H.8.5.5 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~~, B~~~~1~~ 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.
     + - 1. xB1 = ( ( xB1  >>  3 )  <<  3 ) + ( ( xB1  >>  3 ) & 1) \* 7 (H‑236)
  2. 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.
  3. 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. ]
  4. 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) ).
  5. The flag availableIvpMvSearchFlagN is set equal to availableN.
  6. ~~When one of the following conditions is true, N is equal to B~~~~1~~ ~~and ( ( yN  >>  Log2CtbSizeY )  <<  Log2CtbSizeY ) is less than ( ( yCb >> Log2CtbSizeY )  <<  Log2CtbSizeY), availableIvpMvSearchFlagN is set equal to 0.~~
  7. The flag availableFlagIvpMvN is set equal to 0.
  8. 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‑)

mvDisp = MvLXN[ xN ][ yN ] (H‑)

availableDV = 1 (H‑)

* + - * 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 PredMode[ xN ][ yN ] is equal to MODE\_SKIP and IvpMvFlagLX[ xN ][ yN ] is equal to 1, the following applies:

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

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

availableFlagIvpMvN = 1 (H‑)

When availableDV is equal to 0 for ~~each~~ N being A1 ~~and B~~~~1~~, the following applies.

* + When ~~availableDV is equal to 0 and~~ availableFlagIvpMvN is equal to 1, the following applies:
    - 1. mvDisp = ivpMvDispN (H‑)
      2. refViewIdx = refViewIdxN (H‑)
      3. availableDV = 1 (H‑)

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

[Ed. (GT): It is currently an issue under non-CTC that a view with refViewIdx equal to 0 might not be available in the reference picture list. In software a preliminary fix has been incorporated, to enable simulations in IBP configuration. ]

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).
  2. 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 ):

* 1. MvDisp[ x ][ y ] = mvDisp (H‑246)
  2. MvRefinedDisp[ x ][ y ] = mvRefinedDisp (H‑247)
  3. RefViewIdx[ x ][ y ] = refViewIdx (H‑248)
  4. DefaultDispFlag[ x ][ y ] = !availableDV (H‑249)