The proposed working draft modifications are as follows.

H.7.4.8 Coding quadtree semantics

~~The variable anyTempRefPicFlag specifies whether one or more prediction units of the current coding unit utilize a temporal reference picture. The variable anyTempRefPicFlag is initially set equal to 0. When PredMode[ x0 ][ y0 ] is not equal to MODE\_INTRA the following applies for X being replaced by 0 and 1, and Y being equal to 1−X.~~

* 1. ~~anyTempRefPicFlag = anyTempRefPicFlag | |    
     (inter\_pred\_idc[ x0 ][ y0 ] ! = Pred\_LY && refViewIdxLX[ x0 ][ y0 ] = = ViewIdx)  | |    
     (inter\_pred\_idc[ x0 ][ y1 ] ! = Pred\_LY && refViewIdxLX[ x0 ][ y1 ] = = ViewIdx)  | |   
     (inter\_pred\_idc[ x1 ][ y0 ] ! = Pred\_LY && refViewIdxLX[ x1 ][ y0 ] = = ViewIdx)  | |    
     (inter\_pred\_idc[ x1 ][ y1 ] ! = Pred\_LY && refViewIdxLX[ x1 ][ y1 ] = = ViewIdx)   (‑)~~

~~The variable anyIvRefPicFlag specifies whether one or more prediction units of the current coding unit utilize an inter-view reference picture. The variable anyIvRefPicFlag is initially set equal to 0. When PredMode[ x0 ][ y0 ] is not equal to MODE\_INTRA the following applies for X being replaced by 0 and 1, and Y being equal to 1−X.~~

* ~~If DepthFlag is equal to 0, the following applies:~~ 
  1. ~~anyIvRefPicFlag = anyIvRefPicFlag | |    
     (inter\_pred\_idc[ x0 ][ y0 ] ! = Pred\_LY && refViewIdxLX[ x0 ][ y0 ] ! = ViewIdx)  | |    
     (inter\_pred\_idc[ x0 ][ y1 ] ! = Pred\_LY && refViewIdxLX[ x0 ][ y1 ] ! = ViewIdx)  | |   
     (inter\_pred\_idc[ x1 ][ y0 ] ! = Pred\_LY && refViewIdxLX[ x1 ][ y0 ] ! = ViewIdx)  | |    
     (inter\_pred\_idc[ x1 ][ y1 ] ! = Pred\_LY && refViewIdxLX[ x1 ][ y1 ] ! = ViewIdx)   (‑)~~
* ~~Otherwise ( DepthFlag is not equal to 0 ), the following applies:~~

~~for ( m = 0; m < nCbs; m += ( 1 << Log2MinTrafoSize ) )   
 for ( n = 0; n < nCbs; n += ( 1 << Log2MinTrafoSize ) )   
 anyIvRefPicFlag = anyIvRefPicFlag | |    
 (inter\_pred\_idc[ x0 + m ][ y0+n ] ! = Pred\_LY && refViewIdxLX[ x0+ m ][ y0+ n ] ! = ViewIdx)~~

The variable IsIcFlagSignal and RefAttr is defined as follows.

If ViewIdx of each reference picture from RefPicList0 and RefPicList1 is not equal to in ViewIdx of the current view component, IsIcFlagSignal is set to 1 and RefAttr is set to 1.

Else if  ViewIdx of each reference picture from RefPicList0 and RefPicList1 is equal to in ViewIdx of the current view component, IsIcFlagSignal is set to 0 and RefAttr is set to 2.

Else, if PartMode is equal to PART\_2Nx2N, skip\_flag[ x0 ][ y0 ] is equal to 0, and merge\_flag[ x0 ][ y0 ] is equal to 0, IsIcFlagSignal is set to 1 and RefAttr is set to 3.  
Else, IsIcFlagSignal is set to 0 and RefAttr is set to 3.

**ic\_flag** equal to 1 specifies illumination compensation is used for the current coding unit. ic\_flagequal to 0 specifies illumination compensation is not used for the current coding unit. When not present, ic\_flag is inferred to be equal to 0.

The variable ICFlag is set equal to **ic\_flag**.

The variable icEnableFlag specifying whether ic\_flag is present in the bitstream is derived as

~~icEnableFlag = slice\_ic\_enable\_flag &&~~ ~~anyIvRefPicFlag~~

icEnableFlag = slice\_ic\_enable\_flag && IsIcFlagSignal

H.8.5.2.1 Derivation process for motion vector components and reference indices

* 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 IC utilization flag ICFlag, the disparity vector availability flags ivpMvFlagL0 and ivpMvFlagL1, the disparity vectors ivpMvDispL0 and ivpMvDispL1, the flag vspModeFlag, the variable refViewIdx and the flag availableFlagIvMC. .

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 IC utilization flag ICFlag, the disparity vector availability flags ivpMvFlagL0 and ivpMvFlagL1, the disparity vectors ivpMvDispL0 and ivpMvDispL1, the flag vspModeFlag, the variable refViewIdx and the flag availableFlagIvMC.

