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

Input 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 luma location ( xB, yB ) 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 nCS specifying the size of the current luma coding block,
* variables specifying the width and the height of the luma prediction block, nPbW and nPbH,
* a variable partIdx specifying the index of the current prediction unit within the current coding unit.

Outputs of this process are

* luma motion vectors mvL0 and mvL1
* chroma motion vectors mvCL0 and mvCL1,
* reference indices refIdxL0 and refIdxL1,
* prediction list utilization flags predFlagL0 and predFlagL1.

Let ( xP, yP ) specify the top-left sample location of the current luma prediction block relative to the top-left luma sample of the current picture where xP = xC + xB and yP = yC + yB.

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( picX, refIdx, LX ), with X being either 0 or 1, is defined as follows. If the picture with index refIdx from reference picture list LX of the picture picX was marked as "used for long term reference" at the time when picX was the current picture, LongTermRefPic( picX, refIdx, LX ) is equal to 1; otherwise LongTermRefPic( picX, refIdx, LX ) is equal to 0.

The flag vspModeFlag is set equal to 0, the variable refViewIdx is set equal to −1 and the flag availableFlagIvMC is set equal to 0.

For X being replaced by 0 and 1, the variable ivpMvFlagLX and the variable ivpMvDispLX are derived as:

* 1. ivpMvFlagLX = 0 (‑)  
     ivpMvDispLX[ 0 ] = 0 (‑)  
     ivpMvDispLX[ 1 ] = 0 (‑)

For the derivation of the variables mvL0 and mvL1, refIdxL0 and refIdxL1 as well as prefFlagL0 and prefFlagL1, the following applies.

* If PredMode[ xC ][ yC ] is equal to MODE\_SKIP, the derivation process for luma motion vectors for merge mode as specified in subclause is invoked with the luma location ( xC, yC ), the luma location ( xP, yP ), variables nCS, 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 disparity vectors ivpMvDispL0 and ivpMvDispL1, the flag vspModeFlag, the variable refViewIdx, the flag availableFlagIvMC, and the flag cbfAllZero. .
* Otherwise, if PredMode[ xC ][ yC ] is equal to MODE\_INTER and merge\_flag[ xP ][ yP ] is equal to 1, the derivation process for luma motion vectors for merge mode as specified in subclause is invoked with the luma location ( xC, yC ), luma location ( xP, yP ), variables nCS, nPbW and nPbH and the partition index partIdx as inputs and the outputs being the luma motion vectors mvL0 and mvL1, the reference indices refIdxL0 and refIdxL1, the prediction utilization flags predFlagL0 and predFlagL1, the disparity vector availability flags ivpMvFlagL0 and ivpMvFlagL1, the disparity vectors ivpMvDispL0 and ivpMvDispL1, the flag vspModeFlag, the variable refViewIdx, the flag availableFlagIvMC, and the flag cbfAllZero. .
* Otherwise, for X being replaced by either 0 or 1 in the variables predFlagLX, mvLX, refIdxLX and 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[ xP ][ yP ] is equal to Pred\_LX or Pred\_BI,

refIdxLX = ref\_idx\_lX[ xP ][ yP ] (‑71)  
predFlagLX = 1 (‑72)

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

refIdxLX = −1 (‑73)  
predFlagLX = 0 (‑74)

