|  |  |
| --- | --- |
| **Joint Collaborative Team on 3D Video Coding Extensions**  **of ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11**  11th Meeting: Geneva, CH, 12–18 Feb. 2015 | Document: JCT3V-K0041 |

|  |  |  |  |
| --- | --- | --- | --- |
| *Title:* | **CE1-related: Simplification of segmental prediction** | | |
| *Status:* | Input Document to JCT-3V | | |
| *Purpose:* | Proposal | | |
| *Author(s) or Contact(s):* | Takeshi Tsukuba  Tomohiro Ikai  1-9-2 Nakase, Mihama-ku, Chiba-shi, Chiba 261-8520, JAPAN | Tel: Email: | +81-43-299-8526 [tsukuba.takeshi@sharp.co.jp](mailto:tsukuba.takeshi@sharp.co.jp) |
| *Source:* | SHARP Corporation | | |

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

# Abstract

This contribution proposes a simplification of CE1 segmental prediction of JCT3V-K0031. The proposal claims that the buffer memory size utilized in mode value (most occurred value) derivation can be reduced by 50% and 88.4 % with Proposal 1 and Proposal 2, respectively. Proposal 1 uses 1D buffer rather than 2D buffer. Proposal 2 uses 1) sub-sampled pixel position and 2) restricted count value in addition to proposal 1. The experimental result reportedly shows that proposal 1 doesn’t change the coding results and proposal 2’ coding loss is negligible, i.e. 0.XX% in synthesis in CTC, compared to Test 1 of JCT3V-K0031.

# Introduction

In JCT3V-J0032 [1] and the current JCT3V-K0031 [2] proposes segmental prediction for depth coding. However, the technique requires a mode value derivation for each segment, which has the following problems.

* Buffer memory size can be large because it is in proportion to PU size and bit-depth.
* 2D buffer (histogram for each segment) is unnecessarily used.1D buffer (one histogram for both segments) can do the same thing.

As shown in Table 1, the required buffer size can be 3.25 KBytes in 10 bit case, which is considered too large.

**Table 1: Buffer size requirement in CE Test 1**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Bit depth | Block size | Number of segments **S** | Bit-length for each sampleCount[][] **C**  (range of sampleCount[][]) | Range of sampleValue **V** | Required buffer size (=**S**\***C**\***V**)  [bits] |
| 8 | 8x8 | 2 | 7bit (0 to 64) | 256 | 3584 =2\*256\*7 bit |
| 8 | 16x16 | 2 | 9bit (0 to 256) | 256 | 4608 =2\*256\*9 bit |
| 8 | 32x32 | 2 | 11bit (0 to 1024) | 256 | 5632 =2\*256\*11 bit |
| 8 | 64x64 | 2 | 13bit (0 to 4096) | 1024 | 6656 =2\*256\*13 bit |
| 10 | 8x8 | 2 | 7bit (0 to 64) | 1024 | 14336 =2\*1024\*7 bit |
| 10 | 16x16 | 2 | 9bit (0 to 256) | 1024 | 18432 =2\*1024\*9 bit |
| 10 | 32x32 | 2 | 11bit (0 to 1024) | 1024 | 22528 =2\*1024\*11 bit |
| 10 | 64x64 | 2 | 13bit (0 to 4096) | 1024 | 26624 =2\*1024\*13 bit  (= 3.25KB) |

**Buffer memory size**

Since the buffer (histogram) can be implemented as array sampleCount [ segIdx ][ sampleValue ], where segIdx is a segment index, the required 2D buffer size sampleCount[ ][ ] can be calculated as multiplication of number of segments **S**, bit-length for each sampleCount[ ][ ] **C** ( = log2(maximum count of sampleValue) + 1 ), and range of sampleValue **V**. For example, in 64x64 block, the log2(maximum count of sampleValue) + 1 is equal to log2(64\*64) + 1 =13bit. In 10 bit case, the range of sampleValue is equal to 1024. The number of segments is 2. Then the required buffer memory size is calculated as 2\* 13 \* 1024 = 26624 bits = 3.25KBytes.

**Requirement of 10bit ready**

Although the current specification [3] doesn’t include 3D-HEVC Main 10 (10 bit) profile, it is asserted that the basic extension tools should be applicable in 10 bit case. Otherwise the tools may be disabled in 10 bit profile or some special fixed process for 10 bit profile may be needed. It is noted that HEVC Main and HEVC Main 10 applies the same process.

