I 8.5.3.2 Derivation process for motion vector components and reference indices

…

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

IvpMvFlag[ x ][ y ] = ivpMvFlag   
VspModeFlag[ x ][ y ] = vspModeFlag   
DispDerivedDepthFlag[ x ][ y ]  = dispDerivedDepthFlag   
DispDerivedDepthVal[ x ][ y ] = dispDerivedDepthVal

DepthCompensationCalibrationFlag [ x ][ y ]  = depthCompensationCalibrationFlag

…

I 8.5.3.2.1 Derivation process for luma motion vectors for merge mode

…

The variables vspModeFlag, ivpMvFlag, subPbMotionFlag, dispDerivedDepthFlag, ~~and~~ dispDerivedDepthVal and depthCompensationCalibrationFlag are set equal to 0.

…

1. Depending on mpi\_flag[ nuh\_layer\_id ], the following applies:

…

* + - * The derivation process for the disparity derived merging candidates as specified in subclause  is invoked with the luma location ( xPb, yPb ), the variables nPbW and nPbH as inputs, and the outputs are the flag availableFlagD, the prediction utilization flag predFlagLXD, the reference index refIdxLXD, the motion vector mvLXD (with X being 0 or 1, respectively), and the variable dispDerivedDepthVal.
      * The derivation process for the depth compensation calibration merging candidates as specified in subclause  is invoked with the luma location ( xPb, yPb ), the variables nPbW and nPbH as inputs, and the outputs are the flag availableFlagDCC, the prediction utilization flag predFlagLXD, the reference index refIdxLXD, and the motion vector mvLXD (with X being 0 or 1, respectively).

1. The merging candidate list, extMergeCandList, is constructed as follows:

i = 0

if( availableFlagT )

extMergeCandList[ i++ ] = T

if( availableFlagDCC )

extMergeCandList[ i++ ] = DCC

if( availableFlagD )

extMergeCandList[ i++ ] = D

…

1. The variable dispDerivedDepthFlag is derived as follows:
   * + 1. dispDerivedDepthFlag = ( N  = =  D )
2. The variable depthCompensationCalibrationFlag is derived as follows:
   * + 1. depthCompensationCalibrationFlag = ( N  = =  DCC )
          1. Derivation process for the depth compensation calibration merging candidate

Inputs to this process are:

* 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,
* two variables nPbW and nPbH specifying the width and the height of the current prediction block,

Outputs of this process are (with X being replaced by 0 and 1):

* the flag availableFlagDCC specifying whether the depth compensation calibration merging candidate is available,
* the prediction utilization flag predFlagLXDCC,
* the reference index refIdxLXDCC,
* the motion vector mvLXDCC,

Let textPic1 be the picture with PicOrderCntVal and ViewIdx equal to PicOrderCntVal and ViewIdx of the current picture, and DepthFlag being equal to 0 and textPic0 be the picture with PicOrderCntVal and ViewIdx equal to PicOrderCntVal of the current picture and RefViewIdx[ xPb ][ yPb  ], and DepthFlag being equal to 0.

If textPic1 and textPic0 are both available, the variable availableFlagDCC is set equal to 1. Otherwise (textPic1 and textPic0 are not both available), availableFlagDCC is set equal to 0.

If PartMode is not equal to PART\_2Nx2N, availableFlagDCC is set equal to 0.

When availableFlagDCC is equal to 0, the whole process terminates.

The derivation process for a texture merging candidate as specified in subclause  is invoked with the luma location ( xPb, yPb ), the variables nPbW and nPbH, and the variable mvAccFlag being equal to 1 as inputs, and the outputs are the flags availableFlagT, predFlagLXT, the reference index refIdxLXT, and the motion vector mvLXT (with X being replaced by 0 and 1).

