.

3. A method to determine processing order among multiple metadata messages, the method comprising:

receiving an input video bitstream and corresponding input metadata messages defining supplemental processing to be applied to decoded video pictures of the input video bitstream;

parsing the input metadata messages to identify a processing order of metadata messaging (POM) message defining a preferred processing order of selected input metadata messages; and upon detecting the POM message (305):

for an i-th input metadata message among the selected input metadata messages:

detecting (335) whether an importance flag is set to 1 for the i-th input metadata message;

detecting (310) whether a wrapping flag is set to 1 for the i-th input metadata message, and

if the wrapping flag is set to 1, then reading messaging information for the i-th input metadata message; and

determining a preferred processing order for the i-th input metadata message (320).

4. The method of claim 1 or claim 3, further comprising: for a decoded video picture of the input video bitstream, applying the selected input metadata messages with their preferred processing order.

5. The method of claim 1 or claim 3, further comprising: if the wrapping flag for the i-th input metadata message is set to 1, then if its preferred processing order (po_sei_processing_order[i]) is not 0, then input for the i-th input metadata message is intended to be output of the input metadata message which has the preferred processing order equal to po_sei_processing_order[i] − 1.

6. The method of claim 3, further comprising, if a decoder cannot interpret or does not support any indicated input metadata message with its importance flag set to important, then the determined preferred processing order is ignored for all the selected input metadata messages.

7. A method to determine processing order among multiple metadata messages, the method

comprising:

receiving an input video bitstream and corresponding input metadata messages defining supplemental processing to be applied to decoded video pictures of the input video bitstream, wherein the input metadata messages comprise a first class of input metadata messages for a first class of decoders and a second class of input metadata for a second class of decoders;

parsing the input metadata messages to identify a processing order of metadata messaging (POM) message defining a preferred processing order of selected input metadata messages; and upon detecting the POM message:

if a decoder identifies itself as one of the first class of decoders, then ignoring the POM message and all input metadata messages defined in the POM message,

else, if the decoder identifies itself as one of the second class of decoders, then processing the POM message to determine a preferred processing order for selected input metadata messages in the POM message.

8. The method of claim 7, wherein the first class of decoders represents a legacy decoder.

9. The method of claim 7, wherein processing the POM message to determine a preferred processing order further comprises:

reading a first processing-order parameter indicating a total number of the selected input metadata messages;

for an i-th input metadata message among the selected input metadata messages:

reading a preferred processing order for the i-th input metadata message; and

reading messaging information for the i-th input metadata message;

reading a second processing-order parameter indicating a total number of property indications present that apply after completing processing the selected input metadata messages; and

reading messaging information for each of the input metadata messages specified by the second processing-order parameter.

10. The method of claim 7, wherein processing the POM message to determine a preferred processing order further comprises:

reading a first processing-order parameter indicating a total number of selected input metadata messages;

for an i-th input metadata message among the selected input metadata messages:

reading messaging information for the i-th input metadata message;

reading a processing order for the i-th input metadata message;

reading a second processing-order parameter indicating a total number of other input metadata messages with property indications present that apply after completing processing the i-th input metadata message; and

reading messaging information for each of the other input metadata messages specified by the second processing-order parameter.

11. The method of any one of claims 1-10, wherein the input metadata messages comprise supplemental enhancement information (SEI) messages or video user information (VUI) messages.

12. A non-transitory computer-readable storage medium having stored thereon computer-executable instructions for executing with one or more processors a method in accordance with any one of the claims 1-11.

