The added part is highlighted as yellow

The removed part is highlighted as purple

H.8.5.3.2.1. Derivation process for luma motion vectors for merge mode

This process is only invoked when merge\_flag[ xPb ][ yPb ] is equal to 1, where ( xPb, yPb ) specify the top-left sample of the current luma prediction block relative to the top-left luma sample of the current picture.

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 ( xPb, yPb ) of the top-left sample of the current luma prediction block relative to the top-left luma sample of the current picture,
* 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 reference indices refIdxL0 and refIdxL1,
* the prediction list utilization flags predFlagL0 and predFlagL1,
* the disparity vector availability flags ivpMvFlagL0 and ivpMvFlagL1,
* the flag vspModeFlag, specifying, whether the current PU is coded using view synthesis prediction,
* the flag subPbMotionFlag, specifying, whether the motion data of the current PU has sub prediction block size motion accuracy.

[Ed. (GT): In particular two things need to be check in this process: 1.) Are the limits on candidates in the list correct ( e.g. MaxNumMergeCand vs.  5 + NumExtraMergeCand  ) 2.) Is ( xOrigP, yOrigP ) and ( xPb, yPb ) used correctly in all places? ]

The function differentMotion( N, M ) is specified as follows:

* If one of the following conditions is true, differentMotion( N, M ) is equal to 1:
  + predFlagLXN != predFlagLXM (with X being replaced by 0 and 1),
  + mvLXN != mvLXM (with X being replaced by 0 and 1),
  + refIdxLXN != refIdxLXM (with X being replaced by 0 and 1),
* Otherwise, differentMotion( N, M ) is equal to 0.

The motion vectors mvL0 and mvL1, the reference indices refIdxL0 and refIdxL1, and the prediction utilization flags predFlagL0 and predFlagL1 are derived by the following ordered steps:

* 1. The derivation process for the base merge candidate list as specified in subclause H.8.5.3.2.18 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 a modified luma location ( xPb, yPb ), the modified variables nPbW and nPbH, the modified variable partIdx, the luma location ( xOrigP, yOrigP ), the variables nOrigPbW and nOrigPbH, the merge candidate list baseMergeCandList, the luma motion vectors mvL0N and mvL1N, the reference indices refIdxL0N and refIdxL1N, and the prediction list utilization flags predFlagL0N and predFlagL1N, with N being replaced by all elements of baseMergeCandList.
  2. For N being replaced by A1, B1, B0, A0 and B2, the following applies:
     + If N is an element in baseMergeCandList, availableFlagN is set equal to 1.
     + Otherwise (N is not an element in baseMergeCandList), availableFlagN is set equal to 0.
  3. Depending on iv\_mv\_pred\_flag[ nuh\_layer\_id ], the following applies:
     + If iv\_mv\_pred\_flag[ nuh\_layer\_id ] is equal to 0, the flags availableFlagIvMC, availableIvMCShift 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 H.8.5.3.2.10 is invoked with the luma location ( xPb, yPb ), the variables nPbW and nPbH, as the inputs and the output is assigned to the availability flags availableFlagIvMC, availableIvMCShift and availableFlagIvDC, the reference indices refIdxLXIvMC, refIdxLXIvMCShift and refIdxLXIvDC, the prediction list utilization flags predFlagLXIvMC, predFlagLXivMCShift and predFlagLXIvDC, and the motion vectors mvLXIvMC, mvLXIvMCShift and mvLXIvDC (with X being 0 or 1, respectively).
  4. Depending on view\_synthesis\_pred\_flag[ nuh\_layer\_id ], the following applies:
     + If view\_synthesis\_pred\_flag[ nuh\_layer\_id ] is equal to 0, the flag availableFlagVSP is set equal to 0.
     + Otherwise (view\_synthesis\_pred\_flag[ nuh\_layer\_id ] is equal to 1), the derivation process for a view synthesis prediction merge candidate as specified in subclause H.8.5.3.2.13 is invoked with the luma locations ( xCb, yCb ) as input and the outputs are the availability flag availableFlagVSP, the reference indices refIdxL0VSP and refIdxL1VSP, the prediction list utilization flags predFlagL0VSP and predFlagL1VSP, and the motion vectors mvL0VSP and mvL1VSP.
  5. Depending on mpi\_flag[ nuh\_layer\_id ], the following applies:
     + If mpi\_flag[ nuh\_layer\_id ] is equal to 0, the variable availableFlagT is set equal to 0.
     + Otherwise (mpi\_flag[ nuh\_layer\_id ] is equal to 1), the derivation process for the texture merging candidate as specified in subclause H.8.5.3.2.14 is invoked with the luma location ( xPb, yPb ), the variables nPbW and nPbH 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.
  6. The merging candidate list, extMergeCandList, is constructed as follows:

i = 0  
 if( availableFlagT )  
 extMergeCandList[ i++ ] = T  
 if( availableFlagIvMC && ( !availableFlagT  | |  differentMotion( T, IvMC ) ) )  
 extMergeCandList[ i++ ] = IvMC  
 N = DepthFlag ? T : IvMC  
 if( availableFlagA1 && ( !availableFlagN  | |  differentMotion( N, A1 ) ) )  
 extMergeCandList[ i++ ] = A1 if( availableFlagB1 && ( !availableFlagN  | |  differentMotion( N, B1 ) ) )  
 extMergeCandList[ i++ ] = B1 if( availableFlagB0 )  
 extMergeCandList[ i++ ] = B0 (H‑91) if( availableFlagIvDC && ( !availableFlagA1  | |  differentMotion( A1, IvDC ) ) &&   
 ( !availableFlagB1  | |  differentMotion( B1, IvDC ) ) )  
 extMergeCandList[ i++ ] = IvDC  
 if( availableFlagVSP && !ic\_flag ~~&&~~ ~~iv\_res\_pred\_weight\_idx = = 0~~ )  
 extMergeCandList[ i++ ] = VSP  
 if( availableFlagA0 )  
 extMergeCandList[ i++ ] = A0 if( availableFlagB2 )  
 extMergeCandList[ i++ ] = B2 if( availableFlagIvMCShift && i < ( 5 + NumExtraMergeCand ) &&  
 ( !availableFlagIvMC  | |  differentMotion( IvMC, IvMCShift ) ) )  
 extMergeCandList[ i++ ] = IvMCShift

* 1. The variable availableFlagIvDCShift is set equal to 0, and when availableFlagIvMCShift is equal to 0, and i is less than ( 5 + NumExtraMergeCand ), the derivation process for the shifted disparity merging candidate as specified in subclause H.8.5.3.2.15 is invoked with the luma location ( xPb, yPb ), the variables nPbW and nPbH, and the availability flags availableFlagN, 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 extMergeCandList, extMergeCandList, and i as the inputs and the outputs are the flag availableFlagIvDCShift, the prediction utilization flags predFlagL0IvDCShift and predFlagL1IvDCShift, the reference indices refIdxL0IvDCShift and refIdxL1IvDCShift, and the motion vectors mvL0IvDCShift and mvL1IvDCShift.
  2. The merging candidate list, extMergeCandList, is constructed as follows:

if( availableFlagIvDCShift )  
 extMergeCandList[ i++ ] = IvDCShift  
 j = 0  
 while( i < MaxNumMergeCand ) {(H‑92)  
 N = baseMergeCandList[ j++ ]  
 if( N != A1 && N != B1 && N != B0 && N != A0 && N != B2 )  
 extMergeCandList[ i++ ] = N  
 }

* 1. The variable N is set equal to extMergeCandList[ merge\_idx[ xOrigP ][ yOrigP ] ].
  2. The variable subPbMotionFlag is set equal to ( N = = IvMC ).
  3. The following assignments are made with X being replaced by 0 or 1:
     + 1. mvLX = subPbMotionFlag ? 0 : mvLXN (H‑93)
       2. refIdxLX = subPbMotionFlag ? −1 : refIdxLXN (H‑94)
       3. predFlagLX = subPbMotionFlag ? 0 : predFlagLXN (H‑95)
  4. When predFlagL0 is equal to 1 and predFlagL1 is equal to 1, and ( nOrigPbW + nOrigPbH ) is equal to 12, the following applies
     + 1. refIdxL1 = −1 (H‑96)
       2. predFlagL1 = 0 (H‑97)
  5. The derivation process for a view synthesis prediction flag as specified in subclause H.8.5.3.2.17 is invoked with the luma location ( xPb, yPb ), the variables nPbW and nPbH, the merge candidate indicator N as the inputs, and the output is the mergeCandIsVspFlag.
  6. The variable vspModeFlag is derived as specified in the following:
     1. vspModeFlag = mergeCandIsVspFlag && !ic\_flag ~~&& ( iv\_res\_pred\_weight\_idx = = 0 )~~ (H‑98)
  7. The disparity availability flag ivpMvFlagLX is derived as follows (with X being replace by 0 or 1).
     + If DepthFlag is equal to 0 and one of the following conditions is true, ivpMvFlagLX is set equal to 1

[Ed. (GT) There is some redundancy in draft and software since disparities equal for both lists.(#7) ]

* + - * predFlagLXIvMC && extMergeCandList[ merge\_idx[ xPb][ yPb ] ]  =  =  IvMC
      * predFlagLXIvMCShift && extMergeCandList[ merge\_idx[ xPb][ yPb ] ]  = =  IvMCShift

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

* + - Otherwise, ivpMvFlagLX is set equal to 0.

