|  |
| --- |
| **INTERNATIONAL ORGANIZATION FOR STANDARDIZATION ORGANISATION INTERNATIONALE DE NORMALISATION ISO/IEC JTC 1/SC 29/WG 5 MPEG JOINT VIDEO EXPERTS TEAM WITH ITU-T SG 16** |
| **ISO/IEC JTC 1 / SC 29 / WG 5 N 319** |
| **Kemer, TR – 1–8 November 2024** |
| |  |  | | --- | --- | | **Source:** | **Convenor (Jens-Rainer Ohm)** | | **Title:** | **Preliminary Text of ISO/IEC 23002-7:202x/DAM1 Additional SEI messages** | | **Type:** | **Project** | | **Subtype:** | **Other** | | **Status:** | **Approved** | | **Date:** | **2024-12-18** | | **Expected Action:** | **Info** | | **Action due date:** | **N/A** | | **Pages:** | **90** (not including this cover page) | | **Email of convenor:** | **ohm @ ient . rwth-aachen . de** | | **Committee URL:** | **https://sd.iso.org/documents/ui/#!/browse/iso/iso-iec-jtc-1/iso-iec-jtc-1-sc-29/iso-iec-jtc-1-sc-29-wg-5** | |

|  |  |
| --- | --- |
| **Joint Video Experts Team (JVET)**  **of ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29**  36th Meeting: Kemer, TR, 1–8 November 2024 | Document: JVET-AJ2006-v3 |

|  |  |  |  |
| --- | --- | --- | --- |
| *Title:* | **Additional SEI messages for VSEI version 4 (Draft 4)** | | |
| *Status:* | Output document approved by JVET | | |
| *Purpose:* | Draft text | | |
| *Author(s) or Contact(s):* | Jill Boyce Jie Chen Sachin Deshpande Miska M. Hannuksela Sean McCarthy Gary J. Sullivan Hendry Tan Ye-Kui Wang | Email: | jill.boyce@nokia.com jiechen.cj@alibaba-inc.com sdeshpande@sharplabs.com miska.hannuksela@nokia.com sean.mccarthy@dolby.com gary.sullivan@dolby.com dr.hendry@lge.com yekui.wang@bytedance.com |
| *Source:* | Editors | | |

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

# Abstract

This document contains the draft text for changes to the versatile supplemental enhancement information messages for coded video bitstreams (VSEI) standard (Rec. ITU-T H.274 | ISO/IEC 23002-7), to specify additional SEI messages, including encoder optimization information, source picture timing information , object mask information, modality information, text description information, generative face video, generative face video enhancement, digitally signed content initialization, digitally signed content selection, and digitally signed content verification SEI messages and updates to the neural-network post-filter characteristics SEI message.

# Editors’ notes and changes yet to be integrated:

* All automatically generated numbering and cross-reference fields will need to be updated after the changes are integrated into the basis text.Check the use of “pertain”; the word is not used in a similar way anywhere in the existing standard.payloadType values are to be specified the corresponding VVC, HEVC or AVC draft text. The use of prefix SEI NAL unit was enabled for all new SEI messages, whereas the use of suffix SEI NAL unit was enabled for encoder optimization information and object mask information SEI messages, since their content may be determined while the encoder encodes the associated picture.
* (JVET-AH0121 item 5) check/fix the ChromaFormatIdc derivation for the use of the colour transform information SEI message as a part of a processing chain in a similar manner as done for the film grain characteristics SEI message in JVET-AG2027 (to carefully check and communicate offline for correctness, e.g., w.r.t. JVET-AH0047).
* (JVET-AH0121 item 6) check/fix the interface variable derivation for the use of the NNPFC and NNPFA SEI messages as a part of a processing chain in a similar manner as done for the film grain characteristics SEI message in JVET-AG2027 (to carefully check and communicate offline for correctness, e.g., w.r.t. JVET-AH0047).

To be integrated:

* + General editorial improvementsJVET-AJ0185 Editorial updates for NNPFC, SPO and TDI SEI messages
    - Item 2

# Changes that have been integrated:

*A description of the contributions reflected in this document relative to JVET-AI2006:*

* General editorial improvements
  + JVET-AJ0050 Editorial changes to VSEI draft text
  + JVET-AJ0065 Comments On VSEI Version 4
  + JVET-AJ0070 Editorial and technical problems in the draft of VSEI v4 and their solutions
  + JVET-AJ0120 Miscellaneous modifications for SEI messages in the VSEI draft
  + JVET-AJ0185 Editorial updates for NNPFC, SPO and TDI SEI messages
    - Item 1, 3, 4, 5, and 6.
* SEI Processing Order SEI
  + JVET-AJ0045 Handling of processing chains - operations
  + JVET-AJ0046 Handling of processing chains - constraints and special cases
  + JVET-AJ0047 Merging NNPF filtering process into processing chain handling
  + JVET-AJ0048 On SeiProcessingOrderSeiList and SpoProcessSeiList
  + JVET-AJ0128 Adding functionality to po\_processing\_degree\_flag[ i ] of the SPO SEI message
  + JVET-AJ0129 Miscellaneous comments on the SPO SEI message
  + JVET-AJ0105 On signalling of complexity information in SEI processing order SEI message
* Neural Network Post Filter
  + JVET-AJ0049 NNPFC and NNPFA interface text in VVC
  + JVET-AJ0073 A clean-up for the spatial resampling NNPF design
  + JVET-AJ0058 Generalizing text prompt usage in the NNPFC SEI message
  + JVET-AJ0131 On text prompts for NNPF
  + JVET-AJ0234 On Spatial Extrapolation for NNPF
  + JVET-AJ0104 On signalling of prompt in NNFPA SEI message
  + JVET-AJ0114 On signalling input pictures shift in NNPFA SEI message
* Encoder optimization information SEI
  + JVET-AJ0063 On Encoder Optimization Information SEI
  + JVET-AJ0183 Signaling source and added pictures in EOI SEI message
* Source picture timing SEI
  + JVET-AJ0308 Leading pictures and the SPTI SEI message
  + JVET-AJ0252 On SPTI SEI message
* Text description information SEI
  + JVET-AJ0184 On signaling of cancellation, persistency and id in TDI SEI messages
  + JVET-AJ0241 Text description SEI purpose for encoder description
* Generative Face Video and Generative Face Video Enhancement SEI message
  + JVET-AJ0207 Move GFV and GFVE to VSEI draft
  + JVET-AJ0051 On the GFV SEI message
  + JVET-AJ0101 Miscellaneous modifications for SEI messages in the TuC for futhre exensions of VSEI (GFV aspects)
  + JVET-AJ0108 On siganlling of instance count in generative face video SEI message
  + JVET-AJ0135 Refined Methodology for Pupil Position SEI Message for Generaive Face Video
* Digitally signed content SEIs
  + Move to VSEI draft (see notes for JVET-AJ0151)
* Neural Network Post Filter
  + JVET-AJ0142 On potential conflicts between NNPFs and other post-processings

In JVET-AI2006, relative to JVET-AH2006:

*Changes incorporate JVET-AI0180, JVET- AI0146, JVET-AI0059, JVET-AI0061, JVET-AI0070, JVET-AI0071, JVET-AI0073, JVET-AI0098, JVET-AI0120, JVET-AI0341, JVET-AI0212, JVET-AI0202, JVET-AI0116, JVET-AI0153, JVET-AI0120, JVET-AI0061, JVET-AI0117, JVET-AI0214*

*Encoder optimization information SEI modifications (JVET-AI0180, JVET-AI0214)*

*Text description SEI modifications (JVET-AI0059 and JVET-AI0120 Item 10)*

*NNPFC SEI message modifications (JVET-AI0061, JVET-AI0070, JVET-AI0071, JVET-AI0202)*

*Object mask information SEI modifications (JVET-AI0116, JVET-AI0153,* *JVET-AI0120 )*

*SEI processing order SEI and Processing order nesting SEI migration from VVC to VSEI and additional modifications (JVET-AI0146, JVET-AI0071, JVET-AI0073, JVET-AI0098, JVET-AI0341, JVET-AI0212)*

*Source Picture Timing Information SEI modifications (JVET-AI0117, JVET-AI0214 )*

*Source Picture Timing Information SEI Editor action (related to JVET-AI0120 Item 1 and 2)*

*A description of the contributions reflected in this document relative to JVET-AH2006:*

* JVET-AI0180 Add original source picture dimensions to EOI SEI
* JVET- AI0146 Move the SPO and PON SEI messages from VVC to VSEI
* JVET-AI0059 Addition of a global cancel flag, movement of syntax, and define IDs to be purpose specific in Text Descriptions SEI
* JVET-AI0061 Bugfix for spatial extrapolation patch processing
* JVET-AI0070 Byte alignment in NNPFC SEI
* JVET-AI0071 Add two indicators in NNPFC SEI and in SPO SEI
* JVET-AI0073 Add po\_breadth\_first\_flag in the SPO SEI
* JVET-AI0098 Remove condition of the degree flag in SPO SEI
* JVET-AI0341 Semantics for derivation of a PoSeiList
* JVET-AI0212 Cleanups to SPO SEI
* JVET-AI0202 Adding support for removal in spatial extrapolation purpose in NNPFC SEI
* JVET-AI0116 Object permanence for Object Mask Info SEI
* JVET-AI0153 Object Mask Info SEI syntax and semantics updates
* JVET-AI0120 Some updates to Object Mask Info SEI and Text Descriptions SEI
* JVET-AI0061 Signal text prompt for spatial extrapolation.
* JVET-AI0117 Some updates to SPTI SEI
* JVET-AI0214 Syntax element renaming for EOI SEI and some improvements to semantics of SPTI SEI message (Editorial)

In JVET-AH2006, relative to JVET-AG2034:

*Changes incorporate JVET-AH0108, JVET-AH0111, JVET-AH0174, JVET-AH0175, JVET-AH0343, JVET-AH0346*

* *Encoder optimization information SEI modifications (JVET-AH0108, JVET-AH0111)*
* *Text description SEI (JVET-AH0343)*
* *Object mask information SEI modifications (JVET-AH0346)*
* *NNPFC SEI message modifications (JVET-AH0174)*
* *Film grain characteristics SEI message constraints (JVET-AH0175 item 1)*

In JVET-AG2034:

* *Changes incorporated: JVET-AG0070, JVET-AG0081, JVET-AG-0082v2, JVET-AG0086, JVET-AG0089, JVET-AG0148, JVET-AG0188, JVET-AG0322 NNPFC SEI message: Temporal extrapolation purpose (JVET-AG0089).*
* *Source picture timing information SEI message: Include with the following modifications*
  + *various aspects from JVET-AG0070*
    - *(Proposal 1)\_Editorial clarification of NOTE 1*
    - *(Proposal 2) Forbid the value 0 for spti\_num\_units\_in\_elemental\_interval (option 2)*
    - *(Proposal 3)Modify semantic to fix bug in handling the case spti\_source\_type\_present\_flag equal 0*
  + *option 1 of JVET-AG0188*
  + *Modification from JVET-AG0082v2*
* *Encoder optimization information SEI message:*
  + 2-bit indicators eoi\_for\_human\_viewing\_idc and eoi\_for\_machine\_analysis\_idc (JVET-AG0086, item 1)
  + *new optimization type from JVET-AG0081*
* *Object mask information SEI message: Include with modifications from JVET-AG0148*
* *Modality information SEI message:*
* *NNPFC SEI message application information signalling*

# Changes to the specification text:

*Modify subclause 2.3 as follows:*

– Recommendation ITU-T T.35 (in force), Procedure for the allocation of ITU-T defined codes for non standard facilities.

– IETF RFC 1321 (in force), The MD5 Message-Digest Algorithm.

– IETF RFC 4151 (in force), The 'tag' URI Scheme.

– IETF RFC 5646 (in force), Tags for Identifying Languages.

– IETF Standard 66 (in force), Uniform Resource Identifiers (URI): Generic Syntax.

– ISO/CIE 11664-1 (in force), Colorimetry – Part 1: CIE standard colorimetric observers.

– ISO/IEC 10646 (in force), Information technology – Universal coded character set (UCS).

– ISO/IEC 11578:1996, Information technology – Open Systems Interconnection – Remote Procedure Call (RPC).

– ISO/IEC 15938-17 (in force), Information technology – Multimedia content description interface – Part 17: Compression of neural networks for multimedia content description and analysis

– Doc. CTA-766-D (ANSI), CTA, U.S. and Canadian Region Rating Tables (RRT) and Content Advisory Descriptors for Transport of Content Advisory Information Using ATSC Program and System Information Protocol (PSIP), Consumer Technology Association, Arlington, VA, December 11, 2013

– ISO 20473:2007, Optics and photonics – Spectral bands.

– NIST FIPS PUB 180-4, Secure Hash Standard (SHS).

– ISO/IEC 21617-1:2024, Information technology - JPEG Trust - Part 1: Core Foundation.

*In clause 3, add the following definitions (adjust the subclause numbering when adding):*

* 1. **corresponding picture**: For a particular picture picA, the corresponding picture in a picture list is the picture in the picture list that is either picA itself or a processed version of picA generated when the process implied by an SEI message is applied.

NOTE – The particular picture picA could be a picture that is not in the picture list, in which case the corresponding picture in the picture list is a processed version of picA. When picA is in the picture list, it's corresponding picture in the picture list is itself.

* 1. **inserted picture**: A picture that was interpolated or extrapolated when the process implied by an SEI message (e.g., an NNPFA SEI message activating an NNPF with PictureRateUpsamplingFlag or TemporalExtrapolationFlag equal to 1) is applied.
  2. **associated inserted picture**: For a particular picture picA, an associated inserted picture in a picture list is a picture picB in the picture list that is the corresponding picture of an inserted picture generated when applying the process implied by an SEI message to a corresponding picture of picA or to an associated inserted picture of picA.

NOTE – The particular picture picA could be a picture that is not in the picture list.

*In clause 4, add the following abbrevation:*

FGC Film Grain Characteristics

*Modify subclause 6.1 as follows:*

...

Technical specifications that reference this Specification for carrying SEI messages shall specify a container to carry the payload syntax of each specified SEI message, to identify which SEI message is conveyed through a payloadType variable that indicates the SEI message payload type, and to identify the length in bits of the SEI message payload syntax structure. An example of the container and an example of the SEI message payload syntax structure are the sei\_message( ) syntax structure and the sei\_payload( ) syntax structure, respectively, specified in Rec. ITU-T H.266 | ISO/IEC 23090-3 and Rec. ITU-T H.265 | ISO/IEC 23008-2. The design of the container should provide the ability to detect the number of bits in an SEI message and to allow the number of bits to be increased in future versions of this Specification, thus enabling this Specification to provide extensibility by directly appending additional syntax elements to the end of the SEI message payload syntax structure in future versions of this Specification. The syntax of the container of SEI messages as well as the method of identifying which SEI message is outside the scope of this Specification.

The length of the VUI parameters syntax structure or an SEI message payload syntax structure in bits is referred to herein by the variable PayloadBits, which is provided by an external means not specified in this Specification. The number of bytes that contains the payload data is referred to herein by the variable payloadSize, where payloadSize is equal to Ceil( PayloadBits ÷ 8 ).

...

*Modify clause 8.1 as follows:*

*Replace Table 4 with the following:*

| **Table 4 – Persistence scope of SEI messages (informative)** | |
| --- | --- |
| **SEI message** | **Persistence scope** |
| Filler payload | The PU containing the SEI message |
| User data registered by Rec. ITU-T T.35 | Unspecified |
| User data unregistered | Unspecified |
| Film grain characteristics | Specified by the syntax of the SEI message |
| Frame packing arrangement | Specified by the syntax of the SEI message |
| Parameter sets inclusion indication | The CLVS containing the SEI message |
| Decoded picture hash | The PU containing the SEI message |
| Mastering display colour volume | The CLVS containing the SEI message |
| Content light level information | The CLVS containing the SEI message |
| DRAP indication | The picture associated with the SEI message |
| Alternative transfer characteristics | The CLVS containing the SEI message |
| Ambient viewing environment | The CLVS containing the SEI message |
| Content colour volume | Specified by the syntax of the SEI message |
| Equirectangular projection | Specified by the syntax of the SEI message |
| Generalized cubemap projection | Specified by the syntax of the SEI message |
| Sphere rotation | Specified by the syntax of the SEI message |
| Region-wise packing | Specified by the syntax of the SEI message |
| Omnidirectional viewport | Specified by the syntax of the SEI message |
| Frame-field information | The PU containing the SEI message |
| Sample aspect ratio information | Specified by the syntax of the SEI message |
| Annotated regions | Specified by the syntax of the SEI message |
| Scalability dimension information | The CVS containing the SEI message |
| Multiview acquisition information | The CVS containing the SEI message |
| Multiview view position | The CVS containing the SEI message |
| Depth representation information | Specified by the semantics of the SEI message |
| Alpha channel information | Specified by the syntax of the SEI message |
| Extended DRAP indication | The picture associated with the SEI message |
| Display orientation | Specified by the syntax of the SEI message |
| Colour transform information | Specified by the syntax of the SEI message |
| Shutter interval information | The CLVS containing the SEI message |
| Neural-network post-filter characteristics | The CLVS containing the SEI message |
| Neural-network post-filter activation | Specified by the syntax of the SEI message |
| Phase indication | Specified by the semantics of the SEI message |
| SEI processing order | For each value of po\_id, the number of SEI messages and the payloadType codes of the SEI messages indicated within the SEI processing order SEI message persist for the CVS containing the SEI processing order SEI message. |
| Processing order nesting | Depending on the processing-order-nested SEI messages. Each processing-order-nested SEI message has the same persistence scope as if the SEI message was not nested. |
| Encoder optimization information | Specified by the syntax of the SEI message |
| Source picture timing information | Specified by the syntax of the SEI message |
| Object mask information | Specified by the syntax of the SEI message |
| Modality information | Specified by the syntax of the SEI message |
| Text descriptions | Specified by the syntax of the SEI message |

*Renumber NOTE to NOTE 1.*

*Add the following:*

NOTE 2 – When multiple NNPFs specified by NNPFC SEI messages with different nnpfc\_purpose values are activated for the same picture, it is possible that the corresponding NNPF processes have some conflict. Similarly, when an SEI message implying a non-NNPF post-processing process is associated with a picture and an NNPF specified by an NNPFC SEI message with a related purpose is activated for the same picture, it is possible that the corresponding processes have some conflict. Particular care is expected to be exercised in the design of encoders that generate such multiple SEI messages. For example, process-implying SEI messages that have some conflict can be included in different processing chains defined by SEI processing order SEI messages. Alternatively, multiple process-implying SEI messages with related post-processing operations can be used in contexts in which conflicts of usage are unimportant, or not possible, or are managed – e.g., defined or managed in the controlling application or transport specification, or by controlling the environment in which bitstreams are distributed.

*Modify subclause 8.5.2 as follows:*

**fg\_matrix\_coeffs** has the same semantics as specified in clause 7.3 for the vui\_matrix\_coeffs syntax element, except as follows:

– fg\_matrix\_coeffs specifies the matrix coefficients of the film grain characteristics specified in the SEI message, rather than the matrix coefficients used for the CLVS.

– When fg\_matrix\_coeffs is not present in the film grain characteristics SEI message, the value of fg\_matrix\_coeffs is inferred to be equal to vui\_matrix\_coeffs.

– The values allowed for fg\_matrix\_coeffs are not constrained by the chroma format of the decoded video pictures that is indicated by the value of ChromaFormatIdc for the semantics of the VUI parameters.

fg\_matrix\_coeffs shall not be equal to 0 unless fg\_bit\_depth\_luma\_minus8 is equal to fg\_bit\_depth\_chroma\_minus8.

fg\_matrix\_coeffs shall not be equal to 8 unless one of the following conditions is true:

– fg\_bit\_depth\_luma\_minus8 is equal to fg\_bit\_depth\_chroma\_minus8.

– fg\_bit\_depth\_chroma\_minus8 is equal to fg\_bit\_depth\_luma\_minus8 + 1.

*Modify subclause 8.19.2 as follows:*

**sdi\_aux\_id**[ i ] equal to 0 indicates that the i-th layer in the current CVS does not contain auxiliary pictures. sdi\_aux\_id[ i ] greater than 0 indicates the type of auxiliary pictures in the i-th layer in the current CVS as specified in Table 15. When sdi\_auxiliary\_info\_flag is equal to 0, the value of sdi\_aux\_id[ i ] is inferred to be equal to 0.

Table 15 – Mapping of sdi\_aux\_id[ i ] to the type of auxiliary pictures

|  |  |  |
| --- | --- | --- |
| **sdi\_aux\_id[ i ]** | **Name** | **Type of auxiliary pictures** |
| 1 | AUX\_ALPHA | Alpha plane |
| 2 | AUX\_DEPTH | Depth picture |
| 3 | AUX\_OBJECT\_MASK | Object mask picture |
| 4..127 |  | Reserved |
| 128..159 |  | Unspecified |
| 160..255 |  | Reserved |

NOTE 1 – The interpretation of auxiliary pictures associated with sdi\_aux\_id[ i ] in the range of 128 to 159, inclusive, is specified through means other than the sdi\_aux\_id[ i ] value.

sdi\_aux\_id[ i ] shall be in the range of 0 to 3, inclusive, or 128 to 159, inclusive, for bitstreams conforming to this version of this Specification. Although the value of sdi\_aux\_id[ i ] shall be in the range of 0 to 3, inclusive, or 128 to 159, inclusive, in this version of this Specification, decoders shall also allow other values of sdi\_aux\_id[ i ] in the range of 0 to 255, inclusive.

If sdi\_aux\_id[ i ] is equal to 0, the i-th layer is referred to as a primary layer. Otherwise, the i-th layer is referred to as an auxiliary layer. When sdi\_aux\_id[ i ] is equal to 1, the i-th layer is also referred to as an alpha auxiliary layer. When sdi\_aux\_id[ i ] is equal to 2, the i-th layer is also referred to as a depth auxiliary layer. When sdi\_aux\_id[ i ] is equal to 3, the i-th layer is also referred to as an object mask auxiliary layer.

*Modify subclause 8.26.2 (Colour transform information SEI message semantics) to be as follows:*

The colour transform information (CTI) SEI message provides information to enable remapping of the reconstructed colour samples of the output pictures for purposes such as converting the output pictures to a representation that is more suitable for an alternative display. The colour transform model used in the CTI SEI message is composed of a first piece-wise linear function applied to the first colour component. Depending on the values of syntax elements colour\_transform\_cross\_component\_flag, colour\_transform\_cross\_comp\_inferred\_flag, and colour\_transform\_lut2\_present\_flag, one or two additional piece-wise linear functions may be signalled for the second and third colour components.

When ChromaFormatIdc is equal to 0 (monochrome), the CTI SEI message shall not be present, although decoders shall also allow such messages to be present and shall ignore any such CTI SEI messages when present.

**colour\_transform\_id** contains an identifying number that may be used to identify the purpose of the CTI. The value of colour\_transform\_id may be used (in a manner not specified in this Specification) to indicate that the input to the remapping process is the output of some conversion process that is not specified in this Specification, such as a conversion of the picture to some alternative colour representation (e.g., conversion from a YCbCr colour representation to a GBR colour representation). When more than one CTI SEI message is present with the same value of colour\_transform\_id, the content of these CTI SEI messages shall be the same. When CTI SEI messages are present that have more than one value of colour\_transform\_id, this may indicate that the remapping processes indicated by the different values of colour\_transform\_id are alternatives that are provided for different purposes or that a cascading of remapping processes is to be applied in a sequential order (an order that is not specified in this Specification). The value of colour\_transform\_id shall be in the range of 0 to 232 − 2, inclusive.

Values of colour\_transform\_id from 0 to 255 and from 512 to 231 − 1 may be used as determined by the application. Values of colour\_transform\_id from 256 to 511, inclusive, and from 231 to 232 − 2, inclusive, are reserved for future use by ITU-T | ISO/IEC. Decoders shall ignore the CTI SEI messages containing a value of colour\_transform\_id in the range of 256 to 511, inclusive, or in the range of 231 to 232 − 2, inclusive, and bitstreams conforming to this version of this Specification shall not contain colour\_transform\_id with such values.

NOTE – The colour\_transform\_id can be used to support different remapping processes that are suitable for different display scenarios. For example, different values of colour\_transform\_id may correspond to different remapped colour spaces supported by displays.

**colour\_transform\_cancel\_flag** equal to 1 indicates that the CTI SEI message cancels the persistence of any previous CTI SEI message in output order that applies to the current layer. colour\_transform\_cancel\_flag equal to 0 indicates that CTI follows.

**colour\_transform\_persistence\_flag** specifies the persistence of the CTI SEI message for the current layer.

colour\_transform\_persistence\_flag equal to 0 specifies that the CTI SEI message applies to the current decoded picture only.

colour\_transform\_persistence\_flag equal to 1 specifies that the CTI SEI message applies to the current decoded picture and persists for all subsequent pictures of the current layer in output order until one or more of the following conditions are true:

– A new CLVS of the current layer begins.

– The bitstream ends.

– A picture in the current layer in an AU associated with a CTI SEI message is output that follows the current picture in output order.

**colour\_transform\_video\_signal\_info\_present\_flag** equal to 1 specifies that syntax elements colour\_transform\_full\_range\_flag, colour\_transform\_primaries, colour\_transform\_transfer\_function and colour\_transform\_matrix\_coefficients are present. colour\_transform\_video\_signal\_info\_present\_flag equal to 0 specifies that syntax elements colour\_transform\_full\_range\_flag, colour\_transform\_primaries, colour\_transform\_transfer\_function and colour\_transform\_matrix\_coefficients are not present.

**colour\_transform\_full\_range\_flag** has the same semantics as specified in clause 7.3 for the vui\_full\_range\_flag syntax element, except that colour\_transform\_full\_range\_flag identifies the video full range flag of the remapped reconstructed picture, rather than the video full range flag used for the CLVS. When not present, the value of colour\_transform\_full\_range\_flag is inferred to be equal to the value of vui\_full\_range\_flag.

**colour\_transform\_primaries** has the same semantics as specified in clause 7.3 for the vui\_colour\_primaries syntax element, except that colour\_transform\_primaries identifies the colour primaries of the remapped reconstructed picture, rather than the colour primaries used for the CLVS. When not present, the value of colour\_transform\_primaries is inferred to be equal to the value of vui\_colour\_primaries.

**colour\_transform\_transfer\_function** has the same semantics as specified in clause 7.3 for the vui\_transfer\_characteristics syntax element, except that colour\_transform\_transfer\_function identifies the transfer characteristics of the remapped reconstructed picture, rather than the transfer characteristics used for the CLVS. When not present, the value of colour\_transform\_transfer\_function is inferred to be equal to the value of vui\_transfer\_characteristics.

**colour\_transform\_matrix\_coefficients** has the same semantics as specified in clause 7.3 for the vui\_matrix\_coeffs syntax element, except that colour\_transform\_matrix\_coefficients identifies the matrix coefficients of the remapped reconstructed picture, rather than the matrix coefficients used for the CLVS. When not present, the value of colour\_transform\_matrix\_coefficients is inferred to be equal to the value of vui\_matrix\_coeffs.

**colour\_transform\_bit\_depth\_minus8** plus 8 specifies the bit depth of the colour components of the associated pictures for purposes of interpretation of the CTI SEI message. When any CTI SEI message is present with the value of colour\_transform\_bit\_depth plus 8 not equal to the bit depth of the decoded colour components, the SEI message refers to the hypothetical result of a conversion operation performed to convert the decoded colour component samples to the bit depth equal to colour\_transform\_input\_bit\_depth plus 8.

The value of colour\_transform\_bit\_depth plus 8 shall be in the range of 8 to 16, inclusive. Values of colour\_transform\_bit\_depth from in the range of 17 to 23, inclusive, are reserved for future use by ITU-T | ISO/IEC. Decoders shall ignore the CTI SEI messages that contain a value of colour\_transform\_bit\_depth in the range of 17 to 23, inclusive, and bitstreams conforming to this version of this Specification shall not contain colour\_transform\_bit\_depth with such values.

bitDepth is set equal to ( colour\_transform\_bit\_depth + 8 ).

**colour\_transform\_log2\_number\_of\_points\_per\_lut\_minus1** specifies the log2 of the number of pivot points in the piece-wise linear remapping functions minus 1.

log2numLutPoints is set equal to ( colour\_transform\_log2\_number\_of\_points\_per\_lut\_minus1 + 1 ).

numLutPoints is set equal to ( 1 << log2numLutPoints ).

colourTransformSize is set equal to ( numLutPoints + 1 ).

log2distX is set equal to ( bitDepth − log2numLutPoints ).

**colour\_transform\_cross\_component\_flag** equal to 1 indicates that the remapping of the second and third colour components is performed as cross-component remapping based on the first colour component. colour\_transform\_cross\_component\_flag equal to 0 indicates that intra-component remapping is applied to the second and third colour components.

maxIntraComp is set equal to ( 2 \* ( 1 − colour\_transform\_cross\_component\_flag ) ).

**colour\_transform\_cross\_comp\_inferred\_flag** equal to 1 indicates that the remapping piece-wise linear functions of the second and third colour components are inferred from the remapping piece-wise linear function of the first colour component. colour\_transform\_cross\_comp\_inferred\_flag equal to 0 indicates that the remapping piece-wise linear functions of the second and third colour components are signalled. When not present, the value of colour\_transform\_cross\_comp\_inferred\_flag is inferred to be equal to 0.

**colour\_transf\_lut**[ c ][ i ] specifies the piecewise linear remapping function of the colour component of index c. When colour\_transf\_lut[ 1 ][ i ] is present and colour\_transf\_lut[ 2 ][ i ] is not present, the value of colour\_transf\_lut[ 2 ][ i ] is inferred to be equal to colour\_transf\_lut[ 1 ][ i ]. The length of colour\_transf\_lut[ c ][ i ] is 2 + bitDepth – log2numLutPoints bits.

**colour\_transform\_lut2\_present\_flag** equal to 1 specifies that colour\_transf\_lut[ 2 ][ i ] is present in the CTI SEI message. colour\_transform\_lut2\_present\_flag equal to 0 specifies that colour\_transf\_lut[ 2 ][ i ] is not present in the CTI SEI message. When not present, the value of colour\_transform\_lut2\_present\_flag is inferred to be equal to 0.

**colour\_transform\_chroma\_offset** specifies the CTI chroma offset. When not present, colour\_transform\_chroma\_offset is inferred to be equal to 0. The length of colour\_transform\_chroma\_offset is 2 + bitDepth – log2numLutPoints bits.

The remapping process of the input picture components rec[ c ], with width and height equal to picWidth[ c ] and picHeight[ c ], respectively, to the output remapped picture components map[ c ], for c=0..2, is performed as follows.

The array pivotPointX is derived as follows.

– For j=0 ..( numLutPoints − 1 ), pivotPointX[ j ] is set equal to ( j  <<  log2distX ).

For c=0..maxIntraComp, the arrays pivotPointY[ c ] and slope[ c ] are derived as follows:

– pivotPointY[ c ][ 0 ] is set equal to colour\_transf\_lut[ c ][ 0 ]

– For j=1..( numLutPoints − 1 ), pivotPointY[ c ][ j ] is derived as follows:

pivotPointY[ c ][ j ] = pivotPointY[ c ][ j − 1 ] + colour\_transf\_lut[ c ][ j ] (68)

– For j=0..( numLutPoints − 1 ), slope[ c ][ j ] is derived as follows:

slope[ c ][ j ] = ( ( colour\_transf\_lut[ c ][ j + 1 ]  <<  11 ) + ( 1  <<  ( log2distX − 1 ) ) )  >>  log2distX (69)

When colour\_transform\_cross\_component\_flag is equal to 1, the arrays ccPivotPointY[ c ] and ccSlope[ c ] are derived as follows, for c=1..2:

– If colour\_transform\_cross\_comp\_inferred\_flag is equal to 0, ccPivotPointY[ c ] is derived as follows:

– For j=0..numLutPoints, ccPivotPointY[ c ][ j ] is set equal to ( colour\_transf\_lut[ c ][ j ]  <<  ( 11 − log2distX ) ).

– Otherwise (colour\_transform\_cross\_comp\_inferred\_flag is equal to 1), ccPivotPointY[ c ] is derived as follows:

– For j=0..( numLutPoints − 1 ), tmpPivotPt[ j ] is derived as follows:

– If colour\_transf\_lut[ 0 ][ j + 1 ] is equal to 0, tmpPivotPt[ j ] is set equal to ( 1  <<  11 ).

– Otherwise, tmpPivotPt[ j ] is derived as follows:

tmpPivotPt[ j ] = ( colour\_transf\_lut[ 0 ][ j + 1 ] + colour\_transform\_chroma\_offset )  << (70)  
 ( 11 – log2distX )

– The array ccPivotPointY[ c ] is derived as follows:

– For j=1..( numLutPoints − 1 ), ccPivotPointY[ c ][ j ] is derived as follows:

ccPivotPointY[ c ][ j ] = ( tmpPivotPt[ j ] + tmpPivotPt[ j − 1 ] + 1 ) / 2 (71)

– ccPivotPointY[ c ][ 0 ] is set equal to tmpPivotPt[ 0 ].

– ccPivotPointY[ c ][ numLutPoints ] is set equal to tmpPivotPt[ numLutPoints − 1 ].

– For j=0..( numLutPoints − 1 ), the value of ccSlope[ c ][ j ] is set equal to ( ccPivotPointY[ c ][ j + 1 ] − ccPivotPointY[ c ][ j ] ).

For c=0..maxIntraComp, the intra-component remapping process of the input samples picture rec[ c ] into the remapped samples picture map[ c ] is performed as follows.

– for i=0..picWidth[ c ] − 1, j=0..picHeight[ c ] − 1, the following applies:

idx = rec[ c ][ i ][ j ]  >>  log2distX  
map[ c ][ i ][ j ] = Clip3( 0, ( 1  <<  bitDepth ) − 1, pivotPointY[ c ][ idx ] + (72)  
 ( ( slope[ c ][ idx ] \* ( rec[ i ][ j ] − pivotPointX[ idx ] ) + ( 1  <<  10 ) )  >>  11 ) )

When colour\_transform\_cross\_component\_flag is equal to 1, for c=1..2, the cross-component remapping process of the input samples picture rec[ c ] into the remapped samples picture map[ c ] is performed as follows:

– offset is set equal to ( 1  <<  ( bitDepth − 1 ) ).

– subWc and subHc are set equal to ( picWidth[ 0 ] / picWidth[ c ] ) and ( picHeight[ 0 ] / picHeight[ c ] ), respectively.

– For i=0..picWidth[ c ] − 1, j=0..picHeight[ c ] − 1, the following applies:

coloc = rec[ 0 ][ i \* SubWc ][ j \* SubHc ]  
idx = coloc  >>  log2distX (73)  
scale = ccPivotPointY[ c ][ idx ] + ( ( ccSlope[ c ][ idx ] \* ( coloc − pivotPointX[ idx ] ) )  >>  log2distX )  
map[ c ][ i ][ j ] = Clip3( 0, ( 1  <<  bitDepth ) − 1,  
 ( ( offset  <<  11 ) + scale \* ( rec[ c ][ i ][ j ] − offset ) + ( 1  <<  10 ) )  >>  11 )

*Remove subclause 8.28.1 (Note: The text was moved and merged, with updates, to subclause 8.30.3).*

*Renumber clause 8.28.2 to be clause 8.28.1 and modify it to be as follows:*

* + 1. **Neural-network post-filter characteristics SEI message**
       1. **Neural-network post-filter characteristics SEI message syntax**

|  |  |
| --- | --- |
| nn\_post\_filter\_characteristics( payloadSize ) { | **Descriptor** |
| **nnpfc\_purpose** | u(16) |
| **nnpfc\_id** | ue(v) |
| **nnpfc\_base\_flag** | u(1) |
| **nnpfc\_mode\_idc** | ue(v) |
| if( nnpfc\_mode\_idc  = =  1 ) { |  |
| while( !byte\_aligned( ) ) |  |
| **nnpfc\_alignment\_zero\_bit\_a** | f(1) |
| **nnpfc\_tag\_uri** | st(v) |
| **nnpfc\_uri** | st(v) |
| } |  |
| **nnpfc\_property\_present\_flag** | u(1) |
| if( nnpfc\_property\_present\_flag ) { |  |
| /\* input and output formatting \*/ |  |
| **nnpfc\_num\_input\_pics\_minus1** | ue(v) |
| if( nnpfc\_num\_input\_pics\_minus1 > 0 ) { |  |
| for( i = 0; i  <=  nnpfc\_num\_input\_pics\_minus1; i++ ) |  |
| **nnpfc\_input\_pic\_filtering\_flag**[ i ] | u(1) |
| **nnpfc\_absent\_input\_pic\_zero\_flag** | u(1) |
| } |  |
| if( ChromaUpsamplingFlag ) |  |
| **nnpfc\_out\_sub\_c\_flag** | u(1) |
| if( ColourizationFlag ) |  |
| **nnpfc\_out\_colour\_format\_idc** | u(2) |
| if( ResolutionResamplingFlag ) { |  |
| **nnpfc\_pic\_width\_num\_minus1** | ue(v) |
| **nnpfc\_pic\_width\_denom\_minus1** | ue(v) |
| **nnpfc\_pic\_height\_num\_minus1** | ue(v) |
| **nnpfc\_pic\_height\_denom\_minus1** | ue(v) |
| } |  |
| if( PictureRateUpsamplingFlag ) |  |
| for( i = 0; i < nnpfc\_num\_input\_pics\_minus1; i++ ) |  |
| **nnpfc\_interpolated\_pics**[ i ] | ue(v) |
| if( TemporalExtrapolationFlag ) |  |
| **nnpfc\_extrapolated\_pics\_minus1** | ue(v) |
| if( SpatialExtrapolationFlag ) { |  |
| **nnpfc\_spatial\_extrapolation\_left\_offset** | se(v) |
| **nnpfc\_spatial\_extrapolation\_right\_offset** | se(v) |
| **nnpfc\_spatial\_extrapolation\_top\_offset** | se(v) |
| **nnpfc\_spatial\_extrapolation\_bottom\_offset** | se(v) |
| } |  |
| **nnpfc\_component\_last\_flag** | u(1) |
| **nnpfc\_inp\_format\_idc** | ue(v) |
| **nnpfc\_auxiliary\_inp\_idc** | ue(v) |
| if( ( nnpfc\_auxiliary\_inp\_idc & 2 ) > 0 ) { |  |
| **nnpfc\_inband\_prompt\_flag** | u(1) |
| if( nnpfc\_inband\_prompt\_flag ) { |  |
| while( !byte\_aligned( ) ) |  |
| **nnpfc\_alignment\_zero\_bit\_c** | f(1) |
| **nnpfc\_prompt** | st(v) |
| } |  |
| } |  |
| **nnpfc\_inp\_order\_idc** | ue(v) |
| if( nnpfc\_inp\_format\_idc  = =  1 ) { |  |
| if( nnpfc\_inp\_order\_idc  !=  1 ) |  |
| **nnpfc\_inp\_tensor\_luma\_bitdepth\_minus8** | ue(v) |
| if( nnpfc\_inp\_order\_idc > 0 ) |  |
| **nnpfc\_inp\_tensor\_chroma\_bitdepth\_minus8** | ue(v) |
| } |  |
| **nnpfc\_out\_format\_idc** | ue(v) |
| **nnpfc\_out\_order\_idc** | ue(v) |
| if( nnpfc\_out\_format\_idc  = =  1 ) { |  |
| if( nnpfc\_out\_order\_idc  !=  1 ) |  |
| **nnpfc\_out\_tensor\_luma\_bitdepth\_minus8** | ue(v) |
| if( nnpfc\_out\_order\_idc  !=  0 ) |  |
| **nnpfc\_out\_tensor\_chroma\_bitdepth\_minus8** | ue(v) |
| } |  |
| **nnpfc\_separate\_colour\_description\_present\_flag** | u(1) |
| if( nnpfc\_separate\_colour\_description\_present\_flag ) { |  |
| **nnpfc\_colour\_primaries** | u(8) |
| **nnpfc\_transfer\_characteristics** | u(8) |
| if( nnpfc\_out\_format\_idc  = =  1 ) { |  |
| **nnpfc\_matrix\_coeffs** | u(8) |
| **nnpfc\_full\_range\_flag** | u(1) |
| } |  |
| } |  |
| if( nnpfc\_out\_order\_idc > 0 ) |  |
| **nnpfc\_chroma\_loc\_info\_present\_flag** | u(1) |
| if( nnpfc\_chroma\_loc\_info\_present\_flag ) |  |
| **nnpfc\_chroma\_sample\_loc\_type\_frame** | ue(v) |
| if( !SpatialExtrapolationFlag ) { |  |
| **nnpfc\_overlap** | ue(v) |
| **nnpfc\_constant\_patch\_size\_flag** | u(1) |
| } |  |
| if( nnpfc\_constant\_patch\_size\_flag ) { |  |
| **nnpfc\_patch\_width\_minus1** | ue(v) |
| **nnpfc\_patch\_height\_minus1** | ue(v) |
| } else { |  |
| **nnpfc\_extended\_patch\_width\_cd\_delta\_minus1** | ue(v) |
| **nnpfc\_extended\_patch\_height\_cd\_delta\_minus1** | ue(v) |
| } |  |
| **nnpfc\_padding\_type** | ue(v) |
| if( nnpfc\_padding\_type  = =  4 ) { |  |
| if( nnpfc\_inp\_order\_idc  !=  1 ) |  |
| **nnpfc\_luma\_padding\_val** | ue(v) |
| if( nnpfc\_inp\_order\_idc  !=  0 ) { |  |
| **nnpfc\_cb\_padding\_val** | ue(v) |
| **nnpfc\_cr\_padding\_val** | ue(v) |
| } |  |
| } |  |
| **nnpfc\_complexity\_info\_present\_flag** | u(1) |
| if( nnpfc\_complexity\_info\_present\_flag ) { |  |
| **nnpfc\_parameter\_type\_idc** | u(2) |
| if( nnpfc\_parameter\_type\_idc  !=  2 ) |  |
| **nnpfc\_log2\_parameter\_bit\_length\_minus3** | u(2) |
| **nnpfc\_num\_parameters\_idc** | u(6) |
| **nnpfc\_num\_kmac\_operations\_idc** | ue(v) |
| **nnpfc\_total\_kilobyte\_size** | ue(v) |
| } |  |
| **nnpfc\_num\_metadata\_extension\_bits** | ue(v) |
| if( nnpfc\_num\_metadata\_extension\_bits > 0 ) { |  |
| if( nnpfc\_purpose = = 0 ) { |  |
| **nnpfc\_application\_purpose\_tag\_uri\_present\_flag** | u(1) |
| if( nnpfc\_application\_purpose\_tag\_uri\_present\_flag ) { |  |
| while( !byte\_aligned( ) ) |  |
| **nnpfc\_metadata\_alignment\_zero\_bit** | f(1) |
| **nnpfc\_application\_purpose\_tag\_uri** | st(v) |
| } |  |
| } |  |
| if( SpatialExtrapolationFlag  | |  ResolutionResamplingFlag ) |  |
| **nnpfc\_scan\_type\_idc** | u(2) |
| **nnpfc\_for\_human\_viewing\_idc** | u(2) |
| **nnpfc\_for\_machine\_analysis\_idc** | u(2) |
| **nnpfc\_reserved\_metadata\_extension** | u(v) |
| } |  |
| } |  |
| /\* ISO/IEC 15938-17 bitstream \*/ |  |
| if( nnpfc\_mode\_idc  = =  0 ) { |  |
| while( !byte\_aligned( ) ) |  |
| **nnpfc\_alignment\_zero\_bit\_b** | f(1) |
| for( i = 0; more\_data\_in\_payload( ); i++ ) |  |
| **nnpfc\_payload\_byte**[ i ] | b(8) |
| } |  |
| } |  |

* + - 1. **Neural-network post-filter characteristics SEI message semantics**

The neural-network post-filter characteristics (NNPFC) SEI message specifies a neural network that may be used as a post-processing filter. The use of specified neural-network post-processing filters (NNPFs) for specific pictures is indicated with neural-network post-filter activation (NNPFA) SEI messages.