13. An apparatus comprising a processor and configured to perform any one of the methods recited in claims 1-11.

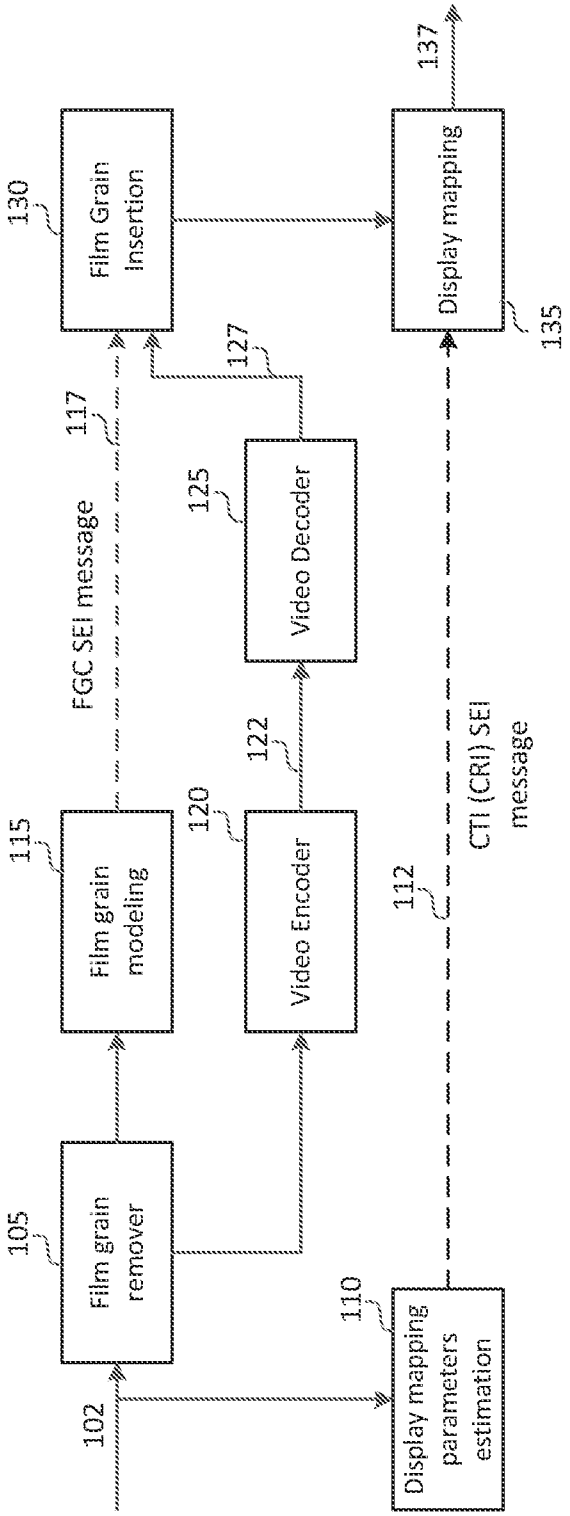

102
105
Film grain remover
110
Display mapping parameters estimation
112
CTI (CRI) SEI message
115
Film grain modeling
117
FGC SEI message
120
Video Encoder
122
125
Video Decoder
127
130
Film Grain Insertion
135
Display mapping
137

