**Derivation of the Inter-view DV merge candidate**

**H.8.5.2.1.11 Derivation process for a disparity inter-view motion vector candidate**

This process is not invoked when iv\_mv\_pred\_flag[ nuh\_layer\_id ] is equal to 0.

Inputs to this process are:

* a luma location ( xP, yP ) of the top-left luma sample of the current prediction unit relative to the top-left luma sample of the current picture,
* variables nPSW and nPSH specifying the width and the height, respectively, of the current prediction unit,
* a prediction list indication X,
* a reference view index refViewIdx,
* a disparity vector mvDisp,
* a prediction list indication lIdxDisp
* a reference index refIdxLlIdxDisp specifying a reference picture in the reference picture list RefPicListlIdxDisp
* a flag mergeFlag specifying whether a merge candidate is derived,
* a reference index refIdxLX specifying a reference picture in the reference picture list RefPicListLX.

Outputs of this process are:

* a flag availableFlagLXInterView specifying whether the disparity inter-view motion vector candidate is available,
* a disparity inter-view motion vector candidate mvLXInterView (if availableFlagLXInterView is equal to 1),
* a reference index refIdxLX specifying a reference picture in the reference picture list RefPicListLX.

The flag availableFlagLXInterView is set equal to 0, both components of mvLXInterView are set equal to 0.

When mergeFlag is equal to 1,

* If X is equal to lIdxDisp, availableFlagLXInterView is set equal to 1, mvLXInterView is set equal to mvDisp, refIdxLX is set equal to refIdxLlIdxDisp.
* Otherwise, the whole decoding process specified in this subclause terminates.

When merge flag is equal to 0,

* When PicOrderCntVal of the picture RefPicListX[refIdxLX] is equal to the PicOrderCntVal of the current picture and ViewIdx(RefPicListX[refIdxLX] ) is equal to refViewIdx, mvLXInterView is set equal to mvDisp.

~~When X is equal to 1 and the current slice is not a B slice the whole decoding process specified in this subclause terminates.~~

~~For each i from ( mergeFlag ? 0 : refIdxLX ) to ( mergeFlag ? num\_ref\_idx\_lX\_active\_minus1 : refIdxLX), inclusive, the following applies:~~

* ~~When PicOrderCntVal of the picture RefPicListX[ i ] is equal to the PicOrderCntVal or the current picture and availableFlagLXInterView is equal to 0, availableFlagLXInterView is set equal to 1 and the following applies:~~
* ~~The motion vector mvLXInterView is derived by~~

~~mvLXInterView[ 0 ] = mvDisp[ 0 ] (‑129)  
mvLXInterView[ 1 ] = 0 (‑130)~~

* ~~When mergeFlag is equal to 1, the reference index refIdxLX is derived by~~

~~refIdxLX = i~~

**Derivation of the VSP merge candidate**

**H.8.5.2.1.12 Derivation process for a view synthesis prediction merge candidate**

Inputs to this process are:

* a luma location ( xC, yC ) of the top-left sample of the current luma coding block relative to the top-left luma sample of the current picture,

Outputs of this process are

* the availability flag availableFlagVSP whether the VSP merge candidate is available,
* the reference indices refIdxL0VSP and refIdxL1VSP ,
* the prediction list utilization flags predFlagL0VSP and predFlagL1VSP,
* the motion vectors mvL0VSP and mvL1VSP.

The variable availableFlagVSP is set equal to 1, the variables predFlagL0VSP and predFlagL1VSP are set equal to 0, the variables refIdxL0VSP and refIdxL1VSP are set equal to −1 and the variable refViewAvailableFlag is set equal to 0.