* + 1. The variable mvdLX is derived as follows.
       - 1. mvdLX[ 0 ] = MvdLX[ xP ][ yP ][ 0 ] (‑75)  
            mvdLX[ 1 ] = MvdLX[ xP ][ yP ][ 1 ] (‑76)
    2. When predFlagLX is equal to 1, the derivation process for luma motion vector prediction in subclause  is invoked with the luma coding block location ( xC, yC ), the coding block size nCS, the luma prediction block location ( xP, yP ), variables nPbW and nPbH, refIdxLX, and the partition index partIdx as the inputs and the output being mvpLX.
    3. When predFlagLX is equal to 1, the luma motion vector mvLX is derived as
       - 1. uLX[ 0 ] = (mvpLX[ 0 ] + mvdLX[ 0 ] + 216) % 216 (‑77)  
            mvLX[ 0 ] = ( uLX[ 0 ] >= 215 ) ? ( uLX[ 0 ] − 216 ) : uLX[ 0 ] (‑78)  
            uLX[ 1 ] = (mvpLX[ 1 ] + mvdLX[ 1 ] + 216) % 216 (‑79)  
            mvLX[ 1 ] = ( uLX[ 1 ] >= 215 ) ? ( uLX[ 1 ] − 216 ) : uLX[ 1 ] (‑80)
         2. NOTE – The resulting values of mvLX[ 0 ] and mvLX[ 1 ] as specified above will always be in the range of −215 to 215 − 1.

When ChromaArrayType is not equal to 0 and predFlagLX (with X being either 0 or 1) is equal to 1, the derivation process for chroma motion vectors in subclause 8.5.2.1.8 is invoked with mvLX and refIdxLX as inputs and the output being mvCLX.

If iv\_res\_pred\_flag[ nuh\_layer\_id ] is equal to 1, availableFlagIvMC is equal to 1, PartMode[ xP ][ yP ] is equal to PART\_2Nx2N, and cbfAllZero is equal to 0, ResPredFlag[ xP ][ yP ] is set equal to 1, otherwise ( iv\_res\_pred\_flag[ nuh\_layer\_id ] is equal to 0 or availableFlagIvMC is equal to 0 or PartMode[ xP ][ yP ] is not equal to PART\_2Nx2N or cbfAllZero is equal to 1 ), ResPredFlag[ xP ][ yP ] is set equal to 0.

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

* 1. IvpMvFlagLX[ x ][ y ] = ivpMvFlagLX (‑81)  
     IvpMvDispLX[ x ][ y ] = ivpMvDispLX (‑82)  
     VspModeFlag[ x ][ y ] = vspModeFlag (‑83)  
     RefViewIdx[ x ][ y ] = refViewIdx (‑84)

[Ed.(GT): The complete derivation process for inter-view motion vectors candidates has the problem (in software as well as in this draft), that it is assumed that disparity vectors are related to the base view. However, in the draft and the software there are some loops looking for an arbitrary non-base view inter-view reference picture for inter-view motion vector prediction in the reference picture lists. This works for CTC, since other inter-view reference pictures apart from the base view are not allowed. However, to enable a more general approach, this should be discussed and possibly made more general in SW and text. (3D-I0007) ]

[Ed.(CY): To fix the above mentioned problem, the derived disparity vector should be associated with a reference view index.]

* + - * 1. Derivation process for luma motion vectors for merge mode

This process is only invoked when PredMode[ xC ][ yC ] is equal to MODE\_SKIP or PredMode[ xC ][ yC ] is equal to MODE\_INTER and merge\_flag [ xP ][ yP ] is equal to 1, where ( xP, yP ) specify the top-left sample of the current luma prediction block relative to the top-left luma sample of the current picture.

Inputs of 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 luma location ( xP, yP ) of the top-left sample of the current luma prediction block relative to the top-left luma sample of the current picture,
* a variable nCS specifying the size of the current luma coding block,
* variables specifying the width and the height of the luma prediction block, nPbW and nPbH,
* 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 reference indices refIdxL0 and refIdxL1,
* the prediction list utilization flags predFlagL0 and predFlagL1,
* the disparity vector availability flags ivpMvFlagL0 and ivpMvFlagL1,
* the disparity vectors ivpMvDispL0 and ivpMvDispL1.
* the flag vspModeFlag, specifying, whether the current PU is coded using view synthesis prediction,
* the variable refViewIdx specifying a reference view for VSP,
* the flag availableFlagIvMC, specifying whether the IvMc candidate is available.
* the flag cbfAllZero specifying whether the reference TU contains one or more transform coefficient levels not equal to 0.

The variables singleMCLFlag is derived as follows.

