|  |  |
| --- | --- |
| **Joint Collaborative Team on 3D Video Coding Extension Development**  **of ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11**  6th Meeting: Geneva, Switzerland, 25 Oct –1 Nov. 2013 | Document: JCT3V- F0157 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| *Title:* | **3D-CE5.h related: Simplified DC predictor improvement for depth intra modes** | | | |
| *Status:* | Input Document | | | |
| *Purpose:* | Proposal | | | |
| *Author(s) or Contact(s):* | Zhouye Gu1  Jianhua Zheng2  Nam Ling1  Philipp Zhang2 | Tel: +1-206-816-2367 Email: zgu@scu.edu  zhengjianhua@hisilicon.com | |  |
| *Source:* Santa Clara University1 and HiSilicon Technologies2 | | |  | |

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

# Abstract

This contribution proposes an improvement of simplified DC prediction for depth intra coding methods including DMM, SDC\_DMM1 and Chain Coding in 3D-HEVC. To take more partition cases into consideration, the performance of simplified DC prediction is further improved in this contribution. It is reported that -0.1% BD-rate gain is achieved under Intra-only test condition on synthesized views.

# Introduction

In 3D-HEVC, partition-based methods including DMM, SDC\_DMM1 and Chain Coding are applied for depth map intra coding. With partition-based depth map intra coding methods, a depth block is partitioned by two parts, i.e., P0 and P1 as illustrated in Figure 1, and each part is represented by a single DC value. Furthermore, the DC value (DC0 and DC1) for each partition (P0 and P1) can be further compensated by a delta DC/residual value which is explicitly signalled in the bitstream.

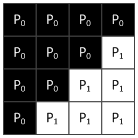
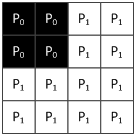
 

Figure 1: Examples of depth block partitioning[2]

## DC prediction in current 3D-HEVC [3]

The DC value of each partition is predicted using one or two reconstructed neighboring reference samples depending on the partition pattern as shown in Figure 2[3]. The reconstructed neighboring reference samples are denoted by pi,j, where i = −1, j = −1..7 and i = 0..7, j = −1 in this example.

Assume the partition value of the left-top sample (c0,0) is X, where X= 0 or 1, given the partition pattern *bPattern*x,y, where x = 0..*N* − 1, y = 0..*N*−1, the predicted DC values, i.e., DC0 and DC1, are derived by follows:

Set bT = (*bPattern*0,0 != *bPattern*N-1,0)? 1 : 0; set bL = (*bPattern*0,0 != *bPattern*0,N-1)? 1 : 0

If bT equals bL

* DCX = (p-1,0 + p0,-1)>>1
* DC1-X = bL ? (p-1,N-1 + pN-1,-1)>>1 : 2B-1

Otherwise

* DCX = bL ? p(N-1)>>1,-1 : p-1, (N-1)>>1
* DC1-X = bL ? p-1,N-1 : pN-1,-1

(a) Case 1 (b) Case 2

(c) Case 3 (d) Case 4

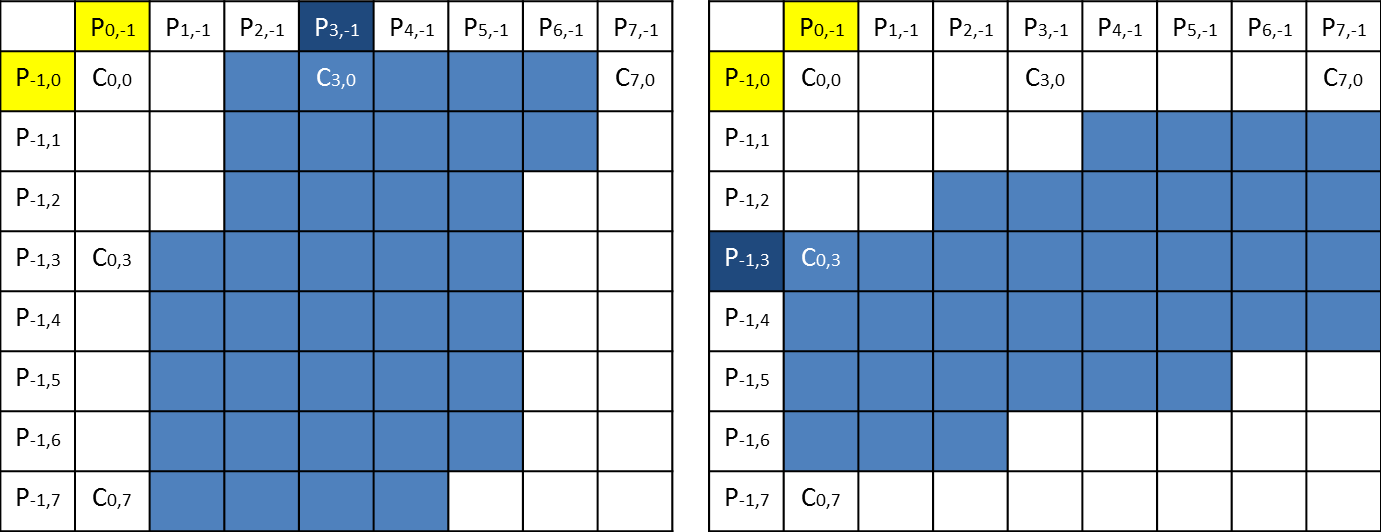
Figure 2: Selection of reference samples for difference partition pattern cases. [3]

