**COMMITTEE DRAFT AMENDMENT****© ISO/IEC 2024 – All rights reserved****Text of ISO/IEC 23002-7:202x (3rd Ed.)/CDAM 1****63****Part 7: Versatile supplemental enhancement information messages for coded video bitstreams, AMENDMENT 1: Additional SEI****Information technology — MPEG video technologies****Élément introductif — Élément central — Partie 7: Titre de la partie****Information technology — MPEG video technologies — Part 7: Versatile supplemental enhancement information messages for coded video bitstreams, AMENDMENT 1: Additional SEI****E****2024-09-05****(30) Committee****ISO/IEC****ISO/IEC J****202x (3rd Ed.)****1****Amendment****International Standard****202x****304****ISO/IEC 23002‑****ISO/IEC 23002‑7****ISO/IEC 23002-7:202x (3rd Ed.)/CDAM 1****JISC****Coding of audio, picture, multimedia and hypermedia information****Information technology****5****29****1** **2****見出し 2****見出し 1****0****2****STD Version 2.1c2****30** **4** **ISO/IEC JTC 1/SC 29 /WG 5 N 304**

Date: **2024-09-05**

**Text of ISO/IEC 23002-7:202x (3rd Ed.)/CDAM 1**

ISO/IEC JTC 1/SC 29/WG 5

Secretariat:  JISC

**Information technology — MPEG video technologies — Part 7: Versatile supplemental enhancement information messages for coded video bitstreams, AMENDMENT 1: Additional SEI**

*Élément introductif — Élément central — Partie 7: Titre de la partie*

|  |
| --- |
| **Warning**  This document is not an ISO International Standard. It is distributed for review and comment. It is subject to change without notice and may not be referred to as an International Standard.  Recipients of this draft are invited to submit, with their comments, notification of any relevant patent rights of which they are aware and to provide supporting documentation |

**Copyright notice**

This ISO document is a working draft or committee draft and is copyright-protected by ISO. While the reproduction of working drafts or committee drafts in any form for use by participants in the ISO standards development process is permitted without prior permission from ISO, neither this document nor any extract from it may be reproduced, stored or transmitted in any form for any other purpose without prior written permission from ISO.

Requests for permission to reproduce this document for the purpose of selling it should be addressed as shown below or to ISO's member body in the country of the requester:

[Indicate the full address, telephone number, fax number, telex number, and electronic mail address, as appropriate, of the Copyright Manger of the ISO member body responsible for the secretariat of the TC or SC within the framework of which the working document has been prepared.]

Reproduction for sales purposes may be subject to royalty payments or a licensing agreement.

Violators may be prosecuted.

Foreword

ISO (the International Organization for Standardization) and IEC (the International Electrotechnical Commission) form the specialized system for worldwide standardization. National bodies that are members of ISO or IEC participate in the development of International Standards through technical committees established by the respective organization to deal with particular fields of technical activity. ISO and IEC technical committees collaborate in fields of mutual interest. Other international organizations, governmental and non-governmental, in liaison with ISO and IEC, also take part in the work.

The procedures used to develop this document and those intended for its further maintenance are described in the ISO/IEC Directives, Part 1. In particular, the different approval criteria needed for the different types of document should be noted. This document was drafted in accordance with the editorial rules of the ISO/IEC Directives, Part 2 (see [www.iso.org/directives](https://www.iso.org/directives-and-policies.html)).

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. ISO and IEC shall not be held responsible for identifying any or all such patent rights. Details of any patent rights identified during the development of the document will be in the Introduction and/or on the ISO list of patent declarations received (see [www.iso.org/patents](https://www.iso.org/iso-standards-and-patents.html)) or the IEC list of patent declarations received (see [http://patents.iec.ch](http://patents.iec.ch/)).

Any trade name used in this document is information given for the convenience of users and does not constitute an endorsement.

For an explanation of the voluntary nature of standards, the meaning of ISO specific terms and expressions related to conformity assessment, as well as information about ISO's adherence to the World Trade Organization (WTO) principles in the Technical Barriers to Trade (TBT), see [www.iso.org/iso/foreword.html](https://www.iso.org/foreword-supplementary-information.html).