* + If log2\_parallel\_merge\_level\_minus2 is greater than 0 and nCS is equal to 8, singleMCLFlag is set to 1.
  + Otherwise, singleMCLFlag is set to 0.

When singleMCLFlag is equal to 1, xP is set equal to xC, yP is set equal to yC, and both nPbW and nPbH are set equal to nCS.

NOTE – When singleMCLFlag is equal to 1, all the prediction units of the current coding unit share a single merge candidate list, which is identical to the merge candidate list of the 2Nx2N prediction unit.

The motion vectors mvL0 and mvL1, the reference indices refIdxL0 and refIdxL1, ~~and~~ the prediction utilization flags predFlagL0 and predFlagL1, the disparity vector availability flags ivpMvFlagL0 and ivpMvFlagL1, and the disparity vectors ivpMvDispL0 and ivpMvDispL1 are derived as specified by the following ordered steps:

* 1. The derivation process for spatial merge ~~ing~~ candidates ~~from neighboring prediction unit partitions~~ as specified in subclause G.8.5.2.1.2 is invoked with the luma coding block location ( xC, yC ), the coding block size nCS, the luma prediction block location ( xP, yP ), the variable singleMCLFlag, the width and the height of the luma prediction block nPbW and nPbH and the partition index partIdx as inputs and the output is assigned to the availability flags availableFlagN, the reference indices refIdxL0N and refIdxL1N, the prediction list utilization flags predFlagL0N and predFlagL1N and the motion vectors mvL0N and mvL1N with N being replaced by A0, A1, B0, B1 or B2.
  2. The reference index for temporal merging candidate refIdxLX (with X being 0 or 1) is set equal to 0.
  3. The derivation process for temporal luma motion vector prediction in subclause is invoked with luma location ( xP, yP ), the width and the height of the luma prediction block nPbW and nPbH, refIdxLX, and mergeFlag being equal to 1 as the inputs and with the output being the availability flag availableFlagLXCol, the temporal motion vector mvLXCol, and the reference index refIdxLXCol. The variables availableFlagCol and predFlagLXCol (with X being 0 or 1, respectively) are derived as specified below.
     + 1. availableFlagCol = availableFlagL0Col | | availableFlagL1Col (‑85)   
          predFlagLXCol = availableFlagLXCol (‑86)
  4. Depending on iv\_mv\_pred\_flag[ nuh\_layer\_id ], the following applies.
     + If multi\_view\_mv\_pred\_flag is equal to 0, the flags availableFlagIvMC and availableFlagIvDC are set equal to 0.
     + Otherwise (iv\_mv\_pred\_flag[ nuh\_layer\_id ] is equal to 1), the derivation process for the inter-view merge candidates as specified in subclause is invoked with the luma locations ( xC, yC ) and ( xP, yP ), the coding block size nCS, the variables nPSW and nPSH, and the partition index partIdx, as the inputs and the output is assigned to the view order index of the reference view refViewIdx, the availability flags availableFlagIvMC and availableFlagIvDC, the reference indices refIdxLXIvMC and refIdxLXIvDC, the prediction list utilization flags predFlagLXIvMC and predFlagLXIvDC, and the motion vectors mvLXIvMC and mvLXIvDC (with X being 0 or 1, respectively), and the disparity vector mvDisp, and the flag cbfAllZero.
  5. Depending on view\_synthesis\_pred\_flag, the following applies.
     1. [ Ed. (GT): The variable view\_synthesis\_pred\_flag is missing in software and HLS. ]
     + If view\_synthesis\_pred\_flag is equal to 0, the flag availableFlagVSP is set equal to 0 and the variable refViewIdx is set equal to −1.
     + Otherwise (view\_synthesis\_pred\_flag is equal to 1), the derivation process for a view synthesis prediction merge candidate as specified in subclause is invoked with the luma locations ( xC, yC ) and ( xP, yP ), the coding block size nCS, the variables nPSW and nPSH, and the partition index partIdx as input and the outputs are the availability flag availableFlagVSP, the reference indices refIdxL0VSP and refIdxL1VSP, the prediction list utilization flags predFlagL0VSP and predFlagL1VSP, the motion vectors mvL0VSP and mvL1VSP, and reference view index refViewIdx.
  6. Depending on DepthFlag, the following applies.
     + If DepthFlag is equal to 0, the variable availableFlagT is set equal to 0.
     + Otherwise ( DepthFlag is equal to 1), the derivation process for the texture merging candidate as specified in subclause is invoked with the luma location ( xP, yP ), the variables nPSW and nPSH as the inputs and the outputs are the flag availableFlagT, the prediction utilization flags predFlagL0T and predFlagL1T, the reference indices refIdxL0T and refIdxL1T, and the motion vectors mvL0T and mvL1T.
  7. The merge candidate list, mergeCandList, is constructed as specified by the following ordered steps.
  8. The variable numMergeCand is set equal to 0.
  9. The variable vspCandInListFlag is set equal to 0.
  10. When availableFlagT is equal to 1, the insertion process for a candidate from a potentially view synthesis predicted neighbour as specified in subclause is invoked with the list mergeCandList, the variable numMergeCand, the variable vspCandInListFlag, the candidate position indicator N being equal to T, the luma location ( xP, yP ), and the variables nPbW and nPbH as the inputs and the outputs are the list mergeCandList, the variable numMergeCand and the flag vspCandInListFlag.
  11. When availableFlagIvMC is equal to 1, the entry mergeCandList[ numMergeCand ] is set equal to IvMC and the variable numMergeCand is increased by 1.
  12. When availableFlagA1 is equal to 1, the following applies:
      + - When the following condition is true,
          * availableFlagT = = 0 && availableFlagIvMC  = = 0,
          1. or one or more of the following conditions are true, with N being replaced by T and IvMC:
          * availableFlagN = = 1 && predFlagLXN ! = predFlagLXA1, (with X being replaced by 0 and 1),
          * availableFlagN = = 1 && mvLXN ! = mvLXA1 (with X being replaced by 0 and 1),
          * availableFlagN = = 1 && refIdxLXN ! = refIdxLXA1 (with X being replaced by 0 and 1),

