|  |
| --- |
| **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 258** |
| **Online – 17–26 January 2024** |
| |  |  | | --- | --- | | **Source:** | **Convenor (Jens-Rainer Ohm)** | | **Title:** | **Preliminary WD: SEI messages for VSEI version 4** | | **Type:** | **Project** | | **Subtype:** | **Other** | | **Status:** | **Approved** | | **Date:** | **2024-03-29** | | **Expected Action:** | **Info** | | **Action due date:** | **N/A** | | **Pages:** | **39** (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**  33rd Meeting, by teleconference, 17–26 January 2024 | Document: JVET-AG2034-v1 |

|  |  |  |  |
| --- | --- | --- | --- |
| *Title:* | **SEI messages for VSEI version 4 (Draft 1)** | | |
| *Status:* | Output document approved by JVET | | |
| *Purpose:* | Draft text | | |
| *Author(s) or Contact(s):* | Sean McCarthy Jie Chen Sachin Deshpande Miska M. Hannuksela Hendry Gary J. Sullivan Ye-Kui Wang | Email: | sean.mccarthy@dolby.com jiechen.cj@alibaba-inc.com sdeshpande@sharplabs.com miska.hannuksela@nokia.com dr.hendry@lge.com gary.sullivan@dolby.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, and modality information SEI messages and updates to the neural-network post-filter characteristics SEI message.

# Changes yet to be integrated:

# Changes that have been integrated:

*Changes incorporate: 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 signaling*
* *payloadType values for the added SEI message (VVC text). [Ed. (MH): 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.]*

# 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

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

*In clause 8.1, 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 |
| 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 |

*Modify subclause 8.28.1 as follows:*

**8.28.1 General** **post-processing filtering process using NNPFs**

**8.28.1.1 General**

Input to this process is a bitstream BitstreamToFilter. Output of this process is a list of NNPF output pictures ListNnpfOutputPics.

First, BitstreamToFilter is decoded, and the list CroppedDecodedPictures is set to be the list of the cropped decoded pictures in output order resulted from decoding BitstreamToFilter.

Second, the filtering process for one picture, as specified in clause 8.28.1.2, is repeatedly invoked, in output order, for each cropped decoded picture that is in CroppedDecodedPictures and for which one or more NNPFs are activated.

The order of the pictures in ListNnpfOutputPics is in output order.

Within ListNnpfOutputPics there shall be no more than one picture pertaining to any particular output time instance. When for any particular picture in CroppedDecodedPictures there are multiple NNPFs activated and only one of the NNPFs is allowed to be chosen to be applied although any of the NNPFs may be chosen, the above constraint shall apply regardless of which NNPF is chosen to be applied to the particular picture.

For any particular pair of pictures inputPicA and inputPicB consecutive in output order in CroppedDecodedPictures, when there are one or more pictures intermediatePicSetA in ListNnpfOutputPics between inputPicA and inputPicB in output order, one and only one of the following shall apply:

– The pictures in intermediatePicSetA shall be among the pictures that were output by applying a particular NNPF nnpfA with PictureRateUpsamplingFlag equal to 1 when a particular picture currPicA in CroppedDecodedPictures was the current picture.

– The pictures in intermediatePicSetA shall be among the pictures that were output by applying a particular NNPF nnpfA with TemporalExtrapolationFlag equal to 1 when a particular picture currPicA in CroppedDecodedPictures was the current picture.

The application of any other NNPF that was used in the filtering process for one picture when currPicA was the current picture or the application of any NNPF (including nnpfA) that was used in the filtering process for one picture when any other picture currPicB in CroppedDecodedPictures was the current picture shall not output any picture between the inputPicA and inputPicB in output order.

NOTE – The intent of the constraints expressed in the above paragraph is to disallow generating NNPF output pictures between any particular pair of consecutive input pictures more than once.

**8.28.1.2 Filtering process for one picture using an NNPF**

The filtering process specified in this clause applies to each cropped decoded picture, referred to as the current picture, that is in CroppedDecodedPictures and for which one or more NNPFs are activated.

