|  |  |
| --- | --- |
| **Joint Collaborative Team on 3D Video Coding Extensions**  **of ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11**  6th Meeting: Geneva, CH, 25 Oct. – 1 Nov. 2013 | Document: JCT3V-F0102 |

|  |  |  |  |
| --- | --- | --- | --- |
| *Title:* | **CE1-related: VSP partitioning for AMP** | | |
| *Status:* | Input Document to JCT-3V | | |
| *Purpose:* | Proposal | | |
| *Author(s) or Contact(s):* | Tomohiro Ikai  Yoshiya Yamamoto  1-9-2 Nakase, Mihama-ku, Chiba-shi, Chiba 261-8520  JAPAN | Tel: Email: | +81-43-299-8526 [ikai.tomohiro@sharp.co.jp](mailto:ikai.tomohiro@sharp.co.jp) |
| *Source:* | Sharp Corporation | | |

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

# Abstract

Adaptive block partitioning for VSP, which uses 8x4 or 4x8 subblock depending on the depth of 8x8 block in JCT3V-E0208, was adopted. However in AMP case, 4x4 block motion compensation process can occur. It is proposed to fix VSP partitioning for AMP to avoid the 4x4 block based process.

# Introduction

In AMP case, sub-partitioning is prevented in 16x4 or 4x16 case (PU width or PU height is less than 8).

However when 16x12 PU (or 12x16) is divided into 8x4 subblock (or 4x8 subblock), 4x4 block motion compensation process can be occured as shown below.



# Proposal

It is proposed to use 8x4 subblock for 16x4 and 16x12 AMP PU (where height % 8 != 0) and to use 4x8 subblock for 4x16 or 12x16 AMP PU (where width % 8 != 0).



Software patch is attached.

#define SHARP\_VSP\_AMP 1

(#define SHARP\_DEPTH\_ACCESS\_RESTRICTION 0

#define SHARP\_DEPTH\_ACCESS\_RESTRICTION\_PLUS 0)

# Proposed Text

Replace the following

(3D-HEVC WD1) The derivation process for a disparity sample array as specified in section H.8.5.5.2 is invoked with the luma location ( xP, yP ), the disparity vector mvDisp, the variable refViewIdx, the variable depthViewIdx, the variable nPSW equal to ( nPbW  = =  12 ? 16 : nPbW ), the variable nPSH equal to ( nPbH  = =  12 ? 16 : nPbH ), and the variable splitFlag equal to ( nPbW > 4  &&  nPbH > 4 ) as the inputs, and the output is the array disparitySamples of size (nPSW)x(nPSH).

with

(Proposal) The derivation process for a disparity sample array as specified in section H.8.5.5.2 is invoked with the luma location ( xP, yP ), the disparity vector mvDisp, the variable refViewIdx, the variable depthViewIdx, the variable nPSW equal to nPbW, the variable nPSH equal to nPbH, and the variable splitFlag equal to 1 as the inputs, and the output is the array disparitySamples of size (nPSW)x(nPSH).

**H.8.5.4.2 Derivation process for a disparity sample array**

Inputs to this process are:

* a luma location ( xP, yP ) relative to the top-left luma sample of the current picture,
* a disparity vector mvDisp,
* a view order index refViewIdx specifying a reference view,
* a view order index depthViewIdx specifying the view the depth should be derived from
* variables nPSW and nPSH specifying a width and a height, respectively
* a variable splitFlag.

Outputs of this process are:

* a (nPSW)x(nPSH) array disparitySamples of disparities values.

Let refDepPic the picture in the current access unit with ViewIdx( refDepPic ) equal to ViewIdx and DepthFlag( refDepPic ) equal to 1.

Let refDepPels be an array of reconstructed depth samples refDepPic. The luma location (xTL, yTL) of top-left luma sample of a block in refDepPels is derived by

* 1. xTL = xP + ( ( mvDisp[ 0 ] + 2 ) >> 2 ) (H‑255)
  2. yTL = yP + ( ( mvDisp[ 1 ] + 2 ) >> 2 ) (H‑256)

1. The variables nSubBlkW and nSubBlkH are derived as specified in the following:
   1. nSubBlkW = (splitFlag && !(nPSW % 8) && !( nPSH % 8) ) \*1 ? 8 : nPSW (H‑257)
   2. nSubBlkH = (splitFlag && !(nPSW % 8) && !( nPSH % 8) ) ? 8 : nPSH (H‑258)

The array disparitySamples of size (nPSW)x(nPSH) is derived as specified in the following:

* For sBy in the range of 0 to ( ( nPSH / nSubBlkH) – 1 ), inclusive, the following applies:
  + For sBx in the range of 0 to ( ( nPSW / nSubBlkW) – 1 ), inclusive, the following applies:
    - The variables xB, yB, xP0, yP0, xP1, yP1, are derived as specified in the following:

xB = sBx \* nSubBlkW  
 yB = sBy \* nSubBlkH  
 xP0 = Clip3( 0, pic\_width\_in\_luma\_samples – 1, xTL + xB )  
 yP0 = Clip3( 0, pic\_height\_in\_luma\_samples – 1, yTL + yB )  
 xP1 = Clip3( 0, pic\_width\_in\_luma\_samples – 1, xTL + xB + nSubBlkW – 1 )  
 yP1 = Clip3( 0, pic\_height\_in\_luma\_samples – 1, yTL + yB + nSubBlkH – 1 )