FIG. 1A

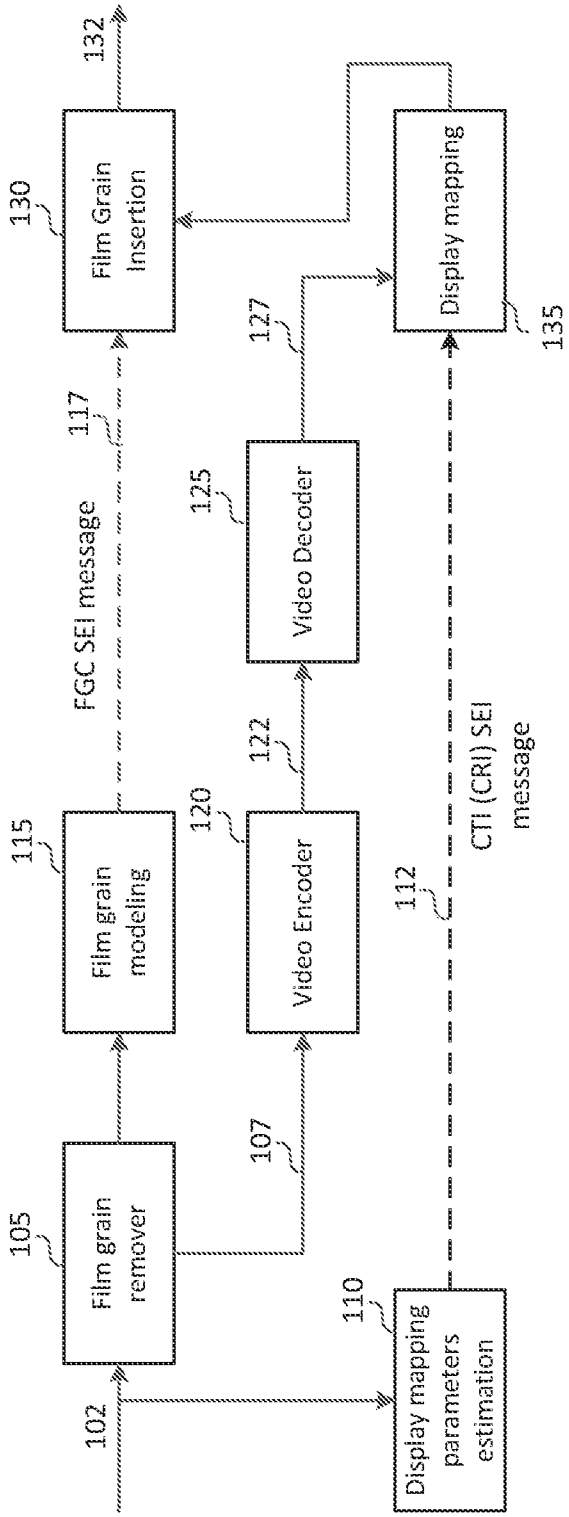
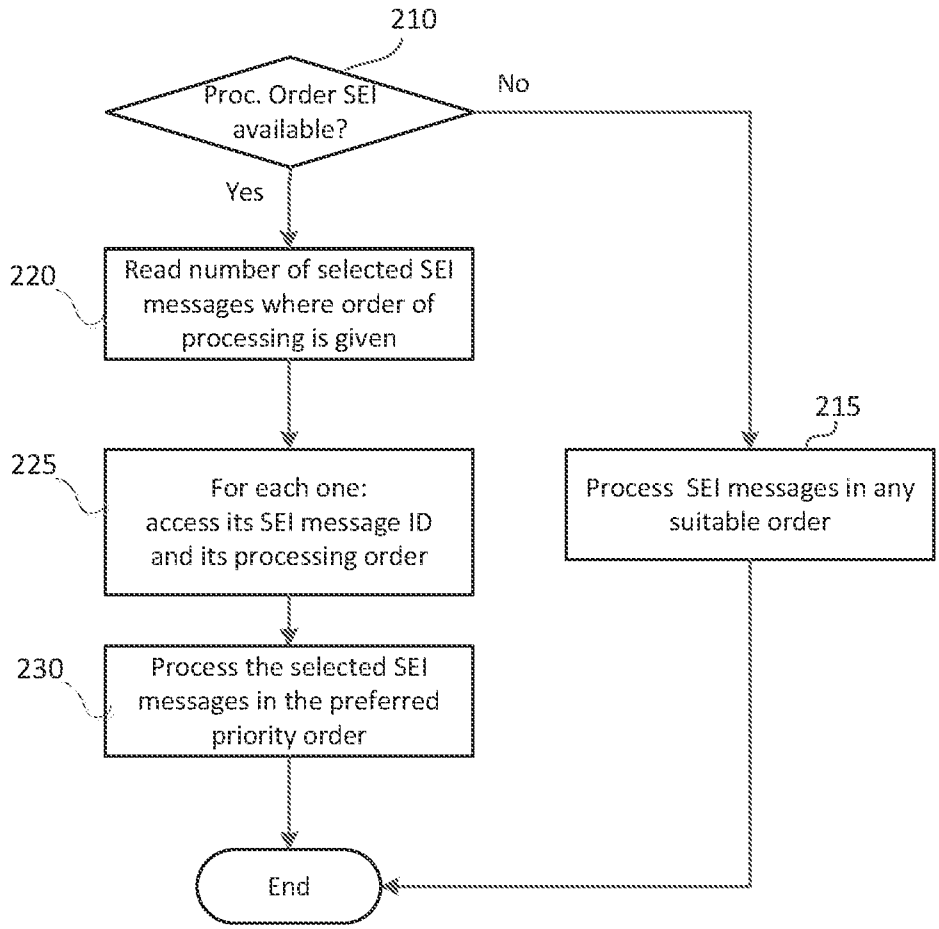

102
105
Film grain remover
107
110
Display mapping parameters estimation
112
CTI (CRI) SEI message
115
Film grain modeling
117
FGC SEI message
120
Video Encoder
122
125
Video Decoder
127
130
Film Grain Insertion
132
135
Display mapping