This document was prepared by Joint Technical Committee ISO/IEC JTC 1, *Information technology*, Subcommittee SC 29, *Coding of audio, picture, multimedia and hypermedia information*, in collaboration with ITU-T. A technically aligned twin text is published as ITU-T H.274.

A list of all parts in the ISO/IEC 23002 series can be found on the ISO website.

Any feedback or questions on this document should be directed to the user’s national standards body. A complete listing of these bodies can be found at [www.iso.org/members.html](https://www.iso.org/members.html).

Information technology — MPEG video technologies — Part 7: Versatile supplemental enhancement information messages for coded video bitstreams, AMENDMENT 1: Additional SEI

*Modify subclause 6.1 as follows:*

...

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

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

...

*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 |
| SEI processing order | For each value of po\_id, the number of SEI messages and the payloadType codes of the SEI messages indicated within the SEI processing order SEI message persist for the CVS containing the SEI processing order SEI message. |
| Processing order nesting | Depending on the processing-order-nested SEI messages. Each processing-order-nested SEI message has the same persistence scope as if the SEI message was not nested. |
| Encoder optimization information | Specified by the syntax of the SEI message |
| Source picture timing information | Specified by the syntax of the SEI message |
| Object mask information | Specified by the syntax of the SEI message |
| Modality information | Specified by the syntax of the SEI message |
| Text descriptions | Specified by the syntax of the SEI message |

*Modify subclause 8.5.2 as follows:*

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

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

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

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

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

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

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

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

*Modify subclause 8.19.2 as follows:*

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

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

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

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

sdi\_aux\_id[ i ] shall be in the range of 0 to 2, inclusive, or 128 to 159, inclusive, for bitstreams conforming to this version of this Specification. Although the value of sdi\_aux\_id[ i ] shall be in the range of 0 to 3, inclusive, or 128 to 159, inclusive, in this version of this Specification, decoders shall also allow other values of sdi\_aux\_id[ i ] in the range of 0 to 255, inclusive.

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

*Modify subclause 8.28.1 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) |
| if( SpatialExtrapolationFlag ) { |  |
| **nnpfc\_spatial\_extrapolation\_left\_offset** | se(v) |
| **nnpfc\_spatial\_extrapolation\_right\_offset** | se(v) |
| **nnpfc\_spatial\_extrapolation\_top\_offset** | se(v) |
| **nnpfc\_spatial\_extrapolation\_bottom\_offset** | se(v) |
| **nnpfc\_spatial\_extrapolation\_prompt\_present\_flag** | u(1) |
| if( nnpfc\_spatial\_extrapolation\_prompt\_present\_flag ) { |  |
| while( !byte\_aligned( ) ) |  |
| **nnpfc\_alignment\_zero\_bit\_c** | u(1) |
| **nnpfc\_prompt** | st(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) |
| if( !SpatialExtrapolationFlag ) { |  |
| **nnpfc\_overlap** | ue(v) |
| **nnpfc\_constant\_patch\_size\_flag** | u(1) |
| if( nnpfc\_constant\_patch\_size\_flag ) { |  |
| **nnpfc\_patch\_width\_minus1** | ue(v) |
| **nnpfc\_patch\_height\_minus1** | ue(v) |
| } else { |  |
| **nnpfc\_extended\_patch\_width\_cd\_delta\_minus1** | ue(v) |
| **nnpfc\_extended\_patch\_height\_cd\_delta\_minus1** | ue(v) |
| } |  |
| } |  |
| **nnpfc\_padding\_type** | ue(v) |
| if( nnpfc\_padding\_type  = =  4 ) { |  |
| if( nnpfc\_inp\_order\_idc  !=  1 ) |  |
| **nnpfc\_luma\_padding\_val** | ue(v) |
| if( nnpfc\_inp\_order\_idc  !=  0 ) { |  |
| **nnpfc\_cb\_padding\_val** | ue(v) |
| **nnpfc\_cr\_padding\_val** | ue(v) |
| } |  |
| } |  |
| **nnpfc\_complexity\_info\_present\_flag** | u(1) |
| if( nnpfc\_complexity\_info\_present\_flag ) { |  |
| **nnpfc\_parameter\_type\_idc** | u(2) |
| if( nnpfc\_parameter\_type\_idc  !=  2 ) |  |
| **nnpfc\_log2\_parameter\_bit\_length\_minus3** | u(2) |
| **nnpfc\_num\_parameters\_idc** | u(6) |
| **nnpfc\_num\_kmac\_operations\_idc** | ue(v) |
| **nnpfc\_total\_kilobyte\_size** | ue(v) |
| } |  |
| **nnpfc\_num\_metadata\_extension\_bits** | ue(v) |
| if( nnpfc\_num\_metadata\_extension\_bits > 0 ) { |  |
| if( nnpfc\_purpose = = 0 ) { |  |
| **nnpfc\_application\_purpose\_tag\_uri\_present\_flag** | u(1) |
| while( !byte\_aligned( ) ) |  |
| **nnpfc\_metadata\_alignment\_zero\_bit** | u(1) |
| if( nnpfc\_application\_purpose\_tag\_uri\_present\_flag ) |  |
| **nnpfc\_application\_purpose\_tag\_uri** | st(v) |
| } |  |
| if( SpatialExtrapolationFlag ) |  |
| **nnpfc\_scan\_type\_idc** | u(2) |
| **nnpfc\_for\_human\_viewing\_idc** | u(2) |
| **nnpfc\_for\_machine\_analysis\_idc** | u(2) |
| **nnpfc\_reserved\_metadata\_extension** | u(v) |
| } |  |
| } |  |
| /\* ISO/IEC 15938-17 bitstream \*/ |  |
| if( nnpfc\_mode\_idc  = =  0 ) { |  |
| while( !byte\_aligned( ) ) |  |
| **nnpfc\_alignment\_zero\_bit\_b** | 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 255, inclusive, in bitstreams conforming to this version of this Specification. Values of 256 to 65 535, inclusive, for nnpfc\_purpose are reserved for future use by ITU-T | ISO/IEC and shall not be present in bitstreams conforming to this version of this Specification. Decoders conforming to this version of this Specification shall ignore NNPFC SEI messages with nnpfc\_purpose in the range of 256 to 65 535, inclusive.