When applying an NNPF to the current picture, the following applies:

– The filtered and/or interpolated pictures are generated by the NNPF by applying the NNPF process specified in the semantics of the NNPFC SEI message, in a patch-wise manner, to the current picture.

– The order of the pictures generated by the NNPF by applying the NNPF process being stored into the output tensor of the NNPF is in output order.

When the applied NNPF is the last NNPF that is applied to the current picture, the pictures generated by the NNPF and output by the NNPF process are included into ListNnpfOutputPics, in the same order as when the pictures are stored into the output tensor of the NNPF.

*Modify clause 8.28.2 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** | u(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) |
| **nnpfc\_component\_last\_flag** | u(1) |
| **nnpfc\_inp\_format\_idc** | ue(v) |
| **nnpfc\_auxiliary\_inp\_idc** | ue(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) |
| **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 ) |  |
| **nnpfc\_application\_purpose\_tag\_uri** | st(v) |
| } |  |
| **nnpfc\_reserved\_metadata\_extension** | u(v) |
| } |  |
| /\* ISO/IEC 15938-17 bitstream \*/ |  |
| if( nnpfc\_mode\_idc  = =  0 ) { |  |
| while( !byte\_aligned( ) ) |  |
| **nnpfc\_alignment\_zero\_bit\_b** | u(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 127, inclusive, in bitstreams conforming to this version of this Specification. Values of 128 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 128 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) |

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

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 variable nnpfcOutputPicWidth, representing the width of the luma sample arrays of the NNPF-generated pictures, is derived as follows:

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

It is a requirement of bitstream conformance that the value of nnpfcOutputPicWidth % 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, representing the height of the luma sample arrays of the NNPF-generated pictures, is derived as follows:

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

It is a requirement of bitstream conformance that the value of nnpfcOutputPicHeight % 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.

**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\_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 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 2 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 2 to 255, inclusive.

**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, one luma matrix and one auxiliary input matrix are present, and the number of channels is 2. |
| 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, two chroma matrices and one auxiliary input matrix are present, and the number of channels is 3. |
| 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, one luma matrix, two chroma matrices and one auxiliary input matrix are present, and the number of channels is 4. |
| 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, four luma matrices, two chroma matrices, and one auxiliary input matrix are present in the input tensor, and the number of channels is 7. 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.

**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.

**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 outPatchWidth \* CroppedWidth shall be equal to nnpfcOutputPicWidth \* inpPatchWidth and outPatchHeight \* CroppedHeight shall be equal to nnpfcOutputPicHeight \* 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 )  
 if( !nnpfc\_component\_last\_flag )  
 inputTensor[ 0 ][ i ][ 1 ][ yPovlp ][ xPovlp ] = strengthControlScaledVal[ i ]  
 else  
 inputTensor[ 0 ][ i ][ yPovlp ][ xPovlp ][ 1 ] = strengthControlScaledVal[ i ]  
 }  
 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 )  
 if( !nnpfc\_component\_last\_flag )  
 inputTensor[ 0 ][ i ][ 2 ][ yPovlp ][ xPovlp ] = strengthControlScaledVal[ i ]  
 else  
 inputTensor[ 0 ][ i ][ yPovlp ][ xPovlp ][ 2 ] = strengthControlScaledVal[ i ]  
 }  
 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 )  
 if( !nnpfc\_component\_last\_flag )  
 inputTensor[ 0 ][ i ][ 3 ][ yPovlp ][ xPovlp ] = strengthControlScaledVal[ i ]  
 else  
 inputTensor[ 0 ][ i ][ yPovlp ][ xPovlp ][ 3 ] = strengthControlScaledVal[ i ]  
 }  
 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 )  
 if( !nnpfc\_component\_last\_flag )  
 inputTensor[ 0 ][ i ][ 6 ][ yPovlp ][ xPovlp ] = strengthControlScaledVal[ i ]  
 else  
 inputTensor[ 0 ][ i ][ yPovlp ][ xPovlp ][ 6 ] = strengthControlScaledVal[ i ]  
 }  
}

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 ) {  
 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)  
 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 ) {  
 inputTensor = DeriveInputTensors( )  
 outputTensor = PostProcessingFilter( inputTensor )  
 StoreOutputTensors( outputTensor )  
 }

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\_reserved\_metadata\_extension is not present. nnpfc\_num\_metadata\_extension\_bits greater than 0 specifies the length, in bits, of nnpfc\_reserved\_metadata\_extension.