* + - The variable nSubSubBlkW is set equal to nSubBlkW and the variable nSubSubBlkH is set equal to nSubBlkH.
    - When nPSH % 8 \*2 is not equal to 0, nSubSubBlkW and nSubSubBlkH are modified as follows:

nSubSubBlkW = 8 (‑240)

nSubSubBlkH = 4 (‑241)

* + - Otherwise, when nPSW % 8 \*3 is not equal to 0, nSubSubBlkW and nSubSubBlkH are modified as follows:

nSubSubBlkW = 4 (‑240)

nSubSubBlkH = 8 (‑241)

* + - Otherwise, When splitFlag is equal to 1, nSubSubBlkW and nSubSubBlkW are modified as follows:
      * The variable horSplitFlag is derived as specified in the following:
        1. horSplitFlag = ( refDepPels[ xP0 ][ yP0 ] > refDepPels[ xP1 ][ yP1 ] )  
            = = ( refDepPels[ xP1 ][ yP0 ] > refDepPels[ xP0 ][ yP1] ) ) (H‑259)
        2. nSubSubBlkW = horSplitFlag ? nSubSubBlkW : ( nSubSubBlkW >> 1 ) (H‑260)
        3. nSubSubBlkH = horSplitFlag ? ( nSubSubBlkH >> 1 ) : nSubSubBlkW (H‑261)
    - The derivation process for a disparity sample block as specified in subclause H.8.5.5.2.1 is invoked with the luma location ( xB, yB ), variables nSubBlkW and nSubBlkH, the array of reconstructed depth samples refDepPels, the luma location ( xTL, yTL ), the variables nSubSubBlkW and nSubSubBlkW, the view order index refViewIdx, and the array disparitySamples as the inputs, and the output is the modified array disparitySamples.

# Simulation results

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | video 0 | video 1 | video 2 | video PSNR / video bitrate | video PSNR / total bitrate | synth PSNR / total bitrate | enc time | dec time | ren time |
| Balloons | 0.0% | 0.1% | 0.0% | 0.0% | 0.0% | 0.0% | 100.5% | 100.4% | 102.1% |
| Kendo | 0.0% | 0.1% | 0.0% | 0.0% | 0.0% | 0.0% | 100.5% | 101.3% | 101.0% |
| Newspaper\_CC | 0.0% | -0.1% | 0.0% | 0.0% | 0.0% | 0.0% | 100.5% | 100.6% | 100.0% |
| GT\_Fly | 0.0% | 0.1% | -0.1% | 0.0% | 0.0% | 0.0% | 99.8% | 100.3% | 100.0% |
| Poznan\_Hall2 | 0.0% | -0.2% | -0.1% | -0.1% | 0.0% | -0.2% | 99.8% | 99.4% | 100.1% |
| Poznan\_Street | 0.0% | 0.1% | 0.1% | 0.0% | 0.0% | 0.0% | 100.1% | 100.2% | 99.6% |
| Undo\_Dancer | 0.0% | 0.0% | -0.1% | 0.0% | 0.0% | -0.1% | 100.0% | 100.7% | 100.0% |
| 1024x768 | 0.0% | 0.0% | 0.0% | 0.0% | 0.0% | 0.0% | 100.5% | 100.8% | 101.1% |
| 1920x1088 | 0.0% | 0.0% | -0.1% | 0.0% | 0.0% | -0.1% | 99.9% | 100.2% | 99.9% |
| **average** | **0.0%** | **0.0%** | **0.0%** | **0.0%** | **0.0%** | **0.0%** | **100.2%** | **100.4%** | **100.4%** |
|  |  |  |  |  |  |  |  |  |  |
| Shark | 0.0% | -0.1% | 0.1% | 0.0% | 0.0% | 0.0% | 99.9% | 95.7% | 98.6% |

# Conclusion

It is proposed to fix VSP partitioning for AMP to avoid the 4x4 block based process.

# References

[1] S. Shimizu, S. Sugimoto, “3D-CE1.h: Adaptive block partitioning for VSP”, JCT3V-E0207, JCT3V 5th Meeting: Vienna, AT, 27 July – 2 Aug. 2013

# Patent rights declaration(s)

**Sharp Corporation may have current or pending patent rights relating to the technology described in this contribution and, conditioned on reciprocity, is prepared to grant licenses under reasonable and non-discriminatory terms as necessary for implementation of the resulting ITU-T Recommendation | ISO/IEC International Standard (per box 2 of the ITU-T/ITU-R/ISO/IEC patent statement and licensing declaration form).**