FIG. 1B



Prior Art

FIG. 2

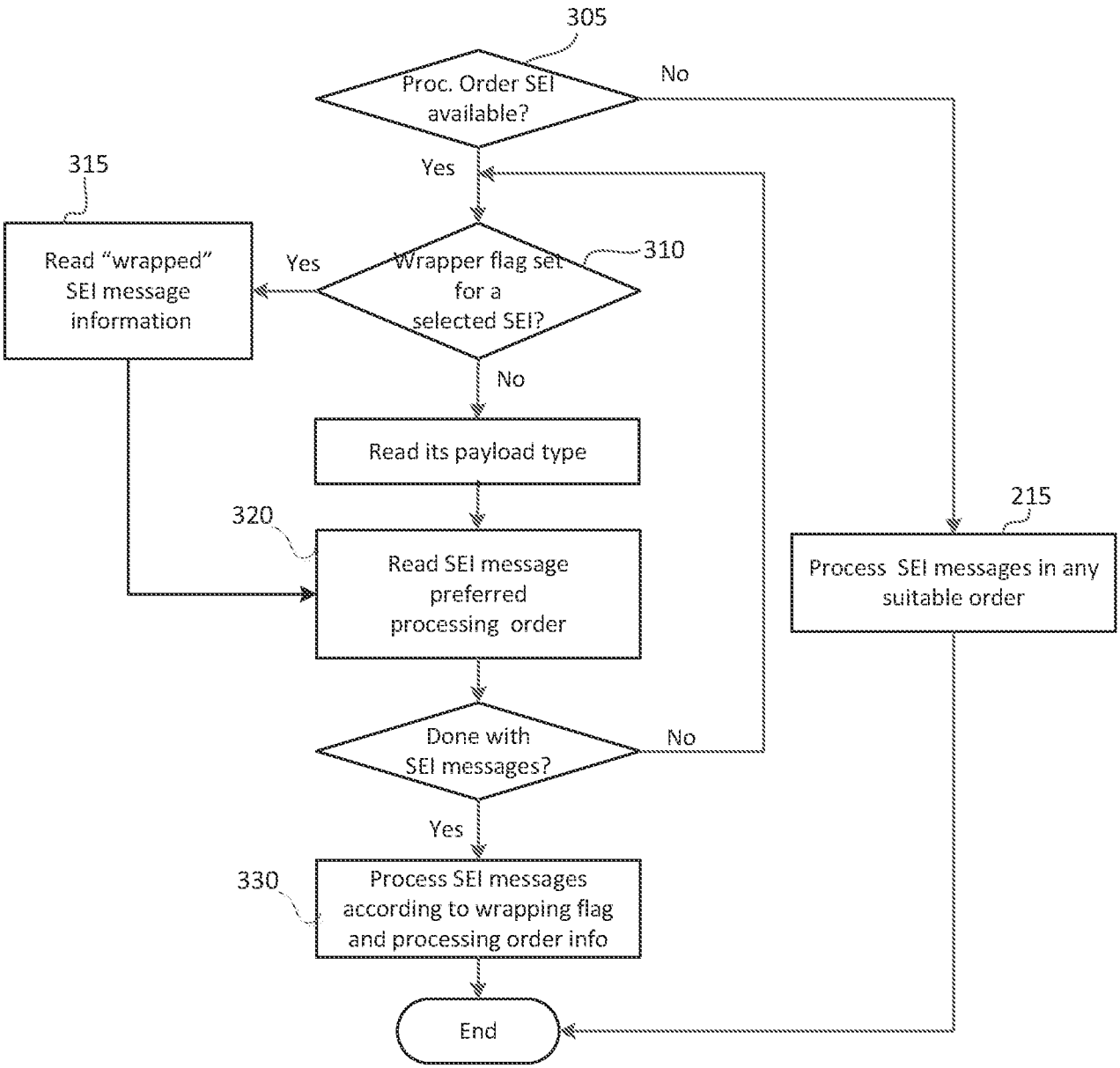

305
Proc. Order SEI available?
No
Yes
315
Read "wrapped" SEI message information
Yes
Wrapper flag set for a selected SEI?
310
No
Read its payload type
320
Read SEI message preferred processing order
215
Process SEI messages in any suitable order
Done with SEI messages?
No
Yes
330
Process SEI messages according to wrapping flag and processing order info
End