**Segmental prediction** with 2 segments [2] consists of the following process:

1. Derive a threshold T.
2. Classify the prediction block into two segments with T.
3. Derive the mode segPred [s] for each segments.

segPred[0] = segPred[1] = 1<<(BitDepthY-1);

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

for (x=0; x<nTbs; x++){

s = segIdx[x,y]

sampleCount[ s ][ p[x,y] ] ++;

if (sampleCount[ s ][ p[x,y] ] > sampleCount[ s ][ segPred[s] ] )

segPred [ s ] = p[x, y];

}

1. Derive a reconstructed sample value R[s] for each segment s.

R[s] = segPred [s] + dcOffset [s]



**Fig. 1: An example of segmental prediction with 2 segments for an 8x8 block [1]**

# Proposal

**Proposal 1:** To reduce the buffer memory size, we firstly propose to use 1D buffer irrespective of the number of segments.

Fig. 2 shows an example of histograms (H0, H1) for each segment (R0, R1) separated by a threshold T. As can be seen in Fig. 2, those histogram Hi(i = 0, 1) don’t overlap each other so that it is possible to utilize the same 1D buffer for segment R0 and segment R1, which can save buffer memory size by 50%.



**Fig. 2: An example of histograms separated by threshold T**

Thus, the derivation process of the mode value described in section 1 can be simplified as follows:

~~segPred[0] = segPred[1] = 1<<(BitDepthY-1);~~

segPred[0] = 0;\*1

segPred[1] = T;\*1

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

for(x=0; x<nTbs; x++){

s = segIdx[x,y];

sampleCount~~[ s ]~~[ p[x,y] ] ++;

if (sampleCount~~[ s ]~~ [ p[x,y] ] > sampleCount~~[ s ]~~ [segPred[s] ] )

segPred [ s ] = p[x, y];

}

\*1: The initial value should be correctly chosen (the value of segPred[0] should be less than T and segPred[1] should be larger or equal to T) if we share the histogram. The initial value is updated with real value when full sampling. (So not realistic one (segPred[0] = 0) is ok for this purpose).

**Proposal 2:** In addition to Proposal 1, we also propose to

1) Sub-sample pixel position to reduce possible maximum count (Step size is NStep) as shown in Fig 3.

2) Restrict the maximum sample value (Max value: CMax)

Where the variable NStep, dX, dY and CMax are derived as follows:

* NStep = nTBS>>2,
* dX = dY = NStep >> 1,
* CMax = 7 ( = 2^(2\*M -1) – 1) where M = 2.



**Fig. 3: An example of sub-sample position**

Thus, the derivation process of the mode value described in Proposal 1 can be modified as follows:

segPred[0] = T;\*2

segPred[1] = T;\*2

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

for(j=0; i<4; i++){

x = i\*NStep + dx, y = j \* NStep + dy;

s = segIdx[x,y];

sampleCount[ p[x,y] ] = min ( CMax, sampleCount[ p[x,y] + 1] );

if ( sampleCount [ p[x,y] ] > sampleCount [segPred[s] ] )

segPred [ s ] = p[x, y];

}

\*2: When subsampling, the initial value may not be updated and can be used as the final mode value. Thus initial value should not be far from the realistic one. The initial value for proposal 1 (segPred[0]) is not good for this purpose. The fact that the initial value of segPred[0] (= T) is not less than T, is practically ok.

**Complexity analysis**

Table 2 shows the number of sampled pixels for deriving the mode value and Table 3 shows the required buffer memory size in CE1 and our proposals. As shown in Table 3, the proposal 1 and proposal 2 decreases buffer memory size by 50% and 88.4%, respectively compared to CE1 Test 1. Proposal 2 uses less than 0.4 Kbytes, which should be OK for implementation.

**Table 2: Number of sampled pixels in CE1 and Proposal**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Block size | CE1 Test1 | CE1 Test2 | Proposal 1 | Proposal 2 |
| 8x8 | 64 | 16 | 64 | 16 |
| 16x16 | 256 | 64 | 256 | 16 |
| 32x32 | 1024 | 256 | 1024 | 16 |
| 64x64 | 4096(=64x64) | 1024(= 32x32) | 4096(=64x64) | 16(=4x4) |

**Table 3: Buffer size requirements in CE1 Test1/CE1 Test2/Proposal**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Bit-depth | Block size | CE1 test1 | CE1 test2 | | Proposal 1 | | Proposal 2 | |
| Buffer size [KB] | Buffer size [KB] | Reduction rate [%] | Buffer size  [KB] | Reduction rate[%] | Buffer size  [KB] | Reduction rate [%] |
| 8 | 64x64 | 0.813KB  (=2\*13\*256) | 0.688KB  (=2\*11\*256) | 15.4% | 0.406KB  (=13\*256) | 50.0% | 0.094KB  (=3\*256) | 88.4% |
| 10 | 64x64 | 3.25KB  (=2\*13\*1024) | 2.75KB  (=2\*11\*1024) | 15.4% | 1.625KB  (=13\*1024) | 50.0% | 0.375KB  (=3\*1024) | 88.4% |

# Experimental results

The proposed methods are integrated on top of CE proposal of JCT3V-K0031, provided by proponents.  Table 5 shows the summary results on CE1 Test 1, CE1 Test 2, Proposal 1 and Proposal 2.

