Modification of WD

Reference picture list 3DVC modification syntax

|  |  |  |
| --- | --- | --- |
| ref\_pic\_list\_3dvc\_modification( ) { | **C** | **Descriptor** |
| if( slice\_type % 5 != 2 && slice\_type % 5 != 4 ) { |  |  |
| **ref\_pic\_list\_modification\_flag\_l0** | 2 | u(1) |
| if( ref\_pic\_list\_modification\_flag\_l0 ) |  |  |
| do { |  |  |
| **modification\_of\_pic\_nums\_idc** | 2 | ue(v) |
| if( modification\_of\_pic\_nums\_idc = = 0 | |  modification\_of\_pic\_nums\_idc = = 1 ) |  |  |
| **abs\_diff\_pic\_num\_minus1** | 2 | ue(v) |
| else if( modification\_of\_pic\_nums\_idc = = 2 ) |  |  |
| **long\_term\_pic\_num** | 2 | ue(v) |
| else if ( modification\_of\_pic\_nums\_idc = = 4 | |  modification\_of\_pic\_nums\_idc = = 5 ) |  |  |
| **abs\_diff\_view\_idx\_minus1** | 2 | ue(v) |
| ~~else if( modification\_of\_pic\_nums\_idc = = 6 )~~ |  |  |
| **~~vsp\_ref\_idx~~** | 2 | ue(v) |
| } while( modification\_of\_pic\_nums\_idc != 3 ) |  |  |
| } |  |  |
| if( slice\_type % 5 = = 1 ) { |  |  |
| **ref\_pic\_list\_modification\_flag\_l1** | 2 | u(1) |
| if( ref\_pic\_list\_modification\_flag\_l1 ) |  |  |
| do { |  |  |
| **modification\_of\_pic\_nums\_idc** | 2 | ue(v) |
| if( modification\_of\_pic\_nums\_idc = = 0 | |  modification\_of\_pic\_nums\_idc = = 1 ) |  |  |
| **abs\_diff\_pic\_num\_minus1** | 2 | ue(v) |
| else if( modification\_of\_pic\_nums\_idc = = 2 ) |  |  |
| **long\_term\_pic\_num** | 2 | ue(v) |
| else if ( modification\_of\_pic\_nums\_idc = = 4 | |  modification\_of\_pic\_nums\_idc = = 5 ) |  |  |
| **abs\_diff\_view\_idx\_minus1** | 2 | ue(v) |
| ~~else if( modification\_of\_pic\_nums\_idc = = 6 )~~ |  |  |
| **~~vsp\_ref\_idx~~** | 2 | ue(v) |
| } while( modification\_of\_pic\_nums\_idc != 3 ) |  |  |
| } |  |  |
| } |  |  |

Modification process for reference picture lists

Input to this process is reference picture list RefPicList0 and, when decoding a B slice, also reference picture list RefPicList1.

Outputs of this process are a possibly modified reference picture list RefPicList0 and, when decoding a B slice, also a possibly modified reference picture list RefPicList1.

When ref\_pic\_list\_modification\_flag\_l0 is equal to 1, the following ordered steps are specified:

1. Let refIdxL0 be an index into the reference picture list RefPicList0. It is initially set equal to 0.
2. The corresponding syntax elements modification\_of\_pic\_nums\_idc are processed in the order they occur in the bitstream. For each of these syntax elements, the following applies:

– If modification\_of\_pic\_nums\_idc is equal to 0 or equal to 1, the process specified in subclause  is invoked with RefPicList0 and refIdxL0 given as input, and the output is assigned to RefPicList0 and refIdxL0.

– Otherwise, if modification\_of\_pic\_nums\_idc is equal to 2, the process specified in subclause  is invoked with RefPicList0 and refIdxL0 given as input, and the output is assigned to RefPicList0 and refIdxL0.

– Otherwise, if modification\_of\_pic\_nums\_idc is equal to 4 or equal to 5, the process specified in subclause  is invoked with RefPicList0 and refIdxL0 given as input, and the output is assigned to RefPicList0 and refIdxL0.

– ~~Otherwise, if modification\_of\_pic\_nums\_idc is equal to 6, the process specified in subclause  is invoked with RefPicList0 and refIdxL0 given as input, and the output is assigned to RefPicList0 and refIdxL0.~~

– Otherwise (modification\_of\_pic\_nums\_idc is equal to 3), the modification process for reference picture list RefPicList0 is finished.