the insertion process for a candidate from a potentially view synthesis predicted neighbour block as specified in subclause is invoked with the list mergeCandList, the variable numMergeCand, the variable vspCandInListFlag, the candidate position indicator N being equal to A1, the luma location ( xP, yP ), and the variables nPbW and nPbH as the inputs and the outputs are the list mergeCandList, the variable numMergeCand and the flag vspCandInListFlag.

* 1. When availableFlagB1 is equal to 1, the following applies:
     + - When the following condition is true ,
         * availableFlagT = = 0 && availableFlagIvMC = = 0,

or one or more of the following conditions is true, with N being replaced by T and IvMC:

* + - * + availableFlagN = = 1 && predFlagLXN ! = predFlagLXB1, (with X being replaced by 0 and 1),
        + availableFlagN = = 1 && mvLXN ! = mvLXB1 (with X being replaced by 0 and 1),
        + availableFlagN = = 1 && refIdxLXN ! = refIdxLXB1 (with X being replaced by 0 and 1),

the insertion process for a candidate from a potentially view synthesis predicted neighbour block as specified in subclause is invoked with the list mergeCandList, the variable numMergeCand, the variable vspCandInListFlag, the candidate position indicator N being equal to B1, the luma location ( xP, yP ), and the variables nPbW and nPbH as the inputs and the outputs are the list mergeCandList, the variable numMergeCand and the flag vspCandInListFlag.