Use of this SEI message requires the definition of the following variables:

– Input picture width and height in units of luma samples, denoted herein by CroppedWidth and CroppedHeight, respectively.

– Luma sample array CroppedYPic[ idx ] and chroma sample arrays CroppedCbPic[ idx ] and CroppedCrPic[ idx ], when present, of the input pictures with index idx in the range of 0 to numInputPics − 1, inclusive, that are used as input for the NNPF.

– Bit depth BitDepthY for the luma sample array of the input pictures.

– Bit depth BitDepthC for the chroma sample arrays, if any, of the input pictures.

– A chroma format indicator, denoted herein by ChromaFormatIdc, as described in clause 7.3.

– When nnpfc\_auxiliary\_inp\_idc is equal to 1, a filtering strength control value array StrengthControlVal[ idx ] that shall contain real numbers in the range of 0 to 1, inclusive, of the input pictures with index idx in the range of 0 to numInputPics − 1, inclusive.

Input picture with index 0 corresponds to the picture for which the NNPF defined by this NNPFC SEI message is activated by an NNPFA SEI message. Input picture with index i in the range of 1 to numInputPics − 1, inclusive, precedes the input picture with index i − 1 in output order.

The variables SubWidthC and SubHeightC are derived from ChromaFormatIdc as specified by Table 2.

NOTE 1 – More than one NNPFC SEI message can be present for the same picture. When more than one NNPFC SEI message with different values of nnpfc\_id is present or activated for the same picture, they can have the same value or different values of nnpfc\_purpose and the same value or different values of nnpfc\_mode\_idc.

**nnpfc\_purpose** indicates the purpose of the NNPF as specified in Table 20, where ( nnpfc\_purpose & bitMask ) not equal to 0 indicates that the NNPF has the purpose associated with the bitMask value in Table 20. When nnpfc\_purpose is greater than 0 and ( nnpfc\_purpose & bitMask ) is equal to 0, the purpose associated with the bitMask value is not applicable to the NNPF. When nnpfc\_pupose is equal to 0, the NNPF may be used as determined by the application and as specified by the nnpfc\_application\_purpose\_tag\_uri.

All NNPFC SEI messages with a particular value of nnpfc\_id within a CLVS shall have the same value of nnpfc\_purpose.

The value of nnpfc\_purpose shall be in the range of 0 to 255, inclusive, in bitstreams conforming to this version of this Specification. Values of 256 to 65 535, inclusive, for nnpfc\_purpose are reserved for future use by ITU-T | ISO/IEC and shall not be present in bitstreams conforming to this version of this Specification. Decoders conforming to this version of this Specification shall ignore NNPFC SEI messages with nnpfc\_purpose in the range of 256 to 65 535, inclusive.

**Table 20 – Definition of nnpfc\_purpose**

|  |  |
| --- | --- |
| **bitMask** | **Interpretation** |
| 0x01 | General visual quality improvement |
| 0x02 | Chroma upsampling (from the 4:2:0 chroma format to the 4:2:2 or 4:4:4 chroma format, or from the 4:2:2 chroma format to the 4:4:4 chroma format) |
| 0x04 | Resolution resampling (increasing or decreasing the width or height) |
| 0x08 | Picture rate upsampling |
| 0x10 | Bit depth upsampling (increasing the luma bit depth or the chroma bit depth) |
| 0x20 | Colourization |
| 0x40 | Temporal extrapolation (i.e., generating one or more future pictures) |
| 0x80 | Spatial extrapolation (i.e., generating content outside of the spatial area of the input pictures), possibly also with removal (i.e. remove partial content from the input pictures) |

The variables ChromaUpsamplingFlag, ResolutionResamplingFlag, PictureRateUpsamplingFlag, BitDepthUpsamplingFlag, ColourizationFlag, and TemporalExtrapolationFlag, specifying whether nnpfc\_purpose indicates the purpose of the NNPF to include chroma upsampling, resolution resampling, picture rate upsampling, bit depth upsampling, colourization, and temporal extrapolation, respectively, are derived as follows:

ChromaUpsamplingFlag = ( ( nnpfc\_purpose & 0x02 ) > 0 ) ? 1 : 0  
ResolutionResamplingFlag = ( ( nnpfc\_purpose & 0x04 ) > 0 ) ? 1 : 0  
PictureRateUpsamplingFlag = ( ( nnpfc\_purpose & 0x08 ) > 0 ) ? 1 : 0 (75)  
BitDepthUpsamplingFlag = ( ( nnpfc\_purpose & 0x10 ) > 0 ) ? 1 : 0  
ColourizationFlag = ( ( nnpfc\_purpose & 0x20 ) > 0 ) ? 1 : 0  
TemporalExtrapolationFlag = ( ( nnpfc\_purpose & 0x40 ) > 0 ) ? 1 : 0  
SpatialExtrapolationFlag = ( ( nnpfc\_purpose & 0x80 ) > 0 ) ? 1 : 0

NOTE 2 – When a reserved value of nnpfc\_purpose is taken into use in the future by ITU-T | ISO/IEC, the syntax of this SEI message could be extended with syntax elements whose presence is conditioned by nnpfc\_purpose being equal to that value or any one of a set of values including that value.

When ChromaFormatIdc is equal to 3, ChromaUpsamplingFlag shall be equal to 0.

When ChromaUpsamplingFlag is equal to 1, ColourizationFlag shall be equal to 0.

When PictureRateUpsamplingFlag or TemporalExtrapolationFlag is equal to 1 and the input picture with index 0 is associated with a frame packing arrangement SEI message with fp\_arrangement\_type equal to 5, all input pictures are associated with a frame packing arrangement SEI message with fp\_arrangement\_type equal to 5 and the same value of fp\_current\_frame\_is\_frame0\_flag.

When TemporalExtrapolationFlag is equal to 1, the extrapolated pictures generated by the NNPF follow all input pictures of the NNPF in output order. When TemporalExtrapolationFlag is equal to 1 and there is a decoded output picture that follows, in output order, the current picture for which the NNPF is activated, the extrapolated pictures generated by the NNPF precede that decoded output picture in output order.

**nnpfc\_id** contains an identifying number that may be used to identify an NNPF. The value of nnpfc\_id shall be in the range of 0 to 232 − 2, inclusive. Values of nnpfc\_id from 256 to 511, inclusive, and from 231 to 232 − 2, inclusive, are reserved for future use by ITU-T | ISO/IEC. Decoders conforming to this version of this Specification encountering an NNPFC SEI message with nnpfc\_id in the range of 256 to 511, inclusive, or in the range of 231 to 232 − 2, inclusive, shall ignore the SEI message.

When an NNPFC SEI message is the first NNPFC SEI message, in decoding order, that has a particular nnpfc\_id value within the current CLVS, the following applies:

– This SEI message specifies a base NNPF.

– This SEI message pertains to the current decoded picture and all subsequent decoded pictures of the current layer, in output order, until the end of the current CLVS.

**nnpfc\_base\_flag** equal to 1 specifies that the SEI message specifies the base NNPF. nnpfc\_base\_flag equal to 0 specifies that the SEI message specifies an update relative to the base NNPF.

The following constraints apply to the value of nnpfc\_base\_flag:

– When an NNPFC SEI message is the first NNPFC SEI message, in decoding order, that has a particular nnpfc\_id value within the current CLVS, the value of nnpfc\_base\_flag shall be equal to 1.

– All NNPFC SEI messages in a CLVS that have a particular nnpfc\_id value and nnpfc\_base\_flag equal to 1 shall have identical SEI payload content.

When nnpfc\_base\_flag is equal to 0, the following applies:

– This SEI message defines an update relative to the preceding base NNPF in decoding order with the same nnpfc\_id value. Updates are not cumulative but rather each update is applied on the base NNPF, which is the NNPF specified by the first NNPFC SEI message, in decoding order, that has a particular nnpfc\_id value within the current CLVS. The NNPF defined by this SEI message is obtained by applying the update defined by this SEI message relative to the base NNPF with the same nnpfc\_id value.

– This SEI message pertains to the current decoded picture and all subsequent decoded pictures of the current layer, in output order, until the end of the current CLVS or up to but excluding the decoded picture that follows the current decoded picture in output order within the current CLVS and is associated with a subsequent NNPFC SEI message, in decoding order, having nnpfc\_base\_flag equal to 0 and that particular nnpfc\_id value within the current CLVS, whichever is earlier.

**nnpfc\_mode\_idc**, when equal to 0, indicates that the neural network information is contained in the NNPFC SEI message, and the neural network information is in the format of an ISO/IEC 15938-17 bitstream. nnpfc\_mode\_idc equal to 1 indicates that the neural network information is identified by the URI indicated by nnpfc\_uri with the format identified by the tag URI nnpfc\_tag\_uri.

The value of nnpfc\_mode\_idc shall be in the range of 0 to 255, inclusive. Values of 2 to 255, inclusive, for nnpfc\_mode\_idc are reserved for future use by ITU-T | ISO/IEC and shall not be present in bitstreams conforming to this version of this Specification. Decoders conforming to this version of this Specification shall ignore NNPFC SEI messages with nnpfc\_mode\_idc in the range of 2 to 255, inclusive.

**nnpfc\_alignment\_zero\_bit\_a** shall be equal to 0.

**nnpfc\_tag\_uri** contains a tag URI with syntax and semantics as specified in IETF RFC 4151 identifying the format and associated information about the neural network used as a base NNPF or an update relative to the base NNPF with the same nnpfc\_id value specified by nnpfc\_uri.

NOTE 3 – nnpfc\_tag\_uri enables uniquely identifying the format of neural network data specified by nnpfc\_uri without needing a central registration authority.

nnpfc\_tag\_uri equal to "tag:iso.org,2023:15938-17" indicates that the neural network data identified by nnpfc\_uri conforms to ISO/IEC 15938-17.

**nnpfc\_uri** contains a URI with syntax and semantics as specified in IETF Internet Standard 66 identifying the neural network used as a base NNPF or an update relative to the base NNPF with the same nnpfc\_id value.

**nnpfc\_property\_present\_flag** equal to 1 specifies that syntax elements related to the filter properties including purpose, input formatting, output formatting, and complexity are present. nnpfc\_property\_present\_flag equal to 0 specifies that no syntax elements related to the filter properties are present.

When nnpfc\_base\_flag is equal to 1, nnpfc\_property\_present\_flag shall be equal to 1.

When nnpfc\_property\_present\_flag is equal to 0, the values of all syntax elements that may be present only when nnpfc\_property\_present\_flag is equal to 1 are inferred to be equal to their corresponding syntax elements, respectively, in the NNPFC SEI message that contains the base NNPF for which this SEI message provides an update.

When an NNPFC SEI message nnpfcCurr is not the first NNPFC SEI message, in decoding order, that has a particular nnpfc\_id value within the current CLVS, is not a repetition of the first NNPFC SEI message with that particular nnpfc\_id value (in this case the value of nnpfc\_base\_flag is equal to 0), and the value of nnpfc\_property\_present\_flag is equal to 1, the following constraints apply:

– The values of syntax elements following nnpfc\_property\_present\_flag and preceding nnpfc\_complexity\_info\_present\_flag, in decoding order, in the NNPFC SEI message shall be the same as the values of corresponding syntax elements in the first NNPFC SEI message, in decoding order, that has that particular nnpfc\_id value within the current CLVS.

– Either nnpfc\_complexity\_info\_present\_flag shall be equal to 0 or both nnpfc\_complexity\_info\_present\_flag shall be equal to 1 in the first NNPFC SEI message, in decoding order, that has that particular nnpfc\_id value within the current CLVS (denoted as nnpfcBase below) and all the following constraints apply:

– nnpfc\_parameter\_type\_idc in nnpfcCurr shall be equal to nnpfc\_parameter\_type\_idc in nnpfcBase.

– nnpfc\_log2\_parameter\_bit\_length\_minus3 in nnpfcCurr, when present, shall be less than or equal to nnpfc\_log2\_parameter\_bit\_length\_minus3 in nnpfcBase.

– If nnpfc\_num\_parameters\_idc in nnpfcBase is equal to 0, nnpfc\_num\_parameters\_idc in nnpfcCurr shall be equal to 0.

– Otherwise (nnpfc\_num\_parameters\_idc in nnpfcBase is greater than 0), nnpfc\_num\_parameters\_idc in nnpfcCurr shall be greater than 0 and less than or equal to nnpfc\_num\_parameters\_idc in nnpfcBase.

– If nnpfc\_num\_kmac\_operations\_idc in nnpfcBase is equal to 0, nnpfc\_num\_kmac\_operations\_idc in nnpfcCurr shall be equal to 0.

– Otherwise (nnpfc\_num\_kmac\_operations\_idc in nnpfcBase is greater than 0), nnpfc\_num\_kmac\_operations\_idc in nnpfcCurr shall be greater than 0 and less than or equal to nnpfc\_num\_kmac\_operations\_idc in nnpfcBase.

– If nnpfc\_total\_kilobyte\_size in nnpfcBase is equal to 0, nnpfc\_total\_kilobyte\_size in nnpfcCurr shall be equal to 0.

– Otherwise (nnpfc\_total\_kilobyte\_size in nnpfcBase is greater than 0), nnpfc\_total\_kilobyte\_size in nnpfcCurr shall be greater than 0 and less than or equal to nnpfc\_total\_kilobyte\_size in nnpfcBase.

**nnpfc\_num\_input\_pics\_minus1** plus 1 specifies the number of pictures used as input for the NNPF. The value of nnpfc\_num\_input\_pics\_minus1 shall be in the range of 0 to 63, inclusive. When PictureRateUpsamplingFlag is equal to 1, the value of nnpfc\_num\_input\_pics\_minus1 shall be greater than 0.

The variable numInputPics, specifying the number of pictures used as input for the NNPF, is derived as follows:

numInputPics = nnpfc\_num\_input\_pics\_minus1 + 1 (76)

**nnpfc\_input\_pic\_filtering\_flag**[ i ] equal to 1 indicates that for the i-th input picture the NNPF generates a corresponding output picture. nnpfc\_input\_pic\_filtering\_flag[ i ] equal to 0 indicates that for the i-th input picture the NNPF does not generate a corresponding output picture. Each NNPF-generated picture is stored in the output tensor of the NNPF. When nnpfc\_num\_input\_pics\_minus1 is equal to 0, nnpfc\_input\_pic\_filtering\_flag[ 0 ] is inferred to be equal to 1. When PictureRateUpsamplingFlag is equal to 0 and nnpfc\_num\_input\_pics\_minus1 is greater than 0, nnpfc\_input\_pic\_filtering\_flag[ i ] shall be equal to 1 for at least one value of i in the range of 0 to nnpfc\_num\_input\_pics\_minus1, inclusive.

**nnpfc\_absent\_input\_pic\_zero\_flag** equal to 1 indicates that the NNPF expects an input picture that is not present in the bitstream to be represented by sample arrays with sample values equal to 0. nnpfc\_absent\_input\_pic\_zero\_flag equal to 0 indicates that the NNPF expects an input picture inputPicA that is not present in the bitstream to be represented by the input picture inputPicB that is the closest to inputPicA in output order and is present in the bitstream.

**nnpfc\_out\_sub\_c\_flag** specifies the values of the variables outSubWidthC and outSubHeightC when ChromaUpsamplingFlag is equal to 1. nnpfc\_out\_sub\_c\_flag equal to 1 specifies that outSubWidthC is equal to 1 and outSubHeightC is equal to 1. nnpfc\_out\_sub\_c\_flag equal to 0 specifies that outSubWidthC is equal to 2 and outSubHeightC is equal to 1. When ChromaFormatIdc is equal to 2 and nnpfc\_out\_sub\_c\_flag is present, the value of nnpfc\_out\_sub\_c\_flag shall be equal to 1.

**nnpfc\_out\_colour\_format\_idc**, when ColourizationFlag is equal to 1, specifies the colour format of the NNPF-generated pictures and consequently the values of the variables outSubWidthC and outSubHeightC. nnpfc\_out\_colour\_format\_idc equal to 1 specifies that the colour format of the NNPF-generated pictures is the 4:2:0 format and outSubWidthC and outSubHeightC are both equal to 2. nnpfc\_out\_colour\_format\_idc equal to 2 specifies that the colour format of the NNPF-generated pictures is the 4:2:2 format and outSubWidthC is equal to 2 and outSubHeightC is equal to 1. nnpfc\_out\_colour\_format\_idc equal to 3 specifies that the colour format of the NNPF-generated pictures is the 4:4:4 format and outSubWidthC and outSubHeightC are both equal to 1. The value of nnpfc\_out\_colour\_format\_idc shall not be equal to 0.

When ChromaUpsamplingFlag and ColourizationFlag are both equal to 0, outSubWidthC and outSubHeightC are inferred to be equal to SubWidthC and SubHeightC, respectively.

**nnpfc\_pic\_width\_num\_minus1** plus 1 and **nnpfc\_pic\_width\_denom\_minus1** plus 1 specify the numerator and denominator, respectively, for the resampling ratio of the width of the NNPF-generated pictures relative to CroppedWidth. Both nnpfc\_pic\_width\_num\_minus1 and nnpfc\_pic\_width\_denom\_minus1 shall be in the range of 0 to 65 535, inclusive.

The value of ( nnpfc\_pic\_width\_num\_minus1 + 1 ) ÷ ( nnpfc\_pic\_width\_denom\_minus1 + 1 ) shall be in the range of 1 ÷ 16 to 16, inclusive. When nnpfc\_pic\_width\_num\_minus1 and nnpfc\_pic\_width\_denom\_minus1 are not present, the values of nnpfc\_pic\_width\_num\_minus1 and nnpfc\_pic\_width\_denom\_minus1 are both inferred to be equal to 0.

The variables nnpfcOutputPicWidth and nnpfcOutputPicHeight respectively represent the width and height of the luma sample arrays of the NNPF-generated pictures when SpatialExtrapolationFlag is equal to 0.

The variables nnpfcOutputPicWidth1 and nnpfcOutputPicHeight1 respectively represent the width and height of the luma sample arrays of the NNPF-generated pictures when SpatialExtrapolationFlag is equal to 1.

The variable nnpfcOutputPicWidthis derived as follows:

nnpfcOutputPicWidth = Ceil( CroppedWidth \*  (77)  
 ( nnpfc\_pic\_width\_num\_minus1 + 1 ) ÷ ( nnpfc\_pic\_width\_denom\_minus1 + 1 ) )

When SpatialExtrapolationFlag is equal to 1, nnpfcOutputPicWidth1 is derived as follows:

nnpfcOutputPicWidth1 = nnpfcOutputPicWidth + outSubWidthC \* (nnpfc\_spatial\_extrapolation\_left\_offset + nnpfc\_spatial\_extrapolation\_right\_offset) (77)

[Ed. (YK): For now don't worry about the equation index values; all fields, including these, will be updated after the changes are integrated into the basis text.]

It is a requirement of bitstream conformance that when SpatialExtrapolationFlag is equal to 0, nnpfcOutputPicWidth shall be greater than 0 and nnpfcOutputPicWidth % outSubWidthC shall be equal to 0.

It is a requirement of bitstream conformance that when SpatialExtrapolationFlag is equal to 1, nnpfcOutputPicWidth1 shall be greater than 0 and nnpfcOutputPicWidth1 % outSubWidthC shall be equal to 0.

**nnpfc\_pic\_height\_num\_minus1** plus 1 and **nnpfc\_pic\_height\_denom\_minus1** plus 1 specify the numerator and denominator, respectively, for the resampling ratio of the height of the NNPF-generated pictures relative to CroppedHeight. Both nnpfc\_pic\_height\_num\_minus1 and nnpfc\_pic\_height\_denom\_minus1 shall be in the range of 0 to 65 535, inclusive.

The value of ( nnpfc\_pic\_height\_num\_minus1 + 1 ) ÷ ( nnpfc\_pic\_height\_denom\_minus1 + 1 ) shall be in the range of 1 ÷ 16 to 16, inclusive. When nnpfc\_pic\_height\_num\_minus1 and nnpfc\_pic\_height\_denom\_minus1 are not present, the values of nnpfc\_pic\_height\_num\_minus1 and nnpfc\_pic\_height\_denom\_minus1 are both inferred to be equal to 0.

The variable nnpfcOutputPicHeight is derived as follows:

nnpfcOutputPicHeight = Ceil( CroppedHeight \* (78)  
 ( nnpfc\_pic\_height\_num\_minus1 + 1 ) ÷ ( nnpfc\_pic\_height\_denom\_minus1 + 1 ) )

When SpatialExtrapolationFlag is equal to 1, nnpfcOutputPicHeight1 is derived as follows:

nnpfcOutputPicHeight1 = nnpfcOutputPicHeight + outSubHeightC \* ( nnpfc\_spatial\_extrapolation\_top\_offset + nnpfc\_spatial\_extrapolation\_bottom\_offset ) (78)

It is a requirement of bitstream conformance that when SpatialExtrapolationFlag is equal to 0, nnpfcOutputPicHeight shall be greater than 0 and nnpfcOutputPicHeight % outSubHeightC shall be equal to 0.

It is a requirement of bitstream conformance that when SpatialExtrapolationFlag is equal to 1, nnpfcOutputPicHeight1 shall be greater than 0 and nnpfcOutputPicHeight1 % outSubHeightC shall be equal to 0.

When ResolutionResamplingFlag is equal to 1, at least one the following conditions shall be true:

– The value of nnpfcOutputPicWidth is not equal to CroppedWidth.

– The value of nnpfcOutputPicHeight is not equal to CroppedHeight.

– SpatialExtrapolationFlag is equal to 1.

**nnpfc\_interpolated\_pics**[ i ] specifies the number of interpolated pictures generated by the NNPF between the i-th and the ( i + 1 )-th input picture for the NNPF. The value of nnpfc\_interpolated\_pics[ i ] shall be in the range of 0 to 63, inclusive. When the nnpfc\_interpolated\_pics[ i ] syntax elements are present, the value of nnpfc\_interpolated\_pics[ i ] shall be greater than 0 for at least one value of i in the range of 0 to nnpfc\_num\_input\_pics\_minus1 − 1, inclusive.

NOTE 4 – When PictureRateUpsamplingFlag is equal to 1 for an NNPF and the NNPFA SEI message that activated this NNPF has nnpfa\_persistence\_flag equal to 1, only for a single value of i in the range of 0 to numInputPics − 1, inclusive, the value of nnpfc\_interpolated\_pics[ i ] is greater than 0.

**nnpfc\_extrapolated\_pics\_minus1** plus 1 specifies the number of extrapolated pictures generated by the NNPF subsequent to all input pictures for the NNPF in output order. The value of nnpfc\_extrapolated\_pics\_minus1 shall be in the range of 0 to 62, inclusive.

The variables NumInpPicsInOutputTensor, specifying the number of pictures that have a corresponding input picture and are present in the output tensor of the NNPF, InpIdx[ idx ], specifying the input picture index, to the list of input pictures in reverse output order, of the idx-th picture that is present in the output tensor of the NNPF and has a corresponding input picture, and numPicsInOutputTensor, specifying the total number of pictures present in the output tensor of the NNPF, are derived as follows:

for( i = 0, numPicsInOutputTensor = 0; i < numInputPics; i++ )  
 if( nnpfc\_input\_pic\_filtering\_flag[ i ] ) {  
 InpIdx[ numPicsInOutputTensor ] = i  
 numPicsInOutputTensor++  
 } (79)  
NumInpPicsInOutputTensor = numPicsInOutputTensor  
if( PictureRateUpsamplingFlag )  
 for( i = 0; i  <=  numInputPics − 2; i++ )  
 numPicsInOutputTensor  +=  nnpfc\_interpolated\_pics[ i ]  
if( TemporalExtrapolationFlag )  
 numPicsInOutputTensor  +=  nnpfc\_extrapolated\_pics + 1

**nnpfc\_spatial\_extrapolation\_left\_offset**, **nnpfc\_spatial\_extrapolation\_right\_offset**, **nnpfc\_spatial\_‌extrapolation\_top\_offset**, and **nnpfc\_spatial\_extrapolation\_bottom\_offset** specify the spatial extrapolation area. When nnpfc\_spatial\_extrapolation\_left\_offset, nnpfc\_spatial\_extrapolation\_right\_offset, nnpfc\_spatial\_extrapolation\_top\_offset, and nnpfc\_spatial\_extrapolation\_bottom\_offset are each greater than or equal to 0, the luma samples with horizontal picture coordinates from outSubWidthC \* nnpfc\_spatial\_extrapolation\_left\_offset to nnpfcOutputPicWidth1 − ( outSubWidthC \* nnpfc\_spatial\_extrapolation\_right\_offset ) and vertical picture coordinates from outSubHeightC \* nnpfc\_spatial\_extrapolation\_top\_offset to nnpfcOutputPicHeight1 − ( outSubHeightC \* nnpfc\_spatial\_extrapolation\_bottom\_offset ) correspond to the spatial area of the input picture. The value of nnpfc\_spatial\_extrapolation\_left\_offset, nnpfc\_spatial\_extrapolation\_right\_offset, nnpfc\_spatial\_**‌**extrapolation\_top\_offset and nnpfc\_spatial\_extrapolation\_bottom\_offset shall be in the range of −65 536 to 65 536, inclusive. At least one of nnpfc\_spatial\_extrapolation\_left\_offset, nnpfc\_spatial\_extrapolation\_right\_offset, nnpfc\_spatial\_extrapolation\_top\_offset and nnpfc\_spatial\_extrapolation\_bottom\_offset shall be greater than 0.

**nnpfc\_component\_last\_flag** equal to 1 indicates that the last dimension in the input tensor inputTensor to the NNPF and the output tensor outputTensor of the NNPF is used for a current channel. nnpfc\_component\_last\_flag equal to 0 indicates that the third dimension in the input tensor inputTensor to the NNPF and the output tensor outputTensor of the NNPF is used for a current channel.

NOTE 5 – The first dimension in the input tensor and in the output tensor is used for the batch index, which is a common practice in some neural network frameworks. While the equations in the semantics of this SEI message use the batch size corresponding to the batch index equal to 0, it is up to the post-processing implementation to determine the batch size used as the input to the neural network inference process.

NOTE 6 – For example, when nnpfc\_inp\_order\_idc is equal to 3 and nnpfc\_auxiliary\_inp\_idc is equal to 1, there are 7 channels in the input tensor, including four luma matrices, two chroma matrices, and one auxiliary input matrix. In this case, the process DeriveInputTensors( ) would derive each of these 7 channels of the input tensor one by one, and when a particular channel of these channels is processed, that channel is referred to as the current channel during the process.

**nnpfc\_inp\_format\_idc** indicates the method of converting a sample value of the input picture to an input value to the NNPF. The value of nnpfc\_inp\_format\_idc shall be in the range of 0 to 255, inclusive. Values of nnpfc\_inp\_format\_idc in the range of 2 to 255, inclusive, are reserved for future specification by ITU-T | ISO/IEC and shall not be present in bitstreams conforming to this version of this Specification. Decoders conforming to this version of this Specification shall ignore NNPFC SEI messages with nnpfc\_inp\_format\_idc in the range of 2 to 255, inclusive.

When nnpfc\_inp\_format\_idc is equal to 0, the input values to the NNPF are real numbers and the functions InpY( ) and InpC( ) are specified as follows:

InpY( x ) = x ÷ ( ( 1  <<  BitDepthY ) − 1 ) (80)

InpC( x )= x ÷ ( ( 1  <<  BitDepthC ) − 1 ) (81)

When nnpfc\_inp\_format\_idc is equal to 1, the input values to the NNPF are unsigned integer numbers and the functions InpY( ) and InpC( ) are specified as follows:

shiftY = BitDepthY − inpTensorBitDepthY  
if( inpTensorBitDepthY >= BitDepthY)  
 InpY( x ) = x  <<  ( inpTensorBitDepthY − BitDepthY ) (82)  
else  
 InpY( x ) = Clip3(0, ( 1  <<  inpTensorBitDepthY ) − 1, ( x + ( 1  <<  ( shiftY − 1 ) ) )  >>  shiftY )

shiftC = BitDepthC − inpTensorBitDepthC  
if( inpTensorBitDepthC >= BitDepthC )  
 InpC( x ) = x  <<  ( inpTensorBitDepthC − BitDepthC ) (83)  
else  
 InpC( x ) = Clip3(0, ( 1  <<  inpTensorBitDepthC ) − 1, ( x + ( 1  <<  ( shiftC − 1 ) ) )  >>  shiftC )

The variable inpTensorBitDepthY is derived from the syntax element nnpfc\_inp\_tensor\_luma\_bitdepth\_minus8 as specified below. The variable inpTensorBitDepthC is derived from the syntax element nnpfc\_inp\_tensor\_chroma\_bitdepth\_minus8 as specified below.

**nnpfc\_auxiliary\_inp\_idc** greater than 0 indicates that auxiliary input data is present in the input tensor of the NNPF. nnpfc\_auxiliary\_inp\_idc equal to 0 indicates that auxiliary input data is not present in the input tensor. nnpfc\_auxiliary\_inp\_idc equal to 1, 2 or 3 specifies that auxiliary input data is derived as specified in Equation 95.

The value of nnpfc\_auxiliary\_inp\_idc shall be in the range of 0 to 255, inclusive. Values of 4 to 255, inclusive, for nnpfc\_auxiliary\_inp\_idc are reserved for future use by ITU-T | ISO/IEC and shall not be present in bitstreams conforming to this version of this Specification. Decoders conforming to this version of this Specification shall ignore NNPFC SEI messages with nnpfc\_auxiliary\_inp\_idc in the range of 4 to 255, inclusive.

When nnpfc\_auxiliary\_inp\_idc is equal to 1 the auxiliary input data consists of strengthControlScaledVal[ i ].

When nnpfc\_auxiliary\_inp\_idc is equal to 2 the auxiliary input data consists of text prompt character values.

When nnpfc\_auxiliary\_inp\_idc is equal to 3, the auxiliary input data consists of strengthControlScaledVal[ i ] and text prompt character values.

**nnpfc\_inband\_prompt\_flag** equal to 1 specifies that the text prompt string to be included in the input tensor is included in this NNPFC SEI message or in an NNPFA SEI message activating the NNPF defined by this NNPFC SEI message. nnpfc\_inband\_prompt\_flag equal to 0 specifies that the text prompt string to be included in the input tensor is provided to the decoding system by external means.

**nnpfc\_alignment\_zero\_bit\_c** shall be equal to 0.

**nnpfc\_prompt** specifies the text string prompt used as input for an NNPF, for example for generating the contents of the spatial extrapolation image area. When nnpfc\_prompt is present, nnpfc\_prompt shall not be a null string.

The variable nnpfcPrompt, specifying the text prompt string to be provided in the input tensor for a particular picture picA for which the NNPF is activated, is derived as follows:

– If nnpfc\_inband\_prompt\_flag is equal to 1 and nnpfa\_prompt\_update\_flag is equal to 1, nnpfcPrompt is set equal to nnpfa\_prompt.

– Otherwise, if nnpfc\_inband\_prompt\_flag is equal to 1 and nnpfa\_prompt\_update\_flag is equal to 0, nnpfcPrompt is set equal to nnpfc\_prompt.

– Otherwise, if nnpfc\_inband\_prompt\_flag is equal to 0 and a text prompt string is provided by external means, nnpfcPrompt is set equal to that text prompt string.

– Otherwise (nnpfc\_inband\_prompt\_flag is equal to 0 and no text prompt string is provided by external means), nnpfcPrompt is set equal to a null string.

**nnpfc\_inp\_order\_idc** indicates the method of ordering the sample arrays of an input picture to form an input tensor to the NNPF.

The value of nnpfc\_inp\_order\_idc shall be in the range of 0 to 255, inclusive. Values of 4 to 255, inclusive, for nnpfc\_inp\_order\_idc are reserved for future use by ITU-T | ISO/IEC and shall not be present in bitstreams conforming to this version of this Specification. Decoders conforming to this version of this Specification shall ignore NNPFC SEI messages with nnpfc\_inp\_order\_idc in the range of 4 to 255, inclusive.

When ChromaFormatIdc is not equal to 1, nnpfc\_inp\_order\_idc shall not be equal to 3.

When ChromaFormatIdc is equal to 0, nnpfc\_inp\_order\_idc shall be equal to 0.

When ChromaUpsamplingFlag is equal to 1, nnpfc\_inp\_order\_idc shall not be equal to 0.

Table 21 contains an informative description of nnpfc\_inp\_order\_idc values.

**Table 21 – Description of nnpfc\_inp\_order\_idc values**

|  |  |
| --- | --- |
| **nnpfc\_inp\_ order\_idc** | **Description** |
| 0 | If nnpfc\_auxiliary\_inp\_idc is equal to 0, one luma matrix is present in the input tensor for each input picture, and the number of channels is 1. Otherwise, when nnpfc\_auxiliary\_inp\_idc is equal to 1 or 2, one luma matrix and one auxiliary input matrix are present, and the number of channels is 2. Otherwise, when nnpfc\_auxiliary\_inp\_idc is equal to 3, one luma matrix and two auxiliary input matrices are present, and the number of channels is 3. |
| 1 | If nnpfc\_auxiliary\_inp\_idc is equal to 0, two chroma matrices are present in the input tensor, and the number of channels is 2. Otherwise, when nnpfc\_auxiliary\_inp\_idc is equal to 1 or 2, two chroma matrices and one auxiliary input matrix are present, and the number of channels is 3. Otherwise, when nnpfc\_auxiliary\_inp\_idc is equal to 3, two chroma matrices and two auxiliary input matrices are present, and the number of channels is 4. |
| 2 | If nnpfc\_auxiliary\_inp\_idc is equal to 0, one luma and two chroma matrices are present in the input tensor, and the number of channels is 3. Otherwise, when nnpfc\_auxiliary\_inp\_idc is equal to 1 or 2, one luma matrix, two chroma matrices and one auxiliary input matrix are present, and the number of channels is 4. Otherwise, when nnpfc\_auxiliary\_inp\_idc is equal to 3, one luma matrix, two chroma matrices and two auxiliary input matrices are present, and the number of channels is 5. |
| 3 | If nnpfc\_auxiliary\_inp\_idc is equal to 0, four luma matrices and two chroma matrices are present in the input tensor, and the number of channels is 6. Otherwise, when nnpfc\_auxiliary\_inp\_idc is equal to 1 or 2, four luma matrices, two chroma matrices, and one auxiliary input matrix are present in the input tensor, and the number of channels is 7. Otherwise, when nnpfc\_auxiliary\_inp\_idc is equal to 3, four luma matrices, two chroma matrices, and two auxiliary input matrices are present in the input tensor, and the number of channels is 8. The luma channels are derived in an interleaved manner as illustrated in Figure 12. This nnpfc\_inp\_order\_idc can only be used when the input chroma format is 4:2:0. |
| 4..255 | Reserved |

A black background with a black square

Description automatically generated

**Figure 12 – Illustration of deriving the four luma channels (right) from the luma component (left) when nnpfc\_inp\_order\_idc is equal to 3**

**nnpfc\_inp\_tensor\_luma\_bitdepth\_minus8** plus 8 specifies the bit depth of luma sample values in the input integer tensor. The value of inpTensorBitDepthY is derived as follows:

inpTensorBitDepthY = nnpfc\_inp\_tensor\_luma\_bitdepth\_minus8 + 8(84)

It is a requirement of bitstream conformance that the value of nnpfc\_inp\_tensor\_luma\_bitdepth\_minus8 shall be in the range of 0 to 24, inclusive.

**nnpfc\_inp\_tensor\_chroma\_bitdepth\_minus8** plus 8 specifies the bit depth of chroma sample values in the input integer tensor. The value of inpTensorBitDepthC is derived as follows:

inpTensorBitDepthC = nnpfc\_inp\_tensor\_chroma\_bitdepth\_minus8 + 8(85)

It is a requirement of bitstream conformance that the value of nnpfc\_inp\_tensor\_chroma\_bitdepth\_minus8 shall be in the range of 0 to 24, inclusive.

**nnpfc\_out\_format\_idc** equal to 0 indicates that the sample values output by the NNPF are real numbers where the value range of 0 to 1, inclusive, maps linearly to the unsigned integer value range of 0 to ( 1  <<  bitDepth ) − 1, inclusive, for any desired bit depth bitDepth for subsequent post-processing or displaying.

nnpfc\_out\_format\_idc equal to 1 indicates that the luma sample values output by the NNPF are unsigned integer numbers in the range of 0 to ( 1  <<  outTensorBitDepthY ) − 1, inclusive, and the chroma sample values output by the NNPF are unsigned integer numbers in the range of 0 to ( 1  <<  outTensorBitDepthC ) − 1, inclusive.

The value of nnpfc\_out\_format\_idc shall be in the range of 0 to 255, inclusive. Values of 2 to 255, inclusive, for nnpfc\_out\_format\_idc are reserved for future specification by ITU-T | ISO/IEC and shall not be present in bitstreams conforming to this version of this Specification. Decoders conforming to this version of this Specification shall ignore NNPFC SEI messages with nnpfc\_out\_format\_idc in the range of 2 to 255, inclusive.

**nnpfc\_out\_order\_idc** indicates the output order of samples resulting from the NNPF.

The value of nnpfc\_out\_order\_idc shall be in the range of 0 to 255, inclusive. Values of 4 to 255, inclusive, for nnpfc\_out\_order\_idc are reserved for future use by ITU-T | ISO/IEC and shall not be present in bitstreams conforming to this version of this Specification. Decoders conforming to this version of this Specification shall ignore NNPFC SEI messages with nnpfc\_out\_order\_idc in the range of 4 to 255, inclusive.

When ChromaUpsamplingFlag is equal to 1, nnpfc\_out\_order\_idc shall not be equal to 0 or 3.

When ColourizationFlag is equal to 1, nnpfc\_out\_order\_idc shall not be equal to 0.

Table 22 contains an informative description of nnpfc\_out\_order\_idc values.

**Table 22 – Description of nnpfc\_out\_order\_idc values**

|  |  |
| --- | --- |
| **nnpfc\_out\_ order\_idc** | **Description** |
| 0 | Only the luma matrix is present in the output tensor, thus the number of channels is 1. |
| 1 | Only the chroma matrices are present in the output tensor, thus the number of channels is 2. |
| 2 | The luma and chroma matrices are present in the output tensor, thus the number of channels is 3. |
| 3 | Four luma matrices and two chroma matrices are present in the output tensor, thus the number of channels is 6. This nnpfc\_out\_order\_idc can only be used when the output chroma format is 4:2:0. |
| 4..255 | Reserved |

**nnpfc\_out\_tensor\_luma\_bitdepth\_minus8** plus 8 specifies the bit depth of luma sample values in the output integer tensor. The value of nnpfc\_out\_tensor\_luma\_bitdepth\_minus8 shall be in the range of 0 to 24, inclusive. The value of outTensorBitDepthY is derived as follows:

outTensorBitDepthY = nnpfc\_out\_tensor\_luma\_bitdepth\_minus8 + 8(86)

**nnpfc\_out\_tensor\_chroma\_bitdepth\_minus8** plus 8 specifies the bit depth of chroma sample values in the output integer tensor. The value of nnpfc\_out\_tensor\_chroma\_bitdepth\_minus8 shall be in the range of 0 to 24, inclusive. The value of outTensorBitDepthC is derived as follows:

outTensorBitDepthC = nnpfc\_out\_tensor\_chroma\_bitdepth\_minus8 + 8(87)

When BitDepthUpsamplingFlag is equal to 1, the value of nnpfc\_out\_format\_idc shall be equal to 1 and at least one of the following conditions shall be true:

– nnpfc\_out\_tensor\_luma\_bitdepth\_minus8 is present and outTensorBitDepthY is greater than BitDepthY.

– nnpfc\_out\_tensor\_chroma\_bitdepth\_minus8 is present and outTensorBitDepthC is greater than BitDepthC.

When nnpfc\_inp\_tensor\_luma\_bitdepth\_minus8, nnpfc\_inp\_tensor\_chroma\_bitdepth\_minus8, nnpfc\_out\_tensor\_luma\_bitdepth\_minus8, and nnpfc\_out\_tensor\_chroma\_bitdepth\_minus8 are present and outTensorBitDepthY is greater than inpTensorBitDepthY, outTensorBitDepthC shall not be less than inpTensorBitDepthC. When nnpfc\_inp\_tensor\_luma\_bitdepth\_minus8, nnpfc\_inp\_tensor\_chroma\_bitdepth\_minus8, nnpfc\_out\_tensor\_luma\_bitdepth\_minus8, and nnpfc\_out\_tensor\_chroma\_bitdepth\_minus8 are present and outTensorBitDepthC is greater than inpTensorBitDepthC, outTensorBitDepthY shall not be less than inpTensorBitDepthY.

**nnpfc\_separate\_colour\_description\_present\_flag** equal to 1 indicates that a distinct combination of colour primaries, transfer characteristics, matrix coefficients, and scaling and offset values applied in association with the matrix coefficients for the picture resulting from the NNPF is specified in the SEI message syntax structure. nnpfc\_separate\_colour\_description\_present\_flag equal to 0 indicates that the combination of colour primaries, transfer characteristics, matrix coefficients, and scaling and offset values applied in association with the matrix coefficients for the picture resulting from the NNPF is the same as implied by the VUI parameters vui\_colour\_primaries, vui\_tranfer\_characteristics, vui\_matrix\_coeffs, and vui\_full\_range\_flag that are indicated or inferred for the CLVS.

**nnpfc\_colour\_primaries** has the same semantics as specified in clause 7.3 for the vui\_colour\_primaries syntax element, except as follows:

– nnpfc\_colour\_primaries specifies the colour primaries of the picture resulting from applying the NNPF specified in the SEI message, rather than the colour primaries used for the CLVS.

– When nnpfc\_colour\_primaries is not present in the NNPFC SEI message, the value of nnpfc\_colour\_primaries is inferred to be equal to vui\_colour\_primaries.

**nnpfc\_transfer\_characteristics** has the same semantics as specified in clause 7.3 for the vui\_transfer\_characteristics syntax element, except as follows:

– nnpfc\_transfer\_characteristics specifies the transfer characteristics of the picture resulting from applying the NNPF specified in the SEI message, rather than the transfer characteristics used for the CLVS.

– When nnpfc\_transfer\_characteristics is not present in the NNPFC SEI message, the value of nnpfc\_transfer\_characteristics is inferred to be equal to vui\_transfer\_characteristics.

**nnpfc\_matrix\_coeffs** describes the equations used in deriving luma and chroma signals from the green, blue, and red, or Y, Z, and X primaries. Its semantics apply to the pictures resulting from applying the NNPF specified in this SEI message and are as specified for MatrixCoefficients in Rec. ITU-T H.273 | ISO/IEC 23091-2 with BitDepthY and BitDepthC being equal to outTensorBitDepthY and outTensorBitDepthC, respectively.

When nnpfc\_matrix\_coeffs is not present in the NNPFC SEI message, the value of nnpfc\_matrix\_coeffs is inferred to be equal to vui\_matrix\_coeffs.

nnpfc\_matrix\_coeffs shall not be equal to 0 unless both of the following conditions are true:

– nnpfc\_out\_tensor\_chroma\_bitdepth\_minus8 is equal to nnpfc\_out\_tensor\_luma\_bitdepth\_minus8.

– nnpfc\_out\_order\_idc is equal to 2, outSubHeightC is equal to 1, and outSubWidthC is equal to 1.

nnpfc\_matrix\_coeffs shall not be equal to 8 unless one of the following conditions is true:

– nnpfc\_out\_tensor\_chroma\_bitdepth\_minus8 is equal to nnpfc\_out\_tensor\_luma\_bitdepth\_minus8.

– nnpfc\_out\_tensor\_chroma\_bitdepth\_minus8 is equal to nnpfc\_out\_tensor\_luma\_bitdepth\_minus8 + 1, nnpfc\_out\_order\_idc is equal to 2, outSubHeightC is equal to 1, and outSubWidthC is equal to 1.

**nnpfc\_full\_range\_flag** indicates the scaling and offset values applied in association with the matrix coefficients as specified by nnpfc\_matrix\_coeffs. Its semantics are as specified for the VideoFullRangeFlag parameter in Rec. ITU-T H.273 | ISO/IEC 23091-2. When not present, the value of nnpfc\_full\_range\_flag is inferred to be equal to 0.