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

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

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

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

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

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

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

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

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

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

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

– This SEI message specifies a base NNPF.

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

The 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 ) )

When SpatialExtrapolation is equal to 1, nnpfcOutputPicWidth is updated as follows:

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

It is a requirement of bitstream conformance that nnpfcOutputPicWidth shall be greater than 0 and 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 ) )

When SpatialExtrapolation is equal to 1, nnpfcOutputPicHeight is updated as follows:

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

It is a requirement of bitstream conformance that nnpfcOutputPicHeight shall be greater than 0 and 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.

– SpatialExtrapolationFlag is equal to 1.

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

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

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

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

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

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

**nnpfc\_spatial\_extrapolation\_prompt\_present\_flag** equal to 1 specifies that nnpfc\_prompt syntax element is present and nnpfc\_alignment\_zero\_bit\_c syntax element may be present. nnpfc\_spatial\_extrapolation\_prompt\_present\_flag equal to 0 specifies that nnpfc\_prompt syntax element and nnpfc\_alignment\_zero\_bit\_c syntax element are not present.

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

**nnpfc\_prompt** specifies the text string prompt used for generating the contents of the spatial extrapolation image area. When nnpfc\_spatial\_extrapolation\_prompt\_present\_flag is equal to 1, nnpfc\_prompt shall not be a null string.

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

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

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

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

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

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

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

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

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

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

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

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

When nnpfc\_auxiliary\_inp\_idc is equal to 2 or 3, nnpfc\_spatial\_extrapolation\_prompt\_present\_flag shall be equal to 1.

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 4 to 255, inclusive.

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

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

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

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

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

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

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

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

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

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

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

A black background with a black square

Description automatically generated

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

**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. When SpatialExtrapolationFlag is equal to 1, nnpfc\_patch\_height\_minus1 is inferred to be equal to CroppedHeight − 1.

