1. * + 1. Derivation process for motion vector components and reference indices

Inputs to this process are:

* a luma location ( xCb, yCb ) of 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 ) of 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,
* two variables nPbW and nPbH specifying the width and the height of the luma prediction block,
* a variable partIdx specifying the index of the current prediction unit within the current coding unit.

Outputs of this process are:

* the luma motion vectors mvL0 and mvL1,
* the chroma motion vectors mvCL0 and mvCL1,
* the reference indices refIdxL0 and refIdxL1,
* the prediction list utilization flags predFlagL0 and predFlagL1.
* the flag subPbMotionFlag, specifying, whether the motion data of the current PU has sub prediction block size motion accuracy.

Let ( xPb, yPb ) specify the top-left sample location of the current luma prediction block relative to the top-left luma sample of the current picture where xPb = xCb + xBl and yPb = yCb + yBl.

Let the variable currPic and ListX be the current picture and RefPicListX, with X being 0 or 1, of the current picture, respectively.

The function LongTermRefPic( aPic, aPb, refIdx, LX ), with X being 0 or 1, is defined as follows:

* If the picture with index refIdx from reference picture list LX of the slice containing prediction block aPb in the picture aPic was marked as "used for long term reference" at the time when aPic was the current picture, LongTermRefPic( aPic, aPb, refIdx, LX ) is equal to 1.
* Otherwise, LongTermRefPic( aPic, aPb, refIdx, LX ) is equal to 0.

The variables vspModeFlag, ivpMvFlagL0, ivpMvFlagL1 and subPbMotionFlag are set equal to 0.

For the derivation of the variables mvL0 and mvL1, refIdxL0 and refIdxL1, as well as predFlagL0 and predFlagL1, the following applies:

* If merge\_flag[ xPb ][ yPb ] is equal to 1, the derivation process for luma motion vectors for merge mode as specified in subclause H.8.5.3.2.1 is invoked with the luma location ( xCb, yCb ), the luma location ( xPb, yPb ), the variables nCbS, nPbW, nPbH, and the partition index partIdx as inputs, and the output being the luma motion vectors mvL0, mvL1, the reference indices refIdxL0, refIdxL1, and the prediction list utilization flags predFlagL0 and predFlagL1,, the disparity vector availability flags ivpMvFlagL0 and ivpMvFlagL1, the flag vspModeFlag, and the flag subPbMotionFlag.
* Otherwise, for X being replaced by either 0 or 1 in the variables predFlagLX, mvLX, and refIdxLX, in PRED\_LX, and in the syntax elements ref\_idx\_lX and MvdLX, the following applies:
  + 1. The variables refIdxLX and predFlagLX are derived as follows:
       - * If inter\_pred\_idc[ xPb ][ yPb ] is equal to PRED\_LX or PRED\_BI,

refIdxLX = ref\_idx\_lX[ xPb ][ yPb ] (H‑79)

predFlagLX = 1 (H‑80)

* + - * + Otherwise, the variables refIdxLX and predFlagLX are specified by:

refIdxLX = −1 (H‑81)

predFlagLX = 0 (H‑82)

* + 1. The variable mvdLX is derived as follows:

mvdLX[ 0 ] = MvdLX[ xPb ][ yPb ][ 0 ] (H‑83)

mvdLX[ 1 ] = MvdLX[ xPb ][ yPb ][ 1 ] (H‑84)

* + 1. When predFlagLX is equal to 1, the derivation process for luma motion vector prediction in subclause 8.5.3.2.5 is invoked with the luma coding block location ( xCb, yCb ), the coding block size nCbS, the luma prediction block location ( xPb, yPb ), the variables nPbW, nPbH, refIdxLX, and the partition index partIdx as inputs, and the output being mvpLX.
    2. When predFlagLX is equal to 1, the luma motion vector mvLX is derived as follows:

uLX[ 0 ] = ( mvpLX[ 0 ] + mvdLX[ 0 ] + 216 ) % 216 (H‑85)

mvLX[ 0 ] = ( uLX[ 0 ] >= 215 ) ? ( uLX[ 0 ] − 216 ) : uLX[ 0 ] (H‑86)

uLX[ 1 ] = ( mvpLX[ 1 ] + mvdLX[ 1 ] + 216 ) % 216 (H‑87)

mvLX[ 1 ] = ( uLX[ 1 ] >= 215 ) ? ( uLX[ 1 ] − 216 ) : uLX[ 1 ] (H‑88)