FIG. 3A

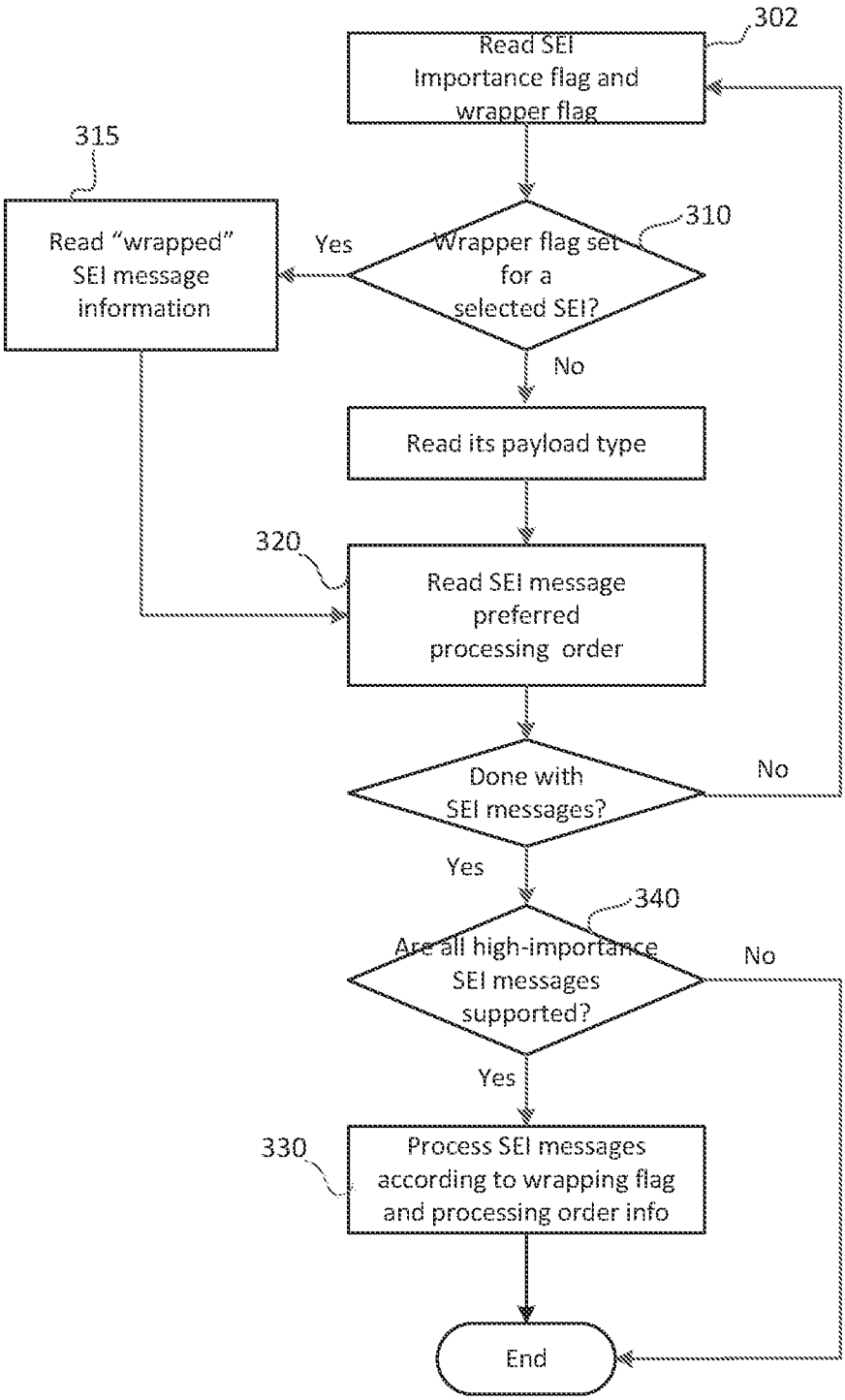

302
Read SEI Importance flag and wrapper flag
315
Read "wrapped" SEI message information
Yes
310
Wrapper flag set for a selected SEI?
No
Read its payload type
320
Read SEI message preferred processing order
Done with SEI messages?
No
Yes
340
Are all high-importance SEI messages supported?
No
Yes
330
Process SEI messages according to wrapping flag and processing order info
End

FIG. 3B

INTERNATIONAL SEARCH REPORT

International application No
PCT/US2024/036717

A. CLASSIFICATION OF SUBJECT MATTER
INV. H04N19/46 H04N19/70 H04N19/85
ADD.

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

B. FIELDS SEARCHED

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

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

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

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	H-M OH ET AL: "Supplemental enhancement information set SEI message", 28. JCT-VC MEETING; 15-7-2017 - 21-7-2017; TORINO; (JOINT COLLABORATIVE TEAM ON VIDEO CODING OF ISO/IEC JTC1/SC29/WG11 AND ITU-T SG.16); URL: HTTP://WFTP3.ITU.INT/AV-ARCH/JCTVC-SITE/,, no. JCTVC-AB0036-v2, 18 July 2017 (2017-07-18), XP030118265, the whole document ----- -/--	1-13

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

[x] See patent family annex.