However, the contour partition generated by DMM4 is not well classified in Figure 2. Moreover, if the reference pixels in the top-right and left-bottom blocks are available, they can also be used for DC prediction. Therefore, in this proposal, we propose an improved simplified DC prediction method below.

# Proposed solution

Situation 1:

For Case 1 in Figure 2 (a), given the block size equal to NxN, if we observe any partition value between the line of (c0,0) and (cN,0) are not the same as the binary partition value of (c0,0), it means this partition pattern is predicted by DMM4 and it could have a partition like what we shown in Figure 3 (a). Similarly, if we observe any partition value between the line of (c0,0) and (c0,N) are not the same as (c0,0), it means this partition pattern is predicted by DMM4 like Figure 3 (b). Under both cases shown in Figure 3, the DC of the pixels in the contour (light blues pixels in Figure 3) is just predicted like bottom-right partition in Case 1 of Figure 2. However, for these cases, the reference pixels between p0,-1 and p7,-1 (or pixels between p-1,0 and p-1,7) could be a better DC predictor such as the reference pixels p3,-1 and p-1,3 shown in Figure 3.



(a) Case 1a: (c0,0), (c7,0) and (c0,7) have the same partition (b) Case 1b: (c0,0), (c7,0) and (c0,7) have the same partition

value which is different from the partition value of (c3,0) value which is different from the partition value of (c0,3)

Figure 3. Selection of reference samples for additional partition pattern cases

Based on analysis above, we estimate the DC of the pixels in the contour by the value of the middle reference pixel p(N-1)>>1,-1 or p-1, (N-1)>>1 depending on the partition pattern we get.

More specifically, under case 1 in Figure 2 (a), if the left-middle partition value, (c0, (N-1)>>1), in the partition block is not the same as the top-left partition value, (c0,0), in the partition block, we will use the left-middle reference pixel, (p-1, (N-1)>>1 ), as the DC predictor of the pixels in the contour. If the top-middle partition value, (c (N-1)>>1,0), in the partition block is not the same as the top-left partition value, (c0,0), in the partition block, we will use the top-middle reference pixel,( p (N-1)>>1,-1), as the DC predictor of the pixels in the contour. If the left-middle partition value, (c0, (N-1)>>1) is the same as the top-middle partition value, (c (N-1)>>1,0), we will use the average of the left-middle reference pixel, (p-1, (N-1)>>1) and top-middle reference pixel, (p (N-1)>>1,-1) as the DC predictor of the pixels in the contour.

We describe this DC estimation algorithm under Case 1 (in Figure 2 (a)) as follows:

Set bTM = (*bPattern*0,0 != *bPattern*(N-1)>>1,0)? 1 : 0; set bLM = (*bPattern*0,0 != *bPattern*0, (N-1)>>1)? 1 : 0

If bTM does not equal to bLM

DC1-X = bLM ? p-1, (N-1)>>1 : p(N-1)>>1,-1;

Otherwise

DC1-X = (p-1, (N-1)>>1 + p(N-1)>>1,-1) >> 1;

Situation 2:

Under case 1 in Figure 2 (a), if both bTM and bLM equal to 0, based on the availability of neighboring reference pixels, the proposed method predicts the bottom-right part of Case 1 as shown in Figure 4. Given the block size equal to NxN, we check if the value of reference pixels (p-1,2\*N-1) and (p2\*N-1,-1) are available. If they are both available, we will calculate the absolute value of difference between (p-1,2\*N-1) and (p-1,0) as abs(p-1,2\*N-1 - p-1,0) and the absolute value of difference between (p2\*N-1,-1) and (p0,-1) as abs(p2\*N-1,-1 - p0,-1). Then if abs(p-1,2\*N-1 - p-1,0) is larger than abs(p2\*N-1,-1 - p0,-1), the value of (p-1,2\*N-1) is used as the DC predictor of the right-bottom partition, while if abs(p-1,2\*N-1 - p-1,0) is smaller than or equal to abs(p2\*N-1,-1 - p0,-1), the value of (p2\*N-1,-1) is used as the DC predictor of the right-bottom partition.