It is observed that Proposal 1 achieves 0.xx% BD-rate saving for the synthesized views compared to HTM-13.0 and also achieves no coding loss (0.xx%) compared to CE1 Test 1.

It is also observed that Proposal 2 achieves 0.XX% BD-rate saving for the synthesized views compared to HTM-13.0 and also achieves negligible coding loss (0.xx%) compared to CE1 Test 1.

**Table 4. Configuration for tests**

|  |  |  |
| --- | --- | --- |
|  | Proposal 1 | Proposal 2 |
| Test 1 | X |  |
| Test 2 | X | X |

**Table 5. Summary of coding performance on CE1 Test 1, CE1 Test 2 and Proposal (CTC) (TBD)**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | video 0 | video 1 | video 2 | video PSNR / video bitrate | video PSNR / total bitrate | synth PSNR / total bitrate |
| CE1 Test 1 | **0.00%** | **-0.10%** | **-0.04%** | **-0.01%** | **0.04%** | **-0.49%** |
| CE1 Test 2 | **0.00%** | **-0.08%** | **0.01%** | **0.00%** | **0.04%** | **-0.46%** |
| Proposal 1 |  |  |  |  |  |  |
| Proposal 2 |  |  |  |  |  |  |

**Table 6: Results on Test 1 (CTC) TBD**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | video 0 | video 1 | video 2 | video PSNR / video bitrate | video PSNR / total bitrate | synth PSNR / total bitrate |
| Balloons |  |  |  |  |  |  |
| Kendo |  |  |  |  |  |  |
| Newspaper\_CC |  |  |  |  |  |  |
| GT\_Fly |  |  |  |  |  |  |
| Poznan\_Hall2 |  |  |  |  |  |  |
| Poznan\_Street |  |  |  |  |  |  |
| Undo\_Dancer |  |  |  |  |  |  |
| Shark |  |  |  |  |  |  |
| 1024x768 |  |  |  |  |  |  |
| 1920x1088 |  |  |  |  |  |  |
| **average** |  |  |  |  |  |  |

**Table 7: Results on Test 2 (CTC) (TBD)**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | video 0 | video 1 | video 2 | video PSNR / video bitrate | video PSNR / total bitrate | synth PSNR / total bitrate |
| Balloons |  |  |  |  |  |  |
| Kendo |  |  |  |  |  |  |
| Newspaper\_CC |  |  |  |  |  |  |
| GT\_Fly |  |  |  |  |  |  |
| Poznan\_Hall2 |  |  |  |  |  |  |
| Poznan\_Street |  |  |  |  |  |  |
| Undo\_Dancer |  |  |  |  |  |  |
| Shark |  |  |  |  |  |  |
| 1024x768 |  |  |  |  |  |  |
| 1920x1088 |  |  |  |  |  |  |
| **average** |  |  |  |  |  |  |

# Conclusion

Because proposal 1, which uses 1D buffer rather than 2D buffer, has obvious benefit (50 % reduction), it is recommend to adopt that aspect. The proposal 2 is also beneficial for future usage in 3D-HEVC. Thus it is recommended to adopt that one, which include proposal 1’s aspect.

# Reference

1. K. Zhang, J. An, X. Zhang, H, Huang, J. Lin, S. Lei, “3D-CE1: Segmental prediction in 3D-HEVC,” JCT3V-J0032, Oct. 2014.
2. K. Zhang, J. An, X. Zhang, H, Huang, J. Lin, S. Lei, “3D-CE1: Segmental prediction in 3D-HEVC,” JCT3V-K0031, Feb. 2015.
3. G. Tech, K. Wegner, Y. Chen and S. Yea, "3D-HEVC Draft Text 6," JCT3V-J001, Oct. 2014.
4. HTM-13.0, <https://hevc.hhi.fraunhofer.de/svn/svn_3DVCSoftware/tags/HTM-13.0>.
5. K. Müller, A. Vetro, “Common test conditions of 3DV core experiments,” JCT3V-G1100, Jan. 2014.

# 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).**