**nnpfc\_chroma\_loc\_info\_present\_flag** equal to 1 indicates the presence of the nnpfc\_chroma\_sample\_loc\_type\_frame syntax element in the NNPFC SEI message. nnpfc\_chroma\_loc\_info\_present\_flag equal to 0 indicates the absence of the nnpfc\_chroma\_sample\_loc\_type\_frame syntax element in the NNPFC SEI message. When nnpfc\_chroma\_loc\_info\_present\_flag is not present, its value is inferred to be equal to 0. When ColourizationFlag is equal to 0 or nnpfc\_out\_colour\_format\_idc is not equal to 1, the value of nnpfc\_chroma\_loc\_info\_present\_flag shall be equal to 0.

**nnpfc\_chroma\_sample\_loc\_type\_frame**, when not equal to 6 and nnpfc\_out\_colour\_format\_idc is equal to 1, specifies the location of chroma samples of the output pictures, as shown in Figure 1. nnpfc\_chroma\_sample\_loc\_type\_frame equal to 6 and nnpfc\_out\_colour\_format\_idc equal to 1 indicates that the location of the chroma samples is unknown or unspecified or specified by other means not specified in this Specification. The value of nnpfc\_chroma\_sample\_loc\_type\_frame shall be in the range of 0 to 6, inclusive.

**nnpfc\_overlap** indicates the overlapping horizontal and vertical sample counts of adjacent input tensors of the NNPF. The value of nnpfc\_overlap shall be in the range of 0 to 16 383, inclusive. When SpatialExtrapolationFlag is equal to 1, nnpfc\_overlap is inferred to be equal to 0.

**nnpfc\_constant\_patch\_size\_flag** equal to 1 indicates that the NNPF accepts exactly the patch size indicated by nnpfc\_patch\_width\_minus1 and nnpfc\_patch\_height\_minus1 as input. nnpfc\_constant\_patch\_size\_flag equal to 0 indicates that the NNPF accepts as input any patch size with width inpPatchWidth and height inpPatchHeight such that the width of an extended patch (i.e., a patch plus the overlapping area), which is equal to inpPatchWidth + 2 \* nnpfc\_overlap, is a positive integer multiple of nnpfc\_extended\_patch\_width\_cd\_delta\_minus1 + 1 + 2 \* nnpfc\_overlap, and the height of the extended patch, which is equal to inpPatchHeight + 2 \* nnpfc\_overlap, is a positive integer multiple of nnpfc\_extended\_patch\_height\_cd\_delta\_minus1 + 1 + 2 \* nnpfc\_overlap. When SpatialExtrapolationFlag is equal to 1, nnpfc\_constant\_patch\_size\_flag is inferred to be equal to 1.

**nnpfc\_patch\_width\_minus1** plus 1, when nnpfc\_constant\_patch\_size\_flag equal to 1, indicates the horizontal sample counts of the patch size required for the input to the NNPF. The value of nnpfc\_patch\_width\_minus1 shall be in the range of 0 to Min( 32 766, CroppedWidth − 1 ), inclusive.

**nnpfc\_patch\_height\_minus1** plus 1, when nnpfc\_constant\_patch\_size\_flag equal to 1, indicates the vertical sample counts of the patch size required for the input to the NNPF. The value of nnpfc\_patch\_height\_minus1 shall be in the range of 0 to Min( 32 766, CroppedHeight − 1 ), inclusive.

**nnpfc\_extended\_patch\_width\_cd\_delta\_minus1** plus 1 plus 2 \* nnpfc\_overlap, when nnpfc\_constant\_patch\_size\_flag equal to 0, indicates a common divisor of all allowed values of the width of an extended patch required for the input to the NNPF. The value of nnpfc\_extended\_patch\_width\_cd\_delta\_minus1 shall be in the range of 0 to Min( 32 766, CroppedWidth − 1 ), inclusive.

**nnpfc\_****extended\_patch\_height\_cd\_delta\_minus1** plus 1 plus 2 \* nnpfc\_overlap, when nnpfc\_constant\_patch\_size\_flag equal to 0, indicates a common divisor of all allowed values of the height of an extended patch required for the input to the NNPF. The value of nnpfc\_extended\_patch\_height\_cd\_delta\_minus1 shall be in the range of 0 to Min( 32 766, CroppedHeight − 1 ), inclusive.

Let the variables inpPatchWidth and inpPatchHeight be the patch size width and the patch size height, respectively.

If nnpfc\_constant\_patch\_size\_flag is equal to 0, the following applies:

– The values of inpPatchWidth and inpPatchHeight are either provided by external means not specified in this Specification or set by the post-processor itself.

– The value of inpPatchWidth + 2 \* nnpfc\_overlap shall be a positive integer multiple of nnpfc\_extended\_patch\_width\_cd\_delta\_minus1 + 1 + 2 \* nnpfc\_overlap and inpPatchWidth shall be less than or equal to CroppedWidth. The value of inpPatchHeight + 2 \* nnpfc\_overlap shall be a positive integer multiple of nnpfc\_extended\_patch\_height\_cd\_delta\_minus1 + 1 + 2 \* nnpfc\_overlap and inpPatchHeight shall be less than or equal to CroppedHeight.

Otherwise (nnpfc\_constant\_patch\_size\_flag is equal to 1), the value of inpPatchWidth is set equal to nnpfc\_patch\_width\_minus1 + 1 and the value of inpPatchHeight is set equal to nnpfc\_patch\_height\_minus1 + 1.

The variables outPatchWidth, outPatchHeight, horCScaling, verCScaling, outPatchCWidth, and outPatchCHeight are derived as follows:

outPatchWidth = ( nnpfcOutputPicWidth \* inpPatchWidth ) / CroppedWidth (88)

outPatchHeight = ( nnpfcOutputPicHeight \* inpPatchHeight ) / CroppedHeight (89)

horCScaling = SubWidthC / outSubWidthC (90)

verCScaling = SubHeightC / outSubHeightC (91)

outPatchCWidth = outPatchWidth \* horCScaling (92)

outPatchCHeight = outPatchHeight \* verCScaling (93)

It is a requirement of bitstream conformance that when SpatialExtrapolationFlag is equal to 0, outPatchWidth \* CroppedWidth shall be equal to nnpfcOutputPicWidth \* inpPatchWidth and outPatchHeight \* CroppedHeight shall be equal to nnpfcOutputPicHeight \* inpPatchHeight.

It is a requirement of bitstream conformance that when SpatialExtrapolationFlag is equal to 1, outPatchWidth \* CroppedWidth shall be equal to nnpfcOutputPicWidth1 \* inpPatchWidth and outPatchHeight \* CroppedHeight shall be equal to nnpfcOutputPicHeight1 \* inpPatchHeight.

**nnpfc\_padding\_type** indicates the process of padding when referencing sample locations outside the boundaries of the input picture as described in Table 23. The value of nnpfc\_padding\_type shall be in the range of 0 to 15, inclusive. Values of 5 to 15, inclusive, for nnpfc\_padding\_type are reserved for future use by ITU-T | ISO/IEC and shall not be present in bitstreams conforming to this version of this Specification. Decoders conforming to this version of this Specification shall ignore NNPFC SEI messages with nnpfc\_padding\_type in the range of 5 to 15, inclusive.

**Table 23 – Informative description of nnpfc\_padding\_type values**

|  |  |
| --- | --- |
| **nnpfc\_padding\_type** | **Description** |
| 0 | Zero padding |
| 1 | Replication padding |
| 2 | Reflection padding |
| 3 | Wrap-around padding |
| 4 | Fixed padding |
| 5..15 | reserved |

**nnpfc\_luma\_padding\_val** indicates the luma value to be used for padding when nnpfc\_padding\_type is equal to 4. The value of nnpfc\_luma\_padding\_val shall be in the range of 0 to ( 1  <<  BitDepthY ) − 1, inclusive.

**nnpfc\_cb\_padding\_val** indicates the Cb value to be used for padding when nnpfc\_padding\_type is equal to 4. The value of nnpfc\_cb\_padding\_val shall be in the range of 0 to ( 1  <<  BitDepthC ) − 1, inclusive.

**nnpfc\_cr\_padding\_val** indicates the Cr value to be used for padding when nnpfc\_padding\_type is equal to 4. The value of nnpfc\_cr\_padding\_val shall be in the range of 0 to ( 1  <<  BitDepthC ) − 1, inclusive.

The function InpSampleVal( y, x, picHeight, picWidth, croppedPic, cIdx ) with inputs being a vertical sample location y, a horizontal sample location x, a picture height picHeight, a picture width picWidth, sample array croppedPic, and component index cIdx (equal to 0 for luma, 1 for Cb, and 2 for Cr) returns the value of sampleVal derived as follows:

NOTE 7 – For the inputs to the function InpSampleVal( ), the vertical location is listed before the horizontal location for compatibility with input tensor conventions of some inference engines.

if( nnpfc\_padding\_type = = 0 )  
 if( y < 0 | | x < 0 | | y >= picHeight | | x >= picWidth )  
 sampleVal = 0  
 else  
 sampleVal = croppedPic[ x ][ y ] (94)  
else if( nnpfc\_padding\_type = = 1 )  
 sampleVal = croppedPic[ Clip3( 0, picWidth − 1, x ) ][ Clip3( 0, picHeight − 1, y ) ]  
else if( nnpfc\_padding\_type = = 2 )   
 sampleVal = croppedPic[ Reflect( picWidth − 1, x ) ][ Reflect( picHeight − 1, y ) ]  
else if( nnpfc\_padding\_type = = 3 )   
 if( y >= 0 && y < picHeight )  
 sampleVal = croppedPic[ Wrap( picWidth − 1, x ) ][ y ]   
else if( nnpfc\_padding\_type = = 4 )   
 if( y < 0 | | x < 0 | | y >= picHeight | | x >= picWidth )  
 sampleVal = ( cIdx = = 0 ? nnpfc\_luma\_padding\_val :   
 ( cIdx = = 1 ? nnpfc\_cb\_padding\_val : nnpfc\_cr\_padding\_val ) )  
 else  
 sampleVal = croppedPic[ x ][ y ]

When nnpfc\_auxiliary\_inp\_idc is equal to 1, the variable strengthControlScaledVal is derived as follows:

for( i = 0; i < numInputPics; i++ )  
 if( nnpfc\_inp\_format\_idc = = 1 ) (95)  
 if( nnpfc\_inp\_order\_idc = = 0 | | nnpfc\_inp\_order\_idc = = 2 | |  
 nnpfc\_inp\_order\_idc = = 3 )  
 strengthControlScaledVal[ i ] =   
 Floor ( StrengthControlVal[ i ] \* ( ( 1  <<  inpTensorBitDepthY ) − 1 ) )  
 else if( nnpfc\_inp\_order\_idc = = 1 )  
 strengthControlScaledVal[ i ] =   
 Floor ( StrengthControlVal[ i ] \* ( ( 1  <<  inpTensorBitDepthC ) − 1 ) )  
 else  
 strengthControlScaledVal[ i ] = StrengthControlVal[ i ]

A patch is a rectangular array of samples from a component (e.g., a luma or chroma component) of a picture.

The process DeriveInputTensors( ), for deriving the input tensor inputTensor for a given vertical sample coordinate cTop and a horizontal sample coordinate cLeft specifying the top-left sample location for the patch of samples included in the input tensor, is specified as follows:

for( i = 0; i < numInputPics; i++ ) {  
 if( nnpfc\_inp\_order\_idc = = 0 )  
 for( yP = −nnpfc\_overlap; yP < inpPatchHeight + nnpfc\_overlap; yP++)  
 for( xP = −nnpfc\_overlap; xP < inpPatchWidth + nnpfc\_overlap; xP++ ) {  
 inpVal = InpY( InpSampleVal( cTop + yP, cLeft + xP, CroppedHeight,  
 CroppedWidth, CroppedYPic[ i ], 0 ) )   
 yPovlp = yP + nnpfc\_overlap  
 xPovlp = xP + nnpfc\_overlap  
 if( !nnpfc\_component\_last\_flag )  
 inputTensor[ 0 ][ i ][ 0 ][ yPovlp ][ xPovlp ] = inpVal  
 else  
 inputTensor[ 0 ][ i ][ yPovlp ][ xPovlp ][ 0 ] = inpVal  
 if( nnpfc\_auxiliary\_inp\_idc = = 1 | | nnpfc\_auxiliary\_inp\_idc = = 3)  
 if( !nnpfc\_component\_last\_flag )  
 inputTensor[ 0 ][ i ][ 1 ][ yPovlp ][ xPovlp ] = strengthControlScaledVal[ i ]  
 else  
 inputTensor[ 0 ][ i ][ yPovlp ][ xPovlp ][ 1 ] = strengthControlScaledVal[ i ]   
 if( nnpfc\_auxiliary\_inp\_idc = = 2 | | nnpfc\_auxiliary\_inp\_idc = = 3) {  
 promptCharVal = utf8ToUInt( nnpfcPrompt )  
  if( !nnpfc\_component\_last\_flag )  
 inputTensor[ 0 ][ i ][ nnpfc\_auxiliary\_inp\_idc − 1 ][ yPovlp ][ xPovlp ] = promptCharVal  
 else  
 inputTensor[ 0 ][ i ][ yPovlp ][ xPovlp ][ nnpfc\_auxiliary\_inp\_idc − 1 ] = promptCharVal  
 }  
 }  
 else if( nnpfc\_inp\_order\_idc = = 1 ) (96)  
 for( yP = −nnpfc\_overlap; yP < inpPatchHeight + nnpfc\_overlap; yP++)  
 for( xP = −nnpfc\_overlap; xP < inpPatchWidth + nnpfc\_overlap; xP++ ) {  
 inpCbVal = InpC( InpSampleVal( cTop + yP, cLeft + xP, CroppedHeight / SubHeightC,  
 CroppedWidth / SubWidthC, CroppedCbPic[ i ], 1 ) )  
 inpCrVal = InpC( InpSampleVal( cTop + yP, cLeft + xP, CroppedHeight / SubHeightC,  
 CroppedWidth / SubWidthC, CroppedCrPic[ i ], 2 ) ) yPovlp = yP + nnpfc\_overlap  
 xPovlp = xP + nnpfc\_overlap  
 if( !nnpfc\_component\_last\_flag ) {  
 inputTensor[ 0 ][ i ][ 0 ][ yPovlp ][ xPovlp ] = inpCbVal  
 inputTensor[ 0 ][ i ][ 1 ][ yPovlp ][ xPovlp ] = inpCrVal  
 } else {  
 inputTensor[ 0 ][ i ][ yPovlp ][ xPovlp ][ 0 ] = inpCbVal  
 inputTensor[ 0 ][ i ][ yPovlp ][ xPovlp ][ 1 ] = inpCrVal  
 }  
 if( nnpfc\_auxiliary\_inp\_idc = = 1 | | nnpfc\_auxiliary\_inp\_idc = = 3)  
 if( !nnpfc\_component\_last\_flag )  
 inputTensor[ 0 ][ i ][ 2 ][ yPovlp ][ xPovlp ] = strengthControlScaledVal[ i ]  
 else  
 inputTensor[ 0 ][ i ][ yPovlp ][ xPovlp ][ 2 ] = strengthControlScaledVal[ i ]   
 if( nnpfc\_auxiliary\_inp\_idc = = 2 | | nnpfc\_auxiliary\_inp\_idc = = 3) {  
 promptCharVal = utf8ToUInt( nnpfcPrompt )   
 if( !nnpfc\_component\_last\_flag )  
 inputTensor[ 0 ][ i ][ nnpfc\_auxiliary\_inp\_idc ][ yPovlp ][ xPovlp ] = promptCharVal  
 else  
 inputTensor[ 0 ][ i ][ yPovlp ][ xPovlp ][ nnpfc\_auxiliary\_inp\_idc ] = promptCharVal  
 }  
 }  
 else if( nnpfc\_inp\_order\_idc = = 2 )  
 for( yP = −nnpfc\_overlap; yP < inpPatchHeight + nnpfc\_overlap; yP++)  
 for( xP = −nnpfc\_overlap; xP < inpPatchWidth + nnpfc\_overlap; xP++ ) {  
 yY = cTop + yP  
 xY = cLeft + xP  
 yC = yY / SubHeightC  
 xC = xY / SubWidthC  
 inpYVal = InpY( InpSampleVal( yY, xY, CroppedHeight,  
 CroppedWidth, CroppedYPic[ i ], 0 ) )  
 inpCbVal = InpC( InpSampleVal( yC, xC, CroppedHeight / SubHeightC,  
 CroppedWidth / SubWidthC, CroppedCbPic[ i ], 1 ) )  
 inpCrVal = InpC( InpSampleVal( yC, xC, CroppedHeight / SubHeightC,  
 CroppedWidth / SubWidthC, CroppedCrPic[ i ], 2 ) ) yPovlp = yP + nnpfc\_overlap  
 xPovlp = xP + nnpfc\_overlap  
 if( !nnpfc\_component\_last\_flag ) {  
 inputTensor[ 0 ][ i ][ 0 ][ yPovlp ][ xPovlp ] = inpYVal  
 inputTensor[ 0 ][ i ][ 1 ][ yPovlp ][ xPovlp ] = inpCbVal  
 inputTensor[ 0 ][ i ][ 2 ][ yPovlp ][ xPovlp ] = inpCrVal  
 } else {  
 inputTensor[ 0 ][ i ][ yPovlp ][ xPovlp ][ 0 ] = inpYVal  
 inputTensor[ 0 ][ i ][ yPovlp ][ xPovlp ][ 1 ] = inpCbVal  
 inputTensor[ 0 ][ i ][ yPovlp ][ xPovlp ][ 2 ] = inpCrVal  
 }  
 if( nnpfc\_auxiliary\_inp\_idc = = 1 | | nnpfc\_auxiliary\_inp\_idc = = 3)  
 if( !nnpfc\_component\_last\_flag )  
 inputTensor[ 0 ][ i ][ 3 ][ yPovlp ][ xPovlp ] = strengthControlScaledVal[ i ]  
 else  
 inputTensor[ 0 ][ i ][ yPovlp ][ xPovlp ][ 3 ] = strengthControlScaledVal[ i ]   
 if( nnpfc\_auxiliary\_inp\_idc = = 2 | | nnpfc\_auxiliary\_inp\_idc = = 3) {  
 promptCharVal = utf8ToUInt( nnpfcPrompt )   
  if( !nnpfc\_component\_last\_flag )  
 inputTensor[ 0 ][ i ][ nnpfc\_auxiliary\_inp\_idc + 1 ][ yPovlp ][ xPovlp ] = promptCharVal  
 else  
 inputTensor[ 0 ][ i ][ yPovlp ][ xPovlp ][ nnpfc\_auxiliary\_inp\_idc + 1 ] = promptCharVal  
 }  
 }  
 else if( nnpfc\_inp\_order\_idc = = 3 )  
 for( yP = −nnpfc\_overlap; yP < inpPatchHeight + nnpfc\_overlap; yP++)  
 for( xP = −nnpfc\_overlap; xP < inpPatchWidth + nnpfc\_overlap; xP++ ) {  
 yTL = cTop + yP \* 2  
 xTL = cLeft + xP \* 2  
 yBR = yTL + 1  
 xBR = xTL + 1  
 yC = cTop / 2 + yP  
 xC = cLeft / 2 + xP  
 inpTLVal = InpY( InpSampleVal( yTL, xTL, CroppedHeight,  
 CroppedWidth, CroppedYPic[ i ], 0 ) )  
 inpTRVal = InpY( InpSampleVal( yTL, xBR, CroppedHeight,  
 CroppedWidth, CroppedYPic[ i ], 0 ) )  
 inpBLVal = InpY( InpSampleVal( yBR, xTL, CroppedHeight,  
 CroppedWidth, CroppedYPic[ i ], 0 ) )  
 inpBRVal = InpY( InpSampleVal( yBR, xBR, CroppedHeight,  
 CroppedWidth, CroppedYPic[ i ], 0 ) )  
 inpCbVal = InpC( InpSampleVal( yC, xC, CroppedHeight / 2,  
 CroppedWidth / 2, CroppedCbPic[ i ], 1 ) )  
 inpCrVal = InpC( InpSampleVal( yC, xC, CroppedHeight / 2,  
 CroppedWidth / 2, CroppedCrPic[ i ], 2 ) )   
 yPovlp = yP + nnpfc\_overlap  
 xPovlp = xP + nnpfc\_overlap  
 if( !nnpfc\_component\_last\_flag ) {  
 inputTensor[ 0 ][ i ][ 0 ][ yPovlp ][ xPovlp ] = inpTLVal  
 inputTensor[ 0 ][ i ][ 1 ][ yPovlp ][ xPovlp ] = inpTRVal  
 inputTensor[ 0 ][ i ][ 2 ][ yPovlp ][ xPovlp ] = inpBLVal  
 inputTensor[ 0 ][ i ][ 3 ][ yPovlp ][ xPovlp ] = inpBRVal  
 inputTensor[ 0 ][ i ][ 4 ][ yPovlp ][ xPovlp ] = inpCbVal  
 inputTensor[ 0 ][ i ][ 5 ][ yPovlp ][ xPovlp ] = inpCrVal  
 } else {  
 inputTensor[ 0 ][ i ][ yPovlp ][ xPovlp ][ 0 ] = inpTLVal  
 inputTensor[ 0 ][ i ][ yPovlp ][ xPovlp ][ 1 ] = inpTRVal  
 inputTensor[ 0 ][ i ][ yPovlp ][ xPovlp ][ 2 ] = inpBLVal  
 inputTensor[ 0 ][ i ][ yPovlp ][ xPovlp ][ 3 ] = inpBRVal  
 inputTensor[ 0 ][ i ][ yPovlp ][ xPovlp ][ 4 ] = inpCbVal  
 inputTensor[ 0 ][ i ][ yPovlp ][ xPovlp ][ 5 ] = inpCrVal  
 }  
 if( nnpfc\_auxiliary\_inp\_idc = = 1 | | nnpfc\_auxiliary\_inp\_idc = = 3)  
 if( !nnpfc\_component\_last\_flag )  
 inputTensor[ 0 ][ i ][ 6 ][ yPovlp ][ xPovlp ] = strengthControlScaledVal[ i ]  
 else  
 inputTensor[ 0 ][ i ][ yPovlp ][ xPovlp ][ 6 ] = strengthControlScaledVal[ i ]   
 if( nnpfc\_auxiliary\_inp\_idc = = 2 | | nnpfc\_auxiliary\_inp\_idc = = 3) {  
 promptCharVal = utf8ToUInt( nnpfcPrompt )  
  if( !nnpfc\_component\_last\_flag )  
 inputTensor[ 0 ][ i ][ nnpfc\_auxiliary\_inp\_idc + 4 ][ yPovlp ][ xPovlp ] = promptCharVal  
 else  
 inputTensor[ 0 ][ i ][ yPovlp ][ xPovlp ][ nnpfc\_auxiliary\_inp\_idc + 4 ] = promptCharVal  
 }  
 }  
}

utf8ToUInt( x ) {  
 result = 0   
 len = 0   
 /\* Check end of text prompt string \*/  
 if( x = = null )  
 return 0   
 /\* Determine the number of bytes in the UTF-8 character \*/  
 if( (x[ 0 ] & 0x80 ) = = 0 )  
 len = 1 /\* 1-byte character \*/  
 else if( (x[ 0 ] & 0xE0 ) = = 0xC0 )  
 len = 2 /\* 2-byte character \*/  
 else if( (x[ 0 ] & 0xF0 ) = = 0xE0 )  
 len = 3 /\* 3-byte character \*/  
 else if( (x[ 0 ] & 0xF8 ) = = 0xF0 )  
 len = 4 /\* 4-byte character \*/  
 else  
 len = 0 /\* Invalid UTF-8 character; this case shall not occur in bitstreams. \*/  
 for( i = 0; i < len; i++ ) /\* Construct an integer from the bytes \*/  
 result = ( result << 8 ) | x[ i ]  
 x = x + len /\* Modifies the input variable, which is a syntax element \*/  
 return result  
}

The process StoreOutputTensors( ), for deriving sample values in the sample arrays FilteredYPic, FilteredCbPic, and FilteredCrPic, for the NNPF-generated pictures, from the output tensor outputTensor for a given vertical sample coordinate cTop and a horizontal sample coordinate cLeft specifying the top-left sample location for the patch of samples included in the input tensor, is specified as follows:

for( i = 0; i < numPicsInOutputTensor; i++ ) {  
 if( nnpfc\_out\_order\_idc = = 0 )  
 for( yP = 0; yP < outPatchHeight; yP++ )  
 for( xP = 0; xP < outPatchWidth; xP++ ) {  
 yY = cTop \* outPatchHeight / inpPatchHeight + yP  
 xY = cLeft \* outPatchWidth / inpPatchWidth + xP  
 if( yY < nnpfcOutputPicHeight && xY < nnpfcOutputPicWidth )  
 if( !nnpfc\_component\_last\_flag )  
 FilteredYPic[ i ][ xY ][yY ] = outputTensor[ 0 ][ i ][ 0 ][ yP ][ xP ]  
 else  
 FilteredYPic[ i ][ xY ][ yY ] = outputTensor[ 0 ][ i ][ yP ][ xP ][ 0 ] }  
 else if( nnpfc\_out\_order\_idc = = 1 ) (97)  
 for( yP = 0; yP < outPatchCHeight; yP++ )  
 for( xP = 0; xP < outPatchCWidth; xP++ ) {  
 xSrc = cLeft \* horCScaling + xP  
 ySrc = cTop \* verCScaling + yP  
 if( ySrc < nnpfcOutputPicHeight / outSubHeightC &&  
 xSrc < nnpfcOutputPicWidth / outSubWidthC )  
 if( !nnpfc\_component\_last\_flag ) {  
 FilteredCbPic[ i ][ xSrc ][ ySrc ] = outputTensor[ 0 ][ i ][ 0 ][ yP ][ xP ]  
 FilteredCrPic[ i ][ xSrc ][ ySrc ] = outputTensor[ 0 ][ i ][ 1 ][ yP ][ xP ]  
 } else {  
 FilteredCbPic[ i ][ xSrc ][ ySrc ] = outputTensor[ 0 ][ i ][ yP ][ xP ][ 0 ]  
 FilteredCrPic[ i ][ xSrc ][ ySrc ] = outputTensor[ 0 ][ i ][ yP ][ xP ][ 1 ]  
 }  
 }  
 else if( nnpfc\_out\_order\_idc = = 2 )  
 for( yP = 0; yP < outPatchHeight; yP++ )  
 for( xP = 0; xP < outPatchWidth; xP++ ) {  
 yY = cTop \* outPatchHeight / inpPatchHeight + yP  
 xY = cLeft \* outPatchWidth / inpPatchWidth + xP  
 yC = yY / outSubHeightC   
 xC = xY / outSubWidthC   
 yPc = ( yP / outSubHeightC ) \* outSubHeightC  
 xPc = ( xP / outSubWidthC ) \* outSubWidthC  
 if( yY < nnpfcOutputPicHeight && xY < nnpfcOutputPicWidth )  
 if( !nnpfc\_component\_last\_flag ) {  
 FilteredYPic[ i ][ xY ][ yY ] = outputTensor[ 0 ][ i ][ 0 ][ yP ][ xP ]  
 FilteredCbPic[ i ][ xC ][ yC ] = outputTensor[ 0 ][ i ][ 1 ][ yPc ][ xPc ]  
 FilteredCrPic[ i ][ xC ][ yC ] = outputTensor[ 0 ][ i ][ 2 ][ yPc ][ xPc ]  
 } else {  
 FilteredYPic[ i ][ xY ][ yY ] = outputTensor[ 0 ][ i ][ yP ][ xP ][ 0 ]  
 FilteredCbPic[ i ][ xC ][ yC ] = outputTensor[ 0 ][ i ][ yPc ][ xPc ][ 1 ]  
 FilteredCrPic[ i ][ xC ][ yC ] = outputTensor[ 0 ][ i ][ yPc ][ xPc ][ 2 ]  
 }  
 }  
 else if( nnpfc\_out\_order\_idc = = 3 )  
 for( yP = 0; yP < outPatchHeight; yP++ )  
 for( xP = 0; xP < outPatchWidth; xP++ ) {  
 ySrc = cTop / 2 \* outPatchHeight / inpPatchHeight + yP  
 xSrc = cLeft / 2 \* outPatchWidth / inpPatchWidth + xP  
 if( ySrc < nnpfcOutputPicHeight / 2 &&  
 xSrc < nnpfcOutputPicWidth / 2 )  
 if( !nnpfc\_component\_last\_flag ) {  
 FilteredYPic[ i ][ xSrc \* 2 ][ ySrc \* 2 ] = outputTensor[ 0 ][ i ][ 0 ][ yP ][ xP ]  
 FilteredYPic[ i ][ xSrc \* 2 + 1 ][ ySrc \* 2 ] = outputTensor[ 0 ][ i ][ 1 ][ yP ][ xP ]  
 FilteredYPic[ i ][ xSrc \* 2 ][ ySrc \* 2 + 1 ] = outputTensor[ 0 ][ i ][ 2 ][ yP ][ xP ]  
 FilteredYPic[ i ][ xSrc \* 2 + 1][ ySrc \* 2 + 1 ] = outputTensor[ 0 ][ i ][ 3 ][ yP ][ xP ]  
 FilteredCbPic[ i ][ xSrc ][ ySrc ] = outputTensor[ 0 ][ i ][ 4 ][ yP ][ xP ]  
 FilteredCrPic[ i ][ xSrc ][ ySrc ] = outputTensor[ 0 ][ i ][ 5 ][ yP ][ xP ]  
 } else {  
 FilteredYPic[ i ][ xSrc \* 2 ][ ySrc \* 2 ] = outputTensor[ 0 ][ i ][ yP ][ xP ][ 0 ]  
 FilteredYPic[ i ][ xSrc \* 2 + 1 ][ ySrc \* 2 ] = outputTensor[ 0 ][ i ][ yP ][ xP ][ 1 ]  
 FilteredYPic[ i ][ xSrc \* 2 ][ ySrc \* 2 + 1 ] = outputTensor[ 0 ][ i ][ yP ][ xP ][ 2 ]  
 FilteredYPic[ i ][ xSrc \* 2 + 1][ ySrc \* 2 + 1 ] = outputTensor[ 0 ][ i ][ yP ][ xP ][ 3 ]  
 FilteredCbPic[ i ][ xSrc ][ ySrc ] = outputTensor[ 0 ][ i ][ yP ][ xP ][ 4 ]  
 FilteredCrPic[ i ][ xSrc ][ ySrc ] = outputTensor[ 0 ][ i ][ yP ][ xP ][ 5 ]  
 }  
 }  
}

An NNPF PostProcessingFilter( ) is the target NNPF as derived in the semantics of the NNPFA SEI message. The following example process may be used, with the NNPF PostProcessingFilter( ), to generate, in a patch-wise manner, the filtered and/or interpolated picture(s), which contain Y, Cb, and Cr sample arrays FilteredYPic, FilteredCbPic, and FilteredCrPic, respectively, as indicated by nnpfc\_out\_order\_idc:

if( nnpfc\_inp\_order\_idc = = 0 | | nnpfc\_inp\_order\_idc = = 2 )  
 for( cTop = 0; cTop < CroppedHeight; cTop += inpPatchHeight )  
 for( cLeft = 0; cLeft < CroppedWidth; cLeft += inpPatchWidth ) {  
 if( SpatialExtrapolationFlag ) {  
  if( cTop = = 0 )  
 outPatchHeight += (outSubHeightC \* nnpfc\_spatial\_extrapolation\_top\_offset)  
 if( cLeft = = 0 )  
 outPatchWidth += (outSubWidthC \* nnpfc\_spatial\_extrapolation\_left\_offset)  
 if( cTop = = (CroppedHeight−inpPatchHeight) )  
 outPatchHeight += (outSubHeightC \* nnpfc\_spatial\_extrapolation\_bottom\_offset)  
 if( cLeft = = (CroppedWidth−inpPatchWidth) )  
 outPatchWidth += (outSubWidthC \* nnpfc\_spatial\_extrapolation\_right\_offset)   
 }  
 inputTensor = DeriveInputTensors( )  
 outputTensor = PostProcessingFilter( inputTensor )  
 StoreOutputTensors( outputTensor )  
 }  
else if( nnpfc\_inp\_order\_idc = = 1 )  
 for( cTop = 0; cTop < CroppedHeight / SubHeightC; cTop += inpPatchHeight )  
 for( cLeft = 0; cLeft < CroppedWidth / SubWidthC; cLeft += inpPatchWidth ) { (98)  
  if( SpatialExtrapolationFlag ) {  
 if( cTop = = 0 )  
 outPatchHeight += (outSubHeightC \* nnpfc\_spatial\_extrapolation\_top\_offset)  
 if( cLeft = = 0 )  
 outPatchWidth += (outSubWidthC \* nnpfc\_spatial\_extrapolation\_left\_offset)  
 if( cTop = = ((CroppedHeight/ SubHeightC)−inpPatchHeight) )  
 outPatchHeight += (outSubHeightC \* nnpfc\_spatial\_extrapolation\_bottom\_offset)  
 if( cLeft = = ((CroppedWidth/ SubWidthC)−inpPatchWidth) )  
 outPatchWidth += (outSubWidthC \* nnpfc\_spatial\_extrapolation\_right\_offset)  
 }  
 inputTensor = DeriveInputTensors( )  
 outputTensor = PostProcessingFilter( inputTensor )  
 StoreOutputTensors( outputTensor )  
 }  
else if( nnpfc\_inp\_order\_idc = = 3 )  
 for( cTop = 0; cTop < CroppedHeight; cTop += inpPatchHeight \* 2 )  
 for( cLeft = 0; cLeft < CroppedWidth; cLeft += inpPatchWidth \* 2 ) {  
 if( SpatialExtrapolationFlag ) {  
 if( cTop = = 0 )  
 outPatchHeight += (outSubHeightC \* nnpfc\_spatial\_extrapolation\_top\_offset)  
 if( cLeft = = 0 )  
 outPatchWidth += (outSubWidthC \* nnpfc\_spatial\_extrapolation\_left\_offset)  
 if( cTop = = (CroppedHeight−2\*inpPatchHeight) )  
 outPatchHeight += (outSubHeightC \* nnpfc\_spatial\_extrapolation\_bottom\_offset)  
 if( cLeft = = (CroppedWidth−2\*inpPatchWidth) )  
 outPatchWidth += (outSubWidthC \* nnpfc\_spatial\_extrapolation\_right\_offset)  
 }  
 inputTensor = DeriveInputTensors( )  
 outputTensor = PostProcessingFilter( inputTensor )  
 StoreOutputTensors( outputTensor )  
 }

NOTE 4 –  For some NNPF purposes (e.g. spatial extrapolation, temporal extrapolation, colourization) the independent processing of patches may result in inconsitency of filtered results across patches, especially at patch boundaries. Thus patch sizes should be set based on the NNPF’s ability to achieve consistency across patches.

[Ed. The equations above need to be checked carefully to make sure they handle the cropping use case correctly.]

It is a requirement of bitstream conformance that outPatchWidth is > 0. It is a requirement of bitstream conformance that outPatchHeight is > 0.

An NNPF-generated picture with index i contains sample arrays FilteredYPic[ i ], FilteredCbPic[ i ], and FilteredCrPic[ i ], when present, that are derived by Equation 98. An NNPF-generated picture does not include the overlap regions.

The NNPF process consists of the process defined by Equation 98 followed by outputting NNPF-generated pictures in their increasing index order, where all NNPF-generated pictures that were interpolated by the NNPF are output and those NNPF-generated pictures that correspond to any input pictures to the NNPF are output as specified in the semantics of the NNPFA SEI message.

**nnpfc\_complexity\_info\_present\_flag** equal to 1 specifies that one or more syntax elements that indicate the complexity of the NNPF associated with the nnpfc\_id are present. nnpfc\_complexity\_info\_present\_flag equal to 0 specifies that no syntax elements that indicates the complexity of the NNPF associated with the nnpfc\_id are present.

**nnpfc\_parameter\_type\_idc** equal to 0 indicates that the neural network uses only integer parameters. nnpfc\_parameter\_type\_idc equal to 1 indicates that the neural network may use floating point or integer parameters. nnpfc\_parameter\_type\_idc equal to 2 indicates that the neural network uses only binary parameters. nnpfc\_parameter\_type\_idc equal to 3 is reserved for future use by ITU-T | ISO/IEC and shall not be present in bitstreams conforming to this version of this Specification. Decoders conforming to this version of this Specification shall ignore NNPFC SEI messages with nnpfc\_parameter\_type\_idc equal to 3.

**nnpfc\_log2\_parameter\_bit\_length\_minus3** equal to 0, 1, 2, and 3 indicates that the neural network does not use parameters of bit length greater than 8, 16, 32, and 64, respectively. When nnpfc\_parameter\_type\_idc is present and nnpfc\_log2\_parameter\_bit\_length\_minus3 is not present, the neural network does not use parameters of bit length greater than 1.

**nnpfc\_num\_parameters\_idc** indicates the maximum number of neural network parameters for the NNPF in units of a power of 2 048. nnpfc\_num\_parameters\_idc equal to 0 indicates that the maximum number of neural network parameters is unknown. The value nnpfc\_num\_parameters\_idc shall be in the range of 0 to 52, inclusive. Values of nnpfc\_num\_parameters\_idc greater than 52 are reserved for future use by ITU-T | ISO/IEC and shall not be present in bitstreams conforming to this version of this Specification. Decoders conforming to this version of this Specification shall ignore NNPFC SEI messages with nnpfc\_num\_parameters\_idc greater than 52.

If the value of nnpfc\_num\_parameters\_idc is greater than zero, the variable maxNumParameters is derived as follows:

maxNumParameters = ( 2 048  <<  nnpfc\_num\_parameters\_idc ) − 1(99)

It is a requirement of bitstream conformance that the number of neural network parameters of the NNPF shall be less than or equal to maxNumParameters.

**nnpfc\_num\_kmac\_operations\_idc** greater than 0 indicates that the maximum number of multiply-accumulate operations per sample of the NNPF is less than or equal to nnpfc\_num\_kmac\_operations\_idc \* 1 000. nnpfc\_num\_kmac\_operations\_idc equal to 0 indicates that the maximum number of multiply-accumulate operations of the network is unknown. The value of nnpfc\_num\_kmac\_operations\_idc shall be in the range of 0 to 232 − 2, inclusive.

**nnpfc\_total\_kilobyte\_size** greater than 0 indicates a total size in kilobytes required to store the uncompressed parameters for the neural network. The total size in bits is a number equal to or greater than the sum of bits used to store each parameter. nnpfc\_total\_kilobyte\_size is the total size in bits divided by 8 000, rounded up. nnpfc\_total\_kilobyte\_size equal to 0 indicates that the total size required to store the parameters for the neural network is unknown. The value of nnpfc\_total\_kilobyte\_size shall be in the range of 0 to 232 − 2, inclusive.

**nnpfc\_num\_metadata\_extension\_bits** equal to 0 specifies that nnpfc\_application\_purpose\_tag\_uri\_present\_flag, nnpfc\_metadata\_alignment\_zero\_bit, nnpfc\_application\_purpose\_tag\_uri, nnpfc\_scan\_type\_idc, nnpfc\_for\_human\_viewing\_idc, nnpfc\_for\_machine\_analysis\_idc and nnpfc\_reserved\_metadata\_extension are not present. When nnpfc\_num\_metadata\_extension\_bits is greater than 0, let the variable numSpecifiedMetadataExtensionBits be the number of bits representing all syntax elements between nnpfc\_num\_metadata\_extension\_bits and nnpfc\_reserved\_metadata\_extension. nnpfc\_num\_metadata\_extension\_bits greater than 0 specifies the sum of numSpecifiedMetadataExtensionBits and the length, in bits, of nnpfc\_reserved\_metadata\_extension.

The value of nnpfc\_num\_metadata\_extension\_bits shall be in the range of numSpecifiedMetadataExtensionBits to 2 048, inclusive. Values in the range of numSpecifiedMetadataExtensionBits + 1 to 2 048, inclusive, for nnpfc\_num\_metadata\_extension\_bits are reserved for future use by ITU-T | ISO/IEC and shall not be present in bitstreams conforming to this version of this Specification. Decoders conforming to this version of this Specification shall allow any value of nnpfc\_num\_metadata\_extension\_bits in the range of 0 to numSpecifiedMetadataExtensionBits + 1 to 2 048, inclusive.

**nnpfc\_application\_purpose\_tag\_uri\_present\_flag** equal to 1indicates that the nnpfc\_application\_purpose\_tag\_uri syntax element is present in this NNPFC SEI message. nnpfc\_application\_purpose\_tag\_uri\_present\_flag equal to 0indicates that the nnpfc\_application\_purpose\_tag\_uri syntax element is not present in this NNPFC SEI message. When not present nnpfc\_application\_purpose\_tag\_uri\_present\_flag is inferred to be equal to 0.

**nnpfc\_metadata\_alignment\_zero\_bit** shall be equal to 0.

**nnpfc\_application\_purpose\_tag\_uri** specifies a tag URI with syntax and semantics as specified in IETF RFC 4151 identifying the application determined purpose of the NNPF, when nnpfc\_purpose is equal to 0.

NOTE 5 – nnpfc\_application\_purpose\_tag\_uri enables uniquely identifying the application determined purpose of NNPF without needing a central registration authority.

**nnpfc\_scan\_type\_idc** equal to 0 indicates that the preferred display method for the pictures output by the NNPF is unknown or unspecified or specified by external means. nnpfc\_scan\_type\_idc equal to 1 indicates that the pictures output by the NNPF are suitable for display using overscan. nnpfc\_scan\_type\_idc equal to 2 indicates that the pictures output by the NNPF contain visually important information in the entire region out to the edges of the picture, such that the pictures output by the NNPF should not be displayed using overscan. Instead, they should be displayed using either an exact match between the display area and the edges, or using underscan. As used in this paragraph, the term "overscan" refers to display processes in which some parts near the borders of the pictures are not visible in the display area. The term "underscan" describes display processes in which the entire pictures are visible in the display area, but they do not cover the entire display area. For display processes that neither use overscan nor underscan, the display area exactly matches the area of the pictures. The value of nnpfc\_scan\_type\_idc shall not be equal to 2. When not present, the value of nnpfc\_scan\_type\_idc is inferred to be equal to 0.

**nnpfc\_for\_human\_viewing\_idc** equal to 3 specifies that the intended optimal usage of the video resulting from the NNPF process includes human viewing. nnpfc\_for\_human\_viewing\_idc equal to 2 specifies that the video resulting from the NNPF process is suitable but not specifically optimized for human viewing. nnpfc\_for\_human\_viewing\_idc equal to 1 specifies that the video resulting by the NNPF process is unsuitable for human viewing. nnpfc\_for\_human\_viewing\_idc equal to 0 specifies that it is unknown if the video resulting by the NNPF process is suitable for human viewing. When not present, nnpfc\_for\_human\_viewing\_idc is inferred to be equal to 0.

**nnpfc\_for\_machine\_analysis\_idc** equal to 3 specifies that the intended optimal usage of the video resulting from the NNPF process includes machine analysis. nnpfc\_for\_machine\_analysis\_idc equal to 2 specifies that the video resulting from the NNPF process is suitable but not specifically optimized for machine analysis. nnpfc\_for\_machine\_analysis\_idc equal to 1 specifies that the video resulting from the NNPF process is unsuitable for machine analysis. nnpfc\_for\_machine\_analysis\_idc equal to 0 specifies that it is unknown if the video resulting from the NNPF process is suitable for machine analysis. When not present, nnpfc\_for\_machine\_analysis\_idc is inferred to be equal to 0.

It is a requirement of bitstream conformance that the value of nnpfc\_for\_human\_viewing\_idc and nnpfc\_for\_machine\_analysis\_idc shall not be both equal to 1.

NOTE 6 – When a decoding system displays the video for human viewing, any NNPF that has nnpfc\_for\_human\_viewing\_idc equal to 1 is suggested to be omitted. When a decoding system performs machine analysis, any NNPF that has nnpfc\_for\_machine\_analysis\_idc equal to 1 is suggested to be omitted.