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

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

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

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

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

if( nnpfc\_inp\_order\_idc = = 0 | | nnpfc\_inp\_order\_idc = = 2 )  
 for( cTop = 0; cTop < CroppedHeight; cTop += inpPatchHeight )  
 for( cLeft = 0; cLeft < CroppedWidth; cLeft += inpPatchWidth ) {  
 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. When nnpfc\_num\_metadata\_extension\_bits is greater than 0, let the variable numSpecifiedMetadataExtensionBits be the number of bits representing all syntax elements between nnpfc\_num\_metadata\_extension\_bits and nnpfc\_reserved\_metadata\_extension. nnpfc\_num\_metadata\_extension\_bits greater than 0 specifies the sum of numSpecifiedMetadataExtensionBits and the length, in bits, of nnpfc\_reserved\_metadata\_extension.

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

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

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

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

NOTE 4 – nnpfc\_application\_purpose\_tag\_uri enables uniquely identifying the application determined purpose of NNPF without needing a central registration authority.

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

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

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

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

NOTE 5 – When a decoding system displays the video for human viewing, any NNPF that has nnpfc\_for\_human\_viewing\_idc equal to 1 is suggested to be omitted. When a decoding system performs machine analysis, any NNPF that has nnpfc\_for\_machine\_analysis\_idc equal to 1 is suggested to be omitted.

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

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

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

*Add clauses 8.30 to 8.35 as follows:*

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

|  |  |  |
| --- | --- | --- |
| sei\_processing\_order( payloadSize ) { | **Descriptor** | |
| **po\_id** | u(8) | |
| **po\_for\_human\_viewing\_idc** | u(2) |
| **po\_for\_machine\_analysis\_idc** | u(2) |
| **po\_reserved\_zero\_4bits** | u(4) |
| **po\_num\_sei\_messages\_minus2** | u(7) |
| **po\_breadth\_first\_flag** | u(1) |
| for( i = 0, i < po\_num\_sei\_messages\_minus2 + 2; i++ ) { |  |
| **po\_sei\_wrapping\_flag**[ i ] | u(1) |
| **po\_sei\_importance\_flag**[ i ] | u(1) |
| **po\_sei\_processing\_degree\_flag**[ i ] | u(1) |
| **po\_sei\_payload\_type**[ i ] | u(12) |
| **po\_sei\_prefix\_flag**[ i ] | u(1) |
| **po\_sei\_processing\_order**[ i ] | u(8) |
| } |  |
| for( i = 0; i < po\_num\_sei\_messages\_minus2 + 2; i++ ) |  |
| if( po\_sei\_prefix\_flag[ i ] ) { |  |
| **po\_num\_bits\_in\_prefix\_indication\_minus1**[ i ] | u(8) |
| for( j = 0; j <= po\_num\_bits\_in\_prefix\_indication\_minus1[ i ]; j++ ) |  |
| **po\_sei\_prefix\_data\_bit**[ i ][ j ] | u(1) |
| while( !byte\_aligned( ) ) |  |
| **po\_byte\_alignment\_bit\_equal\_to\_one** /\* equal to 1 \*/ | f(1) |
| } |  |
| } |  |

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

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

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

– Two lists of payloadType values, SeiProcessingOrderSeiList and SpoProcessSeiList.

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

NOTE 2 – The values of po\_for\_human\_viewing\_idc and po\_for\_machine\_analysis\_idc are in force for the output of a processing chain instead of the respective syntax elements in an encoder optimization information (EOI) SEI message (eoi\_for\_human\_viewing\_idc and eoi\_for\_machine\_analysis\_idc) or a neural-network post-filter characteristics (NNPFC) SEI message (nnpfc\_for\_human\_viewing\_idc and nnpfc\_for\_machine\_analysis\_idc), when the EOI or NNPFC SEI message is associated with the processing chain.

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

**po\_num\_sei\_messages\_minus2** plus 2 indicates the number of types of SEI messages for which the preferred order of processing is indicated in the SPO SEI message.

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

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

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

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

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

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

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

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

NOTE 4 – po\_sei\_wrapping\_flag[ i ] equal to 1 enables SEI messages to be carried within the processing order nesting SEI message to prevent such SEI messages from being incorrectly interpreted by decoders that do not process the SPO SEI message. Thus, po\_sei\_wrapping\_flag[ i ] equal to 1 is intended to be used when po\_sei\_wrapping\_flag[ i ] equal to 0 can lead to unintended results being produced by such decoders.