* X is set equal to lIdxDisp[xC][yC], predFlagLXVSP is set equal to 1, mvLXVSP is set equal to MvDisp[ xC ][ yC ], refIdxLXVSP is set equal to refIdxDisp[xC][yC].
* ~~For X in the range of 0 to 1, inclusive, the following applies:~~ 
  + ~~For i in the range of 0 to NumRefPicsLX – 1, inclusive, the following applies:~~ 
    - ~~When refViewAvailableFlag is equal to 0 and ViewIdx( RefPicListX[ i ] ) is equal to RefViewIdx[ xC ][ yC ], the following applies:~~

~~refViewAvailableFlag = 1 (‑132)  
predFlagLXVSP = 1 (‑133)  
mvLXVSP = MvDisp[ xC ][ yC ] (‑134)  
refIdxLXVSP = i (‑135)  
Y = 1 – X (‑136)~~

~~When the current slice is a B slice and refViewAvailableFlag is equal to 1, refViewAvailableFlag is set equal to 0 and the following applies:~~

* + ~~For i in the range of 0 to NumRefPicsLY – 1, inclusive, the following applies.~~
    - ~~When refViewAvailableFlag is equal to 0 and ViewIdx( RefPicListY[ i ] ) is not equal to RefViewIdx[ xC ][ yC ] and ViewIdx( RefPicListY[ i ] ) is not equal to ViewIdx, the following applies:~~

~~refViewAvailableFlag = 1 (‑137)  
predFlagLYVSP = 1 (‑138)  
mvLYVSP = MvDisp[ xC ][ yC ] (‑139)  
refIdxLYVSP = i~~

**View synthesis prediction process**

8.5.2.2 Decoding process for inter prediction samples

……….

* Otherwise, ( VspModeFlag[ xC + xB ][ yC + yB ] is equal to 1 ), the following applies:
  + X is set equal to 0 if prediction direction of the VSP merge candidate is 1, X is set equal to 1 if prediction direction of the VSP merge candidate is 2, predFlagLXVSP is set equal to 1.
  + For B-slice, if there is an inter-view reference picture RefPicListY[i] (Y= 1- X, i is in the range of 0 to NumRefPicsLY – 1) exists in RefPicListY with view index not equal to ViewIdx (RefPicListX[refIdxLX ] ), predFlagLYVSP is set equal to 1.
  + For X in the range of 0 to 1, inclusive, the following applies.
    - When predFlagLXVSP is equal to 1, the arrays predSamplesL ,predSampleCb, and predSampleCr are derived by invoking the view synthesis prediction process as specified in subclause , with the luma locations ( xC, yC ), ( xB, yB ), the width and the height of the current luma prediction block nPbW, nPbH, ~~the motion vector mvLX,~~ 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 ( xB, yB ), the width and the height of the current luma prediction block nPbW, nPbH, and the sample arrays predSamplesL0L and predSamplesL1L as well as predFlagL0VSP, predFlagL1 VSP, ~~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 ( xB/2, yB/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 predFlagL0VSP, predFlagL1VSP, ~~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 ( xB/2, yB/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 predFlagL0VSP, predFlagL1VSP, ~~refIdxL0, refIdxL1,~~ and cIdx equal to 2 given as input.

8.5.2.2.7 View synthesis prediction process

Inputs to this process are:

* a location ( xC, yC ) specifying the top-left sample of the current luma coding block relative to the top left sample of the current picture,
* a location ( xB, yB ) specifying the top-left sample of the current prediction block relative to the top left sample of the current coding block,
* the width and height of this prediction block, nPbW and nPbH,
* the prediction list indicator X
* ~~a motion vector mvLX~~
* ~~the reference index refIdxLX~~

Outputs of this process are:

– an array predSamplesL of luma prediction samples,

– an array predSamplesCb of chroma prediction samples for the component Cb

– an array predSamplesCr of chroma prediction samples for the component Cr

The location ( xP, yP ) given in full-sample units of the upper-left luma samples of the current prediction block relative to the upper-left luma sample location of the current picture is derived by:

xP = xC + xB (H‑220)  
yP = yC + yB (H‑221)

The prediction list indicator lIdxDisp is set equal to 0 if prediction direction of the VSP merge candidate is 1, X is set equal to 1 if prediction direction of the VSP merge candidate is 2.

If X is equal to lIdxDisp, the reference index refIdxLXVSP is set equal to refIdxLX.

Otherwise, find an inter-view reference picture RefPicListX[i] (i is in the range of 0 to NumRefPicsLX – 1) exists in RefPicListX with view index not equal to ViewIdx (RefPicListlIdxDisp [refIdxL lIdxDisp ] ), the reference index refIdxLXVSP is set equal to i.