NOTE – The resulting values of mvLX[ 0 ] and mvLX[ 1 ] as specified above will always be in the range of −215 to 215 − 1, inclusive.When ChromaArrayType is not equal to 0 and predFlagLX, with X being 0 or 1, is equal to 1, the derivation process for chroma motion vectors in subclause 8.5.3.2.9 is invoked with mvLX as input, and the output being mvCLX.

For use in derivation processes of variables invoked later in the decoding process, the following assignments are made for x = xPb.. ( xPb + nPbW − 1 ), y = yPb..( yPb + nPbH− 1 ) (with X being either 0 or 1):

1. IvpMvFlagLX[ x ][ y ] = ivpMvFlagLX (H‑89)  
   VspModeFlag[ x ][ y ] = vspModeFlag (H‑90)If iv\_res\_pred\_weight\_idx is not equal to 0, the following applies:

* The variables ivRefFlagL0, ivRefFlagL1, ivViewIdxL0, ivViewIdxL1, mvDispL0 and mvDispL1 are derived as:
  + If inter\_pred\_idc[ xPb ][ yPb ] is equal to PRED\_BI, the following applies:
    - For X being each of 0 and 1, the following applies:
      * ivRefFlagX is set equal to ( DiffPicOrderCnt( currPic, RefPicListX[ refIdxLX ] ) = =  0 ).
      * If ivRefFlagX is equal to 1, ivViewIdxLX is set equal to ViewIdx( RefPicListLX[refIdxLX] ) and.mvDispLX is set equal to mvLX,
  + Otherwise, if inter\_pred\_idc[ xPb ][ yPb ] is equal to PRED\_LY, the following applies:
    - For X being each of 0 and 1, the following applies:
      * ivRefFlagX is set equal to ( DiffPicOrderCnt( currPic, RefPicListY[ refIdxLY ] ) = =  0 ).
      * If ivRefFlagX is equal to 1, ivViewIdxLX is set equal to ViewIdx( RefPicListLY[refIdxLY] ) and.mvDispLX is set equal to mvLY,
  + For X being each of 0 and 1, the following applies:
    - If ivRefFlagX is equal to 1, the following applies:
      * The derivation process for a temporal inter-view motion vector candidate as specified in subclause H.8.5.3.2.11 is invoked with the luma location ( xPb, yPb ), the variables nPbW and nPbH, the prediction list indication X , the view order index ivViewIdxLX, the disparity vector mvDispLX, and as the inputs and the outputs are the flag availableFlagLX IvMC, the motion vector mvLXIvMC and the reference index refIdxLXIvMC.
        + If availableFlagLXIvMC is equal to 1, the following applies:

mvLX = mvLXIvMC

refIdxLX = refIdxLXIvMC.

The following assignments are made for x = xPb.. ( xPb + nPbW − 1  ), y = yPb..( yPb + nPbH− 1 ):

MvDisp[ x ][ y ] = mvDispLX

MvRefinedDisp[ x ][ y ] = mvDispLX

RefViewIdx[ x ][ y ] = ivViewIdxLX

IvpMvFlagLX[ x ][ y ] = 1

* + - 1. Decoding process for inter prediction samples
         1. General

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,
* two variables nPbW and nPbH specifying the width and the height of the luma prediction block,
* the luma motion vectors mvL0 and mvL1,
* the chroma motion vectors mvCL0 and mvCL1,
* the reference indices refIdxL0 and refIdxL1,
* the prediction list utilization flags, predFlagL0, and predFlagL1.