* 1. When availableFlagB0 is equal to 1, the insertion process for a candidate from a potentially view synthesis predicted neighbour block as specified in subclause is invoked with the list mergeCandList, the variable numMergeCand, the variable vspCandInListFlag, the candidate position indicator N being equal to B0, the luma location ( xP, yP ), and the variables nPbW and nPbH as the inputs and the outputs are the list mergeCandList, the variable numMergeCand and the flag vspCandInListFlag.
  2. When availableFlagIvDC is equal to 1, and one or more of the following conditions is true,
     + - availableFlagA1 = = 0,
       - predFlagLXA1 ! = predFlagLXIvDC, (with X being replaced by 0 and 1),
       - mvLXA1 ! = mvLXIvDC(with X being replaced by 0 and 1),
       - refIdxLXA1 ! = refIdxLXIvDC(with X being replaced by 0 and 1),

and one or more of the following conditions is true,

* + - * availableFlagB1 = = 0,
      * predFlagLXB1 ! = predFlagLXIvDC, (with X being replaced by 0 and 1),
      * mvLXB1 ! = mvLXIvDC(with X being replaced by 0 and 1),
      * refIdxLXB1 ! = refIdxLXIvDC(with X being replaced by 0 and 1),

the entry mergeCandList[ numMergeCand ] is set equal to IvDC and the variable numMergeCand is increased by 1.

* 1. When availableFlagA0 is equal to 1, the insertion process for a candidate from a potentially view synthesis predicted neighbour block as specified in subclause is invoked with the list mergeCandList, the variable numMergeCand, the variable vspCandInListFlag, the candidate position indicator N being equal to A0, the luma location ( xP, yP ), and the variables nPbW and nPbH as the inputs and the outputs are the list mergeCandList, the variable numMergeCand and the flag vspCandInListFlag.
  2. When availableFlagB2 is equal to 1 and numMergeCand is less than 4 + iv\_mv\_pred\_flag[ nuh\_layer\_id ] + DepthFlag, the insertion process for a candidate from a potentially view synthesis predicted neighbour block as specified in subclause is invoked with the list mergeCandList, the variable numMergeCand, the variable vspCandInListFlag, the candidate position indicator N being equal to B2, the luma location ( xP, yP ), and the variables nPbW and nPbH as the inputs and the outputs are the list mergeCandList, the variable numMergeCand and the flag vspCandInListFlag.
  3. When availableFlagVSP is equal to 1, numMergeCand is less than 4 + iv\_mv\_pred\_flag[ nuh\_layer\_id ] + DepthFlag and vspCandInListFlag is 0, vspCandInListFlag is set equal to 1, the entry mergeCandList[ numMergeCand ] is set equal to VSP and the variable numMergeCand is increased by 1.
  4. When availableFlagCol is equal to 1 and numMergeCand is less than 5 + iv\_mv\_pred\_flag[ nuh\_layer\_id ] + DepthFlag, the entry mergeCandList[ numMergeCand ] is set equal to Col and the variable numMergeCand is increased by 1.