**nnpfc\_reserved\_metadata\_extension** shall not be present in bitstreams conforming to this version of this Specification. However, decoders conforming to this version of this Specification shall ignore the presence and value of nnpfc\_reserved\_metadata\_extension. When present, the length, in bits, of nnpfc\_reserved\_metadata\_extension is equal to nnpfc\_num\_metadata\_extension\_bits − numSpecifiedMetadataExtensionBits.

**nnpfc\_alignment\_zero\_bit\_b** shall be equal to 0.

**nnpfc\_payload\_byte**[ i ] contains the i-th byte of a bitstream conforming to ISO/IEC 15938-17. The byte sequence nnpfc\_payload\_byte[ i ] for all present values of i shall be a complete bitstream that conforms to ISO/IEC 15938-17.

*Renumber clause 8.28.3 to be clause 8.28.2 and modify it to be as follows:*

* + 1. **Neural-network post-filter activation SEI message**
       1. **Neural-network post-filter activation SEI message syntax**

|  |  |
| --- | --- |
| nn\_post\_filter\_activation( payloadSize ) { | **Descriptor** |
| **nnpfa\_target\_id** | ue(v) |
| **nnpfa\_cancel\_flag** | u(1) |
| if( !nnpfa\_cancel\_flag ) { |  |
| **nnpfa\_persistence\_flag** | u(1) |
| **nnpfa\_target\_base\_flag** | u(1) |
| **nnpfa\_no\_prev\_clvs\_flag** | u(1) |
| if( nnpfa\_persistence\_flag ) |  |
| **nnpfa\_no\_foll\_clvs\_flag** |  |
| **nnpfa\_num\_output\_entries** | ue(v) |
| for( i = 0; i < nnpfa\_num\_output\_entries; i++ ) |  |
| **nnpfa\_output\_flag**[ i ] | u(1) |
| if( more\_data\_in\_payload( ) ) { |  |
| **nnpfa\_prompt\_update\_flag** | u(1) |
| if( nnpfa\_prompt\_update\_flag ) { |  |
| while( !byte\_aligned( ) ) |  |
| **nnpfa\_alignment\_zero\_bit** | u(1) |
| **nnpfa\_prompt** | st(v) |
| } |  |
| **nnpfa\_num\_input\_pic\_shift** | ue(v) |
| } |  |
| } |  |
| } |  |

* + - 1. **Neural-network post-filter activation SEI message semantics**

The neural-network post-filter activation (NNPFA) SEI message activates or de-activates the possible use of the target neural-network post-processing filter (NNPF), identified by nnpfa\_target\_id and nnpfa\_target\_base\_flag, for post-processing filtering of a set of pictures. For a particular picture for which the NNPF is activated, the target NNPF is derived as follows:

– If nnpfa\_target\_base\_flag is equal to 1, the target NNPF is the base NNPF with nnpfc\_id equal to nnpfa\_target\_id.

– Otherwise (nnpfa\_target\_base\_flag is equal to 0), the target NNPF is the NNPF specified by the last NNPFC SEI message with nnpfc\_id equal to nnpfa\_target\_id that precedes the first VCL NAL unit of the current picture in decoding order and is not a repetition of the NNPFC SEI message that contains the base NNPF.

NOTE 1 – There can be several NNPFA SEI messages present for the same picture, for example, when the NNPFs are meant for different purposes or for filtering of different colour components.

**nnpfa\_target\_id** indicates the nnpfc\_id of the target NNPF, which is specified by one or more NNPFC SEI messages that pertain to the current picture and have nnpfc\_id equal to nnpfa\_target\_id. The value of nnpfa\_target\_id shall be in the range of 0 to 232 − 2, inclusive.

An NNPFA SEI message with a particular value of nnpfa\_target\_id shall not be present in a current PU unless one or both of the following conditions are true:

– Within the current CLVS there is an NNPFC SEI message with nnpfc\_id equal to the particular value of nnpfa\_target\_id present in a PU preceding the current PU in decoding order.

– There is an NNPFC SEI message with nnpfc\_id equal to the particular value of nnpfa\_target\_id in the current PU.

When a PU contains both an NNPFC SEI message with a particular value of nnpfc\_id and an NNPFA SEI message with nnpfa\_target\_id equal to the particular value of nnpfc\_id, the NNPFC SEI message shall precede the NNPFA SEI message in decoding order.

**nnpfa\_cancel\_flag** equal to 1 indicates that the persistence of the target NNPF established by any previous NNPFA SEI message with the same nnpfa\_target\_id as the current SEI message is cancelled, i.e., the target NNPF is no longer used unless it is activated by another NNPFA SEI message with the same nnpfa\_target\_id as the current SEI message and nnpfa\_cancel\_flag equal to 0. nnpfa\_cancel\_flag equal to 0 indicates that the nnpfa\_persistence\_flag, nnpfa\_target\_base\_flag, nnpfa\_no\_prev\_clvs\_flag, nnpfa\_no\_foll\_clvs\_flag (when nnpfa\_persistence\_flag is equal to 1), and nnpfa\_num\_output\_entries follow.

**nnpfa\_persistence\_flag** specifies the persistence of the target NNPF for the current layer.

nnpfa\_persistence\_flag equal to 0 specifies that the target NNPF may be used for post-processing filtering for the current picture only.

nnpfa\_persistence\_flag equal to 1 specifies that the target NNPF may be used for post-processing filtering for the current picture and all subsequent pictures of the current layer in output order until one or more of the following conditions are true:

– A new CLVS of the current layer begins.

– The bitstream ends.

– A picture in the current layer associated with an NNPFA SEI message with the same nnpfa\_target\_id as the current SEI message is output that follows the current picture in output order.

NOTE 2 – The target NNPF is not applied for this subsequent picture in the current layer associated with an NNPFA SEI message with the same nnpfa\_target\_id as the current SEI message.

Let the nnpfcTargetPictures be the set of pictures to which the NNPFC SEI message corresponding to the target NNPF pertains. Let nnpfaTargetPictures be the set of pictures for which the target NNPF is activated by the current NNPFA SEI message. It is a requirement of bitstream conformance that any picture included in nnpfaTargetPictures shall also be included in nnpfcTargetPictures.

**nnpfa\_target\_base\_flag** equal to 1 specifies that the target NNPF is the base NNPF with nnpfc\_id equal to nnpfa\_target\_id. nnpfa\_target\_base\_flag equal to 0 specifies that the target NNPF is the NNPF specified by the last NNPFC SEI message with nnpfc\_id equal to nnpfa\_target\_id that precedes the first VCL NAL unit of the current picture in decoding order and is not a repetition of the NNPFC SEI message that contains the base NNPF.

NOTE 3 – An NNPFA message can activate a base NNPF with a particular nnpfc\_id value when an update of the base NNPF is active, which switches the target NNPF from the updated NNPF to the base NNPF.

When nnpfa\_target\_base\_flag in an NNPFA SEI message is equal to 0, there shall be at least one NNPFC SEI message with nnpfc\_id equal to nnpfa\_target\_id and nnpfc\_base\_flag equal to 0 that precedes the NNPFA SEI message in decoding order.

**nnpfa\_no\_prev\_clvs\_flag** equal to 1 specifies that the input pictures for the NNPF do not originate from a previous CLVS. nnpfa\_no\_prev\_clvs\_flag equal to 0 specifies that the input pictures for the NNPF may or may not originate from a previous CLVS.

NOTE 4 – The value of nnpfa\_no\_prev\_clvs\_flag can be changed from 0 to 1, when the current CLVS is spliced from another bitstream next to the previous CLVS and this NNPFA SEI message would cause one or more input pictures to be selected from one or more previous CLVSs and therefore is likely to impact the output of the target NNPF negatively.

**nnpfa\_no\_foll\_clvs\_flag** equal to 1 specifies that when this NNPFA SEI message persists for the last PU of a CLVS in output order, the NNPFA SEI message is treated like it persisted for the last PU, in output order, of the current layer within the bitstream. When this NNPFA SEI message does not persist for the last PU, in output order, of a CLVS in output order or nnpfa\_no\_foll\_clvs\_flag is equal to 0, the value of nnpfa\_no\_foll\_clvs\_flag causes no specific impact.

NOTE 5 – The value of nnpfa\_no\_foll\_clvs\_flag can be changed from 0 to 1 for a picture-rate-upsampling NNPF, when the following CLVS is spliced from a different bitstream next to the current CLVS. Consequently, the NNPF process interpolates pictures up to the end of the current CLVS using input pictures originating from the current CLVS only.

**nnpfa\_num\_output\_entries** specifies the number of nnpfa\_output\_flag[ i ] syntax elements present in the NNPFA SEI message. The value of nnpfa\_num\_output\_entries shall be in the range of 0 to NumInpPicsInOutputTensor, inclusive. When PictureRateUpsamplingFlag is equal to 0 and nnpfa\_num\_output\_entries is equal to NumInpPicsInOutputTensor, nnpfa\_output\_flag[ i ] shall be equal to 1 for at least one value of i in the range of 0 to nnpfa\_num\_output\_entries − 1, inclusive.

**nnpfa\_output\_flag**[ i ] equal to 1 specifies that the NNPF-generated picture that corresponds to the input picture having index InpIdx[ i ] is output by the NNPF process activated by this NNPFA SEI message, where the NNPF process is specified in the semantics of the NNPFC SEI message. nnpfa\_output\_flag[ i ] equal to 0 specifies that the NNPF-generated picture that corresponds to the input picture having index InpIdx[ i ] is not output by the NNPF process activated by this NNPFA SEI message. When nnpfa\_num\_output\_entries is less than NumInpPicsInOutputTensor, nnpfa\_output\_flag[ i ] is inferred to be equal to 1 for each value of i in the range of nnpfa\_num\_output\_entries to NumInpPicsInOutputTensor − 1, inclusive.

**nnpfa\_prompt\_update\_flag** equal to 1 specifies that nnpfa\_prompt syntax element is present and nnpfa\_alignment\_zero\_bit syntax element may be present. nnpfa\_prompt\_update\_flag equal to 0 specifies that nnpfa\_prompt syntax element and nnpfa\_alignment\_zero\_bit syntax element are not present. When not present, the value of nnpfa\_prompt\_update\_flag is inferred to be equal to 0.

When nnpfc\_inband\_prompt\_flag is equal to 0, the value of nnpfa\_prompt\_update\_flag, if present, shall be equal to 0.

**nnpfa\_alignment\_zero\_bit** shall be equal to 0.

**nnpfa\_prompt** specifies the text string prompt used as input for the target NNPF. When nnpfa\_prompt\_update\_flag is equal to 1, nnpfa\_prompt shall not be a null string.

**nnpfa\_num\_input\_pic\_shift** specifies the number of input pictures shift in the list of candidate input pictures to get the final input pictures for the target NNPF. When not present, the value of nnpfa\_num\_input\_pic\_shift is inferred to be equal to 0. The value of nnpfa\_num\_input\_pic\_shift shall be in the range from 0 to 63, inclusive.

*Add clauses 8.30 to 8.38 as follows:*

* 1. **SEI processing order and processing order nesting SEI message**
     1. **SEI processing order SEI message**
        1. **SEI processing order SEI message syntax**

|  |  |
| --- | --- |
| sei\_processing\_order( payloadSize ) { | **Descriptor** |
| **po\_id** | u(8) |
| **po\_for\_human\_viewing\_idc** | u(2) |
| **po\_for\_machine\_analysis\_idc** | u(2) |
| **po\_reserved\_zero\_4bits** | u(4) |
| **po\_num\_sei\_messages\_minus2** | u(7) |
| **po\_breadth\_first\_flag** | u(1) |
| for( i = 0, i < po\_num\_sei\_messages\_minus2 + 2; i++ ) { |  |
| **po\_sei\_wrapping\_flag**[ i ] | u(1) |
| **po\_sei\_importance\_flag**[ i ] | u(1) |
| **po\_sei\_processing\_degree\_flag**[ i ] | u(1) |
| **po\_sei\_payload\_type**[ i ] | u(12) |
| **po\_sei\_prefix\_flag**[ i ] | u(1) |
| **po\_sei\_processing\_order**[ i ] | u(8) |
| } |  |
| 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 ] | u(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) |
| } |  |
| **po\_complexity\_info\_present\_flag** | u(1) |
| if( po\_complexity\_info\_present\_flag ) { |  |
| **po\_parameter\_type\_idc** | u(2) |
| if( po\_parameter\_type\_idc  !=  2 ) |  |
| **po\_log2\_parameter\_bit\_length\_minus3** | u(2) |
| **po\_num\_parameters\_idc** | u(6) |
| **po\_num\_kmac\_operations\_idc** | ue(v) |
| **po\_total\_kilobyte\_size** | ue(v) |
| } |  |
| } |  |

* + - 1. **SEI processing order SEI message semantics**

The SEI processing order (SPO) SEI message carries information indicating the preferred processing order, as determined by the encoder (i.e., the content producer), for a group of types of SEI messages that may be present in a CLVS.

Use of this SEI message requires the definition of the following:

– Two lists of payloadType values, SeiProcessingOrderSeiList and SpoProcessSeiList.

The semantics of the SPO SEI message uses the concept of types of SEI messages. SEI messages that have different payloadType values are considered different types of SEI messages. Additionally, different SEI messages that have the same payloadType value but are differentiated by values of syntax elements in the SEI payload are considered different types of SEI messages. Such differentiation by values of syntax elements in the SEI payload is to be performed by comparing values sent using po\_sei\_prefix\_data\_bit[ i ][ j ] syntax elements (when present) or values sent as SEI messages within a processing order nesting SEI message (when present). For example, neural-network post-filter characteristics (NNPFC) SEI messages can be differentiated by having different nnpfc\_id values.

When the i-th SEI message seiA in any SPO SEI message has po\_sei\_wrapping\_flag[ i ] and po\_sei\_prefix\_flag[ i ] both equal to 0, there shall be no other SEI message seiB included in the same SPO SEI message or in a different SPO SEI message in the current CLVS for which all of the following are true:

– The value of po\_sei\_payload\_type[ i ] of seiB is the same as that for seiA.

– The value of po\_sei\_wrapping\_flag[ i ] of seiB is equal to 0.

– The value of po\_sei\_prefix\_flag[ i ] of seiB is equal to 1.

When an SPO SEI message with a particular value of po\_id is present in any picture unit of a CLVS, an SPO SEI message with that particular value of po\_id shall be present in the first picture unit of the CLVS in decoding order. The number of SEI messages and the payloadType codes of the SEI messages indicated within each SPO SEI message with the same value of po\_id persist in decoding order from the current picture unit until the end of the CLVS in output order.

The SPO SEI message can carry one or more SEI prefix indications of a particular payloadType. When present, each SEI prefix indication is a bit string that follows the SEI payload syntax of that value of payloadType and contains a number of complete syntax elements starting from the first syntax element in the SEI payload. These SEI prefix indications should provide sufficient information to determine the specific processing order for types of SEI messages having the same value of payloadType but a different preferred processing order.

**po\_id** contains an identifying number to identify the SPO SEI message.

A processing chain consists of a list of types of SEI messages identified by an SPO SEI message in the preferred processing order indicated in the SPO SEI message. When multiple processing chains are indicated by SPO SEI messages for the same access unit, a decoder should operate only one of the indicated processing chains.

Each type of SEI message in the processing chain indicated by an SPO SEI message is identified by the syntax elements po\_sei\_payload\_type[ i ], po\_sei\_wrapping\_flag[ i ], po\_sei\_processing\_order[ i ] and, when present, po\_num\_bits\_in\_prefix\_indication\_minus1[ i ] and po\_prefix\_data\_bit[ i ][ j ].

An SEI message type is not required to belong to any processing chain and may belong to any number of processing chains identified by SPO SEI messages with different po\_id values.

Each SEI message of an SEI message type identified within the SPO SEI message has the same persistence scope as if the SEI message was carried outside of the SPO SEI message and not identified within an SPO SEI message.

NOTE 1 – When an SEI message specifies a process and is not associated with a processing chain specified by any SPO SEI message, it is implicitly a processing chain by itself. Some standards, such as Rec. ITU-T H.266 | ISO/IEC 23090-3, have specified an implicit processing chain of a super-resolution NNPF followed by another NNPF. Implicitly specified processing chains are treated like processing chains specified by SPO SEI messages when selecting SEI messages to be applied.

**po\_for\_human\_viewing\_idc** equal to 3 specifies that the intended optimal usage of the video resulting from the processing chain specified by this SPO SEI message includes for human viewing. po\_for\_human\_viewing\_idc equal to 2 specifies that that the video resulting from the processing chain specified by this SPO SEI message is suitable but not specifically optimized for human viewing. po\_for\_human\_viewing\_idc equal to 1 specifies that the video resulting from the processing chain specified by this SPO SEI message is unsuitable for human viewing. po\_for\_human\_viewing\_idc equal to 0 specifies that it is unknown if the video resulting from the processing chain specified by this SPO SEI message is suitable for human viewing.

**po\_for\_machine\_analysis\_idc** equal to 3 specifies that the intended optimal usage of the video resulting from the processing chain specified by this SPO SEI message includes machine analysis. po\_for\_machine\_analysis\_idc equal to 2 specifies that the video resulting from the processing chain specified by this SPO SEI message is suitable but not specifically optimized for machine analysis. po\_for\_machine\_analysis\_idc equal to 1 specifies that the video resulting from the processing chain specified by this SPO SEI message is unsuitable for machine analysis. po\_for\_machine\_analysis\_idc equal to 0 specifies that it is unknown if the video resulting from the processing chain specified by this SPO SEI message is suitable for machine analysis.

It is a requirement of bitstream conformance that the value of po\_for\_human\_viewing\_idc and po\_for\_machine\_analysis\_idc shall not be both equal to 1.

NOTE 2 – The values of po\_for\_human\_viewing\_idc and po\_for\_machine\_analysis\_idc are in force for the output of a processing chain.

**po\_reserved\_zero\_4bits** shall be equal to 0. Values greater than 0 for po\_reserved\_zero\_4bits are reserved for future use by ITU-T | ISO/IEC and shall not be present in bitstreams conforming to this version of this Specification. Decoders conforming to this version of this Specification shall allow any value of po\_reserved\_zero\_4bits in the range of 0 to 15, inclusive.

**po\_num\_sei\_messages\_minus2** plus 2 indicates the number of types of SEI messages for which the preferred order of processing is indicated in the SPO SEI message. The variable PoNumProcStgs is set equal to po\_num\_sei\_messages\_minus2 + 2.

**po\_breadth\_first\_flag** equal to 1 specifies that the breadth-first handling of a processing chain specified in clause 8.30.3.2 shall be applied to determine the pictures that are used for interpreting the semantics of the SEI messages applied as a part of the processing chain specified by this SPO SEI message. po\_breadth\_first\_flag equal to 0 specifies that the breadth-first handling of a processing chain specified in clause 8.30.3.2 or the depth-first handling of a processing chain specified in clause 8.30.3.3 shall be applied to determine the pictures that are used for interpreting the semantics of the SEI messages applied as a part of the processing chain specified by this SPO SEI message.

NOTE 3 – When po\_breadth\_first\_flag is equal to 0, the processing chain can be performed for a picture without processing any SEI messages applying to subsequent picture units in output order.

**po\_sei\_wrapping\_flag**[ i ] equal to 1 specifies that an SEI message that applies as the i-th SEI message type in the processing chain specified in this SPO SEI message, if present, is an SEI message that is included in a PON SEI message for which both of the following conditions are true:

– pon\_target\_po\_id[ j ] with any value of j is equal to po\_id.

– There is a k-th loop entry in the processing order nesting SEI message such that the payloadType of the k-th nested SEI message is equal to po\_sei\_payload\_type[ i ] and pon\_processing\_order[ k ] is equal to po\_sei\_processing\_order[ i ].

po\_sei\_wrapping\_flag[ i ] equal to 0 specifies that an SEI message that applies as the i-th SEI message type in the processing chain specified in this SPO SEI message, if present, is an SEI message that is not included in a PON SEI message and for which both of the following conditions are true:

– The payloadType of the SEI message is equal po\_sei\_payload\_type[ i ].

– po\_sei\_prefix\_flag[ i ] is equal to 0, or when po\_sei\_prefix\_flag[ i ] is equal to 1, the payload of the SEI message starts with the values of po\_sei\_prefix\_data\_bit[ i ][ j ].

NOTE 4 – po\_sei\_wrapping\_flag[ i ] equal to 1 enables SEI messages to be carried within the processing order nesting SEI message to prevent such SEI messages from being incorrectly interpreted by decoders that do not process the SPO 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\_importance\_flag**[ i ] affects the derivation of PoSeiList, which is the list of SEI messages that a decoding system should process for a particular picture picA, as specified below.

**po\_sei\_processing\_degree\_flag**[ i ] affects the derivation of PoSeiList as specified below.

A processing chain may contain zero or more sub-chains. A sub-chain includes such SEI message types that either all these SEI message types should be processed by a decoding system or, if the decoding system cannot interpret or does not support one or more of these SEI message types, none of the SEI message types of the sub-chain should be processed. The SEI message types that belong to a sub-chain are determined by the values of po\_importance\_flag[ i ] and po\_processing\_degree\_flag[ i ] as specified below.

Table 24 specifies the interpretation of po\_importance\_flag[ i ] and po\_processing\_degree\_flag[ i ].

**Table 24 – Interpretation of** **po\_importance\_flag[ i ] and po\_processing\_degree\_flag[ i ]**

|  |  |  |
| --- | --- | --- |
| **po\_sei\_­importance\_flag**[ i ] | **po\_sei\_processing\_­degree\_flag**[ i ] | **Interpretation** |
| 1 | 1 | When the decoding system cannot interpret or does not support the i-th SEI message type, it should not apply the processing chain. |
| 0 | 1 | The i-th SEI message type is the last processing stage of a sub-chain. When the decoding system cannot interpret or does not support the i-th SEI message type, it should not apply the sub-chain. |
| 1 | 0 | The i-th SEI message type belongs to a sub-chain. When the decoding system cannot interpret or does not support the i-th SEI message type, it should not apply the sub-chain. |
| 0 | 0 | When the decoding system cannot interpret or does not support the i-th SEI message type, it shall ignore all data associated with the loop variable value of i and exclude the i-th SEI message type from the processing chain performed by the decoding system. |

**po\_sei\_payload\_type**[ i ] specifies the payloadType value of the i-th type of SEI message.

NOTE 5 – An NNPFC SEI message type has the value of po\_sei\_payload\_type[ i ] equal to the payload type value of the NNPFC SEI message. Examples of reasons for including an NNPFC SEI message type in an SPO SEI message include the following:

– The NNPFC SEI message type includes an SEI prefix that includes the nnpfc\_purpose syntax element, which provides a hint to a decoding system which kinds of processing is included in the respective NNPF in the processing chain.

– The NNPF is to be invoked only as a part of a processing chain and hence the NNPFC SEI message type is associated with po\_sei\_wrapping\_flag[ i ] equal to 1 and the respective NNPFC SEI message(s) are included in PON SEI message(s).

When an po\_sei\_payload\_type[ i ] indicates an NNPFC SEI message for a particular NNPF, the same SPO SEI message shall contain po\_sei\_payload\_type[ j ] that indicates a respective NNPFA SEI message for the same NNPF with j greater than i.

**po\_sei\_prefix\_flag**[ i ] equal to 1 specifies that po\_num\_bits\_in\_prefix\_indication\_minus1[ i ] and some 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.

The value of po\_sei\_payload\_type[ i ] for each i in the range of 0 to po\_num\_sei\_messages\_minus2 + 1, inclusive, shall be equal to a value in SeiProcessingOrderSeiList.

When po\_sei\_payload\_type[ i ] is equal to any value in SpoProcessSeiList, the i-th type of SEI message indicates a process.

spoPropertySeiList is set to consist of the payloadType values included in SeiProcessingOrderSeiList excluding the paylaodType values included in SpoProcessSeiList. When po\_sei\_payload\_type[ i ] is equal to any value in spoPropertySeiList, the i-th type of SEI message indicates a property.

**po\_sei\_processing\_order**[ i ] indicates the preferred order of processing of the i-th type of SEI message for which preferred processing order information is provided in the SPO SEI message. For any two different integer values of m and n, po\_sei\_processing\_order[ m ] less than po\_sei\_processing\_order[ n ] indicates that the type of SEI message associated with index m should be processed before the type of SEI message associated with index n, and po\_sei\_processing\_order[ m ] equal to po\_sei\_processing\_order[ n ] indicates that there is no preferred order of processing between the types of SEI messages associated with indexes m and n (e.g., they can indicate different properties that are both applicable at that stage, or one can indicate a property and the other can indicate a process).

For i greater than 0, po\_sei\_processing\_order[ i ] shall be greater than or equal to po\_sei\_processing\_order[ i − 1 ].

Let seiMsgA be an SEI message that applies as the i-th SEI message type in the processing chain specified in this SPO SEI message, persists for a particular picture picA, and is associated with po\_sei\_processing\_order[ i ] equal to poValA.

Let seiMsgSet be a set of SEI messages that consists of each SEI message for which all of the following conditions are true:

– The SEI message applies as the k-th SEI message type in the processing chain specified in this SPO SEI message with any value of k less than i.

– The SEI message persists for picA.

– po\_sei\_processing\_order[ k ] is less than poValA.

– The payloadType value of the SEI message is among the values included in SpoProcessSeiList.

The pictures to which the semantics of seiMsgA apply are specified as follows:

– If seiMsgSet is non-empty, the semantics of seiMsgA apply to all the pictures generated by the process implied by the SEI message that has the greatest value of po\_sei\_processing\_order[ k ] among the SEI messages in seiMsgSet.

– Otherwise, the semantics of seiMsgA apply to picA.

NOTE 6 – When an NNPF process outputs more than one NNPF-generated picture, the semantics of an SEI message that follows the NNPF in the processing order apply to all these NNPF-generated pictures.

**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, with prefix\_sei\_payload\_type replaced by po\_sei\_payload\_type[ i ].

When more than one SPO SEI message with a particular value of po\_id is present in a CLVS, the values of po\_for\_human\_viewing\_idc, po\_for\_machine\_analysis\_idc, po\_num\_sei\_messages\_minus2, po\_breadth\_first\_flag and, for each value of i, the values of po\_sei\_wrapping\_flag[ i ], po\_sei\_importance\_flag[ i ], po\_sei\_processing\_degree\_flag[ i ], po\_sei\_payload\_type[ i ], po\_sei\_prefix\_flag[ i ], po\_sei\_processing\_order[ i ] shall be the same as in the other SPO SEI messages in the CLVS with the same value of po\_id.

**po\_byte\_alignment\_bit\_equal\_to\_one** shall be equal to 1.

The lists PoProcStgIdx, indicating the processing stage indices of the SEI message types in the processing chain, and PoSeiTypeIdx, indicating the SEI message type indices of the processing stages in the processing chain, are derived as follows:

– For each of the SEI message types of in the processing chain, the following applies in a non-decreasing order of the corresponding po\_sei\_processing\_order[ i ] values, with j being set equal to 0 initially:

PoProcStgIdx[ i ] = j  
PoSeiTypeIdx[ j ] = i (xx)  
j++

Where PoProcStgIdx[ i ] indicates the processing stage index of the i-th SEI message type in the processing chain, and PoSeiTypeIdx[ j ] indicates the SEI message type index of the j-th processing stage in the processing chain.

The list poSubChainIdx[ j ] for j ranging from 0 to PoNumProcStgs − 1, inclusive, specifying the sub-chain index of the j-th processing stage of the processing chain, is derived as follows:

poSubChainFlag = 0  
poSubChainPrevIdx = 0  
for( j = 0; j < PoNumProcStgs; j++ ) { (xx)  
 idx = PoSeiTypeIdx[ j ]  
 if( po\_sei\_importance\_flag[ idx ]  = =  1  &&  po\_sei\_processing\_degree\_flag[ idx ]  = =  1 )  
 poSubChainIdx[ j ] = 0  
 else if( po\_sei\_importance\_flag[ idx ]  = =  0  &&  po\_sei\_processing\_degree\_flag[ idx ]  = =  1 ) {  
 poSubChainIdx[ j ] = poSubChainPrevIdx  
 poSubChainFlag = 0  
 } else if( po\_sei\_importance\_flag[ idx ]  = =  1  &&  po\_sei\_processing\_degree\_flag[ idx ]  = =  0 ) {  
 if( poSubChainFlag  = =  0 )  
 poSubChainPrevIdx++  
 poSubChainIdx[ j ] = poSubChainPrevIdx  
 poSubChainFlag = 1  
 } else  
 poSubChainIdx[ j ] = poSubChainFlag \* poSubChainPrevIdx  
}

For a picture, the list PoSeiList, indicting the list of SEI messages that may be applied to the picture, the list PoSeiTypeList, indicating the SEI message type indices of the SEI messages that may be applied to the picture, and the variable PoNumSeiMsgs, indicating the number of SEI messages that may be applied to the picture, are derived as follows:

– PoSeiList is initially empty, and seiListIdx and PoNumSeiMsgs are both initially set equal to 0.

– The following applies in increasing order of j for all values of j in the range of 0 to PoNumProcStgs − 1, inclusive, unless terminated earlier as specified below:

– When an SEI message seiA associated with the PoSeiTypeIdx[ j ]-th SEI message type persists for picA, the following applies:

– If all of the following conditions are true, seiA is added at the end of PoSeiList, PoSeiTypeList[ seiListIdx ] is set equal to PoSeiTypeIdx[ j ], PoNumSeiMsgs is set equal to PoNumSeiMsgs + 1, and seiListIdx is set equal to seiListIdx + 1:

– The decoding system can interpret and supports the functionality indicated by seiA.

– Either of the following conditions is true:

– poSubChainIdx[ j ] is equal to 0.

– The decoding system can interpret and supports the functionality indicated by all SEI message types for all values of k such that poSubChainIdx[ k ] is equal to poSubChainIdx[ j ] and either po\_sei\_importance\_flag[ PoSeiTypeIdx[ k ] ] is equal to 1 or po\_sei\_processing\_degree\_flag[ PoSeiTypeIdx[ k ] ] is equal to 1.

– Otherwise, if po\_sei\_importance\_flag[ PoSeiTypeIdx[ j ] ] is equal to 1 and po\_sei\_processing\_degree\_flag[ PoSeiTypeIdx[ j ] ] is equal to 1, the processing chain specified by this SPO SEI message should not be performed for picA, PoSeiList is set to be empty, PoNumSeiMsgs is set equal to 0, and the derivation of PoSeiList, PoSeiTypeList, and PoNumSeiMsgs is terminated.

**po\_complexity\_info\_present\_flag** equal to 1 specifies that one or more syntax elements that indicate the complexity of invoking NNPFs in the processing chain identified by the sei processing order SEI message are present. po\_complexity\_info\_present\_flag equal to 0 specifies that no syntax elements that indicates the complexity of invoking NNPFs in the processing chain identified by the sei processing order SEI message are present.

**po\_parameter\_type\_idc** equal to 0 indicates that the NNPFs in the processing chain identified by the sei processing order SEI message use only integer parameters. po\_parameter\_type\_flag equal to 1 indicates that the NNPFs in the processing chain identified by the sei processing order SEI message may use floating point or integer parameters. po\_parameter\_type\_idc equal to 2 indicates that the NNPFs in the processing chain identified by the sei processing order SEI message uses only binary parameters. po\_parameter\_type\_idc equal to 3 is reserved for future use by ITU-T | ISO/IEC and shall not be present in bitstreams conforming to this edition of this document. Decoders conforming to this edition of this document shall ignore SPO SEI messages with po\_parameter\_type\_idc equal to 3.

**po\_log2\_parameter\_bit\_length\_minus3** equal to 0, 1, 2, and 3 indicates that the NNPFs in the processing chain identified by the sei processing order SEI message do not use parameters of bit length greater than 8, 16, 32, and 64, respectively. When po\_parameter\_type\_idc is present and po\_log2\_parameter\_bit\_length\_minus3 is not present, the NNPFs in the processing chain identified by the sei processing order SEI message do not use parameters of bit length greater than 1.

**po\_num\_parameters\_idc** indicates the maximum number of parameters needed by NNPFs in the processing chain identified by the sei processing order SEI message in units of a power of 2 048. po\_num\_parameters\_idc equal to 0 indicates that the maximum number of parameters needed by NNPFs in the processing chain identified by the sei processing order SEI message is unknown. The value po\_num\_parameters\_idc shall be in the range of 0 to 52, inclusive. Values of po\_num\_parameters\_idc greater than 52 are reserved for future use by ITU-T | ISO/IEC and shall not be present in bitstreams conforming to this edition of this document. Decoders conforming to this edition of this document shall ignore SPO SEI messages with po\_num\_parameters\_idc greater than 52.

If the value of po\_num\_parameters\_idc is greater than zero, the variable maxNumParameters is derived as follows:

maxNumParameters = ( 2 048  <<  po\_num\_parameters\_idc ) − 1(xx)

It is a requirement of bitstream conformance that the number of parameters shall be less than or equal to maxNumParameters.

**po\_num\_kmac\_operations\_idc** greater than 0 indicates that the maximum number of multiply-accumulate operations per sample of the NNPFs in the processing chain identified by the sei processing order SEI message is less than or equal to po\_num\_kmac\_operations\_idc \* 1 000. po\_num\_kmac\_operations\_idc equal to 0 indicates that the maximum number of multiply-accumulate operations of the NNPFs in the processing chain identified by the sei processing order SEI message is unknown. The value of po\_num\_kmac\_operations\_idc shall be in the range of 0 to 232 − 2, inclusive.

**po\_total\_kilobyte\_size** greater than 0 indicates a total size in kilobytes required to store the uncompressed parameters for the NNPFs in the processing chain identified by the sei processing order SEI message. The total size in bits is a number equal to or greater than the sum of bits used to store each parameter. po\_total\_kilobyte\_size is the total size in bits divided by 8 000, rounded up. po\_total\_kilobyte\_size equal to 0 indicates that the total size required to store the parameters for the NNPFs in the processing chain identified by the sei processing order SEI message is unknown. The value of po\_total\_kilobyte\_size shall be in the range of 0 to 232 − 2, inclusive.

* + 1. **Processing order nesting SEI message**
       1. **Processing order nesting SEI message syntax**

|  |  |
| --- | --- |
| processing\_order\_nesting( payloadSize ) { | **Descriptor** |
| **pon\_num\_po\_ids\_minus1** | u(8) |
| for( i = 0; i <= pon\_num\_po\_ids\_minus1; i++ ) |  |
| **pon\_target\_po\_id**[ i ] | u(8) |
| **pon\_num\_seis\_minus1** | u(8) |
| for( i = 0; i <= pon\_num\_seis\_minus1; i++ ) { |  |
| **pon\_processing\_order**[ i ] | u(8) |
| sei\_pon\_nested\_message( ) |  |
| } |  |
| } |  |

* + - 1. **Processing order nesting SEI message semantics**

The processing order nesting (PON) SEI message includes one or more SEI messages that should be applied only as parts of the processing chain identified by an associated SEI processing order SEI message and should not be applied in a manner that would contradict with the processing chain identified by the associated SEI processing order SEI message.

Use of this SEI message requires the definition of the following:

– The syntax structure of the container of SEI messages, sei\_pon\_nested\_message( )

The SEI messages contained in a PON SEI message are referred to as PON-nested SEI messages.

NOTE – An encoder can include multiple PON SEI messages in the same access unit. For example, a first PON SEI message in an access unit can contain a PON-nested SEI message that applies to multiple processing chains and one or more other PON SEI messages in the same access unit that apply to a single processing chain only.

It is a requirement of bitstream conformance that the semantics and effect of an SEI message that is not a PON-nested SEI message shall not depend on any PON-nested SEI message. Consequences of this constraint include the following specific constraints, in which an associated SEI message is considered to be an SEI message that affects the semantics or effect of a particular SEI message:

– When a neural-network post-filter characteristics SEI message is present with a particular value of nnpfc\_id that is a PON-nested SEI message, any associated neural-network post-filter activation SEI messages with nnpfa\_target\_id equal to that particular value of nnpfc\_id shall also be PON-nested SEI messages.

– When a neural-network post-filter activation (NNPFA) SEI message is present with nnpfa\_persistence\_flag equal to 1 and a particular value of nnpfa\_target\_id that is not a PON-nested SEI message, the next picture in output order in the same CLVS that has an NNPFA SEI message with the same value of nnpfa\_target\_id (if any) shall not have an associated NNPFA SEI message that is a PON-nested SEI message.

– When a film grain characteristics SEI message is present with fg\_characteristics\_persistence\_flag equal to 1 that is not a PON-nested SEI message, there shall not be an associated film grain characteristics SEI message in the same CLVS that is a PON-nested SEI message.

– When a frame packing arrangement SEI message is present with fp\_arrangement\_persistence\_flag equal to 1 that is not a PON-nested SEI message, there shall not be an associated frame packing arrangement SEI message in the same CLVS with fp\_arrangement\_cancel\_flag equal to 1 or the same value of fp\_arrangement\_id that is a PON-nested SEI message.

– When a content colour volume SEI message is present with ccv\_persistence\_flag equal to 1 that is not a PON-nested SEI message, there shall not be an associated frame packing arrangement SEI message in the same CLVS that is a PON-nested SEI message.

– When an equirectangular projection SEI message is present with erp\_persistence\_flag equal to 1 that is not a PON-nested SEI message, there shall not be an associated equirectangular projection SEI message in the same CLVS that is a PON-nested SEI message.

– When a region-wise packing SEI message is present with rwp\_persistence\_flag equal to 1 that is not a PON-nested SEI message, there shall not be an associated region-wise packing SEI message in the same CLVS that is a PON-nested SEI message.

– When a sample aspect ratio SEI message is present with sari\_persistence\_flag equal to 1 that is not a PON-nested SEI message, there shall not be an associated sample aspect ratio SEI message in the same CLVS that is a PON-nested SEI message.

– When an annotated regions SEI message is present that is not a PON-nested SEI message, there shall not be an associated annotated regions SEI message in the same CLVS that is a PON-nested SEI message.

– When an alpha channel information SEI message is present that is not a PON-nested SEI message, there shall not be an associated alpha channel information SEI message in the same CLVS that is a PON-nested SEI message.

– When a display orientation SEI message is present that is not a PON-nested SEI message, there shall not be an associated display orientation SEI message in the same CLVS that is a PON-nested SEI message.

– When a colour transform indication SEI message is present with colour\_transform\_persistence\_flag equal to 1 that is not a PON-nested SEI message, there shall not be an associated colour transform indication SEI message in the same CLVS with colour\_transform\_cancel\_flag equal to 1 or the same value of colour\_transform\_id that is a PON-nested SEI message.

**pon\_num\_po\_ids\_minus1** plus 1 specifies the number of the SEI processing order SEI messages SEI associated with this PON SEI message.

**pon\_target\_po\_id**[ i ] indicates the po\_id of the i-th SEI processing order SEI message associated with this PON SEI message.

**pon\_num\_seis\_minus1** plus 1 specifies the number of the PON-nested SEI messages that are included in this PON SEI message.

**pon\_processing\_order**[ i ] specifies the position of the i-th PON-nested SEI message within the processing order defined by the associated SEI processing order SEI message. When i is greater than 0, pon\_processing\_order[ i ] shall be greater than or equal to pon\_processing\_order[ i − 1 ].

An associated SEI processing order SEI message for the i-th PON-nested SEI message is an SEI processing order SEI message that has an entry k for which all of the following conditions are true:

– po\_sei\_processing\_order[ k ] is equal to pon\_processing\_order[ i ]

– po\_sei\_payload\_type[ k ] is equal to the payloadType value of the i-th PON-nested SEI message.

– When po\_sei\_prefix\_flag[ k ] is equal to 1, po\_sei\_prefix\_data\_bit[ k ][ j ] for j in the range of 0 to po\_num\_bits\_in\_prefix\_indication\_minus1[ k ], inclusive, contain the same content as the po\_num\_bits\_in\_prefix\_indication\_minus1[ k ] plus 1 initial bits of the SEI message payload of the i-th PON-nested SEI message.

The i-th PON-nested SEI message may have any number of associated SEI processing order SEI messages in the range of 0 to pon\_num\_po\_ids\_minus1 + 1, inclusive.

When the i-th PON-nested SEI message has an associated SEI processing order SEI message, the i-th PON-nested SEI message should be applied as the k-th loop entry of the associated SEI processing order SEI message.

The semantics of the i-th PON-nested SEI message applied as the k-th loop entry of the associated SEI processing order SEI message with a particular po\_id value apply without considering any of the PON-nested SEI messages not associated with any SEI processing order SEI message with that particular po\_id value.

For each SEI processing order SEI message that is present in the CLVS and has po\_id equal to pon\_target\_po\_id[ m ] for any value of m in the range of 0 to pon\_num\_po\_ids\_minus1, inclusive, there shall be at least one value n in the range of 0 to pon\_num\_seis\_minus1, inclusive, for which the SEI processing order SEI message is the associated SEI processing order SEI message for the n-th PON-nested SEI message.

* + 1. **Handling of a processing chain**
       1. **General**

Processing chains are alternatives to each other, i.e., at most one processing chain can be chosen to be applied by a decoding system at one time.

An implict NNPF cascading case is defined as the case when two NNPFs are both activated for a picture by NNPFA SEI messages that are not included in PON SEI messages, one of the two NNPFs, referred to as nnpfA, has nnpfc\_purpose equal to 4 and the other, referred to as nnpfB, has multiple input pictures, and the NNPFC SEI messages of the two NNPFs are not included in PON SEI messages. In this case, the two NNPFs are implicitly considered as belonging to one processing chain, and the processing chain only contains these two NNPFs.

NOTE 1 – An implicit NNPF cascading case can be applied when cropped decoded pictures have different spatial resolutions, which can happen, for example, when sps\_ref\_pic\_resampling\_enabled\_flag is equal to 1 in a bitstream conforming to Rec. ITU-T H.266 | ISO/IEC 23090-3. In an example, cropped decoded pictures have two spatial resolutions and the first NNPF in the processing chain for the implicit NNPF cascading case is activated selectively so that low-resolution cropped decoded pictures are upsampled to the resolution of the high-resolution cropped decoded pictures. Consequently, input pictures to the second NNPF in the processing chain have the same spatial resolution.

Except for the implict NNPF cascading case, each processing chain containing multiple SEI message types is indicated by an SPO SEI message with a particular value of po\_id. Except for the implict NNPF cascading case, when the bitstream includes one or more SEI messages that are not included in PON SEI messages and have a particular payloadType and the payloadType is present in SpoProcessSeiList, the SEI message type for the payloadType is in its own processing chain. This case is referred to as the implict single-processing-stage case.

In the implict NNPF cascading case, po\_breadth\_first\_flag is inferred to be equal to either 0 or 1, PoNumProcStgs is set equal to 2, PoNumSeiMsgs is set equal to 2, and the following applies:

1. The SEI message types with index 0 and 1 correspond to nnpfA and nnpfB, respectively.
2. If both of the NNPFs have nnpfc\_purpose equal to 4 and multiple input pictures, either of the two is chosen to be applied first, and for the one chosen to be applied first, PoProcStgIdx[ i ] is set equal to 0, and for the other, PoProcStgIdx[ i ] is set equal to 1.
3. Otherwise, PoProcStgIdx[ 0 ] is set equal to 0, PoProcStgIdx[ 1 ] is set equal to 1.

In the implict single-processing-stage case, po\_breadth\_first\_flag is inferred to be equal to either 0 or 1, PoNumProcStgs is set equal to 1, PoNumSeiMsgs is set equal to 1, PoProcStgIdx[ 0 ] is set equal to 0, and PoSeiList[ 0 ] is set to be the single SEI message.

The PoSeiList for a corresponding picture of picA or for an associated inserted picture of picA is derived to be the same as the PoSeiList derived for picA.

A decoding system may choose and apply a processing chain as follows:

1. The bitstream is decoded, a processing chain is chosen, and the following applies:

– The list PoCdoPicList is set to be the list of the cropped decoded pictures in output order that resulted from decoding the bitstream.