Outputs of this process are:

* an (nCbSL)x(nCbSL) array predSamplesL of luma prediction samples, where nCbSL is derived as specified below,
* an (nCbSC)x(nCbSC) array predSamplesCb of chroma prediction samples for the component Cb, where nCbSC is derived as specified below,
* an (nCbSC)x(nCbSC) array predSamplesCr of chroma residual samples for the component Cr, where nCbSC is derived as specified below.
* The variable nCbSL is set equal to nCbS and the variable nCbSC is set equal to nCbS  >>  1.
* If VspModeFlag[ xCb + xBl ][ yCb + yBl ] is equal to 0, the following ordered steps apply:
  1. Let predSamplesL0L and predSamplesL1L be (nPbW)x(nPbH) arrays of predicted luma sample values and predSampleL0Cb, predSampleL1Cb, predSampleL0Cr, and predSampleL1Cr be (nPbW / 2)x(nPbH / 2) arrays of predicted chroma sample values.
  2. For X being each of 0 and 1, when predFlagLX is equal to 1, the following applies:
     + When predFlagLX is equal to 1, the following applies.
       - The variable resPredFlag is derived as specified in the following: [Ed. (CY): Based on F0123, the only check for resPredFlag is the iv\_res\_pred\_weight\_idx, however F105 introduces other checks for ARP, which may apply to temporal residual prediction. The additional checks in H-195 need to be closely inspected. ]
         * resPredFlag = ( iv\_res\_pred\_weight\_idx != 0 ) && RpRefPicAvailFlagLX &&  
            RefRpRefAvailFlagLX[ RefViewIdx[ xP ][ yP ] ] && ( DiffPicOrderCnt( currPic, RefPicListX[ refIdxLX ] ) !=  0 ) (H‑173)
       - If resPredFlag is equal to 1, the bilinear sample interpolation and residual prediction process as specified in subclause H.8.5.3.3.7 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.
       - Otherwise, ( resPredFlag is equal to 0 ), the following applies:
         * 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 8.5.3.3.2 with refIdxLX as input.
         * If DepthFlag is equal to 0, the arrays predSamplesLXL, predSamplesLXCb, and predSamplesLXCr are derived by invoking the fractional sample interpolation process specified in subclause 8.5.3.3.3 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 and mvCLX, and the reference arrays refPicLXL, refPicLXCb, and refPicLXCr as inputs.
         * Otherwise, (DepthFlag is equal to 1), arrays predSamplesLXL, predSamplesLXCb, and predSamplesLXCr are derived by invoking the full sample interpolation process specified in subclause H.8.5.3.3.5 with the luma locations ( xCb, yCb ), ( xBl, yBl ), the width and the height of the current luma prediction block nPbW, nPbH, the motion vectors mvLX, mvCLX, and the reference arrays with refPicLXL, refPicLXCb and refPicLXCr given as input.
  3. Depending on ic\_flag, the array predSamplesL is derived as specified in the following:
     + If ic\_flag is equal to 0, the following applies.
       - The array predSampleL of the prediction samples of luma component is derived by invoking the weighted sample prediction process specified in subclause 8.5.3.3.4 with the luma prediction block width nPbW, the luma prediction block height nPbH, and the sample arrays predSamplesL0L and predSamplesL1L, and the variables predFlagL0, predFlagL1, refIdxL0, refIdxL1, and cIdx equal to 0 as inputs. [Ed. (GT): There seems to be an issue with the base spec. In this subclause predSampleL s is of size (nCbSL)x(nCbSL), whereas the output of 8.5.3.3.4 is of size (nPbW)x(nPbH). ]
     + Otherwise ( ic\_flag is equal to 1), the following applies.
       - The array predSampleL of the prediction samples of luma component is derived by invoking the illumination compensated sample prediction process specified in subclause H.8.5.3.3.6, with the luma location ( xCb, yCb ), the size of the current luma coding block nCbS, the luma location ( xBl, yBl ), the width and the height of the current luma prediction block nPbW, nPbH, and the sample arrays predSamplesL0L and predSamplesL1L as well as predFlagL0, predFlagL1, refIdxL0, refIdxL1, mvL0, mvL1 and cIdx equal to 0 given as input.
  4. Depending on ic\_flag and nPbW, the arrays predSampleCb, and predSampleCr are derived as specified in the following:
     + If ic\_flag is equal to 0 or nPbW is not greater than 8, the following applies:
       - The array predSampleCb of the prediction samples of component Cb is derived by invoking the weighted sample prediction process specified in subclause 8.5.3.3.4 with the chroma prediction block width nPbWCb set equal to nPbW / 2, the chroma prediction block height nPbHCb set equal to nPbH / 2, the sample arrays predSamplesL0Cb and predSamplesL1Cb, and the variables predFlagL0, predFlagL1, refIdxL0, refIdxL1, and cIdx equal to 1 as inputs.
       - The array predSampleCr of the prediction samples of component Cr is derived by invoking the weighted sample prediction process specified in subclause 8.5.3.3.4 with the chroma prediction block width nPbWCr set equal to nPbW / 2, the chroma prediction block height nPbHCr set equal to nPbH / 2, the sample arrays predSamplesL0Cr and predSamplesL1Cr, and the variables predFlagL0, predFlagL1, refIdxL0, refIdxL1, and cIdx equal to 2 as inputs.
     + Otherwise ( ic\_flag is equal to 1 and nPbW is greater than 8), the following applies:
       - The array predSampleCb of the prediction samples of component Cb is derived by invoking the illumination compensated sample prediction process specified in subclause H.8.5.3.3.6, with the luma location ( xCb, yCb ), the size of the current luma coding block nCbS, with the chroma location ( xBl/2, yBl/2 ), the width and the height of the current chroma prediction block nPbWCb set equal to nPbW / 2, nPbHCb set equal to nPbH / 2, and the sample arrays predSamplesL0Cb and predSamplesL1Cb as well as predFlagL0, predFlagL1, refIdxL0, refIdxL1, mvCL0, mvCL1, and cIdx equal to 1 given as input.
       - The array predSampleCr of the prediction samples of component Cr is derived by invoking the illumination compensated sample prediction process specified in subclause H.8.5.3.3.6, with the luma location ( xCb, yCb ), the size of the current luma coding block nCbS, with the chroma location ( xBl / 2, yBl / 2 ), the width and the height of the current chroma prediction block nPbWCr set equal to nPbW / 2, nPbHCr set equal to nPbH / 2, and the sample arrays predSamplesL0Cr and predSamplesL1Cr as well as predFlagL0, predFlagL1, refIdxL0, refIdxL1, mvCL0, mvCL1, and cIdx equal to 2 given as input.
* Otherwise, ( VspModeFlag[ xCb + xBl ][ yCb + yBl ] is equal to 1 ), the following applies:
  + For X in the range of 0 to 1, inclusive, the following applies.
    - When predFlagLX is equal to 1, the arrays predSamplesL ,predSampleCb, and predSampleCr are derived by invoking the view synthesis prediction process as specified in subclause H.8.5.3.3.7.3, with the luma locations ( xCb, yCb ), ( xBl, yBl ), the width and the height of the current luma prediction block nPbW, nPbH, the prediction list indicator X and the reference index refIdxLX as the inputs and the outputs are the sample arrays predSamplesLXL, predSamplesLXCb, and predSamplesLXCr.
  + The array predSampleL of the prediction samples of luma component is derived by invoking the weighted sample prediction process specified in subclause 8.5.2.2.3 with the luma location ( xBl, yBl ), the width and the height of the current luma prediction block nPbW, nPbH, and the sample arrays predSamplesL0L and predSamplesL1L as well as predFlagL0, predFlagL1, refIdxL0, refIdxL1 and cIdx equal to 0 given as input.
  + The array predSampleCb of the prediction samples of component Cb is derived by invoking the weighted sample prediction process specified in subclause 8.5.2.2.3 with the chroma location ( xBl/2, yBl/2 ), the width and the height of the current chroma prediction block nPbWCb set equal to nPbW / 2, nPbHCb set equal to nPbH / 2, and the sample arrays predSamplesL0Cb and predSamplesL1Cb as well as predFlagL0, predFlagL1, refIdxL0, refIdxL1, and cIdx equal to 1 given as input.
  + The array predSampleCr of the prediction samples of component Cr is derived by invoking the weighted sample prediction process specified in subclause 8.5.2.2.3 with the chroma location ( xBl / 2, yBl / 2 ), the width and the height of the current chroma prediction block nPbWCr set equal to nPbW / 2, nPbHCr set equal to nPbH / 2, and the sample arrays predSamplesL0Cr and predSamplesL1Cr as well as predFlagL0, predFlagL1, refIdxL0, refIdxL1, and cIdx equal to 2 given as input.
    - * 2. 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,

