All the changes are highlighted in green or yellow or ~~red strikethrough~~.

**H.8.3.7 Derivation process for the alternative target reference index for TMVP in merge mode** **and the ARP picture index and availability**

This process is invoked when the current slice is a P or B slice.

The variables AltRefIdxL0 and AltRefIdxL1 are set equal to –1, TempRefPicInListsFlag is set equal to 0 FstRefIdxL0 and FstRefIdxL1 are set equal to –1, and the following applies for X in the range of 0 to 1, inclusive:

* When X is equal to 0 or the current slice is a B slice the following applies:

for( i = 0; i <= num\_ref\_idx\_lX\_active\_minus1 && FstRefIdxLX = = –1; i++)  
 if ( PicOrderCnt(RefPicListX[ i ]) != PicOrderCntVal )  
 FstRefIdxLX = i

TempRefPicInListsFlag = FstRefIdxL0 >= 0 || FstRefIdxL1 >= 0

for( i = 0; i <= num\_ref\_idx\_lX\_default\_active\_minus1; i++ ) {

RefViewIdx = ViewIdx (RefPicListX[i])  
If there is a reference picture A in the DPB with PicOrderCnt( picture A ) equal to PicOrderCntVal, ViewIdx equal to RefViewIdx and marked as “used\_for\_reference”, refIvRefPicAvailable[RefViewIdx] is set equal to 1.

}

zeroIdxLtFlag = RefPicListX[ 0 ] is a short-term reference picture ? 0 : 1  
for( i = 1; i <= num\_ref\_idx\_lX\_active\_minus1 && AltRefIdxLX = = –1; i++)  
 if ( ( zeroIdxLtFlag && RefPicListX[ i ] is a short-term reference picture) | |  
 ( !zeroIdxLtFlag && RefPicListX[ i ] is a long-term reference picture) )   
 AltRefIdxLX = i   
  
 ~~TempRefPicInListsFlag = TempRefPicInListsFlag | | !zeroIdxLtFlag | | ( AltRefIdxLX ! = –1 )~~

**H.8.5.3.3 Decoding process for inter prediction samples**

* + - When predFlagLX is equal to 1, the following applies.
      * The variable resPredFlag is derived as specified in the following:
        1. resPredFlag = ( iv\_res\_pred\_weight\_idx != 0 )   
            ( PicOrderCnt( RefPicListX[ refIdxLX ] ) != PicOrderCntVal ) && FstRefIdxLX >=0 && refIvRefPicAvailable[RefViewIdx[ xP ][ yP ]]

(‑168)

* + - * If resPredFlag is equal to 1, the bilinear sample interpolation and residual prediction process as specified in subclause is invoked with the luma locations ( xCb, yCb ), ( xBl, yBl ), the size of the current luma coding block nCbS, the width and the height of the current luma prediction block nPbW, nPbH, the prediction list indication X, the prediction list utilization flag predFlagLX, the reference index refIdxLX, and the motion vectors mvLX, mvCLX, as the inputs and the outputs are the arrays predSamplesLXL, predSamplesLXCb, and predSamplesLXCr.

**H.8.5.3.3.7 Bilinear sample interpolation and residual prediction process**

The process is only invoked if res\_pred\_flag is equal to 1.

Inputs to this process are:

* a luma location ( xCb, yCb ) specifying the top-left sample of the current luma coding block relative to the top left luma sample of the current picture,
* a luma location ( xBl, yBl ) specifying the top-left sample of the current luma prediction block relative to the top-left sample of the current luma coding block,
* a variable nCbS specifying the size of the current luma coding block,
* variables nPbW and nPbH specifying the width and the height, respectively, of the current prediction unit,prediction list utilization flags, predFlagL0 and predFlagL1,
* the prediction list indication X,
* the prediction list utilization flag predFlagLX,
* the reference index refIdxLX,
* the motion vectors mvLX, mvCLX

Output of this process are:

* the (nPbW)x(nPbH) array predSamplesLXL,
* the (nPbW / 2)x(nPbH / 2) arrays predSamplesLXCb and predSamplesLXCr.

The motion vector mvLX is scaled as specified in the following:

* 1. tx = ( 16384 + ( Abs( td ) >> 1 ) ) / td (‑207)
  2. distScaleFactor = Clip3( −4096, 4095, ( tb \* tx + 32 ) >> 6 ) (‑208)
  3. mvLX = Clip3( −32768, 32767, Sign( distScaleFactor \* mvLX ) \*   
      ( ( Abs( distScaleFactor \* mvLX ) + 127 ) >> 8 ) ) (‑209)

where td and tb are derived as:

* 1. td = Clip3( −128, 127, DiffPicOrderCnt( currPic, RefPicListX[ refIdxLX ] ) ) (‑210)
  2. tb = Clip3( −128, 127, DiffPicOrderCnt( currPic, RefPicListX[ FstRefIdxLX ] ) ) (‑211)

The location ( xP, yP ) is derived by:

* 1. xP = xCb + xBl (‑212)
  2. yP = yCb + yBl (‑213)

The motion vector mvCLX is set equal to mvLX.

The arrays predSamplesLXL, predSamplesLXCb, and predSamplesLXCr  are derived as specified in the following:

* The reference picture consisting of an ordered two-dimensional array refPicLXL of luma samples and two ordered two-dimensional arrays refPicLXCb and refPicLXCr of chroma samples is derived by invoking the process specified in subclause 8.5.2.2.1 with refIdxLX equal to FstRefIdxLX as input.
* The arrays predSamplesLXL, predSamplesLXCb, and predSamplesLXCr are derived by invoking the bilinear sample interpolation process specified in subclause with the luma locations ( xCb, yCb ), ( xBl, yBl ), , the luma prediction block width nPbW, the luma prediction block height nPbH,, the motion vectors mvLX, mvCLX, and the reference arrays with refPicLXL, refPicLXCb and refPicLXCr. as the inputs.

The arrays currIvSamplesLXL, currIvSamplesLXCb, and currIvSamplesLXCr  are derived as specified in the following:

* Let currIvRefPic be the picture with PicOrderCnt( currIvRefPic ) equal to PicOrderCntVal and ViewIdx equal to RefViewIdx[ xP ][ yP ].
* Let the reference picture sample arrays currIvRefPicLXL, currIvRefPicLXCb, and currIvRefPicLXCr corresponding to decoded sample arrays SL, SCb, SCr derived in subclause 8.7 for the previously-decoded picture currIvRefPic.
* The arrays currIvSamplesLXL, currIvSamplesLXCb, and currIvSamplesLXCr are derived by invoking the bilinear sample interpolation process specified in subclause with the luma locations ( xCb, yCb ) and ( xBl, yBl ), the luma prediction block width nPbW, the luma prediction block height nPbH, the motion vectors mvLX equal to MvDisp[ xP ][ yP ] and mvCLX equal to MvDisp[ xP ][ yP ], and the, and the reference arrays with currIvRefPicLXL, currIvRefPicLXCb and currIvRefPicLXCr as the inputs.

The arrays refIvSamplesLXL, refIvSamplesLXCb, and refIvSamplesLXCr  are derived as specified in the following:

* Let refIvRefPic be the picture with PicOrderCnt( refIvRefPic ) equal to PicOrderCnt( RefPicListX[ FstRefIdxLX ] ) and ViewIdx equal to RefViewIdx[ xP ][ yP ].