The value of nnpfc\_num\_metadata\_extension\_bits shall be in the range of 0 to 2 048, inclusive in this edition of this document when nnpfc\_purpose is not equal to 0 and in the range of 1 to 2048 when nnpfc\_prupose is equal to 0. Values in the range of 2049 to 4096 , 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 4096, 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\_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 4 – nnpfc\_application\_purpose\_tag\_uri enables uniquely identifying the application determined purpose of NNPF without needing a central registration authority.

**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, and when nnpfc\_purpose is equal to 0 and nnpfc\_application\_purpose\_tag\_uri\_present\_flag is equal to 1 the length, in bits, of nnpfc\_reserved\_metadata\_extension is equal to nnpfc\_num\_metadata\_extension\_bits – Length of (nnpfc\_application\_purpose\_tag\_uri) – 1. When present and when nnpfc\_purpose is equal to 0 and nnpfc\_application\_purpose\_tag\_uri\_present\_flag is equal to 0 the length, in bits, of nnpfc\_reserved\_metadata\_extension is equal to nnpfc\_num\_metadata\_extension\_bits – 1 bits. When present and when nnpfc\_purpose is not equal to 0 the length, in bits, of nnpfc\_reserved\_metadata\_extension is equal to nnpfc\_num\_metadata\_extension\_bits.

**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.

*Add clauses 8.30 to 8.33 as follows:*

* 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\_type** | u(16) |
| if( EoiObjectBasedFlag ) |  |
| **eoi\_object\_based\_idc** | ue(v) |
| if( EoiTemporalResamplingFlag ) { |  |
| **eoi\_temporal\_resampling\_type\_flag** | u(1) |
| **eoi\_num\_int\_pics** | ue(v) |
| } |  |
| if( EoiPrivacyProtectionFlag ) { |  |
| **eoi\_privacy\_protection\_type\_idc** | u(4) |
| **eoi\_privacy\_protected\_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.

**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 7, inclusive, in bitstreams conforming to this version of this Specification. Values of 8 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 8 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. For example, an encoding system can overwrite areas outside the detected objects with a constant sample value. |

**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\_privacy\_protection\_type\_idc**, when present, indicates the type of privacy protection optimization as specified in Table x3.

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

|  |  |
| --- | --- |
| **eoi\_privacy\_protection\_type\_idc** | **Interpretation** |
| 0 | Unknown. |
| 1 | Blurring; personal information is blurred to make it unidentifiable. |
| 2 | Replacing; personal information is replaced with something different from the original to make it unidentifiable. |
| 3 | Masking; personal information is masked so that it cannot be identified |
| 4…15 | Reserved for future use. |

**eoi\_privacy\_protected\_info\_type**, when present, indicates the types of protected information as specified in Table x4 where ( eoi\_privacy\_protected\_info\_type & bitMask ) not equal to 0 indicates that the information type with the bitMask value in Table x4 has been protected. The value of eoi\_privacy\_protection\_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\_protected\_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\_protected\_info\_type is in the range of 8 to 255, inclusive, decoders conforming to this version of this Specification shall ignore eoi\_privacy\_protected\_info\_type.

**Table x4 – Definition of** **eoi\_privacy\_protection\_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) |
| if( spti\_persistence\_flag ) |  |
| **spti\_max\_sublayers\_minus\_1** | u(3) |
| for( i = 0; 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, e.g., 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. [Ed. Check phrasing of this. “Pertains” is not used in a similar way anywhere in the standard.]

**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. |

The value of ( spti\_source\_type & 0x04 ) & ( spti\_source\_type & 0x08 ) shall be zero (i.e., spti\_source\_type shall not simultaneously indicate high-speed imaging and time-lapse imaging).