We describe this DC estimation algorithm under Case 1 (in Figure 2 (a)) as follows:

If p-1,2\*N-1 and p2\*N-1,-1 are both available

DC1-X = abs(p-1,2\*N-1 - p-1,0) > abs(p2\*N-1,-1 - p0,-1) ? p-1,2\*N-1 : p2\*N-1,-1

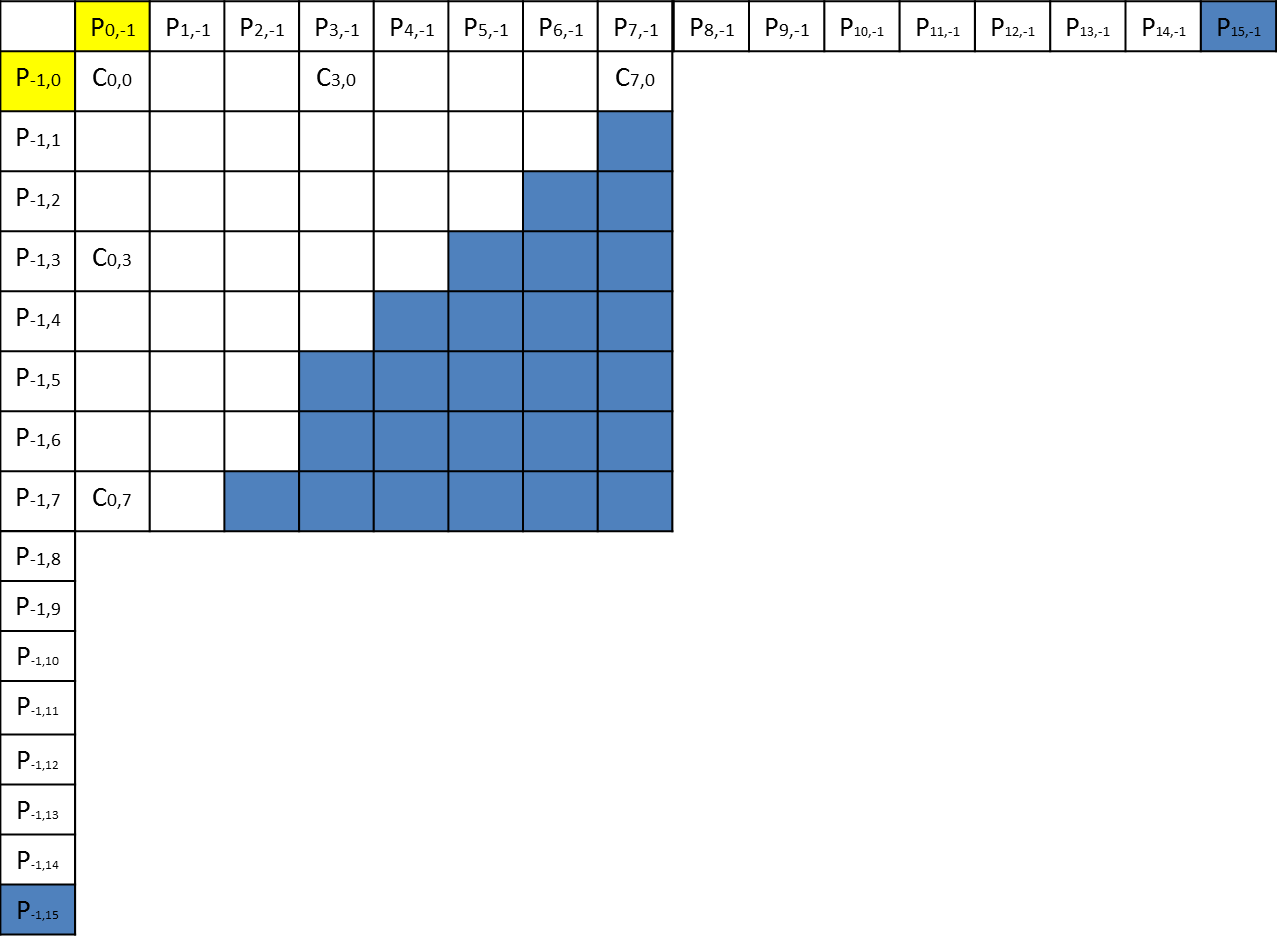


Figure 4. Selection of reference samples from neighboring

# Experimental results

The proposed method is implemented on top of HTM-8.0, and simulations were performed under both “common test condition” [1] and “Intra-only” [2] configurations.