The reference picture consisting of an ordered two-dimensional array refPicL of luma samples and two ordered two-dimensional arrays refPicCb and refPicCr of chroma samples is derived by invoking the process specified in subclause 8.5.2.2.1 with refIdxLXVSP as input.

The variable refViewIdx is set equal to the ViewIdx( RefPicListX[ refIdxLXVSP ] ) and the variable depthViewIdx is set equal to ~~RefViewIdx[ xC + xB ][ yC + yB ]~~ ViewIdx( RefPicListlIdxDisp[ refIdxLlIdxDisp ] ).

The derivation process for disparity sample array as specified in section H.8.5.4.3 is invoked with the luma location ( xP, yP ), the disparity vector mvDisp being equal to mvL~~X~~lIdxDisp, the variable refViewIdx, the variable depthViewIdx, the variables nPSW and nPSH, the variable nSubBlkW being equal 4 and the variable nSubBlkW being equal to 4 as the inputs, and the output is the array disparitySamples of size (nPSW)x(nPSH).

Let ( xIntL, yIntL ) be a luma location given in full-sample units and ( xFracL, yFracL ) be an offset given in quarter-sample units.

For each luma sample location ( xL = 0..nPbW−1, yL = 0..nPbH−1 ) inside the prediction luma sample array predSamplesL, the corresponding prediction luma sample value predSamplesL[ xL ][ yL ] is derived as follows:

* The variables xIntL, yIntL, xFracL, and yFracL are derived by

xIntL = xP + xL+ disparitySamples[ xL ][ yL ] (H‑222)  
yIntL = yP + yL (H‑223)

xFracL = disparitySamples[ xL ][ yL ] & 3 (H‑224)  
yFracL = 0 (H‑225)

* The prediction luma sample value predSamplesL[ xL][ yL ] is derived by invoking the process specified in subclause 8.5.2.2.2.1 with ( xIntL, yIntL ), ( xFracL, yFracL ) and refPicL given as input.

Let ( xIntC, yIntC ) be a chroma location given in full-sample units and ( xFracC, yFracC ) be an offset given in one-eighth sample units.

For each chroma sample location ( xC = 0..nPbW/2−1, yC = 0..nPbH/2−1 ) inside the prediction chroma sample arrays predSamplesCb and predSamplesCr, the corresponding prediction chroma sample values predSampleLXCb[ xC ][ yC ] and predSamplesCr[ xC ][ yC ] are derived as follows:

* The variables xIntC, yIntC, xFracC, and yFracC are derived by

xIntC = ( xP / 2 ) + xC + disparitySamples[ xC << 1 ][ yC << 1 ] (H‑226)  
yIntC = ( yP / 2 ) + yC (H‑227)

xFracC = disparitySamples[ xC << 1][ yC << 1 ] & 7 (H‑228)  
yFracC = 0 (H‑229)

* The prediction sample value predSamplesCb[ xC ][ yC ] is derived by invoking the process specified in subclause 8.5.2.2.2.2 with ( xIntC, yIntC ), ( xFracC, yFracC ) and refPicCb given as input.
* The prediction sample value predSamplesCr[ xC ][ yC ] is derived by invoking the process specified in subclause 8.5.2.2.2.2 with ( xIntC, yIntC ), ( xFracC, yFracC ) and refPicCr given as input.

**Derivation of spatial MVPs for AMVP**

* + - * 1. **Derivation process for motion vector predictor candidates**

Inputs to this process are

* a luma location ( xC, yC ) of the top-left sample of the current luma coding block relative to the top-left luma sample of the current picture,
* a variable nCS specifying the size of the current luma coding block,
* a luma location ( xP, yP ) specifying the top-left sample of the current luma prediction block relative to the top-left luma sample of the current picture,
* variables specifying the width and the height of the luma prediction block, nPbW and nPbH,
* the reference index of the current prediction unit partition refIdxLX (with X being 0 or 1),
* a variable partIdx specifying the index of the current prediction unit within the current coding unit.

Outputs of this process are (with N being replaced by A, or B)

* the motion vectors mvLXN of the neighbouring prediction units,
* the availability flags availableFlagLXN of the neighbouring prediction units.