**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 1 – 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\_max\_sublayers\_minus\_1** 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\_minus\_1 is not present, it is inferred to be equal to TemporalId..

**spti\_sublayer\_interval\_scale\_factor**[ i ], when present, specifies a scale factor used in determining the source picture interval of corresponding pictures in the CLVS having TemporalId equal to i relative to the previous 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 indicated source picture interval associated with an output picture having TemporalId equal to i, relative to the previous output picture with TemporalId less than or 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\_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 2 –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 2 – 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\_aux\_id\_minus128** | ue(v) |
| **omi\_num\_primary\_pic\_layer\_minus1** | ue(v) |
| for( i = 0; i <= omi\_num\_primary\_pic\_layer\_minus1; i++ ) { |  |
| **omi\_primary\_pic\_layer\_id**[ i ] | ue(v) |
| **omi\_num\_aux\_pic**[ i ] | 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\_primary\_pic\_layer\_minus1; i++ ) |  |
| for( j = 0; j < omi\_num\_aux\_pic[ i ]; j++ ) { |  |
| **omi\_mask\_pic\_update\_flag**[ i ][ j ] | f(1) |
| if( omi\_mask\_pic\_update\_flag[ i ][ j ] ) { |  |
| **omi\_num\_mask\_in\_pic\_update**[ i ][ j ] | ue(v) |
| for( k = 0; k < omi\_num\_mask\_in\_pic\_update[ i ][ j ]; k++ ) { |  |
| **omi\_mask\_id**[ i ][ j ][ k ] | u(v) |
| **omi\_aux\_sample\_value**[ i ][ j ][ k ] | u(v) |
| **omi\_mask\_bounding\_box\_present\_flag**[ i ][ j ][ k ] | u(1) |
| if( omi\_mask\_bounding\_box\_present\_flag[ i ][ j ][ k ] ) { |  |
| **omi\_mask\_top**[ i ][ j ][ k ] | u(16) |
| **omi\_mask\_left**[ i ][ j ][ k ] | u(16) |
| **omi\_mask\_width**[ i ][ j ][ k ] | u(16) |
| **omi\_mask\_height**[ i ][ j ][ k ] | u(16) |
| } |  |
| **omi\_mask\_cancel**[ i ][ j ][ k ] | u(1) |
| if( !omi\_mask\_cancel[ i ][ j ][ k ] ) { |  |
| if( omi\_mask\_confidence\_info\_present\_flag ) |  |
| **omi\_mask\_confidence**[ i ][ j ][ k ] | u(v) |
| if( omi\_mask\_depth\_info\_present\_flag ) |  |
| **omi\_mask\_depth**[ i ][ j ][ k ] | u(v) |
| while( !byte\_aligned( ) ) |  |
| **omi\_bit\_equal\_to\_zero** | f(1) |
| if( omi\_mask\_label\_info\_present\_flag ) |  |
| **omi\_mask\_label**[ i ][ j ][ k ] | st(v) |
| } |  |
| } |  |
| } |  |
| } |  |
| } |  |
| } |  |

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

The object mask information (OMI) SEI message provides information about object mask pictures coded as auxiliary pictures. Object mask auxiliary pictures have nuh\_layer\_id equal to sdi\_layer\_id[ i ] and sdi\_aux\_id[ i ] in the range of 128 to 159, inclusive, for any value of i in range of 0 to sid\_max\_layers\_minus1, inclusive.

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

– 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.

When an access unit contains an auxiliary picture picA in a layer, with nuh\_layer\_id equal to nuhLayerIdA, that is indicated as an object mask auxiliary layer by an OMI SEI message, and a primary picture picB in a layer, with nuh\_layer\_id equal to nuhLayerIdB, that is indicated as a primary layer by the OMI SEI message, OMI SEI message persists in output order until one or more of the following conditions are true:

– A CLVS containing the auxiliary picture picA ends.

– A CLVS containing the primary picture picB ends.

– A CVS ends.

– The bitstream ends.

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

**omi\_aux\_id\_minus128** plus 128 indicates the value of sdi\_aux\_id of object mask auxiliary picture layer. om\_aux\_id\_minus128 shall be in the range of 0 to 31, inclusive.