– The list PoDecPicList is set to be the list of the decoded pictures in output order resulted from decoding the bitstream.

1. The following applies for each decoded picture picA in output order:
   1. When the implicit NNPF cascading case was chosen, the following applies for picA:

– PoNumSeiMsgs is initially set equal to 0 for picA.

– When an NNPFA SEI message that corresponds to PoProcStgIdx[ i ] equal to 0 activates an NNPF for picA, PoSeiList[ PoNumSeiMsgs++ ] is set to be equal to that NNPFA SEI message.

– When an NNPFA SEI message that corresponds to PoProcStgIdx[ i ] equal to 1 activates an NNPF for picA, PoSeiList[ PoNumSeiMsgs++ ] is set to be equal to that NNPFA SEI message.

* 1. If PoSeiTypeList[ 0 ] does not correspond to an FGC SEI message, the picture in PoCdoPicList that corresponds to picA is added to the lists CandInputPicList[ i ] for all values of i in the range of 0 to PoSeiTypeList[ 0 ], inclusive.
  2. Otherwise, the picture in PoDecPicList that corresponds to picA is added to the lists CandInputPicList[ i ] for all values of i in the range of 0 to PoSeiTypeList[ 0 ], inclusive.
  3. The picture in PoCdoPicList that corresponds to picA is added to the lists CandInputPicList[ i ] for all values of i in the range of PoSeiTypeList[ 0 ] + 1 to PoNumProcStgs, inclusive.

NOTE 2 – The lists CandInputPicList[ i ] for i in the range of 1 to PoNumProcStgs, inclusive, is possibly updated during the next step. After step 4 below, the list CandInputPicList[ PoNumProcStgs ] stores the final output of the chosen processing chain.

1. If the chosen processing chain is indicated by an SPO SEI message and po\_breadth\_first\_flag in the SPO SEI message is equal to 1, the breadth-first handling of a processing chain specified in clause 8.30.3.2 is invoked. Otherwise, either the depth-first handling of a processing chain specified in clause 8.30.3.3 or the breadth-first handling of a processing chain specified in clause 8.30.3.2 is invoked, and it is a requirement of bitstream conformance that regardless of whether the the depth-first handling of a processing chain specified in clause 8.30.3.3 or the breadth-first handling of a processing chain specified in clause 8.30.3.2 is invoked, the resulting CandInputPicList[ PoNumProcStgs ] shall be the same.
2. The list PoOutputPicList is set to be identical to CandInputPicList[ PoNumProcStgs ].

NOTE 3 – The pictures in PoOutputPicList are in increasing output order and within PoOutputPicList there are no two pictures having the same output order.

For each processing stage with processing stage index i in the range of 0 to PoNumProcStgs − 1, inclusive, the following applies:

– For any particular pair of pictures inputPicA and inputPicB consecutive in output order in CandInputPicList[ i ], which is the list of candidate input pictures for the processing stage, when there are one or more pictures intermediatePicSetA between inputPicA and inputPicB in output order added to CandInputPicList[ i + 1 ] when applying the process implied by a particular SEI message of the processing stage, the application of the process implied by the particular SEI message when another picture other than currPicA was the current picture or the application of the process implied by of another SEI message of the same processing stage when any picture (including currPicA) was the current picture shall not output any picture between the inputPicA and inputPicB in output order.

NOTE 4 – The intent of the constraint expressed above is to disallow generating output pictures between any particular pair of consecutive input pictures more than once within a processing stage.

* + - 1. **Breadth-first handling of a processing chain**

For each of the SEI message types, with SEI message type index i, of the chosen processing chain, the following applies in increasing order of the corresponding processing stage index PoProcStgIdx[ i ] values:

– The following applies for each picture picA in CandInputPicList[ PoProcStgIdx[ i ] ] in output order, when an SEI message associated with the i-th SEI message type is present in PoSeiList of picA:

– When the i-th SEI message type does not correspond to PoSeiList[ 0 ], the following applies for the interpretation of the SEI message:

– The interface variables for purposes of interpretation of the SEI message are derived from picA.

– The semantics of the SEI message, or the semantics of the SEI message and, when the SEI message is an NNPFA SEI message, the associated NNPFC SEI message, apply to pictures in CandInputPicList[ PoProcStgIdx[ i ] ].

– When the SEI payloadType value of the i-th SEI message type is present in SpoProcessSeiList, the process implied by the SEI message is performed and each list CandInputPicList[ PoProcStgIdx[ i ] + j ] with j in the range of 1 to PoNumProcStgs − PoProcStgIdx[ i ], inclusive, is updated by replacing pictures with the corresponding processed pictures, if any, resulting from the process and inserting the other pictures, if any, resulting from the process into CandInputPicList[ PoProcStgIdx[ i ] + j ] so that the output order is obeyed. When the SEI message is an FGC SEI message and PoSeiList[ 0 ], during the replacement of a picture in CandInputPicList[  PoProcStgIdx[ i ] + j  ] with the corresponding processed picture, the corresponding processed picture is first cropped, in the same manner as generating a cropped decoded output picture from the corresponding decoded picture, and the cropped picture is used for the replacement.

* + - 1. **Depth-first handling of a processing chain**

The following is repeatedly applied, in output order, for each picture picA in CandInputPicList[ 0 ]:

– The following applies for each SEI message with SEI message index seiIdx in PoSeiList of picA in an increasing order of list indexes for PoSeiList:

– When the i-th SEI message type does not correspond to PoSeiList[ 0 ], the following applies for the interpretation of the SEI message:

– The interface variables for purposes of interpretation of the SEI message are derived from the pictures in CandInputPicList[ PoProcStgIdx[ PoSeiTypeList[ seiIdx ] ] ].

– The semantics of the SEI message, or of the SEI message and, when the SEI message is an NNPFA SEI message, the associated NNPFC SEI message, apply to pictures in CandInputPicList[ PoProcStgIdx[ PoSeiTypeList[ seiIdx ] ] ].

– When the SEI payloadType value of the SEI message is present in SpoProcessSeiList, the process implied by the SEI message is invoked repeatedly, in output order, for each of the pictures in CandInputPicList[ PoProcStgIdx[ PoSeiTypeList[ seiIdx ] ] ] that is the corresponding picture of picA or an associated inserted picture of picA. After each invocation of the process, each list CandInputPicList[ PoProcStgIdx[ PoSeiTypeList[ seiIdx ] ] + j ] with j in the range of 1 to PoNumProcStgs − PoProcStgIdx[ PoSeiTypeList[ seiIdx ] ], inclusive, is updated by replacing pictures with the corresponding processed pictures, if any, resulting from the process and inserting the other pictures, if any, resulting from the process into CandInputPicList[ PoProcStgIdx[ PoSeiTypeList[ seiIdx ] ] + j ] so that the output order is obeyed. When the SEI message is an FGC SEI message and PoSeiList[ 0 ], during the replacement of a picture in CandInputPicList[ PoSeiTypeList[ seiIdx ] ] + j  ] with the corresponding processed picture, the corresponding processed picture is first cropped, in the same manner as generating a cropped decoded output picture from the corresponding decoded picture, and the cropped picture is used for the replacement.

* 1. **Encoder optimization information SEI message**
     1. **Encoder optimization information SEI message syntax**

|  |  |
| --- | --- |
| encoder\_optimization\_info(payloadSize ) { | **Descriptor** |
| **eoi\_cancel\_flag** | u(1) |
| if( !eoi\_cancel\_flag ) { |  |
| **eoi\_persistence\_flag** | u(1) |
| **eoi\_for\_human\_viewing\_idc** | u(2) |
| **eoi\_for\_machine\_analysis\_idc** | u(2) |
| **eoi\_reserved\_zero\_2bits** | u(2) |
| **eoi\_type** | u(16) |
| if( EoiObjectBasedFlag ) |  |
| **eoi\_object\_based\_idc** | u(16) |
| if( EoiTemporalResamplingFlag ) { |  |
| **eoi\_temporal\_resampling\_type\_flag** | u(1) |
| **eoi\_num\_int\_pics** | ue(v) |
| if( eoi\_temporal\_resampling\_type\_flag  &&  eoi\_num\_int\_pics > 0 ) |  |
| **eoi\_src\_pic\_flag** | u(1) |
| } |  |
| if( EoiSpatialResamplingFlag ) { |  |
| **eoi\_orig\_pic\_dimensions\_flag** | u(1) |
| if( eoi\_orig\_pic\_dimensions\_flag ) { |  |
| **eoi\_orig\_pic\_width** | u(16) |
| **eoi\_orig\_pic\_height** | u(16) |
| } else |  |
| **eoi\_spatial\_resampling\_type\_flag** | u(1) |
| } |  |
| if( EoiPrivacyProtectionFlag ) { |  |
| **eoi\_privacy\_protection\_method\_idc** | u(16) |
| **eoi\_privacy\_info\_type** | u(8) |
| } |  |
| } |  |
| } |  |

* + 1. **Encoder optimization information SEI message semantics**

The encoder optimization information SEI message is used to indicate if the video has been optimized for human viewing or machine analysis and which types of optimization have been applied in pre-processing or encoding.

**eoi\_cancel\_flag** equal to 1 specifies that the persistence of the encoder optimization information SEI message included in any previous PU in output order is cancelled. eoi\_cancel\_flag equal to 0 indicates that information on optimization that has been applied in pre-processing or encoding follows.

**eoi\_persistence\_flag** specifies the persistence of the optimization information provided in this SEI message. eoi\_persistence\_flag equal to 0 specifies that the optimization information applies for the current picture only. eoi\_persistence\_flag equal to 1 specifies that the optimization information applies for the current picture and all subsequent pictures of the current layer in output order until one or more of the following conditions are true:

– A new CLVS of the current layer begins.

– The bitstream ends.

– A picture in the current layer associated with an encoder optimization information SEI message is output that follows the current picture in output order.

**eoi\_for\_human\_viewing\_idc** equal to 3 specifies that purposes for the applied optimization include human viewing. eoi\_for\_human\_viewing\_idc equal to 2 specifies that the video is suitable but not specifically optimized for human viewing. eoi\_for\_huma\_viewing\_idc equal to 1 specifies that the video is unsuitable for human viewing. eoi\_for\_human\_viewing\_idc equal to 0 specifies that it is unknown if the video is suitable for human viewing.

**eoi\_for\_machine\_analysis\_idc** equal to 3 specifies that purposes for the applied optimization include machine analysis. eoi\_for\_machine\_analysis\_idc equal to 2 specifies that the video is suitable but not specifically optimized for machine analysis. eoi\_for\_machine\_analysis\_idc equal to 1 specifies that the video is unsuitable for machine analysis. eoi\_for\_machine\_analysis\_idc equal to 0 specifies that it is unknown if the video is suitable for machine analysis.

It is a requirement of bitstream conformance that the value of eoi\_for\_human\_viewing\_idc and eoi\_for\_machine\_analysis\_idc shall not be both equal to 1.

**eoi\_reserved\_zero\_2bits** shall be equal to 0. Values greater than 0 for eoi\_reserved\_zero\_2bits are reserved for future use by ITU-T | ISO/IEC and shall not be present in bitstreams conforming to this version of this Specification. Decoders conforming to this version of this Specification shall allow any value of eoi\_reserved\_zero\_2bits in the range of 0 to 3, inclusive.

**eoi\_type** indicates the types of optimization method as specified in Table x1 where ( eoi\_type & bitMask ) not equal to 0 indicates that the optimization type with the bitMask value in Table x1 has been applied. When eoi\_type is greater than 0 and ( eoi\_type & bitMask ) is equal to 0, the optimization type with the bitMask value has not been applied. When eoi\_type is equal to 0, optimization as determined by the application has been used.

**Table x1 – Definition of** **eoi\_type**

|  |  |
| --- | --- |
| **bitMask** | **Interpretation** |
| 0x01 | Object-based optimization; the pictures for which this SEI message persists have been pre-processed or encoded so that detected objects in the pictures are optimized with respect to other parts of the pictures for the indicated optimization purposes |
| 0x02 | Temporal resampling optimization |
| 0x04 | Spatial resampling optimization |
| 0x08 | Temporal quality optimization in a manner that quality fluctuates temporally |
| 0x10 | Spatial quality optimization; the pictures for which this SEI message persists have been pre-processed or encoded to reduce unnecessary information or improve the quality of necessary information.(e.g to reduce the amount of noise and remove speckles at the picture-level) |
| 0x20 | Privacy protection optimization; the pictures for which this SEI message persists have been pre-processed or encoded to protect personal information. (e.g. removal or replacing of personal identifiable information, pseudonymization, anonymization) |

The variables EoiObjectBasedFlag, EoiTemporalResamplingFlag, EoiSpatialResamplingFlag, EoiTemporalQualityFlag, EoiSpatialQualityFlag, and EoiPrivacyProtectionFlag, specifying whether eoi\_type indicates the type of the optimization to include object-based optimization, temporal resampling optimization, spatial resampling optimization, temporal quality optimization, spatial quality optimization, and privacy protection optimization, respectively, are derived as follows:

EoiObjectBasedFlag = ( ( eoi\_type & 0x01 ) > 0 ) ? 1 : 0  
EoiTemporalResamplingFlag = ( ( eoi\_type & 0x02 ) > 0 ) ? 1 : 0  
EoiSpatialResamplingFlag = ( ( eoi\_type & 0x04 ) > 0 ) ? 1 : 0 (xx)  
EoiTemporalQualityFlag = ( ( eoi\_type & 0x08 ) > 0 ) ? 1 : 0  
EoiSpatialQualityFlag = ( ( eoi\_type & 0x10 ) > 0 ) ? 1 : 0  
EoiPrivacyProtectionFlag = ( ( eoi\_type & 0x20 ) > 0 ) ? 1 : 0

NOTE – For example, when certain highest temporal sublayers have been encoded with such coarse quantization that human viewers perceive the quality fluctuation annoying, but machine task performance is not compromised, eoi\_for\_human\_viewing\_flag and eoi\_for\_machine\_analaysis\_flag can be set equal to 0 and 1, respectively, and eoi\_type can be set equal to a value that causes EoiTemporalQualityFlag to be equal to 1.

When eoi\_persistence\_flag is equal to 0, it is a requirement of bitstream conformance that EoiTemporalResamplingFlag shall be equal to 0 and EoiTemporalQualityFlag shall be equal to 0.

**eoi\_object\_based\_idc**, when present, indicates the type of object-based optimization as specified in Table x2, where ( eoi\_object\_based\_idc & bitMask ) not equal to 0 indicates that the object-based optimization type associated with the bitMask value in Table x2 has been applied. When eoi\_object\_based\_idc is greater than 0 and ( eoi\_object\_based\_idc & bitMask ) is equal to 0, the object-based optimization type associated with the bitMask value has not been applied. When eoi\_object\_based\_idc is equal to 0, an application-defined type of object-based optimization has been applied. The value of eoi\_object\_based\_idc shall be in the range of 0 to 31, inclusive, in bitstreams conforming to this version of this Specification. Values of 32 to 65 535, inclusive, for eoi\_object\_based\_idc are reserved for future use by ITU-T | ISO/IEC and shall not be present in bitstreams conforming to this version of this Specification. When the value of eoi\_object\_based\_idc is in the range of 32 to 65 535, inclusive, decoders conforming to this version of this Specification shall ignore eoi\_object\_based\_idc.

**Table x2 – Definition of** **eoi\_object\_based\_idc**

|  |  |
| --- | --- |
| **bitMask** | **Interpretation** |
| 0x01 | Areas outside the detected objects have been blurred prior to encoding. |
| 0x02 | Areas outside the detected objects have been encoded with coarser transform-domain quantization than the quantization used for the detected objects. |
| 0x04 | Areas outside the detected objects have been overwritten with a constant sample value. |
| 0x08 | Areas outside the detected objects have been overwritten in some form but not with a constant sample value. |
| 0x10 | Areas in the objects have been treated differently based on the object size. For example, objects are pre-sorted in size and larger objects are coded with coarser quality than smaller objects during encoding. |

**eoi\_temporal\_resampling\_type\_flag** equal to 0 specifies that the temporal resampling optimization is a subsampling operation. eoi\_temporal\_resampling\_type\_flag equal to 1 specifies that the temporal resampling optimization is an upsampling operation.

**eoi\_num\_int\_pics** greater than 0 indicates that the count of pictures that the encoding system excluded between each pair of coded pictures in output order (when eoi\_temporal\_resampling\_type\_flag is equal to 0) or added between each pair of source pictures for encoding (when eoi\_temporal\_resampling\_type\_flag is equal to 1) within the persistence of this SEI message is constant. When eoi\_temporal\_resampling\_type\_flag is equal to 0 and eoi\_num\_int\_pics is greater than 0, eoi\_num\_int\_pics specifies the count of pictures that the encoding system excluded between each pair of coded pictures in output order. When eoi\_temporal\_resampling\_type\_flag is equal to 1 and eoi\_num\_int\_pics is greater than 0, eoi\_num\_int\_pics specifies the count of pictures that the encoding system added between each pair of source pictures for encoding.

eoi\_num\_int\_pics equal to 0 indicates that the count of pictures that the encoding system excluded between each pair of coded pictures in output order (when eoi\_temporal\_resampling\_type\_flag is equal to 0) or added between each pair of source pictures for encoding (when eoi\_temporal\_resampling\_type\_flag is equal to 1) within the persistence of this SEI message is unknown or varying.

The value of eoi\_num\_int\_pics shall be in the range of 0 to 63, inclusive.

**eoi\_src\_pic\_flag** equal to 1 specifies that the picture in the same access unit that contains the EOI SEI message is a source picture for temporal upsampling optimization. eoi\_src\_pic\_flag equal to 0 provides no such indication.

**eoi\_orig\_pic\_dimensions\_flag** equal to 1 specifies that the eoi\_orig\_pic\_width and eoi\_orig\_pic\_height syntax elements are present. eoi\_orig\_pic\_dimensions\_flag equal to 0 specifies that the eoi\_orig\_pic\_width and eoi\_orig\_pic\_height are not present.

**eoi\_orig\_pic\_width** and **eoi\_orig\_pic\_height**, when present, indicate the width and height, respectively, of the original source picture in units of luma samples.

**eoi\_spatial\_resampling\_type\_flag** equal to 0 specifies that the spatial resampling optimization is a subsampling operation. eoi\_spatial\_resampling\_type\_flag equal to 1 specifies that the spatial resampling optimization is an up-sampling operation.

**eoi\_privacy\_protection\_method\_idc**, when present, indicates the method / algorithm that was used to apply privacy protection optimization. eoi\_privacy\_protection\_method\_idc greater than 0 and ( eoi\_privacy\_protection\_method\_idc & bitMask ) not equal to 0 indicates that the method/ algorithm with the bitMask value in Table x3 has been used to apply privacy protection method. When eoi\_privacy\_protection\_method\_idc is equal to 0, the method/ algorithm used for privacy protection is unknown or determined by the application.

The value of eoi\_privacy\_protection\_method\_idc shall be in the range of 0 to 15, inclusive, in bitstreams conforming to this version of this Specification. Values of 16 to 255, inclusive, for eoi\_privacy\_protection\_method\_idc are reserved for future use by ITU-T | ISO/IEC and shall not be present in bitstreams conforming to this version of this Specification. When the value of eoi\_privacy\_protection\_method\_idc is in the range of 16 to 255, inclusive, decoders conforming to this version of this Specification shall ignore eoi\_privacy\_protection\_method\_idc.

**Table x3 – Definition of** **eoi\_privacy\_protection\_method\_idc**

|  |  |
| --- | --- |
| **bitMask** | **Interpretation** |
| 0x01 | Blurring; personal information is blurred to make it unidentifiable. |
| 0x02 | Replacing; personal information is replaced with something different from the original to make it unidentifiable. |
| 0x04 | Masking; personal information is masked so that it cannot be identified |
| 0x08 | Pixelation; personal information is pixelated to make it undiscernible |

**eoi\_privacy\_info\_type**, when present, indicates the types of protected information as specified in Table x4 where eoi\_privacy\_info\_type is greater than 0 and ( eoi\_privacy\_info\_type & bitMask ) not equal to 0 indicates that the information type with the bitMask value in Table x4 has been protected. When eoi\_privacy\_info\_type is equal to 0, an application-defined type of information has been protected. The value of eoi\_privacy\_info\_type shall be in the range of 0 to 7, inclusive, in bitstreams conforming to this version of this Specification. Values of 8 to 255, inclusive, for eoi\_privacy\_info\_type are reserved for future use by ITU-T | ISO/IEC and shall not be present in bitstreams conforming to this version of this Specification. When the value of eoi\_privacy\_info\_type is in the range of 8 to 255, inclusive, decoders conforming to this version of this Specification shall ignore eoi\_privacy\_info\_type.

**Table x4 – Definition of** **eoi\_privacy\_info\_type**

|  |  |
| --- | --- |
| **bitMask** | **Interpretation** |
| 0x01 | Information that identifies a person is protected. For example, the face of the person. |
| 0x02 | Information that can identify vehicles is protected. For example, the license plate of the vehicle. |
| 0x04 | Information that can infer locations is protected. For example text or images on signs. |

* 1. **Source picture timing information SEI message**
     1. **Source picture timing information SEI message syntax**

|  |  |
| --- | --- |
| source\_picture\_timing\_info( payloadSize ) { | **Descriptor** |
| **spti\_cancel\_flag** | u(1) |
| if( !spti\_cancel\_flag ) { |  |
| **spti\_persistence\_flag** | u(1) |
| **spti\_source\_timing\_equals\_output\_timing\_flag** | u(1) |
| if( !spti\_source\_timing\_equals\_output\_timing\_flag ) { |  |
| **spti\_source\_type\_present\_flag** | u(1) |
| if( spti\_source\_type\_present\_flag ) |  |
| **spti\_source\_type** | u(16) |
| **spti\_time\_scale** | u(32) |
| **spti\_num\_units\_in\_elemental\_interval** | u(32) |
| **spti\_direction\_flag** | u(1) |
| if( spti\_persistence\_flag ) |  |
| **spti\_max\_sublayers\_minus1** | u(3) |
| for( i = sptiMinTemporalSublayer; i  <=  spti\_max\_sublayers\_minus1; i++ ) { |  |
| **spti\_sublayer\_interval\_scale\_factor**[ i ] | ue(v) |
| **spti\_sublayer\_synthesized\_picture\_flag**[ i ] | u(1) |
| } |  |
| } |  |
| **}** |  |
| } |  |

* + 1. **Source picture timing information SEI message semantics**

The source picture timing information (SPTI) SEI message indicates the temporal distance between source pictures associated with the corresponding decoded output pictures prior to encoding. For example, for camera-captured content, the temporal distance between source pictures is the difference between the time at which an image sensor was exposed to produce a source picture associated with the current decoded picture and the time at which the image sensor was exposed to produce the source picture associated with a previous decoded picture in output order. The information provided by the SPTI SEI message pertains only for picture(s) starting from the picture in the current layer in the access unit that contains the SPTI SEI message and all subsequent pictures of the current layer in output order based on its persistence. When a frame packing arrangement SEI message with fp\_arrangement\_type equal to 5 (temporal interleaving of constituent pictures as coded pictures) applies to the decoded pictures, the source picture timing information applies to each constituent picture separately.

Use of this SEI message requires the definition of the following variable:

– A temporal sublayer identifier, denoted herein by TemporalId.

**spti\_cancel\_flag** equal to 1 indicates that the SPTI SEI message cancels the persistence of any previous SPTI SEI message in output order that applies to the current layer. spti\_cancel\_flag equal to 0 indicates that source picture timing information follows.

**spti\_persistence\_flag** specifies the persistence of the SPTI SEI message for the current layer.

spti\_persistence\_flag equal to 0 specifies that the SPTI SEI message applies to the current decoded picture only.

spti\_persistence\_flag equal to 1 specifies that the SPTI SEI message applies to the current decoded picture and persists for all subsequent pictures of the current layer in output order until one or more of the following conditions are true:

– A new CLVS of the current layer begins.

– The bitstream ends.

– A picture in the current layer in an AU associated with an SPTI SEI message is output that follows the current picture in output order.

**spti\_source\_timing\_equals\_output\_timing\_flag** equal to 1 indicates the timing of source pictures is the same as the timing of corresponding decoded output pictures. spti\_source\_timing\_equals\_output\_timing\_flag equal to 0 indicates the timing of source pictures might not be the same as the timing of corresponding decoded output pictures.

When spti\_source\_timing\_equals\_output\_timing\_flag is equal to 1 and a picture timing SEI message is present for the current picture, source picture timing could be determined from information conveyed in the picture timing SEI message.

**spti\_source\_type\_present\_flag** equal to 1 indicates the syntax element spti\_source\_type is present in the SEI message. spti\_source\_type\_present\_flag equal to 0 indicates the syntax element spti\_source\_type is not present in the SEI message.

**spti\_source\_type** indicates the timing relationship between source pictures and corresponding decoded output pictures as specified in Table X, where ( spti\_source\_type & bitMask ) not equal to 0 indicates that the timing relationship has the interpretation associated with the bitMask value in Table X. When spti\_source\_type is greater than 0 and ( spti\_source\_type & bitMask ) is equal to 0, the interpretation associated with the bitMask value is not applicable to the SPTI SEI message. When spti\_source\_type is equal to 0, the timing relationship may be specified by the application.

The value of spti\_source\_type shall be in the range of 0 to 127, inclusive, in bitstreams conforming to this edition of this document. Values of 128 to 255, inclusive, for spti\_source\_type are reserved for future use by ITU-T | ISO/IEC and shall not be present in bitstreams conforming to this edition of this document. Decoders conforming to this edition of this document shall ignore SPTI SEI messages with spti\_source\_type in the range of 128 to 255, inclusive.

**Table X – Interpretation of spti\_source\_type**

|  |  |
| --- | --- |
| **bitMask** | **Interpretation** |
| 0x01 | Slow motion: The absolute value of the temporal distance between consecutive source pictures is likely to be less than the temporal distance between corresponding decoded output pictures. |
| 0x02 | Sped-up motion: The absolute value of the temporal distance between consecutive source pictures is likely to be greater than the temporal distance between corresponding decoded output pictures. |
| 0x04 | High-speed imaging: The absolute value of the temporal distance between consecutive source pictures is likely to be less than 1/120 seconds. |
| 0x08 | Time-lapse imaging: The temporal distance between source pictures is likely to be greater than 1.001/24 seconds. |
| 0x10 | Temporal reversal: The absolute value of the temporal distance between consecutive source pictures is indicated to be negative (i.e., decoded pictures are output in reverse temporal order relative to the timing of the corresponding source pictures). |
| 0x20 | Still image / freeze frame: The temporal distance between source pictures is likely to be 0 (i.e., two or more decoded pictures are likely to represent the same source picture). |
| 0x40 | Sporadic or event-driven: The temporal distance between source pictures is likely to be non-constant. |

NOTE 1 – Some combinations of spti\_source\_type values may not be common and thus should not appear together. For example, the following combinations should not be simultaneously indicated in the value of spti\_source\_type: high-speed imaging and time-lapse imaging, slow motion and sped-up motion, still image / freeze frame and other source types.

**spti\_time\_scale** specifies the number of time units that pass in one second. The value of spti\_time\_scale shall not be equal to 0. For example, a time coordinate system that measures time using a 27 MHz clock has an spti\_time\_scale of 27 000 000.

**spti\_num\_units\_in\_elemental\_interval** specifies the number of time units of a clock operating at the frequency spti\_time\_scale Hz that corresponds to the indicated elemental source picture interval of consecutive pictures in output order in the CLVS. The value of spti\_num\_units\_in\_elemental\_interval shall not be equal to 0.

The indicated elemental source picture interval, also to be denoted by the variable ElementalSourcePictureInterval, in units of seconds, is equal to the quotient of spti\_num\_units\_in\_elemental\_interval divided by spti\_time\_scale. For example, to represent an elemental source picture interval equal to 0.04 seconds, spti\_time\_scale may be equal to 27 000 000 and spti\_num\_units\_in\_elemental\_interval may be equal to 1 080 000.

NOTE 2 – The method of indicating the elemental source picture interval is similar to that used for the timing and HRD parameters syntax used in several video coding standards such as Rec. ITU-T H.266 | ISO/IEC 23090-3, with spti\_time\_scale being similar to that syntax’s time\_scale and spti\_num\_units\_in\_elemental\_interval being similar to that syntax’s num\_units\_in\_tick, and thus the variable ElementalSourcePictureInterval being similar to the variable ClockTick in Rec. ITU-T H.266 | ISO/IEC 23090-3.

**spti\_direction\_flag** indicates the direction of the signalled source picture intervals. spti\_direction\_flag equal to 0 indicates that source picture intervals are expressed relative to previous output pictures. spti\_direction\_flag equal to 1 indicates that source picture intervals are expressed relative to future output pictures.

**spti\_max\_sublayers\_minus1** plus 1 specifies the maximum number of temporal sublayers for which picture interval scale factor (spti\_sublayer\_interval\_scale\_factor[ i ]) and synthesized flag (spti\_sublayer\_synthesized\_picture\_flag[ i ]) information is signalled. When spti\_max\_sublayers\_minus1 is not present, it is inferred to be equal to TemporalId. The value of spti\_max\_sublayers\_minus1 shall be equal to or greater than TemporalId of the SPTI SEI message.

The variable sptiMinTemporalSublayer is set to as follows:

– If spti\_persistence\_flag is equal 1, sptiMinTemporalSublayer is equal to 0.

– Otherwise, sptiMinTemporalSublayer is equal to spti\_max\_sublayers\_minus1.

**spti\_sublayer\_interval\_scale\_factor**[ i ], when present, specifies a scale factor used in determining the interval time between the source picture associated with the current picture and the source picture associated with the previous (if spti\_direction\_flag is equal to 0) or next (if spti\_direction\_flag is equal to 1) output picture with TemporalId less than or equal to i. The value 0 may be used to indicate that the source picture corresponding to the current decoded output picture is identical to the source picture corresponding to the previous decoded output picture with TemporalId less than or equal to i. The value of spti\_sublayer\_interval\_scale\_factor[ I ] shall be in the range of 0 to 232 − 2, inclusive.

The indicated source picture interval associated with an output picture having TemporalId equal to i, denoted by the variable SourcePictureInterval[ i ], in units of seconds, is derived as follows:

SourcePictureInterval[ i ] = ElementalSourcePictureInterval \* spti\_sublayer\_interval\_scale\_factor[ i ] \*  
( 1 − 2 \* temporalReversalFlag ) (8-X)

If spti\_direction\_flag is equal to 0, the indicated source picture interval is relative to the previous output picture with TemporalId less than or equal to i. Otherwise (spti\_direction\_flag is equal to 1), the indicated source picture interval is relative to the next output picture with TemporalId less than or equal to i.

If spti\_source\_type\_present\_flag is equal to 1, the variable temporalReversalFlag is equal to ( spti\_source\_type & 0x10 )? 1 : 0. Otherwise (i.e. if spti\_source\_type\_present\_flag is equal to 0), the variable temporalReversalFlag is equal to 0.

NOTE 3 –Since ElementalSourcePictureInterval is multiplied by spti\_sublayer\_interval\_scale\_factor[ i ] when calculating SourcePictureInterval[ i ], it is possible to represent the same value of SourcePictureInterval[ i ] in multiple ways by applying a scale factor to the value of spti\_time\_scale and applying the same scale factor to spti\_num\_units\_in\_elemental\_interval or spti\_sublayer\_interval\_scale\_factor[ i ]. There is no assumption that common scale factors have been removed or that the value of spti\_sublayer\_interval\_scale\_factor[ i ] is equal to 1 for the highest value of i. The reason to allow the same value to be represented in multiple ways is, at least in part, to allow spti\_time\_scale to be chosen to correspond with other timing-related elements used in the system environment, such as the clock rate of 27 MHz used in some multimedia communication systems.

**spti\_sublayer\_synthesized\_picture\_flag**[ i ], when present, equal to 1 indicates that decoded output pictures belonging to the ith temporal sublayer are synthesized and do not correspond to unmodified original source pictures. spti\_sublayer\_synthesized\_picture\_flag[ i ] equal to 0 provides no such indication. When not present, the value of spti\_sublayer\_synthesized\_picture\_flag[ i ] is inferred to be equal to 0.

NOTE 4 – When the TemporalId of an SPTI SEI message is greater than 0, and the SPTI SEI message persists for one or more pictures with lower TemporalId, an encoder can repeat the information of the SPTI SEI message by including it in one or more SPTI SEI messages with lower TemporalId, in order to avoid loss of information when pictures in temporal sublayer(s) are lost or removed.

* 1. **Object mask information SEI message**
     1. **Object mask information SEI message syntax**

|  |  |
| --- | --- |
| object\_mask\_info( payloadSize ) { | **Descriptor** |
| **omi\_cancel\_flag** | u(1) |
| if( !omi\_cancel\_flag ) { |  |
| **omi\_persistence\_flag** | u(1) |
| **omi\_num\_aux\_pic\_layer\_minus1** | ue(v) |
| **omi\_mask\_id\_length\_minus1** | ue(v) |
| **omi\_mask\_sample\_value\_length\_minus8** | ue(v) |
| **omi\_mask\_confidence\_info\_present\_flag** | u(1) |
| if( omi\_mask\_confidence\_info\_present\_flag ) |  |
| **omi\_mask\_confidence\_length\_minus1** | u(4) |
| **omi\_mask\_depth\_info\_present\_flag** | u(1) |
| if( omi\_mask\_depth\_info\_present\_flag ) |  |
| **omi\_mask\_depth\_length\_minus1** | u(4) |
| **omi\_mask\_label\_info\_present\_flag** | u(1) |
| if( omi\_mask\_label\_info\_present\_flag ) { |  |
| **omi\_mask\_label\_language\_present\_flag** | u(1) |
| if( omi\_mask\_label\_language\_present\_flag ) { |  |
| while( !byte\_aligned( ) ) |  |
| **omi\_bit\_equal\_to\_zero** | f(1) |
| **omi\_mask\_label\_language** | st(v) |
| } |  |
| } |  |
| for( i = 0; i < omi\_num\_aux\_pic\_layer; i++ ) { |  |
| **omi\_mask\_pic\_update\_flag**[ i ] | u(1) |
| if( omi\_mask\_pic\_update\_flag[ i ] ) { |  |
| **omi\_num\_mask\_in\_pic\_update**[ i ] | ue(v) |
| for( j = 0; j < omi\_num\_mask\_in\_pic\_update[ i ]; j++ ) { |  |
| **omi\_mask\_id**[ i ][ j ] | u(v) |
| **omi\_mask\_cancel**[ i ][ j ] | u(1) |
| if( !omi\_mask\_cancel[ i ][ j ] ) { |  |
| **omi\_aux\_sample\_value**[ i ][ j ] | u(v) |
| **omi\_mask\_bounding\_box\_present\_flag**[ i ][ j ] | u(1) |
| if( omi\_mask\_bounding\_box\_present\_flag[ i ][ j ] ) { |  |
| **omi\_mask\_top**[ i ][ j ] | u(16) |
| **omi\_mask\_left**[ i ][ j ] | u(16) |
| **omi\_mask\_width**[ i ][ j ] | u(16) |
| **omi\_mask\_height**[ i ][ j ] | u(16) |
| } |  |
| if( omi\_mask\_confidence\_info\_present\_flag ) |  |
| **omi\_mask\_confidence**[ i ][ j ] | u(v) |
| if( omi\_mask\_depth\_info\_present\_flag ) |  |
| **omi\_mask\_depth**[ i ][ j ] | u(v) |
| if( omi\_mask\_label\_info\_present\_flag ) { |  |
| while( !byte\_aligned( ) ) |  |
| **omi\_bit\_equal\_to\_zero** | f(1) |
| **omi\_mask\_label**[ i ][ j ] | st(v) |
| } |  |
| } |  |
| } |  |
| } |  |
| } |  |
| } |  |
| } |  |

* + 1. **Object mask information SEI message semantics**

The object mask information (OMI) SEI message provides object mask information of object mask picture in the auxiliary layer associated with the primary layer, referred to as the current primary layer, in which the SEI message is present. If the OMI SEI message is present, it shall be present in a primary layer. One primary layer can be associated with one or more auxiliary layers. The number of auxiliary layers associated with the current primary layer is equal to omiNumAuxLayer and the layer identifier of the j-th associated auxiliary layer is equal to omiAuxLayerId[ j ]. For each value of j in the range of 0 to omiNumAuxLayer − 1, inclusive, if omiAuxLayerId[ j ] is equal to sdi\_layer\_id[ i ], the value of sdi\_aux\_id[ i ] shall be equal to AUX\_OBJECT\_MASK, for any value of i in the range of 0 to sid\_max\_layers\_minus1, inclusive. When the SDI SEI message is not present in the current CVS, the OMI SEI message should be ignored.

Use of this SEI message requires the definition of the following variables:

– Bit depth BitDepthY for the luma sample array of the current primary picture.

– A cropped picture width and picture height in units of luma samples, denoted herein by CroppedWidth and CroppedHeight, respectively.

– A conformance cropping window left offset, ConfWinLeftOffset

– A conformance cropping window top offset, ConfWinTopOffset

– A chroma format indicator, denoted herein by ChromaFormatIdc, as described in clause 7.3.

The variables SubWidthC and SubHeightC are derived from ChromaFormatIdc as specified by Table 2.

The variable omiNumAuxLayer and omiAuxLayerId[ k ] are derived as follows.

omiNumAuxLayer = 0;  
for( i = 0; i <= sdi\_max\_layers\_minus1; i++ )  
 if( sdi\_aux\_id[ i ] = = AUX\_OBJECT\_MASK )  
 for( j = 0; j <= sdi\_num\_associated\_primary\_layers\_minus1[ i ]; j++ ) (100)  
 if( sdi\_layer\_id[ sdi\_associated\_primary\_layer\_idx[ i ][ j ] ]  = =  omiPrimaryLayerId ) {  
 omiAuxLayerId[ omiNumAuxLayer ] = sdi\_layer\_id[ i ]  
 omiNumAuxLayer++;  
 }

where omiPrimaryLayerId is the layer identifier of the current primary layer.

**omi\_cancel\_flag** equal to 1 indicates that the SEI message cancels the persistence of any previous object mask information SEI message in the same layer, if present, in output order. omi\_cancel\_flag equal to 0 indicates that object mask information follows.

**omi\_persistence\_flag** specifies the persistence of the object mask information provided in this SEI message. omi\_persistence\_flag equal to 0 specifies that the object mask information applies to the current picture only. omi\_persistence\_flag equal to 1 specifies that the object mask information applies to the current picture and all subsequent pictures in the same layer in output order until one or more of the following conditions are true:

– A new CLVS of the current layer begins.

– The bitstream ends.

– A picture in the current layer in a PU containing an object mask information SEI message is output that follows the current picture in output order.

When a CVS does not contain an SDI SEI message with sdi\_aux\_id[ i ] equal to AUX\_OBJECT\_MASK for at least one value of i, the OMI SEI message shall be ignored.

When an AU contains both an SDI SEI message with sdi\_aux\_id[ i ] equal to AUX\_OBJECT\_MASK for at least one value of i and an OMI SEI message, the SDI SEI message shall precede the OMI SEI message in decoding order.

**omi\_num\_aux\_pic\_layer\_minus1** plus 1 indicates the number of auxiliary layers associated with the current primary layer. It is a requirement of bitstream conformance that the value of omi\_num\_aux\_pic\_layer\_minus1 plus 1 shall be equal to omiNumAuxLayer.

**omi\_mask\_id\_length\_minus1** plus 1 specifies the length, in bits, of omi\_mask\_id[ i ][ j ] syntax elements. The value of omi\_mask\_id\_length\_minus1 shall be in the range of 0 to 15, inclusive.

**omi\_mask\_sample\_value\_length\_minus8** plus 8 specifies the length, in bits, of omi\_aux\_sample\_value[ i ][ j ] syntax elements. The value of omi\_mask\_sample\_value\_length\_minus8 shall be in the range of 0 to 8, inclusive. The value of omi\_mask\_sample\_value\_length\_minus8 plus 8 shall be less than or equal to BitDepthY.

**omi\_mask\_confidence\_info\_present\_flag** equal to 1 indicates that omi\_mask\_confidence[ i ][ j ] syntax elements are present. omi\_mask\_confidence\_info\_present\_flag equal to 0 indicates that omi\_mask\_confidence[ i ][ j ]syntax elements are not present.

**omi\_mask\_confidence\_length\_minus1** plus 1 specifies the length, in bits, of the omi\_mask\_confidence[ i ][ j ] syntax elements.

**omi\_mask\_depth\_info\_present\_flag** equal to 1 indicates that omi\_mask\_depth[ i ][ j ] syntax elements are present. omi\_mask\_depth\_info\_present\_flag equal to 0 indicates that omi\_mask\_depth[ i ][ j ]syntax elements are not present.

**omi\_mask\_depth\_length\_minus1** plus 1 specifies the length, in bits, of the omi\_mask\_depth[ i ][ j ] syntax elements.

It is a requirement of bitstream conformance that the value of omi\_num\_aux\_pic\_layer, omi\_mask\_id\_length\_minus1, omi\_mask\_sample\_value\_length\_minus8, omi\_mask\_confidence\_info\_present\_flag, omi\_mask\_confidence\_length\_minus1, if present, omi\_mask\_depth\_info\_present\_flag and omi\_mask\_depth\_length\_minus1, if present, shall be the same for all object\_mask\_info( ) syntax structures within a CLVS.

**omi\_mask\_label\_info\_present\_flag** equal to 1 indicates that omi\_mask\_label\_language\_present\_flag and omi\_mask\_label[ i ][ j ] syntax elements are present. omi\_mask\_label\_info\_present\_flag equal to 0 indicates that omi\_mask\_label\_language\_present\_flag and omi\_mask\_label[ i ][ j ] syntax elements are not present.

**omi\_mask\_label\_language\_present\_flag** equal to 1 indicates that omi\_mask\_label\_language syntax element is present. omi\_mask\_label\_language\_present\_flag equal to 0 indicates that omi\_mask\_label\_language syntax element is not present.

**omi\_bit\_equal\_to\_zero** shall be equal to 0.

**omi\_mask\_label\_language** contains a language tag as specified by IETF RFC 5646 followed by a null termination byte equal to 0x00. The length of the omi\_mask\_label\_language syntax element shall be less than or equal to 255 bytes, not including the null termination byte. When not present, the language of the label is unspecified.

**omi\_mask\_pic\_update\_flag**[ i ] equal to 1 indicates the object mask information of the object mask picture in the i-th auxiliary layer associated with the current primary layer may be updated. omi\_mask\_pic\_update\_flag[ i ] equal to 0 indicates there is no change to the mask information of the object mask picture in the i-th auxiliary layer associated with the current primary layer. When omi\_mask\_pic\_update\_flag[ i ] is equal to 0, the persistence mechanism is used, that is the object mask information is inherited from the last OMI SEI message present in the same layer in decoding order which signals the mask information of the object mask picture in the i-th auxiliary layer associated with the current primary layer.

**omi\_num\_mask\_in\_pic\_update**[ i ] specifies the number of object masks in object mask picture in the i-th auxiliary layer associated with the current primary layer. omi\_num\_mask\_in\_pic\_update [ i ] shall be in the range of 0 to ( 1 << ( omi\_mask\_id\_length\_minus1 + 1 ) ) − 1, inclusive.

**omi\_mask\_id**[ i ][ j ] indicates the identifier of the j-th object mask of the object mask picture in the i-th auxiliary layer associated with the current primary layer. The length of the omi\_mask\_id[ i ][ j ] syntax element is omi\_mask\_id\_length\_minus1 + 1 bits.

The variable maskId[ i ][ j ] specifying the identifier of the j-th object mask of the object mask picture in the i-th auxiliary layer associated with the current primary layer is derived as follows:

for( i = 0; i < omi\_num\_aux\_pic\_layer; i++ )  
 for( j = 0; j < omi\_num\_mask\_in\_pic\_update[ i ]; j++ ) (101)  
 maskId[ i ][ j ] = omi\_mask\_id[ i ][ j ] + ( 1<< ( omi\_mask\_id\_length\_minus1 + 1 ) ) \* i

Let omiA be an OMI SEI message that contains a mask object objectMaskA with maskId[ i0 ][ j0] and omiB be the first OMI SEI message that follows omiA in output order in the same CLVS that contains a mask object objectMaskB with maskId[ i1][ j1 ] and the value of maskId[ i0 ][ j0 ] is equal to maskId[ i1 ][ j1 ], objectMaskA and objectMaskB are object masks of the same object if both of the following condition are true:

– The value of omi\_mask\_cancel[ i0 ][ j0 ] in omiA is equal to 0.