The variable isScaledFlagLX with X being 0 or 1 is set equal to 0.

The function ViewId( picX ) is specified as follows:

ViewId( picX ) = view\_id of the picture picX (‑95)

The variable currViewId is set equal to view\_id[ nuh\_layer\_id ]..

If RefPicListX[ refIdxLX ] is a long-term reference picture, the variable currRefIsLtFlag is set equal to 1, otherwise ( RefPicListX[ refIdxLX ] is not a long-term reference picture ), currRefIsLtFlag is set equal to 0.

The motion vector mvLXA and the availability flag availableFlagLXA are derived in the following ordered steps:

* 1. The sample location (xA0, yA0) is set equal to (xP − 1, yP + nPbH) and the sample location (xA1, yA1) is set equal to (xA0, yA0 − 1).
  2. The availability flag availableFlagLXA is set equal to 0 and the both components of mvLXA are set equal to 0.
  3. The availability derivation process for a prediction block as specified in subclause 6.4.2 is invoked with the luma location ( xC, yC ), the current luma coding block size nCbS set equal to nCS, the luma location ( xP, yP ), the width and the height of the luma prediction block nPbW and nPbH, the luma location ( xN, yN ) set equal to ( xA0, yA0 ) and the partition index partIdx as inputs and the output is assigned to the prediction block availability flag availableA0.
  4. The availability derivation process for a prediction block as specified in subclause 6.4.2 is invoked with the luma location ( xC, yC ), the current luma coding block size nCbS set equal to nCS, the luma location ( xP, yP ), the width and the height of the luma prediction block nPbW and nPbH, the luma location ( xN, yN ) set equal to ( xA1, yA1 ) and the partition index partIdx as inputs and the output is assigned to the prediction block availability flag availableA1.
  5. When availableA0 or availableA1 is equal to TRUE, the variable isScaledFlagLX is set equal to 1.
  6. For ( xAk, yAk ) from ( xA0, yA0 ) to ( xA1, yA1 ), the following applies repeatedly until availableFlagLXA is equal to 1:
     + When availableAk is equal to TRUE, PredMode[ xAk][ yAk ] is not equal to MODE\_INTRA and availableFlagLXA is equal to 0, the following applies.
       - If, PredFlagLX[ xAk ][ yAk ] is equal to 1 and the reference index refIdxLX[ xAk ][ yAk ] is equal to the reference index of the current prediction unit refIdxLX, availableFlagLXA is set equal to 1 and the following assignments are made.

mvLXA = MvLX[ xAk ][ yAk ] (H‑96)  
refIdxA = RefIdxLX[ xAk ][ yAk ] (H‑97)

* + - * Otherwise, the following applies:
        + If RefPicListY[ refIdxLY[ xAk ][ yAk ] is a long-term reference picture, the variable nbRefIsLtFlag is set equal to 1,
        + otherwise ( RefPicListY[ refIdxLY[ xAk ][ yAk ] is not a long-term reference picture ), nbRefIsLtFlag is set equal to 0.

When PredFlagLY[ xAk ][ yAk ] (with Y = !X) is equal to 1 and PicOrderCnt( RefPicListY[ refIdxLY[ xAk ][ yAk ] ] ) is equal to PicOrderCnt( RefPicListX[ refIdxLX ] ), ViewId(RefPicListY[ refIdxLY[ xAk ][ yAk ] ]) is equal to ViewId(RefPicListX[ refIdxLX ] ), and currRefIsLtFlag is equal to nbRefIsLtFlag, availableFlagLXA is set equal to 1 and the following assignments are made.  
mvLXA = MvLY[ xAk ][ yAk ] (H‑98)  
refIdxA = RefIdxLY[ xAk ][ yAk ] (H‑99)

* 1. When availableFlagLXA is equal to 0, for ( xAk, yAk ) from ( xA0, yA0 ) to ( xA1, yA1 ), the following applies repeatedly until availableFlagLXA is equal to 1:
     + When availableAk is equal to TRUE, PredMode[ xAk][ yAk ] is not equal to MODE\_INTRA and availableFlagLXA is equal to 0, the following applies.
       - If PredFlagLX[ xAk ][ yAk ] is equal to 1, availableFlagLXA is set equal to 1 and the following assignments are made.