1. ~~A~~~~1~~~~, if availableFlagA~~~~1~~ ~~is equal to 1~~
2. ~~B~~~~1~~~~, if availableFlagB~~~~1~~ ~~is equal to 1~~
3. ~~B~~~~0~~~~, if availableFlagB~~~~0~~ ~~is equal to 1~~
4. ~~A~~~~0~~~~, if availableFlagA~~~~0~~ ~~is equal to 1~~
5. ~~B~~~~2~~~~, if availableFlagB~~~~2~~ ~~is equal to 1~~
6. ~~Col, if availableFlagCol is equal to 1~~
   1. The variable ~~numMergeCand and~~ numOrigMergeCand ~~are~~ is set equal to numMergeCand ~~to the number of merging candidates in the mergeCandList~~.
   2. When slice\_type is equal to B, the derivation process for combined bi-predictive merging candidates specified in subclause 8.5.2.1.3 is invoked with mergeCandList, the reference indices refIdxL0N and refIdxL1N, the prediction list utilization flags predFlagL0N and predFlagL1N, the motion vectors mvL0N and mvL1N of every candidate N being in mergeCandList, numMergeCand and numOrigMergeCand given as input and the output is assigned to mergeCandList, numMergeCand, the reference indices refIdxL0combCandk and refIdxL1combCandk, the prediction list utilization flags predFlagL0combCandk and predFlagL1combCandk and the motion vectors mvL0combCandk and mvL1combCandk of every new candidate combCandk being added in mergeCandList. The number of candidates being added numCombMergeCand is set equal to ( numMergeCand – numOrigMergeCand ). When numCombMergeCand is greater than 0, k ranges from 0 to numCombMergeCand − 1, inclusive.
   3. The derivation process for zero motion vector merging candidates specified in subclause 8.5.2.1.4 is invoked with the mergeCandList, the reference indices refIdxL0N and refIdxL1N, the prediction list utilization flags predFlagL0N and predFlagL1N, the motion vectors mvL0N and mvL1N of every candidate N being in mergeCandList and the NumMergeCand as the inputs and the output is assigned to mergeCandList, numMergeCand, the reference indices refIdxL0zeroCandm and refIdxL1zeroCandm, the prediction list utilization flags predFlagL0zeroCandm and predFlagL1zeroCandm, the motion vectors mvL0zeroCandm and mvL1zeroCandm of every new candidate zeroCandm being added in mergeCandList. The number of candidates being added numZeroMergeCand is set equal to ( numMergeCand – numOrigMergeCand – numCombMergeCand ). When numZeroMergeCand is greater than 0, m ranges from 0 to numZeroMergeCand − 1, inclusive.
   4. The following assignments are made with N being the candidate at position merge\_idx[ xP][ yP ] in the merging candidate list mergeCandList ( N = mergeCandList[ merge\_idx[ xP][ yP ] ] ) and X being replaced by 0 or 1:
      * 1. mvLX[ 0 ] = mvLXN[ 0 ] (‑87)
        2. mvLX[ 1 ] = mvLXN[ 1 ] (‑88)
        3. refIdxLX = refIdxLXN (‑89)
        4. predFlagLX = predFlagLXN (‑90)
   5. When predFlagL0 is equal to 1 and predFlagL1 is equal to 1, and ( nPbW + nPbH ) is equal to 12, the following applies.
      * 1. refIdxL1 = −1 (‑91)
        2. predFlagL1 = 0 (‑92)
   6. The variable vspModeFlag is set equal to ( mergeCandList[ merge\_idx[ xP][ yP ] ] = = VSP ).

[Ed. (GT): In software VspModeFlag[ x ][ y ] is used in several places (deblocking filter, AMVP) this is not yet reflected in the draft.]

* 1. The disparity availability flag ivpMvFlagLX and the disparity vector ivpMvDispLX are derived as follows (with X being replace by 0 or 1).
     + If all of the following conditions are true, ivpMvFlagLX is set equal to 1 and ivpMvDispLX is set equal to mvDisp [Ed. (GT) There is some redundancy in draft and software since ivpMvDispLX is derived for each list, although it is always equal for both lists.(3D-I0008) ]
       - availableFlagIvMC = = 1
       - merge\_idx[ xP][ yP ] = = 0
       - predFlagLXIvMC = = 1

[Ed. (GT): PredMode[ xC ][ yC ] = = MODE\_SKIP might be added here instead of testing it in the disparity vector derivation process]

* + - Otherwise, ivpMvFlagLX is set equal to 0 and both components of ivpMvDispLX are set equal to 0.

Derivation process for the inter-view motion vector predictor 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 ( 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 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,
* a variable nCS specifying the size of the current luma coding block,
* variables nPSW and nPSH specifying the width and the height, respectively, of the current prediction unit,
* a variable partIdx specifying the index of the current prediction unit within the current coding unit.
* a reference index refIdxLX (with X being equal to 0 or 1) specifying a reference picture in the reference picture list RefPicListLX.

Outputs of this process are:

* a view order index refViewIdx specifying a reference view,
* a flag availableFlagLXInterView specifying whether the inter-view motion vector candidate is available,
* a motion vector candidate mvLXInterView (if availableFlagLXInterView is equal to 1).