The results are summarized in Table 1 and Table 2. As it is reported, the proposed method achieves -0.1% BD-rate coding gain on synthesized views for “Intra-only”.

Table 1: BD rate results for 3-view case under CTC

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | 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% | 0.1% | 0.0% | 0.0% | 0.0% | 100.0% | 96.0% | 95.6% |
| Kendo | 0.0% | -0.1% | -0.2% | 0.0% | 0.0% | -0.1% | 99.7% | 98.3% | 96.0% |
| Newspaper\_CC | 0.0% | 0.0% | 0.2% | 0.0% | 0.0% | 0.0% | 99.9% | 96.6% | 101.2% |
| GT\_Fly | 0.0% | 0.1% | 0.0% | 0.0% | 0.0% | 0.1% | 97.9% | 76.5% | 95.2% |
| Poznan\_Hall2 | 0.0% | 0.0% | 0.1% | 0.0% | 0.0% | -0.1% | 99.8% | 113.1% | 103.6% |
| Poznan\_Street | 0.0% | -0.1% | 0.1% | 0.0% | 0.0% | -0.1% | 99.1% | 91.8% | 91.3% |
| Undo\_Dancer | 0.0% | 0.1% | 0.0% | 0.0% | 0.0% | -0.1% | 97.8% | 106.8% | 99.5% |
| 1024x768 | 0.0% | 0.0% | 0.0% | 0.0% | 0.0% | 0.0% | 99.8% | 97.0% | 97.6% |
| 1920x1088 | 0.0% | 0.0% | 0.0% | 0.0% | 0.0% | 0.0% | 98.7% | 97.0% | 97.4% |
| **average** | **0.0%** | **0.0%** | **0.0%** | **0.0%** | **0.0%** | **0.0%** | **99.2%** | **97.0%** | **97.5%** |
|  |  |  |  |  |  |  |  |  |  |
| Shark | 0.00% | -0.15% | 0.08% | -0.01% | -0.04% | -0.08% | 98.4% | 100.1% | 95.0% |

Table 2: BD rate results for 3-view case under Intra-only

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | 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% | 0.0% | 0.0% | 0.0% | 0.0% | 0.0% | 98.7% | 88.9% | 90.9% |
| Kendo | 0.0% | 0.0% | 0.0% | 0.0% | 0.0% | 0.0% | 99.5% | 85.1% | 95.7% |
| Newspaper\_CC | 0.0% | 0.0% | 0.0% | 0.0% | 0.0% | -0.1% | 98.5% | 89.8% | 97.9% |
| GT\_Fly | 0.0% | 0.0% | 0.0% | 0.0% | -0.1% | -0.1% | 99.3% | 104.0% | 102.5% |
| Poznan\_Hall2 | 0.0% | 0.0% | 0.0% | 0.0% | 0.0% | 0.0% | 99.2% | 94.5% | 100.8% |
| Poznan\_Street | 0.0% | 0.0% | 0.0% | 0.0% | 0.0% | -0.1% | 99.1% | 98.9% | 99.4% |
| Undo\_Dancer | 0.0% | 0.0% | 0.0% | 0.0% | 0.0% | 0.0% | 99.7% | 105.1% | 101.7% |
| 1024x768 | 0.0% | 0.0% | 0.0% | 0.0% | 0.0% | 0.0% | 98.9% | 87.9% | 94.8% |
| 1920x1088 | 0.0% | 0.0% | 0.0% | 0.0% | 0.0% | -0.1% | 99.3% | 100.6% | 101.1% |
| **average** | **0.0%** | **0.0%** | **0.0%** | **0.0%** | **0.0%** | **-0.1%** | **99.1%** | **95.2%** | **98.4%** |
|  |  |  |  |  |  |  |  |  |  |
| Shark | 0.00% | 0.00% | 0.00% | 0.00% | -0.10% | -0.22% | 99.3% | 87.6% | 96.0% |