**po\_sei\_importance\_flag**[ i ] equal to 1 affects the derivation of PoSeiList, which is the list of SEI messages that a decoding system should process for a particular picture picA, as specified below.

po\_sei\_importance\_flag[ i ] equal to 0 specifies that when the decoding system cannot interpret or does not support the functionality indicated by the i-th SEI message type, it shall ignore all data associated with the loop variable value of i and exclude the i-th SEI message type from the processing chain performed by the decoding system.

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

PoSeiList is derived as follows:

– PoSeiList is initially empty.

– The following applies in a non-decreasing order of po\_sei\_processing\_order[ i ] values for all values of i in the range of 0 to po\_num\_sei\_messages\_minus2 + 1, inclusive, unless terminated earlier as specified below:

– When an SEI message seiA associated with the i-th SEI message type persists for picA, the following applies:

– If the decoding system can interpret and supports the functionality indicated by seiA, seiA is added at the end of PoSeiList.

– Otherwise, if po\_sei\_importance\_flag[ i ] is equal to 1 and po\_sei\_processing\_degree\_flag[ i ] is equal to 0, the derivation of PoSeiList is terminated.

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

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

**po\_sei\_prefix\_flag**[ i ] equal to 1 specifies that po\_num\_bits\_in\_prefix\_indication\_minus1[ i ] and some po\_sei\_prefix\_data\_bit[ i ][ j ] syntax elements are present. po\_sei\_prefix\_flag[ i ] equal to 0 specifies that these syntax elements are not present.

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

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

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

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

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

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

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

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

– The SEI message persists for picA.

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

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

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

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

– Otherwise, the semantics of seiMsgA apply to picA.

NOTE 5 – When an NNPF process outputs more than one NNPF-generated picture, the semantics of an SEI message that follows the NNPF in the processing order apply to all these NNPF-generated pictures.

**po\_num\_bits\_in\_prefix\_indication\_minus1**[ i ] and **po\_sei\_prefix\_data\_bit**[ i ][ j ], when present, have the same semantics as the num\_bits\_in\_prefix\_indication\_minus1[ i ] and sei\_prefix\_data\_bit[ i ][ j ] syntax elements of the SEI prefix indication SEI message, with prefix\_sei\_payload\_type replaced by po\_sei\_payload\_type[ i ].

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

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

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

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

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

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

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

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

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

NOTE 1 – An encoder can include multiple PON SEI messages in the same access unit. For example, a first PON SEI message in an access unit can contain a PON-nested SEI message that applies to multiple processing chains and one or more other PON SEI messages in the same access unit that apply to a single processing chain only.

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

A special NNPF cascading case is defined as the case when such two NNPFs are both activated for a picture: the two PPFs are both NNPFs, one of the two NNPFs has nnpfc\_purpose equal to 4 and the other has multiple input pictures, and neither of the two NNPFs is associated with an SPO SEI message. In this case, the two NNPFs are implicitly considered as belonging to one processing chain, and the NNPF with nnpfc\_purpose equal to 4 is applied first.

Except for the special NNPF cascading case, each processing chain containing multiple SEI message types is indicated by an SPO SEI message with a particular value of po\_id. Except for the special NNPF cascading case, any SEI message for which the payloadType is present in SpoProcessSeiList but is not indicated by an SPO SEI message is in its own processing chain.

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

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

– First, the bitstream is decoded, and the list PoPicList is set to be the list of the cropped decoded pictures in output order that resulted from decoding the bitstream, and a processing chain is chosen.

– For each of the SEI message types of the chosen processing chain, the following applies in a non-decreasing order of the corresponding po\_sei\_processing\_order[ i ] values:

– The following applies for each picture picA in PoPicList in output order, when an SEI message associated with the i-th SEI message type is present in PoSeiList derived for picA or a picture for which an NNPF that generated picA was activated by a preceding process in the processing chain:

– When picA is not a cropped decoded picture, the following exceptions apply for the interpretation of the SEI message:

– The interface variables for purposes of interpretation of the SEI message are derived from picA instead of the syntax elements indicating properties for the respective cropped decoded picture.

– The semantics of the SEI message, or the semantics of the SEI message and, when the SEI message is an NNPFA SEI message, the associated NNPFC SEI message, apply to pictures in PoPicList instead of cropped decoded pictures.