– There is no OMI SEI message following omiA and preceding omiB in output order in the same CLVS with omi\_cancel\_flag equal to 1.

**omi\_mask\_cancel**[ i ][ j ] equal to 1 cancels the persistence scope of the j-th object mask of the object mask picture in the i-th auxiliary layer associated with the current primary picture. omi\_mask\_cancel[ i ][ j ] equal to 0 specifies that the information of the j-th object mask of the object mask picture in the i-th auxiliary layer associated with the current primary layer is signalled.

It is a requirement of bitstream conformance that when omi\_mask\_id[ i ][ j ] with a particular value is parsed for the first time in the current CLVS, the value of the corresponding omi\_mask\_cancel[ i ][ j ] shall be equal to 0.

**omi\_aux\_sample\_value**[ i ][ j ] specifies the value of the samples within the area of the j-th object mask of the object mask picture in the i-th auxiliary layer associated with the current primary layer.

**omi\_mask\_bounding\_box\_present\_flag**[ i ][ j ] equal to1 specifies that the syntax elements omi\_mask\_top[ i ][ j ], omi\_mask\_left[ i ][ j ], omi\_mask\_width[ i ][ j ], and omi\_mask\_height[ i ][ j ], are present. omi\_mask\_bounding\_box\_present\_flag[ i ][ j ] equal to 0 indicates syntax elements, omi\_mask\_top[ i ][ j ], omi\_mask\_left[ i ][ j ], omi\_mask\_width[ i ][ j ], and omi\_mask\_height[ i ][ j ], are not present.

**omi\_mask\_top**[ i ][ j ], **omi\_mask\_left**[ i ][ j ], **omi\_mask\_width**[ i ][ j ], and **omi\_mask\_height**[ i ][ j ] specify the coordinates of the top-left corner and the width and height, respectively, of the bounding box of the j-th object mask in the cropped decoded object mask picture in the i-th auxiliary layer associated with the current primary layer, relative to the conformance cropping window specified by the active SPS.

The value of omi\_mask\_left[ i ][ j ] shall be in the range of 0 to ( CroppedWidth / SubWidthC − 1 ), inclusive, CroppedWidth and SubWidthC being associated to the object mask picture in the i-th auxiliary layer associated with the current primary layer. When it is not present, the value of omi\_mask\_left[ i ][ j ] is inferred to be 0.

The value of omi\_mask\_top[ i ][ j ] shall be in the range of 0 to ( CroppedHeight / SubHeightC − 1 ), inclusive, CroppedHeight  and SubHeightC  being associated to the object mask picture in the i-th auxiliary layer associated with the current primary layer. When it is not present, the value of omi\_mask\_top[ i ][ j ] is inferred to be 0.

The value of omi\_mask\_width[ i ][ j ] shall be in the range of 0 to ( CroppedWidth / SubWidthC − omi\_mask\_left[ i ][ j ] ), inclusive. When it is not present, the value of omi\_mask\_width [ i ][ j ] is inferred to be ( CroppedWidth / SubWidthC − omi\_mask\_left[ i ][ j ] ).

The value of omi\_mask\_height[ i ][ j ] shall be in the range of 0 to ( CroppedHeight / SubHeightC − omi\_mask\_top[ i ][ j ] ), inclusive. When it is not present, the value of omi\_mask\_height [ i ][ j ] is inferred to be ( CroppedHeight / SubWidthC − omi\_mask\_top[ i ][ j ] ).

The identified object mask is within a bounding box containing luma samples with horizontal coordinates from SubWidthC \* ( ConfWinLeftOffset + omi\_mask\_left[ i ][ j ] ) to SubWidthC \* ( ConfWinLeftOffset + omi\_mask\_left[ i ][ j ] + omi\_mask\_width[ i ][ j ] ) − 1, inclusive, and vertical coordinates from SubHeightC \* ( ConfWinTopOffset + omi\_mask\_top[ i ][ j ] ) to SubHeightC \* ( ConfWinTopOffset + omi\_mask\_top[ i ][ j ] + omi\_mask\_height[ i ][ j ] ) − 1, inclusive.

Variable pI[ i ] [ x ][ y ] is the decoded value of the sample at the relative sample location (x, y) in the cropped object mask picture in the i-th auxiliary layer associated with the current primary layer. The following process is to determine mask area in a auxiliary picture.

for( i = 0; i < omi\_num\_aux\_pic\_layer; i++ )  
 for( j = 0; j < omi\_num\_mask\_in\_pic\_update[ i ]; j++ )   
  if( !omi\_mask\_cancel[ i ][ j ] )  
 if( pI[ i ][ x ][ y ]  = =  omi\_aux\_sample\_value [ i ][ j ]  
 && x  >=  omi\_mask\_left[ i ][ j ]  
 && x < omi\_mask\_left[ i ][ j ] + omi\_mask\_width[ i ][ j ] (102)  
 && y  >=  omi\_mask\_top[ i ][ j ]  
 && y < omi\_mask\_top[ i ][ j ] + omi\_mask\_height[ i ][ j ] )  
 the sample at location (x, y) is associated with the object mask with the identifier of maskId[ i ][ j ]

**omi\_mask\_confidence**[ i ][ j ] specifies the degree of confidence associated with the j-th object mask of the object mask picture in the i-th auxiliary layer associated with the current primary layer, in units of 2-( omi\_mask\_confidence\_length\_minus1 + 1 ), such that a higher value of omi\_mask\_confidence[ i ][ j ] indicates a higher degree of confidence. The length of the omi\_mask\_confidence[ i ][ j ] syntax element is omi\_mask\_confidence\_length\_minus1 + 1 bits.

**omi\_mask\_depth**[ i ][ j ] specifies the object depth associated with the j-th object mask of the object mask picture in the i-th auxiliary layer associated with the current primary layer. A smaller value of omi\_mask\_depth indicates a shorter distance to the object. The length of the omi\_mask\_depth[ i ][ j ] syntax element is omi\_mask\_depth\_length\_minus1 + 1 bits.

**omi\_mask\_label**[ i ][ j ] specifies the contents of the label associated with the j-th object mask of the object mask picture in the i-th auxiliary layer associated with the current primary layer. The length of the omi\_mask\_label[ i ][ j ] syntax element shall be less than or equal to 255 bytes, not including the null termination byte.

* 1. **Modality information SEI message**
     1. **Modality information SEI message syntax**

|  |  |
| --- | --- |
| modality\_info( payloadSize ) { | **Descriptor** |
| **mi\_modality\_info\_cancel\_flag** | u(1) |
| if( !mi\_modality\_info\_cancel\_flag ) { |  |
| **mi\_modality\_info\_persistence\_flag** | u(1) |
| **mi\_modality\_type** | u(5) |
| **mi\_spectrum\_range\_present\_flag** | u(1) |
| if( mi\_spectrum\_range\_present\_flag ) { |  |
| **mi\_min\_wavelength\_mantissa** | u(11) |
| **mi\_min\_wavelength\_exponent\_plus15** | u(5) |
| **mi\_max\_wavelength\_mantissa** | u(11) |
| **mi\_max\_wavelength\_exponent\_plus15** | u(5) |
| } |  |
| **mi\_modality\_type\_extension\_bits** | ue(v) |
| if( mi\_modality\_type\_extension\_bits > 0 ) |  |
| **mi\_reserved\_modality\_type\_extension** | u(v) |
| } |  |
| } |  |

* + 1. **Modality information SEI message semantics**

The modality information SEI message provides information about the source of optical radiation (such as visible light, infrared, or ultraviolet) used for generating the associated pictures and the wavelength of the spectrum band. As pictures of different modality types serve different purposes, the information conveyed in this SEI message can be used by a receiver to determine the purpose of the associated pictures.

NOTE 1 – The interpretations of mi\_modality\_type and the wavelength of the spectrum band associated with mi\_modality\_type are specified by reference to the division of optical radiation specified in ISO 20473:2007.

**mi\_modality\_info\_cancel\_flag** equal to 1 indicates that the SEI message cancels the persistence of any previous modality information SEI message in output order. mi\_modality\_info\_cancel\_flag equal to 0 indicates that modality information follows.

**mi\_modality\_info\_persistence\_flag** specifies the persistence of the modality information SEI message for the current layer.

mi\_modality\_info\_persistence\_flag equal to 0 specifies that the modality information SEI message applies to the current decoded picture only.

mi\_modality\_info\_persistence\_flag equal to 1 specifies that the modality information SEI message applies to the current decoded picture and persists for all subsequent pictures of the current layer in output order until one or more of the following conditions are true:

– A new CLVS of the current layer begins.

– The bitstream ends.

– A picture in the current layer in an AU associated with a modality information SEI message is output that follows the current picture in output order.

**mi\_modality\_type** indicates the type of modality of the associated pictures as specified in Table 1. When not present, the value of mi\_modality\_type is inferred to be equal to 0, denoting that the modality type of the picture is unknown or unspecified or determined by other means not specified in this Specification. When mi\_modality\_type is equal to 2 or 3, decoders shall ignore vui\_colour\_primaries, vui\_transfer\_characteristics, vui\_matrix\_coeffs, and vui\_full\_range\_flag indicated in the VUI parameters

**Table 1 – Mapping of mi\_modality\_type to the type of picture modalities**

|  |  |
| --- | --- |
| **mi\_modality\_type** | **type of picture modality** |
| 0 | Unspecified |
| 1 | Visible Picture |
| 2 | Infrared Picture |
| 3 | Ultraviolet Picture |
| 4..31 | Reserved for future use |

NOTE 2 – When a reserved value of mi\_modality\_type is taken into use in the future by ITU-T | ISO/IEC, the syntax of this SEI message could be extended with syntax elements whose presence is conditioned by mi\_modality\_type being equal to that value or any one of a set of values including that value.

**mi\_spectrum\_range\_present\_flag** equal to 1 specifies that the spectrum band of the optical radiation wavelength represented by the associated pictures is present in the modality information SEI message. mi\_spectrum\_range\_present\_flag equal to 0 specifies that the spectrum band of the optical radiation wavelength represented by the associated pictures is not present in the modality information SEI message.

**mi\_min\_wavelength\_mantissa** specifies the mantissa part of the minimum wavelength indicating the spectral band of optical radiation represented by the associated pictures. When mi\_min\_wavelength\_mantissa is equal to 0 or is not present, the minimum wavelength indicating the spectral band of optical radiation represented by the associated pictures is unknown or unspecified or determined by other means not specified in this Specification

**mi\_min\_wavelength\_exponent\_plus15** minus 15specifies the exponent part of the minimum wavelength indicating the spectral band of optical radiation represented by the associated pictures. When mi\_min\_wavelength\_mantissa is not present or is equal to 0, decoders shall ignore the value of mi\_min\_wavelength\_exponent\_plus15.

The value of the minimum wavelength indicating the spectral band of optical radiation represented by the associated pictures is derived as follows:

MinWavelength = mi\_min\_wavelength\_mantissa \* 10 mi\_min\_wavelength\_exponent\_plus15 − 15

**mi\_max\_wavelength\_mantissa** specifies the mantissa part of the maximum wavelength indicating the spectral band of optical radiation represented by the associated pictures. When mi\_max\_wavelength\_mantissa is equal to 0 or is not present, the maximum wavelength indicating the spectral band of optical radiation represented by the associated pictures is unknown or unspecified or determined by other means not specified in this Specification.

**mi\_max\_wavelength\_exponent\_plus15** minus 15specifies the exponent part of the maximum wavelength indicating the spectral band of optical radiation represented by the associated pictures. When mi\_max\_wavelength\_mantissa is not present or is equal to 0, decoders shall ignore the value of mi\_max\_wavelength\_exponent\_plus15.

The value of the maximum wavelength indicating the spectral band of optical radiation represented by the associated pictures is derived as follows:

MaxWavelength = mi\_max\_wavelength\_mantissa \* 10 mi\_max\_wavelength\_exponent\_plus15 − 15

MinWavelength and MaxWavelength are in units of metres.

**mi\_modality\_type\_extension\_bits** equal to 0 specifies that mi\_reserved\_modality\_type\_extension is not present. mi\_modality\_type\_extension\_bits greater than 0 specifies the length, in bits, of mi\_reserved\_modality\_type\_extension.

The value of mi\_modality\_type\_extension\_bits shall be in the range of 0 to 2 048, inclusive. Values in the range of 1 to 2 048, inclusive, for mi\_modality\_type\_extension\_bits are reserved for future use by ITU-T | ISO/IEC and shall not be present in bitstreams conforming to this version of this document. Decoders conforming to this version of this document shall allow any value of mi\_modality\_type\_extension\_bits in the range of 0 to 2 048, inclusive, to appear in the syntax.

**mi\_reserved\_modality\_type\_extension** shall not be present in bitstreams conforming to this version of this document. However, decoders conforming to this version of this document shall allow the presence of mi\_reserved\_modality\_type\_extension in the syntax, but ignore the value. When present, the length, in bits, of mi\_reserved\_modality\_type\_extension is equal to mi\_modality\_type\_extension\_bits.

* 1. **Text description information SEI message**
     1. **Text description information SEI message syntax**

|  |  |
| --- | --- |
| text\_description\_info( payloadSize ) { | **Descriptor** |
| **tdi\_descr\_purpose** | u(8) |
| **tdi\_purpose\_cancel\_flag** | u(1) |
| if( !tdi\_cancel\_by\_purpose\_flag ) { |  |
| **tdi\_descr\_id** | u(13) |
| **tdi\_id\_cancel\_flag** | u(1) |
| if( !tdi\_id\_cancel\_flag ) { |  |
| **tdi\_persistence\_flag** | u(1) |
| **tdi\_num\_strings\_minus1** | u(8) |
| for( i = 0; i <= tdi\_num\_strings\_minus1; i++ ) { |  |
| **tdi\_descr\_string\_lang**[ i ] | st(v) |
| **tdi\_descr\_string**[ i ] | st(v) |
| } |  |
| } |  |
| } |  |
| } |  |

* + 1. **Text description information SEI message semantics**

The text description information SEI message provides text description about one or more pictures.

**tdi\_descr\_purpose** indicates the purpose of the text description SEI as specified in Table xx. The value of text\_descr\_purpose shall be in the range of 0 to 5, inclusive. Values in the range of 6 to 255, inclusive, for text\_descr\_purpose are reserved for future use by ITU-T | ISO/IEC and shall not be present in bitstreams conforming to this version of this Specification. Decoders conforming to this version of this Specification shall allow any value of text\_descr\_purpose in the range of 0 to 255, inclusive.

**Table xx – Definition of tdi\_descr\_purpose**

|  |  |
| --- | --- |
| **Value** | **Interpretation** |
| 0 | Application defined |
| 1 | Copyright information |
| 2 | AI marking information |
| 3 | General comment information |
| 4 | Content advisory rating information conforming to US and Canadian Rating Region Tables (RRT) |
| 5 | Tag URI for identifying the bitstream |
| 6 | Encoder description |
| 6..255 | Reserved |

**tdi\_purpose\_cancel\_flag** equal to 1 indicates that the text description information SEI message cancels the persistence of any previous text description information SEI message with the same tdi\_descr\_purpose in output order that applies to the current layer. tdi\_purpose\_cancel\_flagequal to 0 indicates that text description information follows.

**tdi\_descr\_id** indicates the identifier value of this text description information SEI message. The value of tdi\_descr\_id shall be in the range of 1 to 8 191, inclusive. Value 0 and values 8064-8191 are reserved. Text description SEI messages with different values for tdi\_descr\_purpose use separate values spaces for tdi\_descr\_id.

**tdi\_id\_cancel\_flag** equal to 1 indicates that the text description information SEI message cancels the persistence of any previous text description information SEI message with the same tdi\_descr\_id and tdi\_descr\_purpose values as those in the current SEI in output order that applies to the current layer. tdi\_id\_cancel\_flagequal to 0 indicates that text description information syntax elements (tdi\_persistence\_flag, tdi\_num\_strings\_minus1, tdi\_descr\_string\_lang[ i ], tdi\_descr\_string[ i ] ) follow.

**tdi\_persistence\_flag** specifies the persistence of the text description information SEI message with identifier equal to tdi\_descr\_id and purpose equal to tdi\_descr\_purpose for the current layer.

tdi\_persistence\_flag equal to 0 specifies that the text description information SEI message with identifier equal to tdi\_descr\_id and purpose equal to tdi\_descr\_purpose applies to the current decoded picture only.

tdi\_persistence\_flag equal to 1 specifies that the text description information SEI message with identifier equal to tdi\_descr\_id and purpose equal to tdi\_descr\_purpose applies to the current decoded picture and persists for all subsequent pictures of the current layer in output order until one or more of the following conditions are true:

– A new CLVS of the current layer begins.

– The bitstream ends.

– A picture in the current layer in an AU associated with a text description information SEI message with the same values of tdi\_descr\_id and tdi\_descr\_purpose is output that follows the current picture in output order.

**tdi\_num\_strings\_minus1** plus 1 indicates the number of entries for tdi\_descr\_string\_lang[ i ] and tdi\_descr\_string[ i ] that follow.

**tdi\_descr\_string\_lang**[ i ]specifies the language of the tdi\_descr\_string[ i ]. The language of the tdi\_descr\_string[ i ] shall be given by a language tag as defined by IETF RFC 5646. The length of tdi\_descr\_string\_lang[ i ] corresponding to the stringLength variable specified in clause 6.3 for the st(v) parsing process shall be in the range of 0 to 49 bytes, inclusive.

The value of tdi\_descr\_string\_lang[ m ] shall not be equal to the value of tdi\_descr\_string\_lang[ n ] when m is not equal to n, for any values of m and n in the range from 0 to tdi\_num\_strings\_minus1, inclusive.

**tdi\_descr\_string**[ i ] specifies i-th text description information string whose value is interpreted as specified by the tdi\_descr\_purpose.

When tdi\_descr\_purpose is equal to 0, the interpretation of what information is conveyed in the tdi\_descr\_string is application-defined.

When tdi\_descr\_purpose is equal to 1, the tdi\_descr\_string[ i ] specifies copyright information that pertains to the picture(s) in the persistence scope of this SEI message.

When tdi\_descr\_purpose is equal to 2, the tdi\_descr\_string[ i ] specifies, when not a null string, AI marking information that pertains to the picture(s) within the persistence scope of this SEI message.

NOTE: When tdi\_descr\_purpose is equal to 2 the string can contain information about machine-learning-based processing, intended use of the decoded pictures, or other aspects relevant to the associated pictures.

When tdi\_descr\_purpose is equal to 3, the tdi\_descr\_string[ i ] specifies a general text label description that pertains to the picture(s) in the persistence scope of this SEI message.

When tdi\_descr\_purpose is equal to 4, the tdi\_descr\_string[ i ] shall specify content advisory rating information conforming to US and Canadian Rating Region Tables (RRT) as defined by CTA-766-D that pertains to the picture(s) in the persistence scope of this SEI message.

When tdi\_descr\_purpose is equal to 5, tdi\_descr\_string[ i ] shall contain a tag URI with syntax and semantics as specified in IETF RFC 4151 identifying the CLVS.

When tdi\_descr\_purpose is equal to 6, the tdi\_descr\_string[ i ] specifies a description of the encoder used to produce the coded picture(s) in the persistence scope of this SEI.

* 1. **Generative face video SEI message**
     1. **Generative face video SEI message syntax**

|  |  |
| --- | --- |
| generative\_face\_video ( payloadSize ) { | **Descriptor** |
| **gfv\_id** | ue(v) |
| **gfv\_cnt** | ue(v) |
| if( gfv\_cnt  = =  0 ) |  |
| **gfv\_base\_pic\_flag** /\* indicate if the current decoded output picture is a base picture \*/ | u(1) |
| if( gfv\_base\_pic\_flag ) { /\* specify TranslatorNN( ) \*/ |  |
| **gfv\_nn\_present\_flag** | u(1) |
| if( gfv\_nn\_present\_flag ) { |  |
| **gfv\_nn\_mode\_idc** | ue(v) |
| if( gfv\_nn\_mode\_idc  = =  1 ) { |  |
| while( !byte\_aligned( ) ) |  |
| **gfv\_nn\_alignment\_zero\_bit\_a** | u(1) |
| **gfv\_nn\_tag\_uri** | st(v) |
| **gfv\_nn\_uri** | st(v) |
| } |  |
| } |  |
| **gfv\_chroma\_key\_info\_present\_flag** | u(1) |
| if( gfv\_chroma\_key\_info\_present\_flag ) { |  |
| for( c = 0; c < 3; c++ ) { |  |
| **gfv\_chroma\_key\_value\_present\_flag**[ c ] | u(1) |
| if( gfv\_chroma\_key\_value\_present\_flag[ c ] ) |  |
| **gfv\_chroma\_key\_value**[ c ] | u(8) |
| } |  |
| for( i = 0; i < 2; i++ ) { |  |
| **gfv\_chroma\_key\_thr\_present\_flag**[ i ] | u(1) |
| if( gfv\_chroma\_key\_thr\_present\_flag[ i ] ) |  |
| **gfv\_chroma\_key\_thr\_value**[ i ] | ue(v) |
| } |  |
| } |  |
| } else |  |
| **gfv\_drive\_pic\_fusion\_flag** /\* indicate if DrivePicture is input to GenerativeNN( ) \*/ | u(1) |
| **gfv\_low\_confidence\_face\_parameter\_flag** | u(1) |
| **gfv\_coordinate\_present\_flag** | u(1) |
| if( gfv\_coordinate\_present\_flag ) { |  |
| **gfv\_kps\_pred\_flag** | u(1) |
| if( gfv\_base\_pic\_flag  | |  !gfv\_kps\_pred\_flag ) { |  |
| **gfv\_coordinate\_precision\_factor\_minus1** | ue(v) |
| **gfv\_num\_kps\_minus1** | ue(v) |
| **gfv\_coordinate\_z\_present\_flag** | u(1) |
| if(gfv\_coordinate\_z\_present\_flag ) |  |
| **gfv\_coordinate\_z\_max\_value\_minus1** | ue(v) |
| } |  |
| for( i = 0; i  <=  gfv\_num\_kps\_minus1; i++ ) { |  |
| if( !gfv\_kps\_pred\_flag ) { |  |
| **gfv\_coordinate\_x\_abs**[ i ] | ue(v) |
| if( gfv\_coordinate\_x\_abs[ i ] > 0 ) |  |
| **gfv\_coordinate\_x\_sign\_flag**[ i ] | u(1) |
| **gfv\_coordinate\_y\_abs**[ i ] | ue(v) |
| if( gfv\_coordinate\_y\_abs[ i ] > 0 ) |  |
| **gfv\_coordinate\_y\_sign\_flag**[ i ] | u(1) |
| if( gfv\_coordinate\_z\_present\_flag > 0 ) { |  |
| **gfv\_coordinate\_z\_abs**[ i ] | ue(v) |
| if( gfv\_coordinate\_z\_abs[ i ] > 0 ) |  |
| **gfv\_coordinate\_z\_sign\_flag**[ i ] | u(1) |
| } |  |
| } else { |  |
| **gfv\_coordinate\_dx\_abs**[ i ] | ue(v) |
| if( gfv\_coordinate\_dx\_abs[ i ] > 0 ) |  |
| **gfv\_coordinate\_dx\_sign\_flag**[ i ] | u(1) |
| **gfv\_coordinate\_dy\_abs**[ i ] | ue(v) |
| if( gfv\_coordinate\_dy\_abs[ i ] > 0 ) |  |
| **gfv\_coordinate\_dy\_sign\_flag**[ i ] | u(1) |
| if( gfv\_coordinate\_z\_present\_flag ) { |  |
| **gfv\_coordinate\_dz\_abs**[ i ] | ue(v) |
| if( gfv\_coordinate\_dz\_abs[ i ] > 0 ) |  |
| **gfv\_coordinate\_dz\_sign\_flag**[ i ] | u(1) |
| } |  |
| } |  |
| } |  |
| } |  |
| **gfv\_matrix\_present\_flag** | u(1) |
| if(gfv\_matrix\_present\_flag ) { |  |
| if( !gfv\_base\_pic\_flag ) |  |
| **gfv\_matrix\_pred\_flag** | u(1) |
| if( gfv\_base\_pic\_flag  | |  !gfv\_matrix\_pred\_flag ) { |  |
| **gfv\_matrix\_element\_precision\_factor\_minus1** | ue(v) |
| **gfv\_num\_matrix\_types\_minus1** | ue(v) |
| for( i = 0; i  <=  num\_matrix\_types\_minus1; i++ ) { |  |
| **gfv\_matrix\_type\_idx**[ i ] | u(6) |
| if( gfv\_matrix\_type\_idx[ i ]  = =  0  | |  gfv\_matrix\_type\_idx[ i ]  = =  1 ) { |  |
| if( gfv\_coordinate\_present\_flag ) |  |
| **gfv\_num\_matrices\_equal\_to\_num\_kps\_flag**[ i ] | u(1) |
| if(!gfv\_num\_matrices\_equal\_to\_num\_kps\_flag[ i ] ) |  |
| **gfv\_num\_matrices\_info**[ i ] | ue(v) |
| } else if( gfv\_matrix\_type\_idx[ i ]  = = 2  | |  gfv\_matrix\_type\_idx[ i ]  = =  3  | |  gfv\_matrix\_type\_idx[ i ]  >=  7 ) { |  |
| if( gfv\_matrix\_type\_idx[ i ]  >=  7 ) |  |
| **gfv\_num\_matrices\_minus1**[ i ] | ue(v) |
| **gfv\_matrix\_width\_minus1**[ i ] | ue(v) |
| **gfv\_matrix\_height\_minus1**[ i ] | ue(v) |
| } else if( gfv\_matrix\_type\_idx[ i ]  >=  4  &&  gfv\_matrix\_type\_idx[ i ]  <=  6  &&  !gfv\_coordinate\_present\_flag ) |  |
| **gfv\_matrix\_for\_3D\_space\_flag**[ i ] | u(1) |
| } |  |
| } |  |
| for( i = 0; i  <=  num\_matrix\_types\_minus1; i++ ) { |  |
| for( j = 0; j < numMatrices[ i ]; j++ ) |  |
| for( k = 0; k < matrixHeight[ i ]; k++ ) |  |
| for( m = 0; m <matrixWidth[ i ]; m++ ) { |  |
| if( !gfv\_matrix\_pred\_flag ) { |  |
| **gfv\_matrix\_element\_int**[ i ][ j ][ k ][ m ] | ue(v) |
| **gfv\_matrix\_element\_dec**[ i ][ j ][ k ][ m ] | u(v) |
| if( gfv\_matrix\_element\_int[ i][ j ][ k ][ m ]  | |  gfv\_matrix\_element\_dec[ i ][ j ][ k ][ m ] ) |  |
| **gfv\_matrix\_element\_sign\_flag**[ i ][ j ][ k ][ m ] | u(1) |
| } else { |  |
| **gfv\_matrix\_delta\_element\_int**[ i ][ j ][ k ][ m ] | ue(v) |
| **gfv\_matrix\_delta\_element\_dec**[ i ][ j ][ k ][ m ] | ue(v) |
| if( gfv\_matrix\_delta\_element\_int[ i][ j ][ k ][ m ]  | |  gfv\_matrix\_delta\_element\_dec[ i ][ j ][ k ][ m ] ) |  |
| **gfv\_matrix\_delta\_element\_sign\_flag**[ i ][ j ][ k ][ m ] | u(1) |
| } |  |
| } |  |
| } |  |
| } |  |
| if( gfv\_nn\_present\_flag ) |  |
| if( gfv\_nn\_mode\_idc  = =  0 ) { |  |
| while( !byte\_aligned( ) ) |  |
| **gfv\_nn\_alignment\_zero\_bit\_b** | u(1) |
| for( i = 0; more\_data\_in\_payload( ); i++ ) |  |
| **gfv\_nn\_payload\_byte**[ i ] | b(8) |
| } |  |
| } |  |

* + 1. **Generative face video SEI message semantics**

The generative face video (GFV) SEI message carries facial parameters and indicates a facial parameter translator network, denoted as TranslatorNN( ), that may be used to convert various formats of facial parameters signalled in the SEI message into a particular facial parameter format supported by the decoding system. A face picture generator neural network, denoted as GenerativeNN( ), may be used to generate output pictures using the facial parameters translated into the particular format and previously decoded output pictures.

When a picture unit contains a GFV SEI message with a particular gfv\_id value and gfv\_base\_pic\_flag equal to 1, the picture in the picture unit is referred to as a base picture for that particular gfv\_id value.

When a picture unit contains a GFV SEI message with a particular gfv\_id value and gfv\_base\_pic\_flag equal to 0, and the picture unit does not contain a GFV SEI message with that particular gfv\_id value and gfv\_base\_pic\_flag equal to 1, the picture in the picture unit is referred to as a driving picture for that particular gfv\_id value.

When a picture unit contains a GFV SEI message with a particular gfv\_id value, gfv\_base\_pic\_flag equal to 0, and gfv\_drive\_pic\_fusion\_flag equal to 1, and the picture unit does not contain a GFV SEI message with that particular gfv\_id value and gfv\_base\_pic\_flag equal to 1, the picture in the picture unit is referred to as a fusion picture for that particular gfv\_id value.

NOTE 1 – Facial parameters could be determined from source pictures prior to encoding.

NOTE 2 – Previously decoded output pictures input to GenerativeNN( ) may be a base picture (a decoded output picture that provides the reference texture from which the face pictures may be generated) and, optionally, a picture that can be fused by GenerativeNN( ) to improve background texture and facial details. When the current picture is not a base picture, the GFV SEI message may be used to generate a face picture based on the previously decoded base picture, the facial parameters conveyed by the GFV SEI message, and, optionally, the current decoded picture for fusion purpose.

Use of this SEI message requires the definition of the following variables:

* Input and output picture width and height in units of luma samples, denoted herein by CroppedWidth and CroppedHeight, respectively.
* Luma sample array baseCroppedYPic and chroma sample arrays baseCroppedCbPic and baseCroppedCrPic for a decoded output picture, denoted as BasePicture, corresponding to a source base picture.
* Luma sample array driveCroppedYPic and chroma sample arrays driveCroppedCbPic and driveCroppedCrPic for a decoded output picture, denoted as DrivePicture, corresponding to a source driving picture.
* Bit depth BitDepthY for the luma sample array of the input and output pictures.
* Bit depth BitDepthC for the chroma sample arrays, if any, of the input and output pictures.
* A chroma format indicator, denoted herein by ChromaFormatIdc, as described in subclause ‎7.3.

The variables SubWidthC and SubHeightC are derived from ChromaFormatIdc as specified by Table 2.

**gfv\_id** contains an identifying number that may be used to identify face feature information and specify a neural network that may be used as TranslatorNN( ). The value of gfv\_id shall be in the range of 0 to 232 − 2, inclusive. Values of gfv\_id from 256 to 511, inclusive, and from 231 to 232 − 2, inclusive, are reserved for future use by ITU-T | ISO/IEC. Decoders conforming to this edition of this document encountering a GFV SEI message with gfv\_id in the range of 256 to 511, inclusive, or in the range of 231 to 232 − 2, inclusive, shall ignore the SEI message.

**gfv\_cnt** specifies a GFV SEI message instance count value for this gfv\_id value within a picture unit.

The gfv\_cnt of the first GFV SEI message, in decoding order, with a particular value of gfv\_id within a picture unit shall be equal to 0. When gfv\_cnt assigned to currGfvCnt is greater than 0, a GFV SEI message with the same gfv\_id value and gfv\_cnt equal to currGfvCnt − 1 shall be present in the same picture unit and precede the current GFV SEI message in decoding order.

The value of gfv\_cnt shall be in the range of 0 to 65 535, inclusive.

**gfv\_base\_pic\_flag** equal to 1 indicates that the current decoded output picture corresponds to a base picture. gfv\_base\_pic\_flag equal to 0 indicates that the current decoded output picture does not correspond to a base picture or this SEI message does not specify syntax elements for a base picture. When gfv\_base\_pic\_flag is not present, it is inferred to be equal to 0.

When a GFV SEI message is the first GFV SEI message, in decoding order, that has a particular gfv\_id value within the current CLVS, the value of gfv\_base\_pic\_flag shall be equal to 1.

When a GFV SEI message with a particular gfv\_id value has gfv\_base\_pic\_flag equal to 1, the base picture for that particular gfv\_id value, which is the current cropped decoded picture, remains valid for the current decoded picture and all subsequent decoded pictures of the current layer, in output order, until the end of the current CLVS or up to but excluding the decoded picture that is within the current CLVS, follows the current decoded picture in output order, and is associated with a GFV SEI message having that particular gfv\_id value and gfv\_base\_pic\_flag equal to 1, whichever is earlier.

**gfv\_nn\_present\_flag** equal to 1 indicates that a neural network that may be used as a TranslatorNN( ) is contained or indicated by the SEI message. gfv\_nn\_present\_flag equal to 0 indicates that a neural network that may be used as a TranslatorNN( ) is not contained or indicated by the SEI message. When gfv\_nn\_present\_flag is not present, it is inferred to be 0.

When gfv\_nn\_present\_flag is equal to 0 and TranslatorNN is referenced in the semantics of the GFV SEI message, the following constraint applies:

– If gfv\_cnt is equal to 0, there shall be at least one GFV SEI message present in a preceding picture unit in output order in the current CLVS and having the same value of gfv\_id as that in the current GFV SEI message and gfv\_nn\_present\_flag equal to 1.

– Otherwise (gfv\_cnt is greater than 0), there shall be at least one GFV SEI message that is present in either the current picture unit or a preceding picture unit in output order in the current CLVS and has the same value of gfv\_id as that in the current GFV SEI message and gfv\_nn\_present\_flag equal to 1.

When gfv\_nn\_present\_flag is equal to 0 and TranslatorNN is referenced in the semantics of this SEI message, the following applies for deriving the applicable TranslatorNN:

– If gfv\_cnt is greater than 0 and there exists one or more preceding GFV SEI messages in decoding order in the current picture unit that has the same value of gfv\_id as that in the current GFV SEI message and gfv\_nn\_present\_flag equal to 1, the applicable TranslatorNN is defined by the last preceding GFV SEI message in decoding order in the current picture unit that has the same value of gfv\_id as that in the current GFV SEI message and gfv\_nn\_present\_flag equal to 1.

– Otherwise, the applicable TranslatorNN is defined by a GFV SEI message that is present in the last preceding picture unit puB in output order in the current CLVS that has the same value of gfv\_id as the current GFV SEI message and gfv\_nn\_present\_flag equal to 1. When there are multiple such GFV SEI messages present in the picture unit puB that have the same value of gfv\_id as the current GFV SEI message and gfv\_nn\_present\_flag equal to 1, the applicable TranslatorNN is defined by the last of such GFV SEI messages in decoding order.

**gfv\_nn\_mode\_idc**, **gfv\_nn\_alignment\_zero\_bit\_a**, **gfv\_nn\_tag\_uri**, **gfv\_nn\_uri**, **gfv\_nn\_alignment\_zero\_bit\_b**, and **gfv\_nn\_payload\_byte**[ i ] specify a neural network that may be used as a TranslatorNN( ). gfv\_nn\_mode\_idc, gfv\_nn\_alignment\_zero\_bit\_a, gfv\_nn\_tag\_uri, gfv\_nn\_uri, gfv\_nn\_alignment\_zero\_bit\_b, and gfv\_nn\_payload\_byte[ i ] have the same syntax and semantics as nnpfc\_base\_flag, nnpfc\_mode\_idc, nnpfc\_alignment\_zero\_bit\_a, nnpfc\_tag\_uri, nnpfc\_uri, nnpfc\_alignment\_zero\_bit\_b, and nnpfc\_payload\_byte[ i ], respectively.

The GFV SEI messages that are present in the same picture unit and have the same values of gfv\_id and gfv\_cnt shall have the same SEI payload content.

**gfv\_chroma\_key\_info\_present\_flag** equal to 1 indicates that the syntax elements gfv\_chroma\_key\_value\_present\_flag[ c ] and gfv\_chroma\_key\_thr\_present\_flag[ i ] are present and the syntax elements and gfv\_chroma\_key\_value[ c ] and gfv\_chroma\_key\_thr\_value[ i ] might be present. gfv\_chroma\_key\_info\_present\_flag equal to 0 specifies that the syntax elements gfv\_chroma\_key\_value\_present\_flag[ c ], gfv\_chroma\_key\_thr\_present\_flag[ i ], gfv\_chroma\_key\_value[ c ], and gfv\_chroma\_key\_thr\_value[ i ] are not present.

**gfv\_chroma\_key\_value\_present\_flag**[ c ] equal to 1 indicates that the syntax element gfv\_chroma\_key\_value[ c ] is present. gfv\_chroma\_key\_present\_flag[ c ] equal to 0 indicates that the syntax element gfv\_chroma\_key\_value[ c ] is not present.

The variable ChromaKeyDefaultValueFlag is set equal to !( gfv\_chroma\_key\_value\_present\_flag[ 0 ] | | gfv\_chroma\_key\_value\_present\_flag[ 1 ] | | gfv\_chroma\_key\_value\_present\_flag[ 2 ] ).

**gfv\_chroma\_key\_value**[ c ] specifies the chroma key value corresponding to the c-th colour component as follows:

– If ChromaKeyDefaultValueFlag is equal to 1, the variables GfvChromaKeyValue[ c ] are specified as follows:

– GfvChromaKeyValue[ 0 ] is set equal to 50.

– GfvChromaKeyValue[ 1 ] is set equal to 220.

– GfvChromaKeyValue[ 2 ] is set equal to 100.

– Otherwise, ChromaKeyDefaultValueFlag is equal to 0, the following applies:

– If gfv\_chroma\_key\_value\_present\_flag[ c ] is equal to 1, GfvChromaKeyValue[ c ] is set equal to the value of gfv\_chroma\_key\_value[ c ].

– Otherwise, gfv\_chroma\_key\_value\_present\_flag[ c ] is equal to 0, GfvChromaKeyValue[ c ] is not specified by this Specification.

**gfv\_chroma\_key\_thr\_present\_flag**[ i ] equal to 1 indicates that the syntax element gfv\_chroma\_thr\_value[ i ] is present. gfv\_chroma\_key\_thr\_present\_flag[ i ] equal to 0 indicates gfv\_chroma\_key\_thr\_value[ i ] is not present.

**gfv\_chroma\_key\_thr\_value**[ i ], when present, specifies the i-th chroma key threshold value. The value of gfv\_chroma\_key\_thr\_value[ i ] shall be in the range of 0 to 255, inclusive. When not present, the value of gfv\_chroma\_key\_thr\_value[ i ] is inferred as follows:

If i is equal to 0, gfv\_chroma\_key\_thr\_value[ 0 ] is set equal to 48.

Otherwise, i is equal to 1, gfv\_chroma\_key\_thr\_value[ 1 ] is set equal to 75.

NOTE 3 – The syntax elements gfv\_chroma\_key\_value\_present[ c ], gfv\_chroma\_key\_value[ c ], and gfv\_chroma\_key\_thr\_value[ i ] could be used to determine a transparency indicator for fusion of the generated face picture and background picture. For example, a transparency indicator, denoted as alpha[ x ][ y ] , for picture sample value, denoted as I[ c ][ x ][ y ] with bitDepth[ c ], where bitDepth[ 0 ] is equal to BitDepthY, bitDepth[ 1 ] and bitDepth[ 2 ] are equal to BitDepthC, and GfvChromaKey[ c ] values for sample coordinates x, y, and colour components c could be determined as follows:

d[ x ][ y ]= 0  
for( c = 0; c < 3; c++ )  
 if( gfv\_chroma\_key\_value\_present[ c ]  | |  ChromaKeyDefaultValueFlag )  
 d[ x ][ y ]  +=  ( I[ c ][ x ][ y ] / ( 1  <<  (bitDepth[ c ] − 8) ) − GfvChromaKeyValue[ c ] )2  
if( ( d[ x ][ y ] < gfv\_chroma\_key\_thr\_value[ 0 ] )  
 alpha[ x ][ y ] = 0  
else if ( ( d[ x ][ y ] > gfv\_chroma\_key\_thr\_value[ 1 ] )  
 alpha[ x ][ y ] = 1  
else  
 alpha[ x ][ y ] = ( d[ x ][ y ] − gfv\_chroma\_key\_thr\_value[ 0 ] ) ÷  
 ( gfv\_chroma\_key\_thr\_value[ 1 ] − gfv\_chroma\_key\_thr\_value[ 0 ] )

A value of alpha[ x ][ y ] equal to 0 could indicate transparency. A value of alpha[ x ][ y ] equal to 1 could indicate opacity. Intermediate values of alpha[ x ][ y ] could indicate semitransparency.

**gfv\_drive\_pic\_fusion\_flag**, when present, equal to 1 indicates that the current decoded picture, which corresponds to a driving picture that may be used for fusion, may be input to GenerativeNN( ). gfv\_drive\_pic\_fusion\_flag equal to 0 indicates that the current decoded picture should not be input to GenerativeNN( ).

NOTE 4 – A gfv\_drive\_pic\_fusion\_flag value of 1 can be used, for example, to indicate that the current decoded picture can be used to improve face details or handle background changes.

NOTE 5 – When gfv\_base\_pic\_flag is equal to 0 and gfv\_drive\_pic\_fusion\_flag is equal to 1, the GFV process takes three inputs: the base picture, features from keypoints and/or matrices carried in the GFV SEI message, and the current decoded picture that is a fusion picture, and outputs a picture that is generated by the GenerativeNN( ).

NOTE 6 – When gfv\_base\_pic\_flag is equal to 0 and gfv\_drive\_pic\_fusion\_flag is equal to 0, the GFV process takes twoinputs: the base picture and features from keypoints and/or matrices carried in the GFV SEI message, and outputs a picture that is generated by the GenerativeNN( ).

NOTE 7 – When gfv\_base\_pic\_flag is equal to 1, the GFV process directly outputs the cropped decoded picture.

When a GFV SEI message has gfv\_base\_pic\_flag equal to 0 and gfv\_drive\_pic\_fusion\_flag equal to 0, the GFV SEI message pertains to the current decoded picture only.

When a GFV SEI message with a particular gfv\_id value has gfv\_base\_pic\_flag equal to 0 and gfv\_drive\_pic\_fusion\_flag equal to 1, the fusion picture for that particular gfv\_id value, which is the current cropped decoded picture, remains valid for the current decoded picture and all subsequent decoded pictures of the current layer, in output order, until the end of the current CLVS or up to but excluding the decoded picture that is within the current CLVS, follows the current decoded picture in output order, and is associated with a GFV SEI message having that particular gfv\_id value, whichever is earlier.

When a GFV SEI message gfvSeiA with a particular gfv\_id value has gfv\_cnt greater than 0 and a GFV SEI message gfvSeiB with the same gfv\_id value in the same picture unit has gfv\_base\_pic\_flag equal to 1 (i.e., the current decoded picture is a base picture), the GFV SEI message gfvSeiA shall have gfv\_drive\_pic\_fusion\_flag equal to 0.

**gfv\_low\_confidence\_face\_parameter\_flag** equal to 1 indicates the facial parameters have been derived with low confidence. gfv\_low\_confidence\_face\_parameter\_flag equal to 0 indicates the confidence information of the facial parameters is not specified.

**gfv\_coordinate\_present\_flag** equal to 1 indicates that coordinate information of keypoints is present. gfv\_coordinate\_present\_flag equal to 0 indicates that coordinate information of keypoints is not present.

It is a requirement of bitstream conformance that when gfv\_matrix\_type\_idx[ i ] for any i from 0 to gfv\_num\_matrix\_types\_minus1 is equal to 0 or 1, the value of gfv\_coordinate\_present\_flag shall be equal to 1.