Outputs of this process are:

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

The location ( xP, yP ) is derived by:

* + xP = xCb + xBl (H‑212)
  + yP = yCb + yBl (H‑213)

The variable availFlag is set equal to 0.

Depending on RpRefIdxLX, the following applies:

* If RpRefIdxLX is not equal to −1, the variable availFlag is set equal to 1, the variable refIdxLX is set equal to RpRefIdxLX and the residual prediction motion vector scaling process as specified in subclause H.8.5.3.3.7.3 is invoked with the prediction list utilization variable equal to X, the motion vector mvLX, and the RefPicListX[ refIdxLX ] and as inputs and modified mvLX as output.

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 currRefIdx as input.
* The arrays predSamplesLXL, predSamplesLXCb, and predSamplesLXCr are derived by invoking the bilinear sample interpolation process specified in subclause H.8.5.3.3.7.1 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.

When availFlag is equal to 1 and iv\_res\_pred\_weight\_idx is not equal to 0, the following applies:

* The variables rpPic, rpRefPic, mvRp and curRefIdx are derived as specified in the following:
  + Let rpPic be the picture with PicOrderCnt( rpPic ) equal to PicOrderCntVal and ViewIdx equal to RefViewIdx[ xP ][ yP ].
  + Let rpRefPic be the picture with PicOrderCnt( rpRefPic ) equal to RefPicListX[ RpRefIdxLX ] ) and ViewIdx equal to RefViewIdx[ xP ][ yP ],
    - The variable mvRp is set equal to MvDisp[ xP ][ yP ].
  + The variable curRefIdx is set equal to RpRefIdxLX.
* The arrays rpSamplesLXL, rpSamplesLXCb, and rpSamplesLXCr are derived as specified in the following:
  + Let the reference picture sample arrays rpPicLXL, rpPicLXCb, and rpPicLXCr corresponding to decoded sample arrays SL, SCb, SCr derived in subclause 8.7 for the previously-decoded picture rpPic.
  + The arrays rpSamplesLXL, rpSamplesLXCb, and rpSamplesLXCr are derived by invoking the bilinear sample interpolation process specified in subclause H.8.5.3.3.7.1 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 mvRp and mvCLX equal to mvRp, and the reference arrays with rpPicLXL, rpPicLXCb and rpPicLXCr as the inputs.
* The arrays rpRefSamplesLXL, rpRefSamplesLXCb, and rpRefSamplesLXCr are derived as specified in the following:
  + Let the reference picture sample arrays rpRefPicLXL, rpRefPicLXCb, and rpRefPicLXCr corresponding to decoded sample arrays SL, SCb, SCr derived in subclause 8.7 for the previously-decoded picture rpRefPic.
  + The arrays rpRefSamplesLXL, rpRefSamplesLXCb, and rpRefSamplesLXCr are derived by invoking the bilinear sample interpolation process specified in subclause H.8.5.3.3.7.1 with the luma locations ( xCb, yCb ), ( xBl, yBl ), the luma prediction block width nPbW, the luma prediction block height nPbH,, the motion vector mvLX equal to ( mvLX + mvRp ) and the motion vector mvCLX equal to ( mvCLX + mvRp ), and the reference arrays with rpRefPicLXL, rpRefPicLXCb and rpRefPicLXCr as the inputs.