When a CVS does not contain an SDI SEI message with sdi\_aux\_id[ i ] equal to omi\_aux\_id\_minus128 + 128 for at least one value of i, no picture in the CVS shall be associated with an OMI SEI message.

When an AU contains both an SDI SEI message with sdi\_aux\_id[ i ] equal to omi\_aux\_id\_minus128 + 128 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\_primary\_pic\_layer\_minus1** plus 1indicates the number of primary picture layers associated with the object mask auxiliary picture layers to which this SEI message applies. The value of omi\_num\_primary\_pic\_layer\_minus1 shall be in the range of 0 to sdi\_max\_layers\_minus1.

**omi\_primary\_pic\_layer\_id**[ i ] specifies the nuh\_layer\_id value of the i-th primary picture layer that is associated with the object mask auxiliary picture layers to which this OMI SEI message applies. The value of sdi\_aux\_id[ j ] shall be equal to 0 for any value of j in the range of 0 to sid\_max\_layers\_minus1, inclusive, if sdi\_layer\_id[ j ] equal to omi\_primary\_pic\_layer\_id[ i ].

**omi\_num\_aux\_pic**[ i ] indicates the number of auxiliary picture layers associated with the i-th primary picture layer that is associated with the object mask auxiliary picture layers.It is a requirement of bitstream conformance that the value of omi\_num\_aux\_pic[ i ] shall be equal to numAuxLayer[omi\_primary\_pic\_layer\_id[ i ]] for i form 0 to omi\_num\_primary\_pic\_layer\_minus1, inclusive, where the variable numAuxLayer[ primaryLayerId ] indicating the number of the object mask auxiliary picture layers associated with primary picture layer with nuh\_layer\_id equal to primaryLayerId is derived as follows.

for( i = 0; i <= sdi\_max\_max\_layers\_minus1; i++ )  
 numAuxLayer[ sdi\_layer\_id[ i ] ] = 0;  
for( i = 0; i <= sdi\_max\_layers\_minus1; i++ ) {  
 if( sdi\_aux\_id[ i ] = = omi\_aux\_id\_minus128 + 128 ) {  
 for( j = 0; j <= sdi\_num\_associated\_primary\_layers\_minus1[ i ]; j++ ) { (xx)  
 primaryLayerId = sdi\_layer\_id[ sdi\_associated\_primary\_layer\_idx[ i ][ j ] ]; numAuxLayer[ primaryLayerId ]++;  
 }  
 }  
}

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

**omi\_mask\_sample\_value\_length\_minus8** plus 8 specifies the length, in bits, of omi\_aux\_sample\_value[ i ][ j ][ k ] syntax elements. The value of omi\_mask\_sample\_value\_length\_minus8 shall be in the range of 0 to 8. **omi\_mask\_confidence\_info\_present\_flag** equal to 1 indicates that omi\_mask\_confidence[ i ][ j ][ k ] syntax elements are present. omi\_mask\_confidence\_info\_present\_flag equal to 0 indicates that omi\_mask\_confidence[ i ][ j ][ k ]syntax elements are not present.

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

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

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

It is a requirement of bitstream conformance that the value of omi\_aux\_id\_minus128, omi\_num\_primary\_pic\_layer\_minus1, omi\_primary\_pic\_layer\_id[ i ], omi\_num\_aux\_pic[ i ], omi\_mask\_id\_length\_minus1 and omi\_mask\_sample\_value\_length\_minus8, omi\_mask\_confidence\_info\_present\_flag, omi\_mask\_confidence\_length\_minus1, omi\_mask\_depth\_info\_present\_flag and omi\_mask\_depth\_length\_minus1 shall be the same for all object\_mask\_info( ) syntax structures within a CVS.