* Special categories of cited documents :

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

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

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

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

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

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

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

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

"&" document member of the same patent family

Date of the actual completion of the international search	Date of mailing of the international search report
14 October 2024	28/10/2024

Name and mailing address of the ISA/	Authorized officer
European Patent Office, P.B. 5818 Patentlaan 2 NL - 2280 HV Rijswijk Tel. (+31-70) 340-2040, Fax: (+31-70) 340-3016	Kopilovic, Ivan

2

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

C(Continuation). DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X,P	SULLIVAN (DOLBY) G J ET AL: "AHG9: Message wrapping and importance indication for the SEI processing order SEI message", 31. JVET MEETING; 20230711 - 20230719; GENEVA; (THE JOINT VIDEO EXPLORATION TEAM OF ISO/IEC JTC1/SC29/WG11 AND ITU-T SG.16), , no. JVET-AE0156 11 July 2023 (2023-07-11), XP030311461, Retrieved from the Internet: URL:https://jvet-experts.org/doc_end_user/documents/31_Geneva/wg11/JVET-AE0156-v2.zip JVET-AE0156-v2.docx [retrieved on 2023-07-11] sections 3 and 4	1-13

X,P	SULLIVAN (OUTLOOK) G J ET AL: "AHG9: Proposed modifications of the draft SEI processing order SEI message in VVC", 32. JVET MEETING; 20231013 - 20231020; HANNOVER; (THE JOINT VIDEO EXPLORATION TEAM OF ISO/IEC JTC1/SC29/WG11 AND ITU-T SG.16), , no. JVET-AF0189 6 October 2023 (2023-10-06), XP030312364, Retrieved from the Internet: URL:https://jvet-experts.org/doc_end_user/documents/32_Hannover/wg11/JVET-AF0189-v1.zip JVET-AF0189-v1.docx [retrieved on 2023-10-06] section D.11 SEI processing order SEI message	1-13

A	HANNUKSELA (NOKIA) M M ET AL: "AHG9: On post-filter SEI", 22. JVET MEETING; 20210420 - 20210428; TELECONFERENCE; (THE JOINT VIDEO EXPLORATION TEAM OF ISO/IEC JTC1/SC29/WG11 AND ITU-T SG.16), , no. JVET-V0058 ; m56451 13 April 2021 (2021-04-13), XP030294069, Retrieved from the Internet: URL:https://jvet-experts.org/doc_end_user/documents/22_Teleconference/wg11/JVET-V0058-v1.zip JVET-V0058.docx [retrieved on 2021-04-13] the whole document	1-13

	-/--	

C(Continuation). DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
A	WO 2023/278302 A1 (DOLBY LABORATORIES LICENSING CORP [US]) 5 January 2023 (2023-01-05) cited in the application abstract; figures 1,2 paragraph [0022] - paragraph [0023]; table 1	1-13
A	US 2018/278964 A1 (WANG YEKUI [US] ET AL) 27 September 2018 (2018-09-27) paragraph [0093] - paragraph [0115]	1-13

INTERNATIONAL SEARCH REPORT

Information on patent family members

International application No
PCT/US2024/036717

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
WO 2023278302 A1	05-01-2023	AU 2022300855 A1	22-02-2024
		BR 112023025098 A2	20-02-2024
		CA 3221431 A1	05-01-2023
		CN 117597925 A	23-02-2024
		EP 4364411 A1	08-05-2024
		IL 308874 A	01-01-2024
		JP 2024525177 A	10-07-2024
		KR 20240025660 A	27-02-2024
		US 2024283978 A1	22-08-2024
		WO 2023278302 A1	05-01-2023
US 2018278964 A1	27-09-2018	AU 2018237153 A1	29-08-2019
		BR 112019019250 A2	14-04-2020
		CN 110419223 A	05-11-2019
		EP 3603075 A1	05-02-2020
		JP 2020511861 A	16-04-2020
		KR 20190122867 A	30-10-2019
		TW 201904297 A	16-01-2019
		US 2018278964 A1	27-09-2018
		WO 2018175609 A1	27-09-2018