# References

1. D. Rusanovskyy, K. Müller, A. Vetro, “Common Test Conditions of 3DV Core Experiments,” JCT3V-E1100, Vienna, AT, 27 July – 2 Aug. 2013
2. F. Jäger, “Description of Core Experiment 5 (CE5) on Depth Intra Coding,” JCT3V-E1105, Vienna, AT, 27 July – 2 Aug. 2013.
3. X. Zhao, L. Zhang, Y. Chen and M. Karczewicz “CE6.h related: Simplified DC predictor for depth intra modes,” JCT3V-D0183, Incheon, KR, 20–26 Apr. 2013

# Change in WD

**H.8.4.4.2.11 Depth partition value derivation and assignment process**

– the neighbouring samples p[ x ][ y ], with x = −1, y = −1..nTbS \* 2 − 1 and x = 0..nTbS \* 2 − 1, y = −1,

– a binary array partitionPattern[ x ][ y ], with x, y =0..nTbS − 1, specifying a partitioning of the prediction block in a partition 0 and a partition 1.

– a variable nTbS specifying the transform block size,

– a flag dcOffsetAvailFlag, specifying whether DC Offset values are available,

– a flag intraChainFlag, specifying whether the current intra prediction mode is equal to INTRA\_CHAIN

– the variables dcOffsetP0 and dcOffsetP1, specifying the DC offsets for the block partitions

Output of this process is:

– the predicted samples predSamples[ x ][ y ], with x, y = 0..nTbS − 1.

The variables vertEdgeFlag and horEdgeFlag are derived as specified in the following:

vertEdgeFlag = ( partitionPattern[ 0 ][ 0 ] != partitionPattern[ nTbS − 1 ][ 0 ] ) ? 1 : 0 (H 45)

horEdgeFlag = ( partitionPattern[ 0 ][ 0 ] != partitionPattern[ 0 ][ nTbS − 1 ] ) ? 1 : 0 (H 46)

The variables dcVal0 and dcVal1 are derived as specified in the following:

* If vertEdgeFlag is equal to horEdgeFlag, the following applies:

dcValBR = horEdgeFlag ?

( ( p[ −1 ][ nTbS − 1 ] + p[ nTbS − 1 ][ −1 ] ) >> 1 ) : ( 1 << ( BitDepthY − 1 ) ) (H 47)

dcValLT = ( p[ −1 ][ 0 ] + p[ 0 ][ −1 ] ) >> 1 (H‑48)

* + If horEdgeFlag is equal to 0

vertMidEdgeFlag = ( partitionPattern[ 0 ][ 0 ] != partitionPattern[( nTbS – 1 ) >> 1 ][ 0 ] ) ? 1 : 0

horMidEdgeFlag = ( partitionPattern[ 0 ][ 0 ] != partitionPattern[ 0 ][ ( nTbS – 1 ) >> 1 ] ) ? 1 : 0

* + - If vertMidEdgeFlag is not equal to horMidEdgeFlag

dcValBR = horMidEdgeFlag ? p[ -1 ][ ( nTbS – 1 ) >> 1 ] : p[( nTbS – 1 ) >> 1 ][ -1 ]

* + - Otherwise,

dcValBR = (p[ -1 ][ ( nTbS – 1 ) >> 1 ] + p[( nTbS – 1 ) >> 1 ][ -1 ]) >> 1

* + - If vertMidEdgeFlag is equal to 0 and horMidEdgeFlag is equal to 0
      * if p [ nTbS\*2 − 1 ][ -1 ] is available and p [ -1 ][ nTbS\*2 − 1 ] is available

dcValBR = abs(p [ 0 ][ -1 ] - p [ nTbS\*2 − 1 ][ -1 ]) > abs(p [ -1 ][ 0 ] - p [ -1 ][ nTbS\*2 − 1 ]) ?

p [ nTbS\*2 − 1 ][ -1 ] : p [ -1 ][ nTbS\*2 − 1 ]

* Otherwise ( horEdgeFlag is not equal to vertEdgeFlag), the following applies:
  + 1. dcValBR = horEdgeFlag ? p[ −1 ][ nTbS − 1 ] : p[ nTbS − 1 ][ −1 ] (H‑49)
    2. dcValLT = horEdgeFlag ? p[ ( nTbS – 1 ) >> 1 ][ −1 ] : p[ −1 ][ ( nTbS – 1 ) >> 1 ] (H‑50)

# Patent rights declaration(s)

**HiSilicon Technologies, Santa Clara University and Huawei Technologies 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).**