* The variable shiftVal is set equal to ( iv\_res\_pred\_weight\_idx − 1 ).
* The modified prediction samples predSamplesLXL[ x ][ y ] with x = 0..( nPbW ) − 1 and y = 0..( nPbH ) − 1 are derived as specified in the following:
  + predSamplesLXL[ x ][ y ] = predSamplesLXL[ x ][ y ] +   
     ( ( rpSamplesLXL[ x ][ y ] − rpRefSamplesLXL[ x ][ y ] )  >>  shiftVal ) (H‑214)
* The modified prediction samples predSamplesLXCb[ x ][ y ] with x = 0..( nPbW /2 ) − 1 and y = 0..( nPbH /2 )−1 are derived as specified in the following:
  + predSamplesLXCb[ x ][ y ] = predSamplesLXCb[ x ][ y ] +   
     ( ( rpSamplesLXCb[ x ][ y ] − rpRefSamplesLXCb[ x ][ y ] )  >>  shiftVal ) (H‑215)
* The modified prediction samples predSamplesLXCr[ x ][ y ] with x = 0..( nPbW /2 ) − 1 and y = 0..( nPbH /2 ) − 1 are derived as specified in the following:
  + predSamplesLXCr[ x ][ y ] = predSamplesLXCr[ x ][ y ] +   
     ( ( rpSamplesLXCr[ x ][ y ] − rpRefamplesLXCr[ x ][ y ] )  >>  shiftVal ) (H‑216)

* + 1. Derivation process for disparity vectors

Inputs to this process are:

a luma location ( xCb, yCb ) of the top-left sample of the current luma coding block relative to the top-left luma sample of the current picture,

a variable nCbS specifying the size of the current luma coding block,

The flag availableDV is set equal to 0, and both components of the disparity vector mvDisp are set equal to 0.

The variable checkParallelMergeFlag is derived as follows:

If one or more of the following conditions are true, checkParallelMergeFlag is set equal to 1.

PredMode[ xCb ][ yCb ] is equal to MODE\_SKIP.

PredMode[ xCb ][ yCb ] is equal to MODE\_INTER and merge\_flag[ xCb ][ yCb ] is equal to 1.

Otherwise, checkParallelMergeFlag is set equal to 0.

The derivation process for a disparity vector from temporal neighbour block as specified in subclause H.8.5.5.1 is invoked with the luma location ( xCb, yCb ), and the variable nCbS as inputs, and the outputs are the flag availableDV, the disparity vector mvDisp and the reference view order index refViewIdx.

When availableDV is equal to 0, for each N being A1, B1 and ( xN, yN ) being ( xCb − 1,  yCb + nCbS − 1 ), ( xCb + nCbS − 1,  yCb − 1 ), respectively, the following ordered steps apply.

* 1. When yCb − 1 is less than ( ( yCb  >>  Log2CtbSizeY )  <<  Log2CtbSizeY ), the following applies.

xB1 = ( ( xB1  >>  3 )  <<  3 ) + ( ( xB1  >>  3 ) & 1) \* 7 (H‑251)

The derivation process for z-scan order block availability as specified in subclause 6.4.1 is invoked with ( xCurr, yCurr ) set equal to the ( xCb, yCb ) and the luma location ( xN, yN ) as the input and the output assigned to availableN.