The derivation process for a disparity vector as specified in subclause is invoked with the luma locations ( xC, yC ) and ( xP, yP ), the coding block size nCS, the variables nPSW and nPSH, the partition index partIdx and the variable deriveFromDepthFlag being equal to 1, as the inputs and the view order index refViewIdx, a flag availableDV and a disparity vector mvDisp as the outputs. The variable refPicViewIdx is set equal to the variable ViewIdx of the RefPicListLX[ refIdxLX ].

[Ed. (GT): refViewIdx might be used here to further restrict the reference views]

Depending on the value of refPicViewIdx, the following applies:

* If refPicViewIdx is not equal to ViewIdx, the derivation process for a disparity inter-view motion vector candidate as specified in subclause is invoked with the luma location ( xP, yP ), the variables nPSW and nPSH, the prediction list indication being equal to X, the view identifier refViewIdx, the disparity vector mvDisp, the flag mergeFlag being equal to 0, and the reference index refIdxLX, as the inputs and the outputs are the flag availableFlagLXInterView, the motion vector candidate mvLXInterView, and the reference index refIdxLX.
* Otherwise (refPicViewIdx is equal to ViewIdx), the derivation process for a temporal inter-view motion vector candidate as specified in subclause is invoked with the luma location ( xP, yP ), the variables nPSW and nPSH, the prediction list indication being equal to X, the reference view identifier refViewIdx, the disparity vector mvDisp, the flag mergeFlag being equal to 0, and the reference index refIdxLX, as the inputs and the outputs are the flag availableFlagLXInterView, the motion vector candidate mvLXInterView, and the flag cbfAllZero.
  + - * 2. Derivation process for inter-view merge candidates

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 ( 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 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,
* a variable nCS specifying the size of the current luma coding block,
* variables nPSW and nPSH specifying the width and the height, respectively, of the current prediction unit,
* a variable partIdx specifying the index of the current prediction unit within the current coding unit.

Outputs of this process are (with X being 0 or 1, respectively)

* a view order index refViewIdx specifying a reference view.
* the availability flags availableFlagIvMC and availableFlagIvDC specifying whether the inter-view merge candidates are available,
* the reference indices refIdxLXIvMC and refIdxLXIvDC,
* the prediction list utilization flags predFlagLXIvMC and predFlagLXIvDC,
* the motion vectors mvLXIvMC and mvLXIvDC,
* the disparity vector mvDisp.
* the flag cbfAllZero specifying whether the reference TU contains one or more transform coefficient levels not equal to 0.

The derivation process for a disparity vector as specified in subclause H.8.5.4 is invoked with the luma locations ( xC, yC ) and ( xP, yP ), the coding block size nCS, the variables nPSW and nPSH, the partition index partIdx, the variable deriveFromDepthFlag being equal to 1, as the inputs and the outputs are and the view order index of the reference view refViewIdx, the flag availableDV and the disparity vector mvDisp.

The temporal inter-view motion vector merging candidate is derived by the following ordered steps.

* 1. For the prediction list indication X being 0 and 1 the following applies.
     + The derivation process for a temporal inter-view motion vector candidate as specified in subclause H.8.5.2.1.10 is invoked with the luma location ( xP, yP ), the variables nPSW and nPSH, the prediction list indication X , the view identifier refViewIdx, the disparity vector mvDisp, the flag mergeFlag being equal to 1, and the reference index refIdxLX being equal to −1, and as the inputs and the outputs are the flag availableFlagLXIvMC, the motion vector mvLXIvMC, the reference index refIdxLXIvMC, and the flag cbfAllZero.
  2. The availability flag availableFlagIvMC, and the prediction utilization flags predFlagL0IvMC and predFlagL1IvMC are derived by
     + 1. availableFlagIvMC = availableFlagL0IvMC | | availableFlagL1IvMC (H‑)  
          predFlagL0IvMC = availableFlagL0IvMC (H‑)  
          predFlagL1IvMC = availableFlagL1IvMC (H‑)

The disparity inter-view motion vector merging candidate is derived by the following ordered steps.