**omi\_mask\_label\_info\_present\_flag** equal to 1 indicates that omi\_mask\_label\_language\_present\_flag and omi\_mask\_label[ i ][ j ][ k ] 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 ][ k ] 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 ][ j ] equal to 1 indicates the mask information of j-th object mask auxiliary picture associated with i-th primary picture is signalled. omi\_mask\_pic\_update\_flag[ i ][ j ] equal to 0 indicates the mask information of j-th object mask auxiliary picture associated with i-th primary picture is not signalled. When the mask information of j-th object mask auxiliary picture associated with i-th primary picture is not present, the persistence mechanism is used, that is the information is inherited from the last OMI SEI message which signals the mask information of j-th object mask auxiliary picture associated with i-th primary picture.

**omi\_num\_mask\_in\_pic\_update**[ i ][ j ] indicates the number of object masks of which the information is signalled in the j-th object mask auxiliary picture associated with i-th primary picture. omi\_num\_mask\_in\_pic\_update [ i ][ j ] shall be in the range of 0 to (1<<(omi\_mask\_id\_length\_minus1 + 1)) – 1, inclusive.

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

The variable maskId[ i ][ j ][ k ] specifying the global identifier of k-th signaled object mask in the j-th object mask auxiliary picture associated with i-th primary picture in the SEI message is derived as follows:

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

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

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

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

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

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

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

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

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

for( i = 0; i <= omi\_num\_primary\_pic\_layer\_minus1; i++ ) {  
 for( j = 0; j < omi\_num\_aux\_pic[ i ]; j++ ) {   
 for( k = 0; k < omi\_num\_mask\_in\_pic\_update[ i ][ j ]; k++ ) { (xx)  
 if( pI[ i ][ j ][ x ][ y ] == omi\_aux\_sample\_value [ i ][ j ][ k ]  
 && x >= omi\_mask\_left[ i ][ j ][ k ]   
 && x < omi\_mask\_left[ i ][ j ][ k ] + omi\_mask\_width[ i ][ j ][ k ]  
 && y >= omi\_mask\_top[ i ][ j ][ k ]   
 && y < omi\_mask\_top[ i ][ j ][ k ] + omi\_mask\_height[ i ][ j ][ k ] )  
 The sample at location (x, y) in the cropped j-th object mask auxiliary picture associated   
 with the i-th primary picture is associated with the object mask with the identifier of  
 maskId[ i ][ j ][ k ]  
 }  
 }  
}

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

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

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

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

**omi\_mask\_label**[ i ][ j ][ k ] specifies the contents of the label associated with k-th signaled object mask in the j-th object mask auxiliary picture associated with i-th primary picture. The length of the omi\_mask\_label[ i ][ j ][ k ] 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 meters as specified in ISO/IEC 80000.

**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.

*In the syntax table of subclause D.2.1, add payloadType values 215 to 218 to a future edition or ammendment of Recommendation ITU-T H.266 | International Standard ISO/IEC 23090-3 as follows:*