H.8.5.3.3..8 View synthesis prediction process

Inputs to this process are:

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

* 1. xP = xCb + xBl (‑233)
  2. yP = yCb + yBl (‑234)

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 refIdxLX as input.

The variable refViewIdx is set equal to the ViewIdx( RefPicListX[ refIdxLX ] ) and the variable depthViewIdx is set equal to RefViewIdx[ xCb + xBl ][ yCb + yBl ]. The variable mvDisp is set equal to MvDisp[ xCb + xBl ][ yCb + yBl ]

The derivation process for a disparity sample array as specified in section is invoked with the luma location ( xP, yP ), the disparity vector mvDisp, the variable refViewIdx, the variable depthViewIdx, the variable nPSW, the variable nPSH, and the variable splitFlag equal to 1 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:

* set the motion vector dv equal to (disparitySamples[ xL ][ yL ], 0)
* The derivation process for a motion vector from a reference block for residual prediction as specified in subclause is invoked with ( xP, yP ), nPbW and nPbH, refPicL, and the motion vector dv, as inputs, and availFlag, motion vector mvT and prediction list utilization variable Y as outputs.
* When availFlag is equal to 0 and RpRefIdxLX is not equal to −1, availFlag is set equal to 1, mvT is set equal to (0, 0), Y is set equal to X.
* The variables xIntL, yIntL, xFracL, and yFracL are derived by
  1. xIntL = xP + xL+ disparitySamples[ xL ][ yL ] (‑235)
  2. yIntL = yP + yL (‑236)
  3. xFracL = disparitySamples[ xL ][ yL ] & 3 (‑237)
  4. yFracL = 0 (‑238)
* The prediction luma sample value predSamplesL[ xL][ yL ] is derived by invoking the process specified in subclause 8.5.3.3.3.2 with ( xIntL, yIntL ), ( xFracL, yFracL ) and refPicL given as input.

[Ed. (GT): As for inter prediction the treatment of colour planes for depth needs to be discussed. In software colour planes are set to 128 in VSP process. (#12)]

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
  1. xIntC = ( xP / 2 ) + xC + disparitySamples[ xC  <<  1 ][ yC  <<  1 ] (‑239)
  2. yIntC = ( yP / 2 ) + yC (‑240)
  3. xFracC = disparitySamples[ xC  <<  1][ yC   <<  1 ] & 7 (‑241)
  4. yFracC = 0 (‑242)
* The prediction sample value predSamplesCb[ xC ][ yC ] is derived by invoking the process specified in subclause 8.5.3.3.3.3 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 8.5.3.3.3.3 with ( xIntC, yIntC ), ( xFracC, yFracC ) and refPicCr given as input.
* When availFlag is equal to 1 and iv\_res\_pred\_weight\_idx is not equal to 0, the following applies:
  + Let rpPic be the picture RefPicListY[ RpRefIdxLY ].
  + Let rpRefPic be the picture with PicOrderCnt( rpRefPic ) equal to PicOrderCnt( rpPic ) and ViewIdx equal to refViewIdx
  + The variable mvRp is set equal to mvT.
  + The variable currRefIdx is set equal to RpRefIdxLY.
  + 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 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 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 ( dv + mvRp ) and the motion vector mvCLX equal to ( dv + 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:
  1. predSamplesLXL[ x ][ y ] = predSamplesLXL[ x ][ y ] +   
      ( ( rpSamplesLXL[ x ][ y ] − rpRefSamplesLXL[ x ][ y ] )  >>  shiftVal ) (‑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:
  1. predSamplesLXCb[ x ][ y ] = predSamplesLXCb[ x ][ y ] +   
      ( ( rpSamplesLXCb[ x ][ y ] − rpRefSamplesLXCb[ x ][ y ] )  >>  shiftVal ) (‑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:
  1. predSamplesLXCr[ x ][ y ] = predSamplesLXCr[ x ][ y ] +   
      ( ( rpSamplesLXCr[ x ][ y ] − rpRefamplesLXCr[ x ][ y ] )  >>  shiftVal ) (‑216)