H.8.5.2.1.1 Derivation process for luma motion vectors for merge mode

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 IC utilization flag ICFlag,
* 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.

…

11. 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. ICFlag = ICFlagN  (‑91)

H.8.5.2.1.2 Derivation process for spatial merge candidates

The specifications in subclause 8.5.2.1.2 apply, with the following changes:

The following paragraph

* N is equal to B2 and availableFlagA0 + availableFlagA1 + availableFlagB0 + availableFlagB1 is equal to 4.

is removed.

* the IC flags ICFlagA0, ICFlagA1, ICFlagB0, ICFlagB1, ICFlagB2 of the neighbouring prediction units.

is inserted as outputs.

The variable ICFlagS (with S being A0, A1, B0, B1 or B2 ) is derived as

If availableFlagS is set equal to 0, ICFlagS is set to 0.

Otherwise, ICFlagS = ICFlag[ xS ][ yS ].

H.8.5.2.1.3 Derivation process for combined bi-predictive merging candidates

The specifications in subclause 8.5.2.1.3 apply.

* IC flags ICFlagcombCandk of every new candidate combCandk being added in mergeCandList during the invokation of this process,

is inserted as outputs.

ICFlagcombCandk is derived as,

ICFlagcombCandk = ICFlagl0Cand | ICFlagl1Cand.

H 8.5.2.1.7 Derivation process for temporal luma motion vector prediction

Outputs of this process are

* the motion vector prediction mvLXCol,
* the availability flag availableFlagLXCol,
* The IC flag ICFlagCol,
* the reference index refIdxLX.

…

* + Otherwise, the following ordered steps apply.
    1. The motion vector mvCol, the reference index refIdxCol, the IC flag ICFlagCol and the reference list identifier listCol are derived as follows.

ICFlagCol is set equal to ICFlag [ xPCol ][ yPCol ].

H 8.5.2.1.9 Derivation process for inter-view merge candidates

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 IC flag ICFlagIvMC and ICFlagIvDC
* the disparity vector mvDisp.

CFlagIvMC is set to zero.

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, the variable deriveFromDepthFlag being equal to 1, as the inputs and the outputs are the view order index of the reference view refViewIdx, the IC flag ICFlagIvDC, the flag availableDV and the disparity vector mvDisp.

H.8.5.2.2 Decoding process for inter prediction samples

The variable nCSL is set equal to nCS and the variable nCSC is set equal to nCS >> 1.

The variable useICFlag is derived as

UseICFlag = ic\_flag | ( RefAttr == 3 & ICFlag & ( skip\_flag[ xC ][ yC ] | merge\_flag[ xB ][ yB ]))

* If VspModeFlag[ xC + xB ][ yC + yB ] is equal to 0, the following ordered steps apply:
  1. …
  2. …
  3. Depending on ~~ic\_flag~~ useICFlag, the arrays predSamplesL ,predSampleCb, and predSampleCr are derived as specified in the following:
     + If ~~ic\_flag~~ useICFlag is equal to 0, the following applies.

…

* + - Otherwise (~~ic\_flag~~ useICFlag is equal to 1), the following applies.

H.8.5.4 Derivation process for a disparity vector

Outputs of this process are:

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

...

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

refViewIdx = ViewIdx( RefPicListX[ RefIdxLX[ xN ][ yN ] ] ) (H‑274)  
mvDisp = MvLXN[ xN ][ yN ] (H‑275)  
availableDV = 1 (H‑276)

ICFlagDV = ICFlagN[ xN ][ yN ] (‑277)

H.8.5.4 .1 Derivation process for a disparity vector from temporal neighbour blocks

Outputs of this process are

* the disparity vector mvDisp,
* the availability flag availableFlag.
* the IC flag ICFlagDV.

…

For i from 0 to NumDdvCandPics, the following ordered steps apply and the whole decoding process of this sub-clause terminates once availableFlag is set to 1.

1. When availableRb is equal to 1, the derivation process for a disparity vector in a block of a candidate picture as specified in subclause H.8.5.4.2 is invoked with candidate picture DdvCandPicsList[ i ], luma location ( xPRb , yPRb ) as inputs, and the flag availableFlag , the flag ICFlagDV, and the disparity vector mvDisp as outputs.
2. The derivation process for a disparity vector in a block of a candidate picture as specified in subclause H.8.5.4.2 is invoked with candidate picture DdvCandPicsList[ i ], luma location (xPCtr, yPCtr) as inputs, and the flag availableFlag, the flag ICFlagDV, and the disparity vector mvDisp as outputs.

H.8.5.4 .2 Derivation process for a disparity vector in a block of a candidate picture

Outputs of this process are:

– a flag availableDV specifying whether the disparity vector is available,

– a disparity vector mvDisp.

– IC flag ICflagDV.

1. The flag availableDV is set equal to 0, ICflagDV is set equal to 0, and both components of mvDisp are set equal to 0.
2. …
   * + 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 , the following applies:
       - 1. mvDisp = candMV[ xPCol ][ yPCol ] (H-289)  
            availableDV = 1 (H-290)

ICflagDV = ICFlag[ xPCol ][ yPCol ] (H-291)