mvLXA = MvLX[ xAk ][ yAk ] (H‑100)  
refIdxA = RefIdxLX[ xAk ][ yAk ] (H‑101)  
refPicListA = RefPicListX (H‑102)

* + - * Otherwise, if PredFlagLY[ xAk ][ yAk ] (with Y = !X) is equal to 1, availableFlagLXA is set equal to 1 and the following assignments are made.

mvLXA = MvLY[ xAk ][ yAk ] (H‑103)  
refIdxA = RefIdxLY[ xAk ][ yAk ] (H‑104)  
refPicListA = RefPicListY (H‑105)

* + - If RefPicListA[ refIdxA ] is a long-term reference picture, the variable nbRefIsLtFlag is set equal to 1, otherwise ( RefPicListA[ refIdxA ] is not a long-term reference picture ), nbRefIsLtFlag is set equal to 0.
    - When none of the following conditions is true,
      * + availableFlagLXA is equal to 0,
        + nbRefIsLtFlag is not equal to currRefIsLtFlag,
        + currRefIsLtFlag is equal to 0 and PicOrderCnt( refPicListA[ refIdxA ] ) is equal to PicOrderCnt( RefPicListX[ refIdxLX ] )
        + currRefIsLtFlag is equal to 1, and ViewId( refPicListA[ refIdxA ] ) is equal to ViewId( refPicListX[ refIdxX ] ).
        + currRefIsLtFlag is equal to 1 and ( ( ViewIdx != 0 ) && iv\_mv\_scaling\_flag ) is equal to 0.

mvLXA is derived as specified below.

tx = ( 16384 + ( Abs( td ) >> 1 ) ) / td (H‑106)

distScaleFactor = Clip3( −4096, 4095, ( tb \* tx + 32 ) >> 6 ) (H‑107)

mvLXA = Clip3( −32768, 32767, Sign2( distScaleFactor \* mvLXA ) \*    
 ( (Abs( distScaleFactor \* mvLXA ) + 127 ) >> 8 ) ) (H‑108)

where td and tb are derived as

* + - * + If RefPicListX[ refIdxLX ] is short-term reference picture,

td = Clip3( −128, 127, PicOrderCntVal – PicOrderCnt( refPicListA[ refIdxA ] ) ) (H‑109)

tb = Clip3( −128, 127, PicOrderCntVal – PicOrderCnt( RefPicListX[ refIdxLX ] ) ) (H‑110)

* + - * + Otherwise ( RefPicListX[ refIdxLX ] is not a short-term reference picture)

td = Clip3( −128, 127, currViewId – ViewId( refPicListA[ refIdxA ] ) ) (‑111)

tb = Clip3( −128, 127, currViewId – ViewId( RefPicListX[ refIdxLX ] ) ) (‑112)

The motion vector mvLXB and the availability flag availableFlagLXB are derived in the following ordered steps:

* 1. Let a set of three sample location (xBk, yBk), with k = 0,1,2, specifies sample locations with xB0 = xP + nPbW, xB1 = xB0− 1, xB2 = xP − 1 and yBk = yP − 1. The set of sample locations ( xBk, yBk ) represent the sample locations immediately to the upper side of the above partition boundary and its extended line.
  2. When yP−1 is less than (( yC >> Log2CtbSizeY ) << Log2CtbSizeY), the following applies.

xB0 = (xB0>>3)<<3) + ((xB0>>3)&1)\*7 (H‑113)  
xB1 = (xB1>>3)<<3) + ((xB1>>3)&1)\*7 (H‑114)  
xB2 = (xB2>>3)<<3) + ((xB2>>3)&1)\*7 (H‑115)