|  |  |
| --- | --- |
| sei\_payload( payloadType, payloadSize ) { | **Descriptor** |
| SeiExtensionBitsPresentFlag = 0 |  |
| if( nal\_unit\_type = = PREFIX\_SEI\_NUT ) |  |
| if( payloadType = = 0 ) |  |
| buffering\_period( payloadSize ) |  |
| else if( payloadType = = 1 ) |  |
| pic\_timing( payloadSize ) |  |
| … |  |
| else if( payloadType = = 213 ) |  |
| sei\_processing\_order( payloadSize ) |  |
| else if( payloadType = = 214 ) |  |
| processing\_order\_nesting( payloadSize ) |  |
| else if( payloadType = = 215 ) /\* Specified in Rec. ITU-T H.274 | ISO/IEC 23002-7 \*/ |  |
| encoder\_optimization\_info( payloadSize ) |  |
| else if( payloadType = = 216 ) /\* Specified in Rec. ITU-T H.274 | ISO/IEC 23002-7 \*/ |  |
| source\_picture\_timing\_info( payloadSize ) |  |
| else if( payloadType = = 217 ) /\* Specified in Rec. ITU-T H.274 | ISO/IEC 23002-7 \*/ |  |
| object\_mask\_info( payloadSize ) |  |
| else if( payloadType = = 218 ) /\* Specified in Rec. ITU-T H.274 | ISO/IEC 23002-7 \*/ |  |
| modality\_info( payloadSize ) |  |
| else /\* Specified in Rec. ITU-T H.274 | ISO/IEC 23002-7 \*/ |  |
| reserved\_message( payloadSize ) |  |
| else /\* nal\_unit\_type = = SUFFIX\_SEI\_NUT \*/ |  |
| if( payloadType = = 3 ) /\* Specified in Rec. ITU-T H.274 | ISO/IEC 23002-7 \*/ |  |
| filler\_payload( payloadSize ) |  |
| else if( payloadType = = 4 ) /\* Specified in Rec. ITU-T H.274 | ISO/IEC 23002-7 \*/ |  |
| user\_data\_registered\_itu\_t\_t35( payloadSize ) |  |
| else if( payloadType = = 5 ) /\* Specified in Rec. ITU-T H.274 | ISO/IEC 23002-7 \*/ |  |
| user\_data\_unregistered( payloadSize ) |  |
| else if( payloadType = = 132 ) /\* Specified in Rec. ITU-T H.274 | ISO/IEC 23002-7 \*/ |  |
| decoded\_picture\_hash( payloadSize ) |  |
| else if( payloadType = = 133 ) |  |
| scalable\_nesting( payloadSize ) |  |
| else if( payloadType = = 210 ) /\* Specified in Rec. ITU-T H.274 | ISO/IEC 23002-7 \*/ |  |
| nn\_post\_filter\_characteristics( payloadSize ) |  |
| else if( payloadType = = 211 ) /\* Specified in Rec. ITU-T H.274 | ISO/IEC 23002-7 \*/ |  |
| nn\_post\_filter\_activation( payloadSize ) |  |
| else if( payloadType = = 214 ) |  |
| processing\_order\_nesting( payloadSize ) |  |
| else if( payloadType = = 215 ) /\* Specified in Rec. ITU-T H.274 | ISO/IEC 23002-7 \*/ |  |
| encoder\_optimization\_info( payloadSize ) |  |
| else if( payloadType = = 217 ) /\* Specified in Rec. ITU-T H.274 | ISO/IEC 23002-7 \*/ |  |
| object\_mask\_info( payloadSize ) |  |
| else /\* Specified in Rec. ITU-T H.274 | ISO/IEC 23002-7 \*/ |  |
| reserved\_message( payloadSize ) |  |
| if( SeiExtensionBitsPresentFlag | | more\_data\_in\_payload( ) ) { |  |
| if( payload\_extension\_present( ) ) |  |
| **sei\_reserved\_payload\_extension\_data** | u(v) |
| **sei\_payload\_bit\_equal\_to\_one** /\* equal to 1 \*/ | f(1) |
| while( !byte\_aligned( ) ) |  |
| **sei\_payload\_bit\_equal\_to\_zero** /\* equal to 0 \*/ | f(1) |
| } |  |
| } |  |

*Text to be added to a future edition or ammendment of Recommendation ITU-T H.266 | International Standard ISO/IEC 23090-3:*

**Use of the source picture timing SEI message in VVC**

For purposes of interpretation of the source picture timing SEI message, the following variable is specified:

– TemporalId is set equal to TemporalId.

*Text to be added to a future edition or ammendment of Recommendation ITU-T H.265 | International Standard ISO/IEC 23008-2:*

**Use of the source picture timing SEI message in HEVC**

For purposes of interpretation of the source picture timing SEI message, the following variable is specified:

– TemporalId is set equal to TemporalId.

*Text to be added to a future edition or ammendment of Recommendation ITU-T H.264 | International Standard ISO/IEC 14496-10:*

**Use of the source picture timing SEI message in AVC**

For purposes of interpretation of the source picture timing SEI message, the following variable is specified:

– If the bitstream conforms to any of the profiles defined in Annex G, H, I, or J, TemporalId is set equal to temporal\_id.

– Otherwise (the bitstream conforms to a profile defined in Annex A), TemporalId is set equal to ( nal\_ref\_idc = = 0 ? 1 : 0 ).