When ref\_pic\_list\_modification\_flag\_l1 is equal to 1, the following ordered steps are specified:

1. Let refIdxL1 be an index into the reference picture list RefPicList1. It is initially set equal to 0.
2. The corresponding syntax elements modification\_of\_pic\_nums\_idc are processed in the order they occur in the bitstream. For each of these syntax elements, the following applies:

– If modification\_of\_pic\_nums\_idc is equal to 0 or equal to 1, the process specified in subclause  is invoked with RefPicList1 and refIdxL1 given as input, and the output is assigned to RefPicList1 and refIdxL1.

– Otherwise, if modification\_of\_pic\_nums\_idc is equal to 2, the process specified in subclause  is invoked with RefPicList1 and refIdxL1 given as input, and the output is assigned to RefPicList1 and refIdxL1.

– Otherwise, if modification\_of\_pic\_nums\_idc is equal to 4 or equal to 5, the process specified in subclause  is invoked with RefPicList1 and refIdxL1 given as input, and the output is assigned to RefPicList1 and refIdxL1.

– ~~Otherwise, if modification\_of\_pic\_nums\_idc is equal to 6, the process specified in subclause  is invoked with RefPicList1 and refIdxL1 given as input, and the output is assigned to RefPicList1 and refIdxL1.~~

– Otherwise (modification\_of\_pic\_nums\_idc is equal to 3), the modification process for reference picture list RefPicList1 is finished.

~~Modification process for reference picture lists for synthetic reference components~~

~~Inputs to this process are an index refIdxLX and a reference picture list RefPicListX (with X being 0 or 1).~~

~~Outputs of this process are an incremented index refIdxLX and a modified reference picture list RefPicListX.~~

~~The following procedure is conducted to place a synthetic reference component with VSP reference index equal to vsp\_ref\_idx into the index position refIdxLX, shift the position of any other remaining pictures to later in the list, and increment the value of refIdxLX:~~

~~for( cIdx = num\_ref\_idx\_lX\_active\_minus1 + 1; cIdx > refIdxLX; cIdx− − )  
 RefPicListX[ cIdx ] = RefPicListX[ cIdx − 1]  
RefPicListX[ refIdxLX++ ] = synthetic reference component with VSP reference index equal to vsp\_ref\_idx~~

Derivation process for luma motion vectors for ~~VSP~~ Interview skipped macroblocks in P and SP and ~~VSP~~ Interview skipped/direct in B slices

This process is invoked when mb\_type is equal to P\_Skip, nal\_unit\_type is equal to 21, DepthFlag is equal to 0, and MbVSSkipFlag is equal to 1.

~~The reference index refIdxLX for a VSP skipped/direct macroblock is derived as the synthetic reference component that appears first in the reference picture list X, with X being replaced by 0 or 1.~~ ~~When there is no synthetic picture in the reference picture list X, refIdxLX is set equal to 0.~~

~~The motion vector mvLX is set equal to zero motion vector, with X being replaced by 0 or 1.~~

Outputs of this process are the motion vectors mvL0 \_i with i = 0, 1, 2, 3, … , 14, 15 and the reference index refIdxL0.

The reference index refIdxL0 and reference picture InterViewPic for a interview skipped macroblock are derived as the interview picture that appears first in RefPicList0.

For the derivation of the motion vector mvL0\_i of a P\_Skip macroblock type, the following ordered steps are specified:

* Inter-view reference picture InterViewPic and an offset vector dv are derived as specified by the following ordered steps:

– The inverse macroblock scanning process as specified in subclause 6.4.1 is invoked with CurrMbAddr as the input and the output is assigned to ( x1, y1 ).

– The inverse macroblock partition scanning process specified in subclause 6.4.2.1 is invoked with mbPartIdx as the input and the output assigned to ( dx1, dy1 ).

– The inverse sub-macroblock partition scanning process specified in subclause 6.4.2.2 is invoked with mbPartIdx and subMbPartIdx as the input and the output assigned to ( dx2, dy2 ).

– The process specified in subclause  is invoked with depthPic set to DepthCurrPic, dbx1 set to x1 + dx1 + dx2, dby1 set to y1 + dy1 + dy2 and listSuffixFlag as input and InterViewPic, an offset vector dv and an variable InterViewAvailable as outputs.

* The motion vector mvL0\_i with i = (4 \* mbPartIdx + subMbPartIdx) is set to the offset dv.

After motion compensation:

The reference index refIdxL0 is set as not available

The motion vector mvL0\_i with i = 0, 1, 2, 3, … , 14, 15 is set equal to zero motion vector.