* 1. The availability flag availableFlagLXB is set equal to 0 and the both components of mvLXB are set equal to 0.
  2. For ( xBk, yBk ) from ( xB0, yB0 ) to ( xB2, yB2 ) where xB0 = xP + nPbW, xB1 = xB0 − 1, and xB2 =  xP − 1, the following applies repeatedly until availableFlagLXB is equal to 1:
     + The availability derivation process for a prediction block as specified in subclause 6.4.2 is invoked with the luma location ( xC, yC ), the current luma coding block size nCbS set equal to nCS, the luma location ( xP, yP ), the width and the height of the luma prediction block nPbW and nPbH, the luma location ( xN, yN ) set equal to ( xBk, yBk ) and the partition index partIdx as inputs and the output is assigned to the prediction block availability flag availableBk.
     + When availableBk is equal to TRUE and availableFlagLXB is equal to 0, the following applies.
       - If PredFlagLX[ xBk ][ yBk ] is equal to 1, and the reference index refIdxLX[ xBk ][ yBk ] is equal to the reference index of the current prediction unit refIdxLX, availableFlagLXB is set equal to 1 and the following assignments are made.

mvLXB = MvLX[ xBk ][ yBk ] (H‑116)  
refIdxB = RefIdxLX[ xBk ][ yBk ] (H‑117)

the following applies:

* + - * + If RefPicListY[ refIdxLY[ xAk ][ yAk ] is a long-term reference picture, the variable nbRefIsLtFlag is set equal to 1, otherwise ( RefPicListY[ refIdxLY[ xAk ][ yAk ] is not a long-term reference picture ), nbRefIsLtFlag is set equal to 0.
        + When, if PredFlagLY[ xBk ][ yBk ] (with Y = !X) is equal to 1 and PicOrderCnt( RefPicListY[ refIdxLY[ xBk ][ yBk ] ] ) is equal to PicOrderCnt( RefPicListX[ refIdxLX ] ), ViewId(RefPicListY[ refIdxLY[ xBk ][ yBk ] ]) is equal to ViewId(RefPicListX[ refIdxLX ] ), and currRefIsLtFlag is equal to nbRefIsLtFlag, availableFlagLXB is set equal to 1 and the following assignments are made.

mvLXB = MvLY[ xBk ][ yBk ] (H‑118)  
refIdxB = RefIdxLY[ xBk ][ yBk ] (H‑119)

* 1. When isScaledFlagLX is equal to 0 and availableFlagLXB is equal to 1, availableFlagLXA is set equal to 1 and the following assignments are made.

mvLXA = mvLXB (H‑120)  
refIdxA = refIdxLXB (H‑121)

* 1. When isScaledFlagLX is equal to 0, availableFlagLXB is set equal to 0 and for ( xBk, yBk ) from ( xB0, yB0 ) to ( xB2, yB2 ) where xB0 = xP + nPbW, xB1 = xB0 − 1, and xB2 = xP − 1, the following applies repeatedly until availableFlagLXB is equal to 1:
     + The availability derivation process for a prediction block as specified in subclause 6.4.2 is invoked with the luma location ( xC, yC ), the current luma coding block size nCbS set equal to nCS, the luma location ( xP, yP ), the width and the height of the luma prediction block nPbW and nPbH, the luma location ( xN, yN ) set equal to ( xBk, yBk ) and the partition index partIdx as inputs and the output is assigned to the prediction block availability flag availableBk.
     + When availableBk is equal to TRUE and availableFlagLXB is equal to 0, the following applies.
       - If PredFlagLX[ xBk ][ yBk ] is equal to 1, availableFlagLXB is set equal to 1 and the following assignments are made.

mvLXB = MvLX[ xBk ][ yBk ] (H‑122)  
refIdxB = RefIdxLX[ xBk ][ yBk ] (H‑123)  
refPicListB = RefPicListX (H‑124)

* + - * Otherwise, if PredFlagLY[ xBk ][ yBk ] (with Y = !X) is equal to 1, availableFlagLXB is set equal to 1 and the following assignments are made.

mvLXB = MvLY[ xBk ][ yBk ] (H‑125)  
refIdxB = RefIdxLY[ xBk ][ yBk ] (H‑126)  
refPicListB = RefPicListY (H‑127)