– When the i-th SEI message type is present in SpoProcessingList, the process implied by the SEI message is performed and PoPicList is updated by replacing pictures with the corresponding processed pictures, if any, resulting from the process and inserting the other pictures, if any, resulting from the process into PoPicList so that the output order is obeyed.

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

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

– First, the bitstream is decoded, and the list PoPicList is set to be the list of the cropped decoded pictures in output order that resulted from decoding the bitstream, and a processing chain is chosen.

– The following is repeatedly applied, in output order, for each picture picA in PoPicList:

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

– When the current SEI message is not the first in the set of the SEI messages, the following exceptions apply for the interpretation of the SEI message:

– The interface variables for purposes of interpretation of the SEI message are derived from the pictures in the updated PoPicList instead of the syntax elements indicating properties for the respective cropped decoded pictures.

– The semantics of the SEI message, or of the SEI message and, when the SEI message is an NNPFA SEI message, the associated NNPFC SEI message, apply to pictures in PoPicList instead of cropped decoded pictures.

– The process implied by the SEI message is invoked repeatedly, in output order, for picA and each of the pictures in PoPicList that are, or correspond to, interpolated or extrapolated pictures generated by the application of the process implied by any preceding SEI message, if any, to picA. After each invocation of the process, PoPicList is updated by replacing pictures with the corresponding processed pictures, if any, resulting from the process and inserting the other pictures, if any, resulting from the process into PoPicList so that the output order is obeyed.

* 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( EoiSpatialResamplingFlag ) { |  |
| **eoi\_orig\_pic\_dimensions\_flag** | u(1) |
| if( eoi\_orig\_pic\_dimensions\_flag ) { |  |
| **eoi\_orig\_pic\_width** | u(16) |
| **eoi\_orig\_pic\_height** | u(16) |
| } else |  |
| **eoi\_spatial\_resampling\_type\_flag** | u(1) |
| } |  |
| if( EoiPrivacyProtectionFlag ) { |  |
| **eoi\_privacy\_protection\_method\_idc** | u(4) |
| **eoi\_privacy\_info\_type** | u(8) |
| } |  |
| } |  |
| } |  |

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

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

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

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

– A new CLVS of the current layer begins.

– The bitstream ends.

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

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

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

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

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

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

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

**eoi\_privacy\_protection\_method\_idc**, when present, indicates the method / algorithm that was used to apply privacy protection optimization as specified in Table x3.

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

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

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

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

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

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

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

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

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

– A temporal sublayer identifier, denoted herein by TemporalId.

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

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

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

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

– A new CLVS of the current layer begins.

– The bitstream ends.

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

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

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

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

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

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

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

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

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

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

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

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

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

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

The variable sptiMinTemporalSublayer is set to as follows:

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

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

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

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

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

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

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

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

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

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

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

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

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

– A conformance cropping window left offset, ConfWinLeftOffset

– A conformance cropping window top offset, ConfWinTopOffset

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

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

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

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

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

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

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

– A new CLVS of the current layer begins.

– The bitstream ends.

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

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

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

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

**omi\_mask\_id\_length\_minus1** plus 1 specifies the length, in bits, of omi\_mask\_id[ i ][ j ] syntax elements. The value of omi\_mask\_id\_length\_minus1 shall be in the range of 0 to 31, inclusive.

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

– A new CLVS of the current layer begins.

– The bitstream ends.

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

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

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

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

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

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

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

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

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

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

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

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

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

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

MinWavelength and MaxWavelength are in units of metres.

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

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

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

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

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

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

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

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

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

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

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

**txt\_descr\_id** indicates the identifier value of this text description information SEI message. The value of txt\_descr\_id shall be in the range of 1 to 16383, inclusive. Value 0 is reserved. Text description SEI messages with different values for txt\_descr\_purpose use separate values spaces for txt\_descr\_id.

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

**txt\_persistence\_flag** specifies the persistence of the text information description SEI message for the current layer.

txt\_persistence\_flag equal to 0 specifies that the text description information applies to the current decoded picture only.

txt\_persistence\_flag equal to 1 specifies that the text description 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 text description information SEI message with the same txt\_descr\_id is output that follows the current picture in output order.

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

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

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

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

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

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

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

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

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

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

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

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