**gfv\_kps\_pred\_flag** equal to 1 indicates that the syntax elements gfv\_coordinate\_dx\_abs[ i ] ,gfv\_coordinate\_dy\_abs[ i ], and gfv\_coordinate\_dz\_abs[ i ] are present and the syntax elements gfv\_coordinate\_dx\_sign\_flag[ i ], gfv\_coordinate\_dy\_sign\_flag[ i ] and gfv\_coordinate\_dz\_sign\_flag[ i ] may be present. gfv\_kps\_pred\_flag equal to 0 indicates that the syntax elements gfv\_coordinate\_x\_abs[ i ], gfv\_coordinate\_y\_abs[ i ], and gfv\_coordinate\_z\_abs[ i ] are present and the syntax elements gfv\_coordinate\_x\_sign\_flag[ i ], gfv\_coordinate\_y\_sign\_flag[ i ] and gfv\_coordinate\_z\_sign\_flag[ i ] may be present.

When gfv\_coordinate\_present\_flag is equal to 1, gfv\_base\_pic\_flag is equal to 0, and gfv\_kps\_pred\_flag is equal to 1, there shall be a previous GFV SEI message in decoding order with the same gfv\_id as the current GFV SEI message and gfv\_base\_pic\_flag equal to 1 in the current CLVS.

**gfv\_coordinate\_precision\_factor\_minus1** plus 1 indicates the precision of key point coordinates signgalled in the SEI message. The value of gfv\_coordinate\_precision\_factor\_minus1 shall be in the range of 0 to 31, inclusive. When gfv\_coordinate\_present\_flag is equal to 1, gfv\_base\_pic\_flag is equal to 0, and gfv\_kps\_pred\_flag is equal to 1, the value of gfv\_coordinate\_precision\_factor\_minus1 is inferred to be equal to the gfv\_coordinate\_precision\_factor\_minus1 of the previous GFV SEI message in decoding order with the same gfv\_id as the current GFV SEI message and gfv\_base\_pic\_flag equal to 1.

**gfv\_num\_kps\_minus1** plus 1 indicates the number of keypoints. The value of gfv\_num\_kps\_minus1 shall be in the range of 0 to 210 – 1, inclusive. When gfv\_coordinate\_present\_flag is equal to 1, gfv\_base\_pic\_flag is equal to 0, and gfv\_kps\_pred\_flag is equal to 1, the value of gfv\_num\_kps\_minus1 is inferred to be equal to the gfv\_num\_kps\_minus1 of the previous GFV SEI message in decoding order with the same gfv\_id as the current GFV SEI message and gfv\_base\_pic\_flag equal to 1.

**gfv\_coordinate\_z\_present\_flag** equal to 1 indicates that z-axis coordinate information of the keypoints is present. gfv\_coordinate\_z\_present\_flagequal to 0 indicates that the z-axis coordinate information of the keypoints is not present. When gfv\_coordinate\_present\_flag is equal to 1, gfv\_base\_pic\_flag is equal to 0, and gfv\_kps\_pred\_flag is equal to 1, the value of coordinate\_z\_present\_flag is inferred to be equal to the coordinate\_z\_present\_flag of the previous GFV SEI message in decoding order with the same gfv\_id as the current GFV SEI message and gfv\_base\_pic\_flag equal to 1.

**gfv\_coordinate\_z\_max\_value\_minus1** plus 1 indicates the maximum absolute value of z-axis coordinates of keypoints. The value of gfv\_coordinate\_z\_max\_value\_minus1 shall be in the range of 0 to 216 − 1, inclusive. When gfv\_coordinate\_present\_flag is equal to 1, gfv\_base\_pic\_flag is equal to 0, and gfv\_kps\_pred\_flag is equal to 1, the value of gfv\_coordinate\_z\_max\_value\_minus1 is inferred to be equal to the gfv\_coordinate\_z\_max\_value\_minus1, when present, in the previous GFV SEI message in decoding order with the same gfv\_id as the current GFV SEI message and gfv\_base\_pic\_flag equal to 1.

**gfv\_coordinate\_x\_abs**[ i ] is used to derive the x-axis coordinate of the i-th keypoint. The value of gfv\_coordinate\_x\_abs[ i ] shall be in the range of 0 to 2gfv\_coordinate\_precision\_factor\_minus1 + 1, inclusive

**gfv\_coordinate\_x\_sign\_flag**[ i ] specifies the sign of the x-axis coordinate of the i-th keypoint. When gfv\_coordinate\_x\_sign\_flag[ i ] is not present, it is inferred to be equal to 0.

**gfv\_coordinate\_y\_abs**[ i ] is used to derive y-axis coordinate of i-th keypoint. The value of gfv\_coordinate\_y\_abs[ i ] shall be in the range of 0 to 2gfv\_coordinate\_precision\_factor\_minus1 + 1, inclusive.

**gfv\_coordinate\_y\_sign\_flag**[ i ] specifies the sign of the y-axis coordinate of the i-th keypoint. When gfv\_coordinate\_y\_sign\_flag[i] is not present, it is inferred to be equal to 0.

**gfv\_coordinate\_z\_abs**[ i ] is used to derive z-axis coordinate of the i-th keypoint. The value of gfv\_coordinate\_z\_abs[ i ] shall be in the range of 0 to 2gfv\_coordinate\_precision\_factor\_minus1 + 1, inclusive.

**gfv\_coordinate\_z\_sign\_flag**[ i ] specifies the sign of the z-axis coordinate of the i-th key point. When gfv\_coordinate\_z\_sign\_flag[ i ] is not present, it is inferred to be equal to 0.

**gfv\_coordinate\_dx\_abs**[ i ] specifies a difference value that is used to derive x-axis coordinate of the i-th keypoint. The value of gfv\_coordinate\_dx\_abs[ i ] shall be in the range of 0 to 2gfv\_coordinate\_precision\_factor\_minus1 + 2, inclusive.

**gfv\_coordinate\_dx\_sign\_flag**[ i ] specifies the sign of the difference value of the x-axis coordinate of the i-th keypoint. When gfv\_coordinate\_dx\_sign\_flag[ i ] is not present, it is inferred to be equal to 0.

**gfv\_coordinate\_dy\_abs**[ i ] specifies a difference value that is used to derive y-axis coordinate of the i-th keypoint. The value of gfv\_coordinate\_dy\_abs[ i ] shall be in the range of 0 to 2gfv\_coordinate\_precision\_factor\_minus1 + 2, inclusive.

**gfv\_coordinate\_dy\_sign\_flag**[ i ] specifies the sign of the difference value of the y-axis coordinate of the i-th keypoint. When gfv\_coordinate\_yd\_sign\_flag[i] is not present, it is inferred to be equal to 0.

**gfv\_coordinate\_dz\_abs**[ i ] specifies a difference value that is used to derive z-axis coordinate of the i-th keypoint. The value of gfv\_coordinate\_dz\_abs[ i ] shall be in the range of 0 to 2gfv\_coordinate\_precision\_factor\_minus1 + 2, inclusive.

**gfv\_coordinate\_dz\_sign\_flag**[ i ] specifies the sign of the difference value of the z-axis coordinate of the i-th key point. When gfv\_coordinate\_dz\_sign\_flag[ i ] is not present, it is inferred to be equal to 0.

If gfv\_coordinate\_z\_max\_value\_minus1 is present**,** the variable CroppedDepth is set equal to gfv\_coordinate\_z\_max\_value\_minus1 + 1. Otherwise, CroppedDepth is set equal to 0.

When gfv\_kps\_pred\_flag is equal to 1, the variables coordinateDeltaX[ i ], coordinateDeltaY[ i ] and coordinateDeltaZ[ i ] indicating the delta x-axis coordinate, delta y-axis coordinate and delta z-axis coordinate of the i-th keypoint, respectively, are derived as follows:

coordinateDeltaX[ i ] = ( 1 − 2 \* gfv\_coordinate\_dx\_sign\_flag[ i ] ) \* gfv\_coordinate\_dx\_abs[ i ] ÷  
 ( 1  <<  ( gfv\_coordinate\_precision\_factor\_minus1 + 1 ) )  
coordinateDeltaY[ i ] = ( 1 − 2 \* gfv\_coordinate\_dy\_sign\_flag[ i ] ) \* gfv\_coordinate\_dy\_abs[ i ] ÷  
 ( 1  <<  ( gfv\_coordinate\_precision\_factor\_minus1 + 1 ) )  
if( gfv\_coordinate\_z\_present\_flag )  
 coordinateDeltaZ[ i ] = ( 1 − 2 \* gfv\_coordinate\_dz\_sign\_flag[ i ] ) \* gfv\_coordinate\_dz\_abs[ i ] ÷  
 ( 1  <<  ( gfv\_coordinate\_precision\_factor\_minus1 + 1 ) )

The variables coordinateX[ i ], coordinateY[ i ], and, when gfv\_coordinate\_z\_present\_flag is equal to 1, coordinateZ[ i ] indicating the x-axis coordinate, y-axis coordinate and z-axis coordinate of the i-th keypoint, respectively, are derived as follows:

If gfv\_kps\_pred\_flag is equal to 0, the following applies:

coordinateX[ i ] = ( 1 − 2 \* gfv\_coordinate\_x\_sign\_flag[ i ] ) \* gfv\_coordinate\_x\_abs[ i ] ÷  
 ( 1  <<  ( gfv\_coordinate\_precision\_factor\_minus1 + 1 ) )  
coordinateY[ i ] = ( 1 − 2 \* gfv\_coordinate\_y\_sign\_flag[ i ] ) \* gfv\_coordinate\_y\_abs[ i ] ÷  
 ( 1  <<  ( gfv\_coordinate\_precision\_factor\_minus1 + 1 ) )  
if (gfv\_coordinate\_z\_present\_flag )  
 coordinateZ[ i ] = ( 1 − 2 \* gfv\_coordinate\_z\_sign\_flag[ i ] ) \* gfv\_coordinate\_z\_abs[ i ] ÷  
 ( 1  <<  ( gfv\_coordinate\_precision\_factor\_minus1 + 1 ) )

Otherwise (gfv\_kps\_pred\_flag is equal to 1), the following applies:

if( gfv\_base\_pic\_flag ) {  
 coordinateX[ i ] = (( i > 0 ) ? coordinateX[ i − 1 ] : 0 ) + coordinateDeltaX[ i ]  
 coordinateY[ i ] = (( i > 0 ) ? coordinateY[ i − 1 ] : 0 ) + coordinateDeltaY[ i ]  
 if (gfv\_coordinate\_z\_present\_flag )   
 coordinateZ[ i ] = (( i > 0 ) ? coordinateZ[ i − 1 ] : 0 ) + coordinateDeltaZ[ i ]  
} else if( gfv\_cnt  = =  0 ) {  
 coordinateX[ i ] = BaseKpCoordinateX[ i ] + coordinateDeltaX[ i ]  
 coordinateY[ i ] = BaseKpCoordinateY[ i ] + coordinateDeltaY[ i ]  
 if (gfv\_coordinate\_z\_present\_flag )   
 coordinateZ[ i ] = BaseKpCoordinateZ[ i ] + coordinateDeltaZ[ i ]  
} else {  
 coordinateX[ i ] = PrevKpCoordinateX[ i ] + coordinateDeltaX[ i ]  
 coordinateY[ i ] = PrevKpCoordinateY[ i ] + coordinateDeltaY[ i ]  
 coordinateZ[ i ] = PrevKpCoordinateZ[ i ] + coordinateDeltaZ[ i ]  
}

The following applies for derivation of the variables BaseKpCoordinateX[ i ], BaseKpCoordinateY[ i ], BaseKpCoordinateZ[ i ], PrevKpCoordinateX[ i ], PrevKpCoordinateY[ i ], and PrevKpCoordinateZ[ i ]:

if( gfv\_base\_pic\_flag ) {  
 PrevKpCoordinateX[ i ] = BaseKpCoordinateX[ i ] = coordinateX[ i ]  
 PrevKpCoordinateY[ i ] = BaseKpCoordinateY[ i ] = coordinateY[ i ]  
 if (gfv\_coordinate\_z\_present\_flag )  
 PrevKpCoordinateZ[ i ] = BaseKpCoordinateZ[ i ] = coordinateZ[ i ]  
} else {  
 PrevKpCoordinateX[ i ] = coordinateX[ i ]  
 PrevKpCoordinateY[ i ] = coordinateY[ i ]  
 PrevKpCoordinateZ[ i ] = coordinateZ[ i ]  
}

**gfv\_matrix\_present\_flag** equal to 1 indicates that matrix parameters are present. gfv\_matrix\_present\_flag equal to 0 indicates that matrix parameters are not present. When gfv\_coordinate\_present\_flag is equal to 0, gfv\_matrix\_present\_flag shall be equal to 1.

**gfv\_matrix\_pred\_flag** equal to 1 indicates that the syntax elements gfv\_matrix\_element\_int[ i ][ j ][ k ][ m ] and gfv\_matrix\_element\_dec[ i ][ j ][ k ][ m ] are present and the syntax element gfv\_matrix\_element\_sign\_flag[ i ][ j ][ k ][ m ] may be present. gfv\_matrix\_pred\_flag equal to 0 indicates that the syntax elements gfv\_matrix\_delta\_element\_int[ i ][ j ][ k ][ m ] and gfv\_matrix\_delta\_element\_dec[ i ][ j ][ k ][ m ] are present and the syntax element gfv\_matrix\_delta\_element\_sign\_flag [ i ][ j ][ k ][ m ] may be present. When gfv\_matrix\_pred\_flag is not present, it is inferred to be 0.

When gfv\_matrix\_present\_flag is equal to 1, gfv\_base\_pic\_flag is equal to 0, and gfv\_matrix\_pred\_flag is equal to 1, there shall be a previous GFV SEI message in decoding order with the same gfv\_id as the current GFV SEI message and gfv\_base\_pic\_flag equal to 1 in the current CLVS.

**gfv\_matrix\_element\_precision\_factor\_minus1** plus 1 indicates the precision of matrix elements signalled in the SEI message. The value of gfv\_matrix\_element\_precision\_factor\_minus1 shall be in the range of 0 to 31, inclusive. When gfv\_matrix\_present\_flag is equal to 1, gfv\_base\_pic\_flag is equal to 0, and gfv\_matrix\_pred\_flag is equal to 1, the value of gfv\_matrix\_element\_precision\_factor\_minus1 is inferred to be equal to the gfv\_matrix\_element\_precision\_factor\_minus1 of the previous GFV SEI message in decoding order with the same gfv\_id as the current GFV SEI message and gfv\_base\_pic\_flag equal to 1.

**gfv\_num\_matrix\_types\_minus1** plus 1indicates the number of matrix types signalled in the SEI message. The value of gfv\_num\_matrix\_types\_minus1 shall be in the range of 0 to 26 − 1, inclusive. It is a requirement of bitstream conformance that when gfv\_matrix\_pred\_flag is equal to 1 and gfv\_base\_pic\_flag is equal to 0, the value of gfv\_num\_matrix\_types\_minus1 shall be equal to the value of gfv\_num\_matrix\_types\_minus1 in each of the preceding GFV SEI message in decoding order in the current CLVS which has the same gfv\_id value as the gfv\_id value in the current SEI and has gfv\_base\_pic\_flag equal to 1. When gfv\_matrix\_present\_flag is equal to 1, gfv\_base\_pic\_flag is equal to 0, and gfv\_matrix\_pred\_flag is equal to 1, the value of gfv\_matrix\_type\_num\_minus1 is inferred to be equal to the gfv\_matrix\_type\_num\_minus1 of the previous GFV SEI message in decoding order with the same gfv\_id as the current GFV SEI message and gfv\_base\_pic\_flag equal to 1.

**gfv\_matrix\_type\_idx**[ i ]indicates the index of the i-th matrix type as specified in Table 1. The value of gfv\_matrix\_type\_idx[ i ] shall be in the range of 0 to 63, inclusive. In bitstreams conforming to this version of this Specification, the value of gfv\_matrix\_type\_idx[ i ] shall be in the range of 0 to 31, inclusive. Decoders conforming to this version of this Specification shall allow gfv\_matrix\_type\_idx[ i ] to be greater than 31 to appear in the bitstream and the decoder shall ignore all information for the i-th type of matrix for which gfv\_matrix\_type\_idx[ i ] is greater than 31.

**Table 1- Specification of gfv\_matrix\_type\_idx[ i ]**

|  |  |
| --- | --- |
| **Value** | **Specification** |
| 0 | Affine translation matrix with the size of 2\*2 or 3\*3. |
| 1 | Covariance matrix with size of 2\*2 or 3\*3. |
| 2 | Mouth matrix representing mouth motion. |
| 3 | Eye matrix representing the open-close status and level of eyes. |
| 4 | Head rotation paramters with the size of 2\*2 or 3\*3 representing the head rotation in 2D space or 3D space. |
| 5 | Head translation matrix with the size of 1\*2 or 1\*3 representing head translationin 2D space or 3D space. |
| 6 | Head location matrix with size of 1\*2 or 1\*3 representing the head location in 2D space or 3D space. |
| 7 | Compact feature matrix with the size being specified by gfv\_matrix\_width\_minus1[i] and gfv\_matrix\_height\_minus1[i]. |
| 8…31 | Other matrix that may be used as determined by the application with the size being specified by gfv\_matrix\_width\_minus1[i] and gfv\_matrix\_height\_minus1[i]. |
| 32…63 | Reserved |

NOTE 8 – The undefined matrxi type is used to represent the matrxi type rather than affine translation matrix, covariance matrix, rotation matrix, translation matrix and compact feature matrix. It can be used by the user to extend the matrix type.

**gfv\_num\_matrices\_equal\_to\_num\_kps\_flag**[ i ] equal to 1 indicates that the number of matrices of the i-th matrix type is equal to gfv\_num\_kps\_minus1 + 1. gfv\_num\_matrices\_equal\_to\_num\_kps\_flag[ i ] equal to 0 indicates the number of matrices of the i-th matrix type is not equal to gfv\_num\_kps\_minus1 + 1. If gfv\_matrix\_present\_flag is equal to 1, gfv\_base\_pic\_flag is equal to 0, gfv\_matrix\_pred\_flag is equal to 1, gfv\_matrix\_type\_idx[ i ] is equal to 0 or 1, and gfv\_coordinate\_present\_flag is equal to 1, the value of gfv\_num\_matrices\_equal\_to\_num\_kps\_flag[ i ] is inferred to be equal to the gfv\_num\_matrices\_equal\_to\_num\_kps\_flag[ i ], when present, in the previous GFV SEI message in decoding order with the same gfv\_id as the current GFV SEI message and gfv\_base\_pic\_flag equal to 1. Otherwise, when gfv\_num\_matrices\_equal\_to\_num\_kps\_flag[ i ] is not present, its vlaue is inferred to be equal to 0.

**gfv\_num\_matrices\_info**[ i ] provides information to derive the number of the matrices of the i-th matrix type. The value of gfv\_num\_matrices\_info[ i ] shall be in the range of 0 to 210 − 1, inclusive. When gfv\_matrix\_present\_flag is equal to 1, gfv\_base\_pic\_flag is equal to 0, gfv\_matrix\_pred\_flag is equal to 1, gfv\_matrix\_type\_idx[ i ] is equal to 0 or 1, and either gfv\_coordinate\_present\_flag is equal to 0 or gfv\_num\_matrix\_equal\_to\_num\_kps\_flag[ i ] is equal to 0, the value of gfv\_num\_matrices\_info[ i ] is inferred to be equal to the gfv\_num\_matrices\_info[ i ], when present, in the previous GFV SEI message in decoding order with the same gfv\_id as the current GFV SEI message and gfv\_base\_pic\_flag equal to 1.

**gfv\_matrix\_width\_minus1**[ i ] plus 1 indicates the width of the matrix of the i-th matrix type. The value of gfv\_matrix\_width\_minus1[ i ] shall be in the range of 0 to 210 − 1, inclusive. When gfv\_matrix\_present\_flag is equal to 1, gfv\_matrix\_pred\_flag is equal to 0, gfv\_matrix\_pred\_flag is equal to 1, and gfv\_matrix\_type\_idx[ i ] is equal to 2 or 3 or is greater than or equal to 7, the value of gfv\_matrix\_width\_minus1[ i ] is inferred to be equal to the gfv\_matrix\_width\_minus1[ i ], when present, in the previous GFV SEI message in decoding order with the same gfv\_id as the current GFV SEI message and gfv\_base\_pic\_flag equal to 1.

**gfv\_matrix\_height\_minus1**[ i ] plus 1 indicates the height of the matrix of the i-th matrix type. The value of gfv\_matrix\_height\_minus1[ i ] shall be in the range of 0 to 210 − 1, inclusive. When gfv\_matrix\_present\_flag is equal to 1, gfv\_base\_pic\_flag is equal to 0, gfv\_matrix\_pred\_flag is equal to 1, and gfv\_matrix\_type\_idx[ i ] is equal to 2 or 3 or is greater than or equal to 7, the value of gfv\_matrix\_height\_minus1[ i ] is inferred to be equal to the gfv\_matrix\_height\_minus1[ i ], when present, in the previous GFV SEI message in decoding order with the same gfv\_id as the current GFV SEI message and gfv\_base\_pic\_flag equal to 1.

**gfv\_matrix\_for\_3D\_space\_flag**[ i ] equal to 1 indicates the matrix of the i-th matrix type is a matrix defined in three-dimensional space. gfv\_matrix\_for\_3D\_space\_flag[ i ] equal to 0 indicates the matrix of the i-th matrix type is a matrix defined in two-dimensional space. When gfv\_matrix\_present\_flag is equal to 1, gfv\_base\_pic\_flag is equal to 0, gfv\_matrix\_pred\_flag is equal to 1, gfv\_matrix\_type\_idx[ i ] is equal to 4, 5, or 6, and gfv\_coordinate\_present\_flag is equal to 0, the value of gfv\_matrix\_for\_3D\_space\_flag[ i ] is inferred to be equal to the gfv\_matrix\_for\_3D\_space\_flag[ i ], when present, in the previous GFV SEI message in decoding order with the same gfv\_id as the current GFV SEI message and gfv\_base\_pic\_flag equal to 1.

When gfv\_matrix\_width\_minus1[ i ] is not present, it is inferred as follows:

* If gfv\_matrix\_type\_idx[ i ] is equal to 0, 1 or 4, and one of coordinate\_z\_present\_flag and gfv\_matrix\_for\_3D\_space\_flag[ i ] is present and equal to 1, gfv\_matrix\_width\_minus1[i] is inferred to be equal to 2.
* Otherwise, if gfv\_matrix\_type\_idx[ i ] is equal to 0, 1 or 4, and one of coordinate\_z\_present\_flag and gfv\_matrix\_for\_3D\_space\_flag[ i ] is present and equal to 0, gfv\_matrix\_width\_minus1[ i ] is inferred to be equal to 1.
* Otherwise (gfv\_matrix\_type\_idx[ i ] is equal to 5 or 6), gfv\_matrix\_width\_minus1[ i ] is inferred to be equal to 0.

When gfv\_matrix\_height\_minus1[ i ] is not present, it is inferred as follows:

* If gfv\_matrix\_type\_idx is equal to 0, 1, 4, 5 or 6, and one of gfv\_coordinate\_z\_present\_flag and gfv\_matrix\_for\_3D\_space\_flag[ i ] is present and equal to 1, gfv\_matrix\_height\_minus1[ i ] is inferred to be equal to 2.
* Otherwise (gfv\_matrix\_type\_idx is equal to 0, 1, 4, 5 or 6, and one of gfv\_coordinate\_z\_present\_flag and gfv\_matrix\_for\_3D\_space\_flag[ i ] is 0), gfv\_matrix\_height \_minus1[ i ] is inferred to be equal to 1.

The variables matrixWidth[ i ] and matrixHeight[ i ] indicating the width and height of the matrix of the i-th matrix type are derived as follows:

if( gfv\_matrix\_pred\_flag ) {  
 matrixWidth[ i ] = BaseMatrixWidth[ i ]  
 matrixHeight[ i ] = BaseMatrixHeight[ i ]  
} else {  
 matrixWidth[ i ] = gfv\_matrix\_width\_minus1[ i ] + 1  
 matrixHeight[ i ] = gfv\_matrix\_height\_minus1[ i ] + 1  
}  
if( gfv\_base\_pic\_flag ) {  
 BaseMatrixWidth[ i ] = matrixWidth[ i ]  
 BaseMatrixHeight[ i ] = matrixHeight[ i ]  
}

**gfv\_num\_matrices\_minus1**[ i ] plus 1 indicates the number of matrices of the i-th matrix type. The value of gfv\_num\_matrices\_minus1[ i ] shall be in the range of 0 to 210 − 1, inclusive. When gfv\_matrix\_present\_flag is equal to 1, gfv\_base\_pic\_flag is equal to 0, gfv\_matrix\_pred\_flag is equal to 1, and gfv\_matrix\_type\_idx[ i ] is greater than or equal to 7, the value of gfv\_num\_matrices\_minus1[ i ] is inferred to be equal to the gfv\_num\_matrices\_minus1[ i ], when present, in the previous GFV SEI message in decoding order with the same gfv\_id as the current GFV SEI message and gfv\_base\_pic\_flag equal to 1.

The variable numMatrices[ i ] indicating the number of the matrices of the i-th matrix type is derived as follows:

if( gfv\_matrix\_pred\_flag )  
 numMatrices[ i ] = BaseNumMatrices[ i ]  
else if( gfv\_matrix\_type\_idx[ i ]  = =  0  | |  gfv\_matrix\_type\_idx[ i ]  = =  1 ) {  
 if( gfv\_coordinate\_present\_flag )  
 numMatrices[ i ] = gfv\_num\_matrices\_equal\_to\_num\_kps\_flag[ i ] ? gfv\_num\_kps\_minus1 + 1:  
 ( gfv\_num\_matrices\_info[ i ] < gfv\_num\_kps\_minus1 ? gfv\_num\_matrices\_info [ i ] + 1:  
 gfv\_num\_matrices\_info [ i ] + 2 )  
 else  
 numMatrices[ i ] = gfv\_num\_matrices\_info[ i ] + 1  
} else if( gfv\_matrix\_type\_idx[ i ]  >=  2  &&  gfv\_matrix\_type\_idx[ i ] < 7 )  
 numMatrices[ i ] = 1  
else  
 numMatrices[ i ] = gfv\_num\_matrices\_minus1[ i ] + 1  
if( gfv\_base\_pic\_flag )  
 BaseNumMatrices[ i ] = numMatrices[ i ]

It is a requirement of bitstream conformance that when gfv\_matrix\_pred\_flag is equal to 1 and gfv\_base\_pic\_flag is equal to 0, the values of numMatrices[ i ], matrixWidth[ i ], and matrixHeight[ i ] for i in the range of 0 to gfv\_num\_matrix\_types\_minus1, inclusive shall be respectively equal to the values of numMatrices[ i ], matrixWidth[ i ], and matrixHeight[ i ] for i in the range of 0 to gfv\_num\_matrix\_types\_minus1, inclusive in each of the preceding GFV SEI message in decoding order in the current CLVS which has the same gfv\_id value as the gfv\_id value in the current SEI and has gfv\_base\_pic\_flag equal to 1.

**gfv\_matrix\_element\_int**[ i ][ j ][ k ][ m ] indicates the integer part of the value of the matrix element at position (m, k) of the j-th matrix of the i-th matrix type. The value of gfv\_matrix\_element\_int[ i ][ j ][ k ][ m ] shall be in the range of 0 to 232 − 2, inclusive.

**gfv\_matrix\_element\_dec**[ i ][ j ][ k ][ m ] indicates the decimal part of the value of the matrix element at position (m, k) of the j-th matrix of the i-th matrix type. The length of gfv\_matrix\_element\_dec[ i ][ j ][ k ][ m ] is gfv\_matrix\_element\_precision\_factor\_minus1 + 1 bits.

**gfv\_matrix\_element\_sign\_flag**[ i ][ j ][ k ][ m ] indicates the sign of the matrix element at position (m, k) of the j-th matrix of the i-th matrix type. When gfv\_matrix\_element\_sign\_flag[ i ][ j ][ k ][ m ]is not present, it is inferred to be equal to 0.

**gfv\_matrix\_delta\_element\_int**[ i ][ j ][ k ][ m ] indicates the integer part of the difference value of the matrix element at position (m, k) of the j-th matrix of the i-th matrix type. The value of gfv\_matrix\_delta\_element\_int[ i ][ j ][ k ][ m ] shall be in the range of 0 to 232 − 2, inclusive.

**gfv\_matrix\_delta\_element\_dec**[ i ][ j ][ k ][ m ] indicates the decimal part of the difference value of the matrix element at position (m, k) of the j-th matrix of the i-th matrix type. The value of gfv\_matrix\_delta\_element\_dec[ i ][ j ][ k ][ m ] shall be in the range of 0 to 232 − 2, inclusive.

**gfv\_matrix\_delta\_element\_sign\_flag**[ i ][ j ][ k ][ m ] indicates the sign of the difference value of the matrix element at position (m, k) of the j-th matrix of the i-th matrix type. When gfv\_matrix\_element\_sign\_flag[ i ][ j ][ k ][ m ]is not present, it is inferred to be equal to 0.

When gfv\_matrix\_pred\_flag is equal to 1, the variable matrixElementDeltaVal[ i ][ j][ k ][ m ] representing the difference value of the matrix element at position (m, k) of the j-th matrix of the i-th matrix type is derived as follows:

matrixElementDeltaVal[ i][ j ][ k ][ m ] =   
The variable matrixElementVal[ i ][ j][ k ][ m ] representing the value of the matrix element at position (m, k) of the j-th matrix of the i-th matrix type is derived as follows:

If gfv\_matrix\_pred\_flag is equal to 0, the following applies:

matrixElementVal[ i][ j ][ k ][ m ] =   
  
if( gfv\_base\_pic\_flag )  
 BaseMatrixElementVal[ i][ j ][ k ][ m ] = matrixElementVal[ i][ j ][ k ][ m ]

Otherwise (gfv\_matrix\_pred\_flag is equal to 1), the following applies:

if( gfv\_cnt  = =  0 )  
 matrixElementVal[ i][ j ][ k ][ m ] = BaseMatrixElementVal[ i][ j ][ k ][ m ] +  
 matrixElementDeltaVal[ i][ j ][ k ][ m ]  
else  
 matrixElementVal[ i][ j ][ k ][ m ] = PrevMatrixElementVal[ i][ j ][ k ][ m ] +  
 matrixElementDeltaVal[ i][ j ][ k ][ m ]

The following applies:

if( gfv\_base\_pic\_flag )  
 PrevMatrixElementVal[ i][ j ][ k ][ m ] = BaseMatrixElementVal[ i][ j ][ k ][ m ] =  
 matrixElementVal[ i][ j ][ k ][ m ]  
else  
 PrevMatrixElementVal[ i][ j ][ k ][ m ] = matrixElementVal[ i][ j ][ k ][ m ]

For a particular gfv\_id value, the following process is used in increasing order of gfv\_cnt to generate a video picture per each GFV SEI message that has gfv\_base\_pic\_flag equal to 0 and a unique value of gfv\_cnt within a picture unit:

DeriveSigParam( )  
TranslatorNN( sigKeyPoint, sigMatrix )  
DeriveInputTensors( )  
if( gfv\_base\_pic\_flag  = =  0  &&  gfv\_drive\_pic\_fusion\_flag  = =  0 ) {  
 if( ChromaFormatIdc  = =  0 )  
 GenerativeNN( inputBaseY, inputBaseKeyPoint, inputBaseMatrix, inputDriveKeyPoint,  
 inputDriveMatrix, CroppedWidth, CroppedHeight, CroppedDepth )  
 else  
 GenerativeNN( inputBaseY, inputBaseCb, inputBaseCr, inputBaseKeyPoint, inputBaseMatrix,  
 inputDriveKeyPoint, inputDriveMatrix, CroppedWidth, CroppedHeight, CroppedDepth )  
} else if( gfv\_base\_pic\_flag  = =  0  &&  gfv\_drive\_pic\_fusion\_flag  = =  1 ) {  
 if( ChromaFormatIdc  = =  0 )  
 GenerativeNN( inputBaseY, inputDriveY, inputBaseKeyPoint, inputBaseMatrix, inputDriveKeyPoint,  
 inputDriveMatrix, CroppedWidth, CroppedHeight, CroppedDepth )  
 else  
 GenerativeNN( inputBaseY, inputBaseCb, inputBaseCr, inputDriveY, inputDriveCb, inputDriveCr,  
 inputBaseKeyPoint, inputBaseMatrix,, inputDriveKeyPoint, inputDriveMatrix,  
 CroppedWidth, CroppedHeight, CroppedDepth )  
}  
StoreOutputTensors( )

The process DeriveSigParam( ) for deriving the inputs of TranslatorNN( ) is specified as follows:

The keypoint coordinate array sigKeyPoint and the matrix sigMatrix are derived as follows:

if( gfv\_coordinate\_present\_flag )  
 for( i = 0; i  <=  gfv\_num\_kps\_minus1; i++ ) {  
 sigKeyPoint[ i ][ 0 ] = coordinateX[ i ]  
 sigKeyPoint[ i ][ 1 ] = coordinateY[ i ]  
 if( gfv\_coordinate\_z\_present\_flag )  
 sigKeyPoint[ i ][ 2 ] = coordinateZ[ i ]  
 }  
else  
 for( i = 0; i  <=  gfv\_num\_kps\_minus1; i++ ) {  
 sigKeyPoint[ i ][ 0 ] = 0  
 sigKeyPoint[ i ][ 1 ] = 0  
 if ( gfv\_coordinate\_z\_present\_flag )  
 sigKeyPoint[ i ][ 2 ] = 0  
 }  
if( gfv\_matrix\_present\_flag )  
 for ( i = 0; i  <=  gfv\_num\_matrix\_types\_minus1; i++ )  
 for ( j = 0; j < numMatrices[ i ]; j++ )  
 for( k = 0; k < matrixHeight [ i ]; k++ )  
 for( l = 0;l < matrixWidth [ i ]; l++)  
 sigMatrix[ i ][ j ][ k ][ l ] = matrixElementVal[ i ][ j][ k][ l ]  
else  
 for( i = 0; i  <=  gfv\_num\_matrix\_types\_minus1; i++ )  
 for ( j = 0; j < numMatrices[ i ]; j++ )  
 for( k = 0; k < matrixHeight [ i ]; k++ )  
 for( l = 0;l < matrixWidth [ i ]; l++)  
 sigMatrix[ i ][ j ][ k ][ l ] = 0

TranslatorNN( ) is a process to translate the various formats of the facial parameters carried in the SEI message to the fixed format of the facial parameters to be input to the generative network to generate the output picture

Inputs to TranslatorNN( ) are:

* sigKeyPoint and sigMatrix

Outputs of TranslatorNN( ) are:

* convKeyPoint and convNumKeyPoint
* convMatrix and convNumMatrix, convMatrixWidth, convMatrixHeight

The process DeriveInputTensors( ) for deriving the inputs of GenerativeNN( ) is specified as follows:

When gfv\_base\_pic\_flag is equal to 1, the BasePicture input tensor inputBaseY, inputBaseCb and inputBaseCr are derived as follows:

for( x = 0; x < CroppedWidth; x++ )  
 for ( y = 0; y < CroppedHeight; y++ )  
 inputBaseY[ x ][ y ] = InpY( baseCroppedYPic[ x ][ y ] )  
if( ChromaFormatIdc  !=  0 )  
 for( x = 0; x < CroppedWidth / SubWidthC; x++ )  
 for ( y = 0; y < CroppedHeight / SubHeightC; y++ ) {  
 inputBaseCb[ x][ y ] = InpC( baseCroppedCbPic[ x][ y ] )  
 inputBaseCr[ x][ y ] = InpC( baseCroppedCrPic[ x ][ y ] )  
 }

When gfv\_base\_pic\_flag is equal to 0 and gfv\_drive\_pic\_fusion\_flag is equal to 1, the DrivePicture luma sample array inputDriveY, inputDriveCb and input DriveCr are derived as follows:

for( x = 0; x< CroppedWidth; x++ )  
 for ( y = 0; y< CroppedHeight; y++ )  
 inputDriveY[ x ][ y ] = InpY( driveCroppedYPic[ x ][ y ] )  
if( ChromaFormatIdc  !=  0 )  
 for( x = 0; x< CroppedWidth / SubWidthC; x++ )  
 for( y = 0; y < CroppedHeight / SubHeightC; y++ ) {  
 InputDriveCb[ x][ y ] = InpC( driveCroppedCbPic[ x][ y ] )  
 InputDriveCr[ x][ y ] = InpC( driveCroppedCrPic[ x ][ y ] )  
 }

When gfv\_base\_pic\_flag is equal to 0, the keypoint coordinate array inputDriveKeyPoint and the matrix inputDriveMatrix for the current picture are derived as follows:

for( i = 0; i < = convNumKeyPoint; i++ ) {  
 inputDriveKeyPoint[ i ][ 0 ] = convKeyPoint[ i ][ 0 ]  
 inputDriveKeyPoint [ i ][ 1 ] = convKeyPoint[ i ][ 1 ]  
 inputDriveKeyPoint [ i ][ 2 ] = convKeyPoint[ i ][ 2 ]  
}  
for( j = 0; j < convNumMatrix; j++ )  
 for( k = 0; k < convMatrixHeight; k++ )  
 for( m = 0; m < convMatrixWidth; m++ )  
 inputDriveMatrix[ j ][ k ][ m ] = convMatrix [ j ][ k ][ m ]

When gfv\_base\_pic\_flag is equal to 1, the keypoint coordinate array inputBaseKeyPoint and the matrix inputBaseMatrix for the base picture are derived as follows:

for( i = 0; i  <=  convNumKeyPoint; i++ ) {  
 inputBaseKeyPoint[ i ][ 0 ] = convKeyPoint[ i ][ 0 ]  
 inputBaseKeyPoint [ i ][ 1 ] = convKeyPoint[ i ][ 1 ]  
 inputBaseKeyPoint [ i ][ 2 ] = convKeyPoint[ i ][ 2 ]  
}  
for( j = 0; j < convNumMatrix; j++ )  
 for( k = 0; k < convMatrixHeight; k++ )  
 for( l = 0; l < convMatrixWidth; l++ )  
 inputBaseMatrix[ j ][ k ][ l ] = convMatrix [ j ][ k ][ l ]

Where the functions InpY( ) and InpC( ) are specified as follows:

InpY( x ) = x ÷ ( ( 1  <<  BitDepthY ) − 1 )  
InpC( x ) = x ÷ ( ( 1  <<  BitDepthC ) − 1 )

GenerativeNN( ) is a process to generate the sample values of an output picture corresponding to a driving picture. It is only invoked when gfc\_base\_pic\_flag is equal to 0. Input values to GenerativeNN( ) and output values from GenerativeNN( ) are real numbers.

Inputs to GenerativeNN( ) are:

* When gfv\_base\_pic\_flag is equal to 0, gfv\_drive\_pic\_fusion\_flag is equal to 0, and ChromaFormatIdc is equal to 0: inputBaseY, inputBaseKeyPoint, inputBaseMatrix, inputDriveKeyPoint, inputDriveMatrix, CroppedWidth, CroppedHeight,and CroppedDepth.
* When gfv\_base\_pic\_flag is equal to 0, gfv\_drive\_pic\_fusion\_flag is equal to 0, and ChromaFormatIdc is not equal to 0: inputBaseY, inputBaseCb, inputBaseCr, inputBaseKeyPoint, inputBaseMatrix, inputDriveKeyPoint, inputDriveMatrix, CroppedWidth, CroppedHeight, and CroppedDepth.
* When gfv\_base\_pic\_flag is equal to 0, gfv\_drive\_pic\_fusion\_flag is equal to 1, gfv\_chroma\_key\_info\_present\_flag is equal to 0, and ChromaFormatIdc is equal to 0: inputBaseY, inputDriveY, inputBaseKeyPoint, inputBaseMatrix, inputDriveKeyPoint, inputDriveMatrix, CroppedWidth, CroppedHeight, and CroppedDepth.
* When gfv\_base\_pic\_flag is equal to 0, gfv\_drive\_pic\_fusion\_flag is equal to 1, gfv\_chroma\_key\_info\_present\_flag is equal to 0, and ChromaFormatIdc is not equal to 0: inputBaseY, inputBaseCb, inputBaseCr, inputDriveY, inputDriveCb, inputDriveCr , inputBaseKeyPoint, inputBaseMatrix,, inputDriveKeyPoint, inputDriveMatrix, CroppedWidth, CroppedHeight, and CroppedDepth.
* When gfv\_base\_pic\_flag is equal to 0, gfv\_drive\_pic\_fusion\_flag is equal to 1, gfv\_chroma\_key\_info\_present\_flag is equal to 1, and ChromaFormatIdc is equal to 0: inputBaseY, inputDriveY, inputBaseKeyPoint, inputBaseMatrix, inputDriveKeyPoint, inputDriveMatrix, CroppedWidth, CroppedHeight, CroppedDepth, gfv\_chroma\_key\_thr\_value[ 0 ], gfv\_chroma\_key\_thr\_value[ 1 ], and GfvChromaKeyValue[ 0 ].
* When gfv\_base\_pic\_flag is equal to 0, gfv\_drive\_pic\_fusion\_flag is equal to 1, gfv\_chroma\_key\_info\_present\_flag is equal to 0, and ChromaFormatIdc is not equal to 0: inputBaseY, inputBaseCb, inputBaseCr, inputDriveY, inputDriveCb, inputDriveCr , inputBaseKeyPoint, inputBaseMatrix, inputDriveKeyPoint, inputDriveMatrix, CroppedWidth, CroppedHeight, CroppedDepth, gfv\_chroma\_key\_thr\_value[ 0 ], gfv\_chroma\_key\_thr\_value[ 1 ], and, when specified, GfvChromaKeyValue[ 0 ], GfvChromaKeyValue[ 1 ], and GfvChromaKeyValue[ 2 ].

Outputs of GenerativeNN( ) are:

* A luma sample array genY
* When ChromaFormatIdc is not equal to 0, two chroma sample arrays genCb and genCr.

The process StoreOutputTensors( ) for deriving the output is specified as follows:

When gfv\_base\_pic\_flag is equal to 0, the output sample array outYPic[ x ][ y ], outCbPic[ x ][ y ], and outCrPic[ x ][ y ] are derived as follows:

for( x = 0; x < CroppedWidth; x++ )  
 for( y = 0; y < CroppedHeight; y++ )  
 outputYPic[ x ][ y ] = OutY( genY[ x ][ y ] )  
if( ChromaFormatIdc  !=  0 )  
 for( x = 0; x < CroppedWidth / SubWidthC; x++ )  
 for( y = 0; y < CroppedHeight / SubHeightC; y++ ) {  
 outputCbPic[ x ][ y ] = OutC( genCb[ x ][ y ] )  
 outputCrPic[ x][ y ] = OutC( genCr[ x ][ y ] )  
 }

When gfv\_base\_pic\_flag is equal to 1, the output sample array outYPic[ x ][ y ], outCbPic[ x ][ y ], and outCrPic[ x ][ y ] are derived as follows (each output picture derived by the process StoreOutputTensors( ) is referred to as a GFV-generated picture):

for( x = 0; x< CroppedWidth; x++ )  
 for( y = 0; y< CroppedHeight; y++ )  
 outputYPic[ x ][ y ] = baseCroppedYPic[ x ][ y ]  
if( ChromaFormatIdc  !=  0 )  
 for( x = 0; x< CroppedWidth / SubWidthC; x++ )  
 for( y = 0; y< CroppedHeight / SubHeightC; y++ ) {  
 outputCbPic[ x ][ y ] = baseCroppedCbPic[ x ][ y ]  
 outputCrPic[ x][ y ] = baseCroppedCbPic[ x ][ y ]  
 }

Where the functions OutY( ) and OutC( ) are specified as follows:

OutY( x ) = Clip3( 0, ( 1  <<  BitDepthY ) − 1 , x \* ( ( 1  <<  BitDepthY ) − 1 )  
OutC( x ) = Clip3( 0, ( 1  <<  BitDepthC ) − 1 , x \* ( ( 1  <<  BitDepthC ) − 1 )

The output order of GFV-generated pictures corresponding to the GFV SEI messages in a picture unit with the same gfv\_id value and different gfv\_cnt values shall be in increasing order of the gfv\_cnt values. For any two pictures picA and picB wherein picA precedes picB in output order, any GFV-generated picture corresponding to a GFV SEI message with a particular gfv\_id value and associated with picA shall precede, in output order, any GFV-generated picture corresponding to a GFV SEI message with the particular gfv\_id value and associated with picB.