* 1. For the prediction list indication X being 0 and 1 the following applies.
     + The derivation process for the disparity inter-view motion vector candidate as specified in subclause H.8.5.2.1.11 is invoked with the luma location ( xP, yP ), the variables nPSW and nPSH, the view identifier refViewIdx, the disparity vector mvDisp, the prediction list indication X, the flag mergeFlag being equal to 1, and the reference index refIdxLX being equal to −1, as the inputs and the outputs are the flag availableFlagLXIvDC, the motion vector mvLXIvDC, and the reference index refIdxLXIvDC.
  2. The availability flag availableFlagIvDC, and the prediction utilization flags predFlagL0IvDC and predFlagL1IvDC are derived by
     + 1. availableFlagIvDC = availableFlagL0IvDC | | availableFlagL1IvDC  (H‑)  
          predFlagL0IvDC = availableFlagL0IvDC (H‑)  
          predFlagL1IvDC = availableFlagL1IvDC (H‑)
          1. Derivation process for a temporal 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 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 temporal inter-view motion vector candidate is available,
* a temporal 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,
* a flag cbfAllZero specifying whether the reference TU contains one or more transform coefficient levels not equal to 0.

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

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

The reference layer luma location ( xRef, yRef ) is derived by

* 1. xRef = Clip3( 0, PicWidthInSamplesL – 1, xP + ( ( nPSW – 1 ) >> 1 ) + ( ( mvDisp[ 0 ] + 2 ) >> 2 ) ) (‑161)  
     yRef = Clip3( 0, PicHeightInSamplesL – 1, yP + ( ( nPSH – 1 ) >> 1 ) + ( ( mvDisp[ 1 ] + 2 ) >> 2 )) (‑162)

Let refCU be the coding unit that covers the luma location ( xRef, yRef ) in the view component with ViewIdx equal to refViewIdx.

Let refTU be the transform unit that covers the luma location (xRef, yRef) in the view component with ViewIdx equal to refViewIdx.

The flag cbfAllZero is set to 0.

[Ed. (GT:) In software refCU is the coding unit that covers the luma location ( xRef, yRef ) in the view component that closest (in terms of ViewIdx) to the current view component and that is included in one of the reference lists of the current view component.(3D-I0010)]

[Ed. (CY): What is implemented in the software has no impact on the final results in common test condition, so it is preferred to have the current text and software changes may be desirable. ]

When the variable PredMode for the coding unit refCU is equal to MODE\_SKIP or MODE\_INTER, the following ordered steps apply, for Y in the range of X to (1 – X), inclusive:

* 1. The variable refPredFlagLY is set equal to the prediction utilization flag predFlagLY of the prediction unit refPU.
  2. The variable refRefIdxLY, is set equal to the reference index refIdxLY of the prediction unit refPU.
  3. The variable refMvLY is set equal to the motion vector mvLY of the prediction unit refPU.
  4. The variable refRefPicListLY, is set equal to the reference picture list RefPicListLY of the prediction unit refCU.
  5. When refPredFlagLY is equal to 1, the following applies for each i from ( mergeFlag ? 0 : refIdxLX ) to ( mergeFlag ? num\_ref\_idx\_lX\_active\_minus1 : refIdxLX), inclusive:
     + When availableFlagLXInterView is equal to 0, and the picture order count of the picture refRefPicListLY[ refRefIdxLY ] is equal to the picture order count of the picture RefPicListLX[ i ], the flag availableFlagLXInterView is set equal to 1 and the following applies.
       - The motion vector mvLXInterView is derived by:
         1. mvLXInterView[ 0 ] = refMvLY[ 0 ] (‑)  
            mvLXInterView[ 1 ] = refMvLY[ 1 ] (‑)
       - When mergeFlag is equal to 1, the reference index refIdxLX is derived by:
         1. refIdxLX = i (‑)
       - If the cbf flags for luma, Cb, and Cr component of refTU are all zero, then the cbfAllZero flag is set equal to 1.