If availableFlagT is equal to 1, for X in the range of 0 to 1, inclusive, when X is equal to 0 or the current slice is a B slice, the following applies:

* + 1. mvLXDCC = mvLXT
    2. refIdxLXDCC = refIdxLXT
    3. predFlagLXDCC = predFlagLXT

Otherwise (availableFlagT is equal to 0), for X in the range of 0 to 1, inclusive, when X is equal to 0 or the current slice is a B slice, the following applies:

* + 1. mvLXDCC = mvLXbaseMergeCandList[ 0 ]
    2. refIdxLXDCC = refIdxLX baseMergeCandList[ 0 ]

predFlagLXDCC = predFlagLX baseMergeCandList[ 0 ]

I.8.5.3 Decoding process for prediction units in inter prediction mode

I.8.5.3.1 General

…

2. Depending on subPbMotionFlag, and dbbp\_flag[ xCb ][ yCb ] the following applies:

…

3. When dispDerivedDepthVal [ xCb][ yCb ] is equal to 1, the decoding process for depth compensation calibration as specified in subclause I.8.5.3.3.10 is invoked with the luma coding block location ( xCb, yCb ), the luma coding block size block nCbS and the inter prediction samples (predSamples) that are an (nCbSL)x(nCbSL) array predSamplesL of prediction luma samples as inputs. And the output is assigned to the array predSamplesL of prediction luma samples.

…

**I.8.5.3.3.10 The decoding process for depth compensation calibration**

Inputs to this process are:

* a luma location ( xCb, yCb ) specifying 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,
* an (nCbS)x(nCbS) array predSamples of luma prediction samples.

Outputs of this process is:

* a modified (nCbS)x(nCbS) array predSamplesof luma prediction samples.

1. For xB in the range of 0 to nCbS/4-1, inclusive, the following applies:

* For yB in the range of 0 to nCbS/4-1, inclusive, the following applies:
  + The derivation process for calibrated depth value as specified in subclause I.8.5.3.3.10.1 is invoked with the luma coding block location ( xCb, yCb ), the calibration block offset (4\*xB, 4\*yB), and the predSamples as inputs, and a calibrated depth value calibratedDepthValue as the output.
  + The inter prediction samples (predSamples) are modified as follows
    - for ( y = 0; y < 4; y++ )
    - for( x = 0; x < nCbSL; x++ )
    - predSamples [ 4\*xB +x ][ 4\*yB +y ] = calibratedDepthValue.

**I.8.5.3.3.10.1 The derivation process for calibrated depth value**

Inputs to this process are:

* a luma location ( xCb, yCb ) specifying the top-left sample of the current luma coding block relative to the top-left luma sample of the current picture,
* a calibration block offset (xCl, yCl) specifying the top-left sample of the current calibration block offset relative to ( xCb, yCb ),
* an (nCbS)x(nCbS) array predSamples of luma prediction samples.

Outputs of this process is:

* a calibrated depth value calibratedDepthValue.

A variable nCandNum is set equal to 0.

A variable nSeedCandNum is set equal to 0.

A position list seedCandPosList is assigned as seedCandPosList[0]=(2,2), seedCandPosList[1]=(0,0), seedCandPosList[2]=(3,0), seedCandPosList[3]=(0,3) and seedCandPosList[4]=(3,3).

A seed depth value list seedValueList is assigned as follows:

For a variable iSeedCandIdx from 0 or 4 inclusively, a variable pPotentialSeedVaule is set equal to predSamples[4\*xCl+seedCandPosList[iSeedCandIdx][0]][4\*yCl+seedCandPosList[iSeedCandIdx][1]]. When pPotentialSeedVaule is not equal to seedValueList[k]with k from 0 to nSeedCandNum-1 inclusively, seedValueList[nSeedCandNum]= pPotentialSeedVaule and nSeedCandNum++.

A check list valueCheckList is initialized as follows:

valueCheckList[ k ] are set equal to 0 for all k from 0 to (1<<BitDepthY)-1.

calibratedDepthValue is set equal to 1<<(BitDepthY-1).