* 1. **Generative face video enhancement SEI message**
     1. **Generative face video enhancement SEI message syntax**

|  |  |
| --- | --- |
| generative\_face\_video\_enhancement ( payloadSize ) { | **Descriptor** |
| **gfve\_id** | ue(v) |
| **gfve\_gfv\_id** | ue(v) |
| **gfve\_gfv\_cnt** | ue(v) |
| if( gfve\_gfv\_cnt==0) { |  |
| **gfve\_base\_pic\_flag** |  |
| if (gfve\_base\_pic\_flag) { |  |
| **gfve\_nn\_present\_flag** | u(1) |
| if( gfve\_nn\_present\_flag ) { |  |
| **gfve\_nn\_mode\_idc** | ue(v) |
| if( gfve\_nn\_mode\_idc  = =  1 ) { |  |
| while( !byte\_aligned( ) ) |  |
| **gfve\_nn\_alignment\_zero\_bit\_a** | u(1) |
| **gfve\_nn\_tag\_uri** | st(v) |
| **gfve\_nn\_uri** | st(v) |
| } |  |
| } |  |
| } |  |
| **gfve\_matrix\_element\_precision\_factor** | ue(v) |
| **gfve\_num\_matrices\_minus1** | ue(v) |
| for(i=0; i <= gfve\_num\_matrices\_minus1; i++){ |  |
| **gfve\_matrix\_height\_minus1**[ i ] | ue(v) |
| **gfve\_matrix\_width\_minus1**[ i ] | ue(v) |
| for( j = 0; j <= gfve\_matrix\_height\_minus1[ i ]; j++ ) |  |
| for( k = 0; k <= gfve\_matrix\_width\_minus1[ i ]; k++ ) { |  |
| **gfve\_matrix\_element**[ i ][ j ][ k ] | ue(v) |
| if( !gfve\_matrix\_element[ i ][ j ][ k ]) |  |
| **gfve\_matrix\_element\_sign\_flag**[ i ][ j ][ k ] | u(1) |
| } |  |
| } |  |
| **gfve\_pupil\_coordinate\_present\_idx** | u(2) |
| if( gfve\_pupil\_coordinate\_present\_idx  !=  0 ) { |  |
| if( gfve\_gfv\_base\_pic\_flag ) |  |
| **gfve\_pupil\_coordinate\_precision\_factor\_minus1** | ue(v) |
| if( gfve\_pupil\_coordinate\_present\_idx  = =  1 || gfve\_pupil\_coordinate\_present\_idx  = =  3) { |  |
| **gfve\_pupil\_left\_eye\_dx\_coordinate\_abs** | ue(v) |
| if(gfve\_pupil\_left\_eye\_dx\_coordinate\_abs) |  |
| **gfve\_pupil\_left\_eye\_dx\_coordinate\_sign\_flag** | u(1) |
| **gfve\_pupil\_left\_eye\_dy\_coordinate\_abs** | ue(v) |
| if(gfve\_pupil\_left\_eye\_dy\_coordinate\_abs) |  |
| **gfve\_pupil\_left\_eye\_dy\_coordinate\_sign\_flag** | u(1) |
| } |  |
| if( gfve\_pupil\_coordinate\_present\_idx  = =  2 || gfve\_pupil\_coordinate\_present\_idx  = =  3) { |  |
| **gfve\_pupil\_right\_eye\_dx\_coordinate\_abs** | ue(v) |
| if(gfve\_pupil\_right\_eye\_dx\_coordinate\_abs) |  |
| **gfve\_pupil\_right\_eye\_dx\_coordinate\_sign\_flag** | u(1) |
| **gfve\_pupil\_right\_eye\_dy\_coordinate\_abs** | ue(v) |
| if(gfve\_pupil\_right\_eye\_dy\_coordinate\_abs) |  |
| **gfve\_pupil\_right\_eye\_dy\_coordinate\_sign\_flag** | u(1) |
| } |  |
| } |  |
| if( gfve\_nn\_present\_flag ) |  |
| if( gfve\_nn\_mode\_idc  = =  0 ) { |  |
| while( !byte\_aligned( ) ) |  |
| **gfve\_nn\_alignment\_zero\_bit\_b** | u(1) |
| for( i = 0; more\_data\_in\_payload( ); i++ ) |  |
| **gfve\_nn\_payload\_byte**[ i ] | b(8) |
| } |  |
| } |  |

* + 1. **Generative face video enhancement SEI message semantics**

The generative face video enhancement (GFVE) SEI message indicates enhancement facial parameters and specifies an enhancement network, denoted as EnhancerNN( ), that may be used to enhance the visual quality of the face pictures generated with GFV SEI message.

NOTE 1 – Enhancement facial parameters could be determined from source pictures prior to encoding.

NOTE 2 – When the current picture is not a base picture, the GFV SEI message may be used to generate a face picture based the facial parameters conveyed by the GFV SEI message, and the GFVE SEI message may be further used to enhance the generated face picture to improve the visual quality.

Use of this SEI message requires the definition of the following variables:

* Input and output picture width and height in units of luma samples, denoted herein by CroppedWidth and CroppedHeight, respectively.
* Luma sample array baseCroppedYPic and chroma sample arrays baseCroppedCbPic and baseCroppedCrPic for a decoded output picture, denoted as BasePicture, corresponding to a base picture.
* Luma sample array genCroppedYPic and chroma sample arrays genCroppedCbPic and genCroppedCrPic for a generated picture with associated GFV SEI message, denoted as GenPicture, corresponding to a driving picture.
* Bit depth BitDepthY for the luma sample array of the input and output pictures.
* Bit depth BitDepthC for the chroma sample arrays, if any, of the input and output pictures.
* A chroma format indicator, denoted herein by ChromaFormatIdc, as described in subclause ‎7.3.

The variables SubWidthC and SubHeightC are derived from ChromaFormatIdc as specified by Table 2.

**gfve\_id** contains an identifying number that may be used to identify GFVE SEI message and specify a neural network that may be used as EnhancerNN( ). The value of gfve\_id shall be in the range of 0 to 232 − 2, inclusive. Values of gfve\_id from 256 to 511, inclusive, and from 231 to 232 − 2, inclusive, are reserved for future use by ITU-T | ISO/IEC. Decoders conforming to this edition of this document encountering a GFVE SEI message with gfve\_id in the range of 256 to 511, inclusive, or in the range of 231 to 232 − 2, inclusive, shall ignore the SEI message.

**gfve\_gfv\_id** and **gfve\_gfv\_cnt** specifies gfv\_id and gfv\_cnt of the associated GFV SEI message. The associated GFV SEI message is a GFV SEI message in the same picture unit with the GFVE SEI message having gfv\_id equal to gfve\_gfv\_id, gfv\_cnt equal to gfve\_gfv\_cnt. The GFVE message is used to enhance the picture generated with the associated GFV SEI message.

For a GFVE SEI message, the following applies:

* The GFVE SEI message shall be present in the same picture unit with the associated GFV SEI message. And the associated GFV SEI message shall be precede the GFVE SEI message in the decoding order. When the associated GFV SEI message is not present in the picture unit containing the GFVE SEI message, the GFVE SEI message shall be ignored.
* If a GFV SEI message A is associated with a GFVE SEI message A, and a GFV SEI message B is associated with a GFVE SEI message B, and GFV SEI message A precede the GFV SEI message B in decoding order, the GFVE SEI A shall also precede GFVE SEI message B in decoding order.
* If a GFVE SEI message A with gfve\_gfv\_id equal to gfveGfvIdA, gfve\_gfv\_cnt value equal to gfveGfvCntA, and a GFVE SEI message B with gfve\_gfv\_id equal to gfveGfvIdB, gfve\_gfv\_cnt value equal to gfveGfvCntB, are present in the same picture unit, and gfveGfvIdA is equal to gfveGfvIdB and gfveGfvCntA is less than gfveGfvCntB, the GFVE SEI message A shall precede the GFVE SEI message B in decoding order.
* If a GFVE SEI message A with gfve\_gfv\_id equal to gfveGfvIdA, gfve\_gfv\_cnt value equal to gfveGfvCntA, and a GFV SEI message B with gfv\_id equal to gfvIdB, gfv\_cnt value equal to gfvCntB, are present in the same picture unit, if gfveGfvIdA is equal to gfvIdB, gfveGfvCntA is less than gfvCntB, the GFVE SEI message A shall precede the GFV SEI message B in decoding order.
* The GFVE SEI messages that are present in the same picture unit and have the same values of gfve\_id, gfve\_gfv\_id, and gfve\_gfv\_cnt shall have the same SEI payload content

**gfve\_base\_pic\_flag** equal to 1 indicates that the current decoded output picture corresponds to a base picture. gfv\_base\_pic\_flag equal to 0 indicates that the current decoded output picture does not correspond to a base picture or this SEI message does not specify syntax elements for a base picture. When gfv\_base\_pic\_flag is not present, it is inferred to be equal to 0. It is a requirement of bitstream conformance that the value of gfv\_base\_pic\_flag shall be equal to the value of gfv\_base\_pic\_flag of the associated GFV SEI message.

**gfve\_nn\_present\_flag** equal to 1 indicatesa neural network that may be used as a EnhancerNN( ) is contained or indicated by the SEI message. gfve\_nn\_present\_flag equal to 0 indicatesa neural network that may be used as a EnhancerNN( ) is not contained or indicated by the SEI message.

**gfve\_nn\_mode\_idc**, **gfve\_nn\_alignment\_zero\_bit\_a**, **gfve\_nn\_tag\_uri**, **gfve\_nn\_uri**, **gfve\_nn\_alignment\_zero\_bit\_b**, and **gfve\_nn\_payload\_byte**[ i ] specify a neural network that may be used as a EnhancerNN( ). gfve\_nn\_mode\_idc, gfve\_nn\_alignment\_zero\_bit\_a, gfve\_nn\_tag\_uri, gfve\_nn\_uri, gfve\_nn\_alignment\_zero\_bit\_b, and gfve\_nn\_payload\_byte[ i ] have the same syntax and semantics as nnpfc\_base\_flag, nnpfc\_mode\_idc, nnpfc\_alignment\_zero\_bit\_a, nnpfc\_tag\_uri, nnpfc\_uri, nnpfc\_alignment\_zero\_bit\_b, and nnpfc\_payload\_byte[ i ], respectively.

**gfve\_matrix\_element\_precision\_factor** indicates quantization factor of matrix elements signalled in the SEI message. The value of gfve\_matrix\_element\_precision\_factor shall be in the range of 0 to 32, inclusive.

**gfve\_num\_matrices\_minus1** plus 1 specifies the number of matrices signalled in the SEI message. The value of gfv\_num\_matrices\_minus1 shall be in the range of 0 to 210 − 1, inclusive.

**gfve\_matrix\_height\_minus1**[ i ] plus 1 indicates the height of the i-th matrix. The value of gfve\_matrix\_height\_minus1[ i ] shall be in the range of 0 to 210 − 1, inclusive.

**gfve\_matrix\_width\_minus1**[ i ] plus 1 indicates the width of the i-th matrix. The value of gfve\_matrix\_width\_minus1[ i ] shall be in the range of 0 to 210 − 1, inclusive.

**gfve\_matrix\_element**[ i ][ j ][ k ] is used to derive the value of the element at position (k, j) of the i-th matrix. The value of gfve\_matrix\_element[ i ][ j ][ k ] shall be in the range of 0 to 232 − 2, inclusive.

**gfve\_matrix\_element\_sign\_flag**[ i ][ j ][ k ] indicates the sign of the matrix element at position (k, j) of the i-th matrix.

The variable matrixElementVal[ i ][ j][ k ] representing the value of the matrix element at position (k, j) of the i-th matrix is derived as follows:

matrixElementVal[ i][ j ][ k ] =

**gfve\_pupil\_coordinate\_present\_idx** equal to 0 indicates the pupil coordinate is not present. gfve\_pupil\_coordinate\_present\_idx equal to 1 indicates the pupil coordinate information of the left eye is present. gfve\_pupil\_coordinate\_present\_idx equal to 2 indicates the pupil coordinate information of the right eye is present. gfve\_pupil\_coordinate\_present\_idx equal to 3 indicates pupil coordinate information of both the left eye and the right eye is present.

**gfve\_pupil\_coordinate\_precision\_factor\_minus1** plus 1 plus 1 indicates the precision of pupil coordinates signgalled in the SEI message. The value of gfve\_pupil\_coordinate\_precision \_factor\_minus1 shall be in the range of 0 to 31, inclusive. When gfve\_pupil\_coordinate\_present\_idx is not equal to 0 and gfve\_gfv\_base\_pic\_flag is equal to 0, the value of gfve\_pupil\_coordinate\_precision\_factor\_minus1 is inferred to be equal to the gfve\_pupil\_coordinate\_precision \_factor\_minus1 of the previous GFVE SEI message in decoding order with the same gfve\_id as the current GFVE SEI message and gfve\_gfv\_base\_pic\_flag equal to 1.

**gfve\_pupil\_left\_eye\_dx\_coordinate\_abs** specifies a difference value that is used to derive the x-axis coordinate of left eye. When gfve\_pupil\_left\_eye\_dx\_coordinate\_abs is not present, it is inferred to be equal to 0.[Ed (JC): value range missing]

**gfve\_pupil\_left\_eye\_dx\_coordinate\_sign\_flag** specifies the sign of the difference value of the x-axis coordinate of the left eye. When gfve\_pupil\_left\_eye\_dx\_coordinate\_sign\_flag is not present, it is inferred to be equal to 0.

**gfve\_pupil\_left\_eye\_dy\_coordinate\_abs** specifies a difference value that is used to derive the y-axis coordinate of the left eye. When gfve\_pupil\_left\_eye\_dy\_coordinate\_abs is not present, it is inferred to be equal to 0. [Ed (JC): value range missing]

**gfve\_pupil\_left\_eye\_dy\_coordinate\_sign\_flag** specifies the sign of the difference value of the y-axis coordinate of the left eye. When gfve\_pupil\_left\_eye\_dy\_coordinate\_sign\_flag is not present, it is inferred to be equal to 0.

When gfve\_pupil\_coordinate\_present\_idx is equal to 1 or 3, the variables leftPupilCoordinateDeltaX and leftPupilCoordinateDeltaY indicating the difference value of x-axis coordinate and y-axis coordinate of the left pupil center position, respectively, are derived as follows:

leftPupilCoordinateDeltaX = ( 1 − 2 \* gfve\_pupil\_left\_eye\_dx\_coordinate\_sign\_flag ) \*   
 gfve\_pupil\_left\_eye\_dx\_coordinate\_abs ÷ ( 1 << ( gfve\_pupil\_coordinate\_precision\_factor\_minus1 + 1 ) )   
leftPupilCoordinateDeltaY = ( 1 − 2 \* gfve\_pupil\_left\_eye\_dy\_coordinate\_sign\_flag ) \*   
 gfve\_pupil\_left\_eye\_dy\_coordinate\_abs ÷ ( 1 << ( gfve\_pupil\_coordinate\_precision\_factor\_minus1 + 1 ) )

**gfve\_pupil\_right\_eye\_dx\_coordinate\_abs** indicates a difference value that is used to derive the x-axis coordinate of the right eye. When gfve\_pupil\_right\_eye\_dx\_coordinate\_abs is not present, it is inferred to be equal to 0. [Ed (JC): value range missing]

**gfve\_pupil\_right\_eye\_dx\_coordinate\_sign\_flag** specifies the sign of the difference value of the x-axis coordinate of the right eye. When gfve\_pupil\_right\_eye\_dx\_coordinate\_sign\_flag is not present, it is inferred to be equal to 0.

**gfve\_pupil\_right\_eye\_dy\_coordinate\_abs** indicates a difference value that is used to derive the y-axis coordinate of the right eye. When gfve\_pupil\_right\_eye\_dy\_coordinate\_abs is not present, it is inferred to be equal to 0. [Ed (JC): value range missing]

**gfve\_pupil\_right\_eye\_dy\_coordinate\_sign\_flag** specifies the sign of the difference value of the y-axis coordinate of the right eye. When gfve\_pupil\_right\_eye\_dy\_coordinate\_sign\_flag is not present, it is inferred to be equal to 0.

When gfve\_pupil\_coordinate\_present\_idx is equal to 2 or 3, the variables rightPupilCoordinateDeltaX and rightPupilCoordinateDeltaY indicating the the difference value of x-axis coordinate and y-axis coordinate of the right pupil center position, respectively, are derived as follows:

rightPupilCoordinateDeltaX = ( 1 − 2 \* gfve\_pupil\_right\_eye\_dx\_coordinate\_sign\_flag ) \*   
 gfve\_pupil\_right\_eye\_dx\_coordinate\_abs ÷ ( 1 << ( gfve\_pupil\_coordinate\_precision\_factor\_minus1 + 1 ) )   
rightPupilCoordinateDeltaY = ( 1 − 2 \* gfve\_pupil\_right\_eye\_dy\_coordinate\_sign\_flag ) \*   
 gfve\_pupil\_right\_eye\_dy\_coordinate\_abs ÷ ( 1 << ( gfve\_pupil\_coordinate\_precision\_factor\_minus1 + 1 ) )

The variables leftPupilCoordinateX, leftPupilCoordinateY, rightPupilCoordinateX and rightPupilCoordinateY indicating the x-axis coordinate and y-axis coordinate of the left and right pupil center position, respectively, are derived as follows:

if (gfve\_gfv\_cnt == 0){   
 if (gfve\_gfv\_base\_pic\_flag){   
 leftPupilCoordinateX =leftPupilCoordinateDeltaX   
 leftPupilCoordinateY = leftPupilCoordinateDeltaY  
 rightPupilCoordinateX = rightPupilCoordinateDeltaX + leftPupilCoordinateDeltaX  
 rightPupilCoordinateY = rightPupilCoordinateDeltaY + leftPupilCoordinateDeltaY  
 }else{  
 if (gfve\_pupil\_coordinate\_present\_idx == 1 || gfve\_pupil\_coordinate\_present\_idx == 3){   
 leftPupilCoordinateX = leftPupilCoordinateDeltaX + BaseLeftPupilCoordinateX  
 leftPupilCoordinateY = leftPupilCoordinateDeltaY + BaseLeftPupilCoordinateY  
 }  
 if(gfve\_pupil\_coordinate\_present\_idx == 2 || gfve\_pupil\_coordinate\_present\_idx == 3){   
 rightPupilCoordinateX = rightPupilCoordinateDeltaX + BaseRightPupilCoordinateX  
 rightPupilCoordinateY = rightPupilCoordinateDeltaY + BaseRightPupilCoordinateY  
  }  
 }  
} else {  
 if (gfve\_pupil\_coordinate\_present\_idx == 1 || gfve\_pupil\_coordinate\_present\_idx == 3){   
 leftPupilCoordinateX = leftPupilCoordinateDeltaX + PrevLeftPupilCoordinateX  
 leftPupilCoordinateY = leftPupilCoordinateDeltaY + PrevLeftPupilCoordinateY  
 }  
 if(gfve\_pupil\_coordinate\_present\_idx == 2 || gfve\_pupil\_coordinate\_present\_idx == 3){   
 rightPupilCoordinateX = rightPupilCoordinateDeltaX + PrevRightPupilCoordinateX  
 rightPupilCoordinateY = rightPupilCoordinateDeltaY + PrevRightPupilCoordinateY  
 }  
}

The following applies for derivation of the variables PrevLeftPupilCoordinateX, PrevLeftPupilCoordinateY, PrevRightPupilCoordinateX, PrevRightPupilCoordinateY:

if (gfve\_gfv\_base\_pic\_flag){   
 PrevLeftPupilCoordinateX = BaseLeftPupilCoordinateX = leftPupilCoordinateX  
 PrevLeftPupilCoordinateY = BaseLeftPupilCoordinateY = leftPupilCoordinateY  
 PrevRightPupilCoordinateX = BaseRightPupilCoordinateX = rightPupilCoordinateX  
 PrevRightPupilCoordinateY = BaseRightPupilCoordinateY = rightPupilCoordinateY  
}else{  
 PrevLeftPupilCoordinateX = leftPupilCoordinateX  
 PrevLeftPupilCoordinateY = leftPupilCoordinateY  
 PrevRightPupilCoordinateX = rightPupilCoordinateX  
 PrevRightPupilCoordinateY = rightPupilCoordinateY  
}

The following process is invoked for each GFVE SEI message to enhance the picture generated with the associated GFV SEI message:

DeriveInputTensors( )  
if( ChromaFormatIdc  = =  0 )  
 EnhancerNN( inputBaseY, inputGenY, inputMatrix)  
else  
 EnhancerNN( inputBaseY, inputBaseCb, inputBaseCr, inputGenY, inputGenCb, inputGenCr, inputMatrix)  
StoreOutputTensors( )

The process DeriveInputTensors( ) for deriving the inputs of EnhancerNN( ) is specified as follows:

The BasePicture input tensor inputBaseY, inputBaseCb and inputBaseCr are derived as follows:

for( x = 0; x < CroppedWidth; x++ )  
 for ( y = 0; y < CroppedHeight; y++ )  
 inputBaseY[ x ][ y ] = InpY( baseCroppedYPic[ x ][ y ] )  
if( ChromaFormatIdc  !=  0 )  
 for( x = 0; x < CroppedWidth/ SubWidthC; x++ )  
 for ( y = 0; y < CroppedHeight/ SubHeightC; y++ ) {  
 inputBaseCb[ x][ y ] = InpC( baseCroppedCbPic[ x][ y ] )  
 inputBaseCr[ x][ y ] = InpC( baseCroppedCrPic[ x ][ y ] )  
 }

The GenPicture input tensor inputGenY, inputGenCb and inputGenCr are derived as follows:

for( x = 0; x < CroppedWidth; x++ )  
 for ( y = 0; y < CroppedHeight; y++ )  
 inputGenY[ x ][ y ] = InpY( genCroppedYPic[ x ][ y ] )  
if( ChromaFormatIdc  !=  0 )  
 for( x = 0; x < CroppedWidth / SubWidthC; x++ ) {  
 for ( y = 0; y < CroppedHeight / SubHeightC; y++ ) {  
 inputGenCb[ x][ y ] = InpC( genCroppedCbPic[ x][ y ] )  
 inputGenCr[ x][ y ] = InpC( genCroppedCrPic[ x ][ y ] )  
 }

The matrix input tensor inputMatrix is derived as follows:

for( i = 0; i  <=  gfve\_num\_matrices\_minus1; i++ )  
 for ( j = 0; j  <=  gfve\_matrix\_height\_minus1[ i ]; j++)  
 for( k = 0; k  <=  gfve\_matrix\_width\_minus1[ i ]; k++ )  
 inputMatrix[ i ][ j ][ k ] = matrixElementVal[ i ][ j][ k ]

Where the functions InpY( ) and InpC( ) are specified as follows:

InpY( x ) = x ÷ ( ( 1  <<  BitDepthY ) − 1 )  
InpC( x ) = x ÷ ( ( 1  <<  BitDepthC ) − 1 )

EnhancerNN( ) is a process to enhance the sample values of a generated picture that is generated with the associated GFV SEI message. Input values to EnhancerNN( ) and output values from EnhancerNN( ) are real numbers.

Inputs to EnhancerNN( ) are:

* When ChromaFormatIdc is equal to 0: inputBaseY, inputGenY, inputMatrix
* When ChromaFormatIdc is not equal to 1: inputBaseY, inputBaseCb, inputBaseCr, inputGenY, inputGenCb, inputGenCr, inputMatrix

Outputs of EnhancerNN( ) are:

* A luma sample array enhanceY
* When ChromaFormatIdc is not equal to 0, two chroma sample arrays enhanceCb and enhanceCr.

The process StoreOutputTensors( ) for deriving the output is specified as follows:

The output sample array outYPic[ x ][ y ], outCbPic[ x ][ y ], and outCrPic[ x ][ y ] are derived as follows:

for( x=0; x< CroppedWidth; x++ )  
 for( y=0; y< CroppedHeight; y++ )  
 outputYPic[ x ][ y ] = OutY( enhanceY[ x ][ y ] )  
if( ChromaFormatIdc  !=  0 )  
 for( x = 0; x< CroppedWidth / SubWidthC; x++ )  
 for( y = 0; y< CroppedHeight / SubHeightC; y++ ) {  
 outputCbPic[ x ][ y ] = OutC( enhanceCb[ x ][ y ] )  
 outputCrPic[ x][ y ] = OutC( enhanceCr[ x ][ y ] )  
 }

Where the functions OutY( ) and OutC( ) are specified as follows:

OutY( x ) = Clip3( 0, ( 1  <<  BitDepthY ) − 1 , x \* ( ( 1  <<  BitDepthY ) − 1 )  
OutC( x ) = Clip3( 0, ( 1  <<  BitDepthC ) − 1 , x \* ( ( 1  <<  BitDepthC ) − 1 )

* 1. **Digitally signed content initialization SEI message**
     1. **Digitally signed content initialization SEI message syntax**

|  |  |
| --- | --- |
| digitally\_signed\_content\_initialization( payloadSize ) { | **Descriptor** |
| **dsci\_hash\_method\_type** | u(8) |
| **dsci\_key\_source\_uri** | st(v) |
| **dsci\_num\_verification\_substreams\_minus1** | ue(v) |
| **dsci\_key\_retrieval\_mode\_idc** | ue(v) |
| if( dsci\_key\_retrieval\_mode\_idc = = 1){ |  |
| **dsci\_use\_key\_register\_idx\_flag** | u(1) |
| if( dsci\_use\_key\_register\_idx\_flag ) |  |
| **dsci\_key\_register\_idx** | ue(v) |
| } |  |
| **dsci\_content\_uuid\_present\_flag** | u(1) |
| if( dsci\_content\_uuid\_present\_flag) |  |
| **dsci\_content\_uuid** | u(128) |
| } |  |

* + 1. **Digitally signed content initialization SEI message semantics**

Use of this SEI message requires the definition of the following:

– A lists of non-VCL NAL unit type identifiers nonVclDigitallySignedNalUnitsList.

The digitally signed content initialization SEI message, digitally signed content selection SEI message, and digitally signed content verification SEI message provide a mechanism for verifying that the coded video has been produced by a content provider that identifies itself via the digital certificate that is referenced in the digitally signed content initialization SEI message. This SEI message also provides information about the secure hash algorithm used for calculating message digests, which are used together with the digital signature present in digitally signed content verification SEI messages to verify the trustworthiness of non-VCL NAL units with a NAL unit type identifier among the values in nonVclDigitallySignedNalUnitsList and VCL NAL units present in the coded video sequence. It further provides information about the digital signature algorithm used and the public key of the content provider. The digitally signed content initialization SEI message may provide the public key of the content provider either by providing an URI that identifies a trust record, as specified in ISO/IEC 21617-1, that contains the certificate of the content provider or by providing an URI that directly identifies the certificate.

When a digitally signed content initialization SEI message is present in any AU of a CVS, a digitally signed content initialization SEI message shall be present for all IDR, CRA and GDR PUs of the CVS.

When a digitally signed content initialization SEI message is present in any AU of a CVS, it shall precede all non-VCL NAL units with a NAL unit type identifier among the values in nonVclDigitallySignedNalUnitsList and VCL NAL units of the AU.

The digitally signed content initialization SEI message applies to the current coded picture and all following coded pictures until one or more of the following conditions are true:

* The bitstream ends.
* A new CVS begins.
* A new digitally signed content initialization SEI message is received.

When a digitally signed content initialization SEI message is present in an AU of a CVS, a digitally signed content verification SEI message shall be present for each of the substreams that a NAL unit is assigned to. The signed content verification SEI message shall be present in the bitstream before one or more of the following conditions are true:

* The bitstream ends.
* A new CVS begins.
* A new digitally signed content initialization SEI message is received.

**dsci\_hash\_method\_type** indicates the secure hash algorithm that is used to calculate message digests for subsets of non-VCL NAL units with a NAL unit type identifier among the values in nonVclDigitallySignedNalUnitsList and VCL NAL units of the coded video sequence. Based on these message digests and the digital signatures present in digitally signed content verification SEI messages, a decoder can verify that the coded video was produced by the content originator indicated by the syntax elements dsci\_key\_source\_uri, dsci\_use\_key\_register\_idx\_flag and, if dsci\_use\_key\_register\_idx\_flag flag is equal to 1, dsci\_key\_register\_idx. The supported values for the syntax element dsci\_hash\_method\_type, the block size used for calculating the message digest, and the size of the calculated message digests are specified in. Values of dsci\_hash\_method\_type that are not listed in the Table XXXare reserved for future use by ITU-T | ISO/IEC and shall not be present in payload data conforming to this version of this Specification. Decoders shall ignore trustworthy initialization SEI messages that contain reserved values for dsci\_hash\_method\_type. The secure hash algorithms listed in Table XXX are specified in the “Secure Hash Standard” NIST FIPS PUB 180-4.

**Table XXX – Supported values of dsci\_hash\_method\_type**

|  |  |  |  |
| --- | --- | --- | --- |
| **dsci\_hash\_method\_type** | **Hash method** | **Block size (bits)** | **Message digest size (bits)** |
| 0 | SHA-1 | 512 | 160 |
| 1 | SHA-224 | 512 | 224 |
| 2 | SHA-256 | 512 | 256 |
| 3 | SHA-384 | 1024 | 384 |
| 4 | SHA-512 | 1024 | 512 |
| 5 | SHA-512/224 | 1024 | 224 |
| 6 | SHA-512/256 | 1024 | 256 |

**dsci\_key\_source\_uri** contains a URI with syntax and semantics as specified in IETF Internet Standard 66. If dsci\_key\_retrieval\_mode\_idc is equal to 0, dsci\_key\_source\_uri specifies a trust record, as specified in ISO/IEC 21617-1. If dsci\_key\_retrieval\_mode\_idc is equal to 1, the following applies:

* If dsci\_use\_key\_register\_idx\_flag is equal to 0, the URI identifies the certificate of the content provider that can be used for verifying the signatures present in following digitally signed content verification SEI messages;
* Otherwise (if dsci\_use\_key\_register\_idx\_flag is equal to 1), the URI identifies a register of certificates and the certificate of the content provider that can be used for verifying the signatures present in following digitally signed content verification SEI messages as indicated by dsci\_key\_register\_idx.

**dsci\_num\_verification\_substreams\_minus1** plus 1 indicates the number of substreams for which message digests are calculated and signatures may be present in following digitally signed content verification SEI messages.

The variable NumVerificationSubstream is derived as:

NumVerificationSubstream = dsci\_num\_verification\_substreams\_minus1 + 1.

**dsci\_key\_retrieval\_mode\_idc** equal to 0 indicates that the URI contained in dsci\_key\_source\_uri specifies a trust record, as specified in ISO/IEC 21617-1,. dsci\_key\_retrieval\_mode\_idc equal to 1 indicates that the URI contained in dsci\_key\_source\_uri and, when present, dsci\_key\_register\_idx specify a certificate. In this version of this Specification dsci\_key\_retrieval\_mode\_idc shall be in the range of 0 to 1. Decoders shall also allow other values of dsci\_key\_retrieval\_mode\_idc, but shall ignore the content of the digitally signed content initialization SEI message, associated digitally signed content selection SEI messages and associated digitally signed content verification SEI messages.

**dsci\_use\_key\_register\_idx\_flag** equal to 1 indicates that the URI contained in dsci\_key\_source\_uri specifies a register of certificates and the syntax element dsci\_key\_register\_idx is present in the SEI message. dsci\_use\_key\_register\_idx\_flag equal to 0 indicates that the URI contained in dsci\_key\_source\_uri specifies a certificate and the syntax element dsci\_key\_register\_idx is not present in the SEI message.

When dsci\_key\_retrieval\_mode\_idc is equal to 0, the media asset for which the last trust manifest within the trust record, as specified in ISO/IEC 21617-1 provides content binding is the digitally signed content initialization SEI message. The following constraints apply to the trust record, as specified in ISO/IEC 21617-1, identified by the dsci\_key\_source\_uri:

* The last trust manifest within the trust record, as specified in ISO/IEC 21617-1, shall contain exactly one hard binding data hash assertion with a label equal to c2pa.hash.data.
* The schema for data hash assertion is defined by the data-hash-map rule in the following CDDL Definition:

|  |
| --- |
| ; The data structure used to store the cryptographic hash of some or all of the asset's data  ; and additional information required to compute the hash.  data-hash-map = {  ? "exclusions": [1\* EXCLUSION\_RANGE-map], ; Ranges have monotonically increasing `start` values, and no two ranges may overlap.  ? "alg":tstr .size (1..max-tstr-length), ; A string identifying the cryptographic hash algorithm used to compute the hash in this assertion.  "hash": bstr, ; byte string of the hash value  "pad": bstr, ; zero-filled byte string used for filling up space  ? "pad2": bstr, ; optional zero-filled byte string used for filling up space  ? "name": tstr .size (1..max-tstr-length), ; (optional) a human-readable description of what this hash cover  ? "url": uri, ; Unused and deprecated.  }  EXCLUSION\_RANGE-map = {  "start": int, ; Starting byte of the range  "length": int, ; Number of bytes of data to exclude  } |

* The exclusion range specifying the data in the digitally signed content initialization SEI message that is excluded when computing the hash value and indicated in the data hash assertion shall match the dsci\_key\_source\_uri bytes in the digitally signed content initialization SEI message.

**dsci\_key\_register\_idx**, when present, contains an index that specifies the certificate of the content provider, in the certificate register indicated by dsci\_key\_source\_uri, which can be used for verifying the signatures present in following digitally signed content verification SEI messages.

The certificate indicated by the syntax elements dsci\_key\_retrieval\_mode\_idc, dsci\_use\_key\_register\_idx\_flag, dsci\_key\_source\_uri, and, if dsci\_use\_key\_register\_idx\_flag is equal to 1, dsci\_key\_register\_idx shall specify a digital signature method, with associated parameters (if applicable), and the public key of the content provider. When dsci\_key\_retrieval\_mode\_idc is equal to 1, the format in which this information is provided is outside the scope of this specification. It is suggested that a digital signature algorithm conforming to the “Digital Signature Standard” NIST FIPS 186-5 is used.

**dsci\_content\_uuid\_present\_flag e**qual to 1 specifies that the syntax element dsci\_content\_uuid is present. dsci\_content\_uuid\_present\_flag equal to 0 specifies that the syntax element dsci\_content\_uuid is not present. When dsci\_key\_retrieval\_mode\_idc is equal to 0, dsci\_content\_uuid\_present\_flag shall be equal to 1.

**dsci\_content\_uuid**, when present, indicates an identifier for the video content and shall have a value specified as a UUID according to the procedures of ISO/IEC 11578:1996, Annex A.

When a digitally signed content initialization SEI message is present in an AU, the calculation of NumVerificationSubstream message digests is initialized according to the specification in NIST FIPS PUB 180-4 for the specified dsci\_hash\_method\_type. Each non-VCL NAL unit with a NAL unit type identifier among the values in nonVclDigitallySignedNalUnitsList and VCL NAL unit following the digitally signed content initialization SEI message is associated to one of the NumVerificationSubstream message digests; the verification substream id is either indicated by the digitally signed content selection SEI message or, if no digitally signed content selection SEI message is present for a PU, inferred to be equal to 0. The message used for calculating the k-th message digest, with k being in the range from 0 to dsci\_num\_verification\_substreams\_minus1, inclusive, is obtained by concatenating all non-VCL NAL units with a NAL unit type identifier among the values in nonVclDigitallySignedNalUnitsList and VCL NAL units associated with the k-th verification substream. The calculation of the message digests is conducted based on blocks, where the block size is specified in Table XXX depending on the value of dsci\_hash\_method\_type. For each non-VCL NAL unit with a NAL unit type identifier among the values in nonVclDigitallySignedNalUnitsList and VCL NAL unit, the associated message digest is updated according to the algorithm specified in NIST FIPS PUB 180-4 for the specified dsci\_hash\_method\_type. Note that, since the message digests are calculated for the concatenation of all non-VCL NAL units with a NAL unit type identifier among the values in nonVclDigitallySignedNalUnitsList and VCL NAL units for a verification substream, some of the processing blocks typically span over two or more successive NAL units.

* 1. **Digitally signed content selection SEI message**
     1. **Digitally signed content selection SEI message syntax**

|  |  |
| --- | --- |
| digitally\_signed\_content\_selection( payloadSize ) { | **Descriptor** |
| **dscs\_verification\_substream\_id** | u(8) |
| } |  |

* + 1. **Digitally signed content selection SEI message semantics**

Use of this SEI message requires the definition of the following:

– A lists of non-VCL NAL unit type identifiers nonVclDigitallySignedNalUnitsList.

The digitally signed content selection SEI message provides a mechanism for associating coded pictures with one of the verification substreams indicated in a digitally signed content initialization SEI message.

When an AU contains both a digitally signed content initialization SEI message and a digitally signed content selection SEI message, the digitally signed content initialization SEI message shall precede the digitally signed content selection SEI message in decoding order.

When a CVS does not contain a digitally signed content initialization SEI message, CLVSs of the CVS shall not contain a digitally signed content selection SEI message.

When a digitally signed content selection SEI message is present in any AU of a CVS, it shall precede all non-VCL NAL units with a NAL unit type identifier among the values in nonVclDigitallySignedNalUnitsList and VCL NAL units of the AU.

**dscs\_verification\_substream\_id** indicates the verification substream to which the non-VCL NAL units with a NAL unit type identifier among the values in nonVclDigitallySignedNalUnitsList and VCL NAL units of the current coded picture are assigned to. When a digitally signed content initialization SEI message was present in the current coded video sequence, but no digitally signed content selection SEI message is present for a coded picture, the value of dscs\_verification\_substream\_id is inferred to be equal to 0. The value of dscs\_verification\_substream\_id shall be in the range from 0 to dsci\_num\_verification\_substreams\_minus1, inclusive.

As specified in section 3.1.2, the message digest for the verification substream with id equal to dscs\_verification\_substream\_id is updated with the non-VCL NAL units with a NAL unit type identifier among the values in nonVclDigitallySignedNalUnitsList and VCL NAL units of the current coded picture according to the dsci\_hash\_method\_type specified in the preceding digitally signed content initialization SEI message.

* 1. **Digitally signed content verification SEI message**
     1. **Digitally signed content verification SEI message syntax**

|  |  |
| --- | --- |
| digitally\_signed\_content\_verification( payloadSize ) { | **Descriptor** |
| **dscv\_verification\_substream\_id** | u(8) |
| **dscv\_signature\_length\_in\_octets\_minus1** | u(16) |
| **dscv\_signature** | u(v) |
| } |  |

* + 1. **Digitally signed content verification SEI message semantics**

Use of this SEI message requires the definition of the following:

– A lists of non-VCL NAL unit type identifiers nonVclDigitallySignedNalUnitsList.

The digitally signed content verification SEI message provides a mechanism for verifying the digital signature of video content.

When a CVS does not contain a digitally signed content initialization SEI message, CLVSs of the CVS shall not contain digitally signed content verification SEI message.

When an AU contains both a digitally signed content initialization SEI message and a digitally signed content verification SEI message, the digitally signed content initialization SEI message shall precede the digitally signed content verification SEI message. When a PU contains both a digitally signed content selection SEI message and a digitally signed content verification SEI message, the digitally signed content selection SEI message shall precede the digitally signed content verification SEI message.

When a digitally signed content verification SEI message is present in a PU of a CVS, no non-VCL NAL units with a NAL unit type identifier among the values in nonVclDigitallySignedNalUnitsList or VCL NAL unit shall be assigned to the substream that is indicated by dscv\_verification\_substream\_id, unless one or more of the following conditions are true:

* The bitstream ends.
* A new CVS begins.
* A new digitally signed content initialization SEI message is received.

**dscv\_verification\_substream\_id** indicates the verification substream to which the SEI message applies.

**dscv\_signature\_length\_in\_octets\_minus1** plus 1 specifies the length of the syntax element dscv\_signature in octets (one octet consists of 8 bits).

**dscv\_signature** contains the digital signature for the verification substream indicated by dscv\_verification\_substream\_id.

The verification of the bitstream signature consists of the following ordered steps:

1. The calculation of the message digest referred to as CurrDigest is finalized as follows:
   * The concatenation of the non-VCL NAL units with a NAL unit type identifier among the values in nonVclDigitallySignedNalUnitsList and VCL NAL units for the verification substream with id equal to dscv\_verification\_substream\_id is padded according to the specification in NIST FIPS PUB 180-4. Note that it is sufficient to pad the last NAL unit of the verification substream.
   * The calculation of the message digest CurrDigest is finalized according to the specification in NIST FIPS PUB 180-4. The length (in bits) of the message digest is given in Table XXX.
2. The reference message digest RefDigest is determined as follows:
   * If dscv\_verification\_substream\_id is greater than 0, the reference message digest RefDigest is the last calculated message digest for the verification substream with id equal to dscv\_verification\_substream\_id – 1. It is a requirement of bitstream conformance that any digitally signed content verification SEI associated with verification substream id equal to dscv\_verification\_substream\_id – 1 is present before the digitally signed content verification SEI message with verification substream id equal to dscv\_verification\_substream\_id.
   * Otherwise, if the current digitally signed content verification SEI message is the first digitally signed content verification SEI with verification id equal to 0 in the coded video sequence and the preceding coded video sequence did not contain any digitally signed content initialization SEI message (this includes the case that the current coded video sequence is the first coded video sequence in the bitstream), the RefDigest is set equal to a bitstring that consists of DigestSize bits equal to 1, where DigestSize is the size of the message digest as specified in Table XXX.
   * Otherwise, the reference message digest RefDigest is the last calculated message digest for the verification substream with id equal to 0.
3. The identification string IdString is constructed by concatenating the binary representations of the reference message digest RefDigest, the current message digest, and the dsci\_hash\_method\_type and, when present, the dsci\_content\_uuid, as illustrated in Figure XXX.

|  |  |  |  |
| --- | --- | --- | --- |
| RefDigest | CurrDigest | dsci\_hash\_method\_type | dsci\_content\_uuid (when present) |

Figure XXX – Construction of identification string IdString

The number of bits for RefDigest is determined by the value of dsci\_hash\_method\_type which was valid when calculating the value of RefDigest, the number of bits for CurrDigest is determined by the current value of dsci\_hash\_method\_type, and the value of dsci\_hash\_method\_type is represented with 8 bits and, when present, the value of dsci\_content\_uuid is represented with 128 bits.

1. The identification string IdString represents the message used for verifying the signature. The signature verification algorithm and the public key used for verifying the signature are indicated by the syntax elements dsci\_use\_key\_register\_idx\_flag, dsci\_key\_source\_uri, and, if dsci\_use\_key\_register\_idx\_flag is equal to 1, dsci\_key\_register\_idx.

NOTE 1 – Since the bitstring used for signature verification includes the RefDigest, it cannot only be verified that the non-VCL NAL units with a NAL unit type identifier among the values in nonVclDigitallySignedNalUnitsList and VCL NAL units used for calculating the current message digest are correct, but it can additionally be verified that neither additional non-VCL NAL units with a NAL unit type identifier among the values in nonVclDigitallySignedNalUnitsList and VCL NAL units were added to the bitstream nor non-VCL NAL units with a NAL unit type identifier among the values in nonVclDigitallySignedNalUnitsList and VCL NAL units were removed from the bitstream.

NOTE 2 – When a decoder tunes into a bitstream, the IdString constructed for the first digitally signed content verification SEI message cannot be verified, because the value of RefDigest cannot be calculated correctly. But starting from the second digitally signed content verification SEI message, the signatures can be verified.

After verification, the message digest for the verification substream with id equal to dscv\_verification\_substream\_id is reinitialized according to the specification in NIST FIPS PUB 180-4 for the specified dsci\_hash\_method\_type.