When availableN is equal to 1 and PredMode[ xN ][ yN ] is equal to MODE\_INTRA, availableN is set equal to 0. [Ed. (GT): 2+3 correspond to 6.4.2 for CU when ( xN, yN ) outside CU. Cross-check appreciated. ]

When all of the following conditions are true, availableN is set equal to 0.

checkParallelMergeFlag is equal to 1

( xCb  >>  ( log2\_parallel\_merge\_level\_minus2 + 2) ) is equal to  
( xN  >>  ( log2\_parallel\_merge\_level\_minus2 + 2) )

( yCb  >>  ( log2\_parallel\_merge\_level\_minus2 + 2) ) is equal to ( yN  >>  ( log2\_parallel\_merge\_level\_minus2 + 2) ).

The flag availableIvpMvSearchFlagN is set equal to availableN.

When one of the following conditions is true, N is equal to B1 and ( ( yN  >>  Log2CtbSizeY )  <<  Log2CtbSizeY ) is less than ( ( yCb >> Log2CtbSizeY )  <<  Log2CtbSizeY), availableIvpMvSearchFlagN is set equal to 0.

The flag availableFlagIvpMvN is set equal to 0.

For each X from 0 to 1, the following applies:

When availableDV is equal to 0, availableN is equal to 1, RefIdxLX[ xN ][ yN ] is greater than or equal to 0, and PredFlagLX[ xN ][ yN ] is equal to 1, the following applies:

If RefPicListX[ RefIdxLX[ xN ][ yN ] ] is an inter-view reference picture of the current picture, the following applies:

refViewIdx = ViewIdx( RefPicListX[ RefIdxLX[ xN ][ yN ] ] ) (H‑252)

mvDisp = MvLXN[ xN ][ yN ] (H‑253)

availableDV = 1 (H‑254)

Otherwise (RefPicListX[ RefIdxLX[ xN ][ yN ] ] is not an inter-view reference picture), the following applies:

When availableIvpMvSearchFlagN is equal to 1, availableFlagIvpMvN is equal to 0, and IvpMvFlagLX[ xN ][ yN ] is equal to 1, the following applies:

ivpMvDispN = MvRefinedDisp[ xN ][ yN ] (H‑255)

refViewIdxN = RefViewIdx[ xN ][ yN ] (H‑256)

availableFlagIvpMvN = 1 (H‑257)

When availableDV is equal to 0 for each N being A1 and B1, the following applies.

When availableDV is equal to 0 and availableFlagIvpMvN is equal to 1, the following applies:

mvDisp = ivpMvDispN (H‑258)

refViewIdx = refViewIdxN (H‑259)

availableDV = 1 (H‑260)

When availableDV is equal to 0, refViewIdx is set equal to DefaultViewIdx, and mvDisp is set equal to ( 0, 0 ). The variable mvRefinedDisp is set equal to mvDisp.

When depth\_refinement\_flag[ nuh\_layer\_id ]is equal to 1, the following ordered steps apply:

* 1. The derivation process for a disparity sample array as specified in subclause H.8.5.5.2 is invoked with the luma locations xCb, yCb, the disparity vector mvDisp, the view identifier refViewIdx, the variable nPSW equal to nCbS, the variable nPSH equal to nCbS, and the variable splitFlag equal to 0 as the inputs, and the output is the array disparitySamples of size (nCbS)x(nCbS).

The horizontal component of the disparity vector mvRefinedDisp[ 0 ] is set equal to disparitySamples[ 0 ][ 0 ].

For use in derivation processes of variables invoked later in the decoding process, the following assignments are made for x = xCb.. ( xCb + nCbS − 1 ), y = yCb..( yCb + nCbS− 1 ):

MvDisp[ x ][ y ] = mvDisp (H‑261)

MvRefinedDisp[ x ][ y ] = mvRefinedDisp (H‑262)

RefViewIdx[ x ][ y ] = refViewIdx (H‑263)

DefaultDispFlag[ x ][ y ] = !availableDV (H‑264)