A variable minDiff is set equal to (1<<BitDepthY)\*32.

A variable numCheckNum is set equal to 0;

For a variable nOffset from 0 to 4 inclusively, the following applies:

For iSeedCandIdx from 0 or nSeedCandNum-1 inclusively, the following applies:

For a variable offSign from 0 or 1 inclusively, the following applies:

* A variable depthValueCand is set equal to seedValueList[iSeedCandIdx]+(1- 2\* offSign ) \* nOffset. When valueCheckList[depthValueCand ] is equal to 0, the following applies:
  + - The derivation process for texture inter-view difference as specified in subclause I.8.5.3.3.10.1.1 is invoked with the luma coding block location ( xCb, yCb ), the calibration block offset (xB, yB), and the candidate depth value depthValueCand as input, and the output is the inter-view difference ivDiff.
    - When ivDiff is less than minDiff, minDiff is set equal to ivDiff and calibratedDepthValue is set equal to depthValueCand
    - If numCheckNum is lower than 16, numCheckNum++. Otherwise, the whole process terminates.

**I.8.5.3.3.10.1.1 The derivation process for texture inter-view difference**

Inputs to this process are:

* a luma location ( xCb, yCb ) specifying the top-left sample of the current luma coding block relative to the top-left luma sample of the current picture,
* a calibration block offset (xCl, yCl) specifying the top-left sample of the current calibration block offset relative to ( xCb, yCb ),
* a candidate depth value depthValueCand.

Outputs of this process is:

* a texture difference value ivDiff.

A disparity vector DV is calculated as DV [ 0 ] = DepthToDisparityB[ RefViewIdx[ xCb ][ yCb  ]] [ depthValueCand ] and DV[ 1 ] = 0. The disparity vector DVC is set equal to DV.

Let textPicCur be the picture with PicOrderCntVal and ViewIdx equal to PicOrderCntVal and ViewIdx of the current picture, and DepthFlag being equal to 0 and textPicRef be the picture with PicOrderCntVal and ViewIdx equal to PicOrderCntVal of the current picture and RefViewIdx[ xPb ][ yPb  ], and DepthFlag being equal to 0.

The arrays predSamplesLXL, predSamplesLXCb, and predSamplesLXCr are derived as specified in the following:

* The arrays predSamplesL, predSamplesCb, and predSamplesCr are derived by invoking the bilinear sample interpolation process specified in subclause  with the luma locations ( xCb, yCb ), ( 4\*xCl, 4\*yCl), the luma prediction block width 4, the luma prediction block height 4, the motion vectors DV, DVC, and the reference arrays with textPicRefL, textPicRefCb and textPicRef Cr. as inputs.
* The arrays curSamplesL, curSamplesCb, and curSamplesCr are derived by invoking the bilinear sample interpolation process specified in subclause  with the luma locations ( xCb, yCb ), ( 4\*xCl, 4\*yCl), the luma prediction block width 4, the luma prediction block height 4, the motion vectors (0,0), the motion vectors (0,0), and the reference arrays with textPicCurL, textPicCurCb and textPicCurCr. as inputs.

The variable nDiffL is calculated as follows:

nDiffL = 0;

for( x = 0; x < 4; x++ )

for ( y = 0; y < 4; y++ )

nDiffL += abs( curSamplesL[ x ][ y ] - predSamplesL[ x ][ y ])

The variable textSubWidthC and textSubHeightC are set equal to SubWidthC and SubHeightC of textPicCur respectively.

The variable nDiffX with X equal to Cb and Cr are calculated as follows:

nDiffX = 0;

for( x = 0; x < 4/ textSubWidthC; x++ )

for ( y = 0; y < 4/ textSubHeightC; y++ )

nDiffX += abs( curSamplesX[ x ][ y ] – predSamplesX[ x ][ y ])

ivDiff is set equal to nDiffL+ (( nDiffCb + nDiffCr)<<2)