* + - If RefPicListB[ refIdxB ] is a long-term reference picture, the variable nbIsLtFlag is set equal to 1, otherwise ( RefPicListB[ refIdxB ] is not a long-term reference picture ), nbIsLtFlag is set equal to 0.
    - When none of the following conditions is true,
      * + availableFlagLXB is equal to 0,
        + nbRefIsLtFlag is not equal to currRefIsLtFlag,
        + currRefIsLtFlag is equal to 0 and PicOrderCnt( refPicListB[ refIdxB ] ) is equal to PicOrderCnt( RefPicListX[ refIdxLX ] )
        + currRefIsLtFlag is equal to 1, and ViewId( refPicListB[ refIdxB ] ) is equal to ViewId( refPicListX[ refIdxX ] ).
        + currRefIsLtFlag is equal to 1 and ( ( ViewIdx != 0 ) && iv\_mv\_scaling\_flag ) is equal to 0.

mvLXB is derived as specified below.

tx = ( 16384 + ( Abs( td ) >> 1 ) ) / td (H‑128)

distScaleFactor = Clip3( −4096, 4095, ( tb \* tx + 32 ) >> 6 ) (H‑129)

mvLXB =Clip3( −32768, 32767, Sign2( distScaleFactor \* mvLXB ) \*   
 ( (Abs( distScaleFactor \* mvLXB ) + 127 ) >> 8 ) ) (H‑130)

where td and tb are derived as

* + - * + If RefPicListX[ refIdxLX ] is short-term reference picture,

td = Clip3( −128, 127, PicOrderCntVal – PicOrderCnt( refPicListB[ refIdxB ] ) ) (H‑131)

tb = Clip3( −128, 127, PicOrderCntVal – PicOrderCnt( RefPicListX[ refIdxLX ] ) ) (H‑132)

* + - * + Otherwise ( RefPicListX[ refIdxLX ] is not a short-term reference picture)

td = Clip3( −128, 127, currViewId – ViewId( refPicListA[ refIdxB ] ) ) (‑133)

tb = Clip3( −128, 127, currViewId – ViewId( RefPicListX[ refIdxLX ] ) ) (‑134)

**DV derivation from the temporal neighboring blocks**

**8.5.4.2 Derivation process for a disparity vector in a block of a candidate picture**

Inputs to this process are:

* a candidate picture candPic,
* a luma location ( xPCand , yPCand ) relative to the top-left luma sample of the candidate picture.

Outputs of this process are:

* a flag availableDV specifying whether the disparity vector is available,
* a disparity vector mvDisp
* a reference view order index refViewIdx.

The motion vector mvDisp and the availability flag availableDV are derived as specified by the following ordered steps and the whole decoding process of this sub-clause terminates once availableDV is set equal to 1.

* 1. Let colPu the prediction unit in candPic covering the position ( ( xPCand  >>  4 ) <<4 ,  ( yPCand  >>  4 ) <<4 ).
  2. The position ( xPCol, yPCol ) is set equal to the position of the top-left sample of colPu relative to the top-left luma sample of the candPic.
  3. If slice\_type is equal to B, the variable dir is set equal to collocated\_from\_l0\_flag, otherwise, dir is set equal to 1 – collocated\_from\_l0\_flag.
  4. The flag availableDV is set equal to 0, and both components of mvDisp are set equal to 0.
  5. For each X from dir to 1 – dir, inclusive, the following applies:
     + Let candPicRefPicList be the reference picture list RefPicListX of candPic.
     + The variable candPredFlag is set equal to the prediction list utilization flag PredFlagLX of candPic.
     + The variable candRefIdx is set equal to the reference indices RefIdxLX of candPic.
     + The variable candMV is set equal to the motion vectors MvLX of candPic.
     + When the ViewIdx of candPicRefPicList[ candRefIdx[ xPCol ][ yPCol ] ] is not equal to the ViewIdx of candPic ~~and~~, candPredFlag[ xPCol ][ yPCol ] is equal to 1 and there is an inter-view reference picture with view index equal to view index of candPicRefPicList[ candRefIdx[ xPCol ][ yPCol ] ] exists in the reference lists of current CU, the following applies:

mvDisp = candMV[ xPCol ][ yPCol ] (H‑250)  
availableDV = 1 (H‑251)  
refViewIdx = ViewIdx of candPicRefPicList[ candRefIdx[ xPCol ][ yPCol ] ] (H‑252)