|  |  |
| --- | --- |
| **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**  10th Meeting: Strasbourg, FR, 18–24 Oct. 2014 | Document: JCT3V-J0065 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| *Title:* | **On  Single Depth Mode Simplification** | | | |
| *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 a simplified candidate pixel selection for Single Depth Mode. It not only saves the candidate list construction process but also reduces the worst case of candidate pixel check from 5 to 2. Experiment result show that this proposal simplified candidate pixel selection simplifies Single Depth Mode with negligible BD-rate change under both All Intra and CTC configuration. Moreover, about 7% decoding time saving is oberserved under All Intra configuration.

# Introduction

For current Single Depth Mode coding, to reconstruct a CU coded as single depth mode, a sample candidate list is first constructed by inserting the sample candidates derived from the neighboring samples of current CU in a predefined order into the candidate list[2]. As shown in 1(each circle represents a sample), the spatial neighboring samples are those five reconstructed samples around the current CU and are inserted into the candidate list in the order: (An/2, Bn/2, B0, A0 and B-1). In the proposed scheme, the size of the sample candidate list is fixed to 2. If empty entry still exists in the candidate list after the pruning and the derivation of candidates from the spatial neighboring samples, offset candidates which are derived by adding offsets to first available sample candidate will be used to fill in those empty entries. After the sample candidate list is constructed, a candidate index is signalled to indicate which sample candidate is selected to fill the current CU. If the current CU is coded with the single depth mode, it is noted that, no residual is further coded.



Figure 1:The spatial sample candidates used to construct the sample candidate list. [1]

However, the candidate list construction process is complex. To make the decoding process more hardware friendly, we propose a new candidate selection algorithm without candidate list construction process.

# Proposed solution

**2.1. Method 1**

We design two pattern, T1 and T2, with two fix candidate positions as shown in Figure 2. The flag single\_sample\_idx specifies which pattern to use. More specifically, at the decoding process, when flag single\_sample\_idx = 0, T1 will be used for decoding; when flag single\_sample\_idx = 1, T2 will be used for decoding. How to derive the pixel from the pattern pair is described with example below:

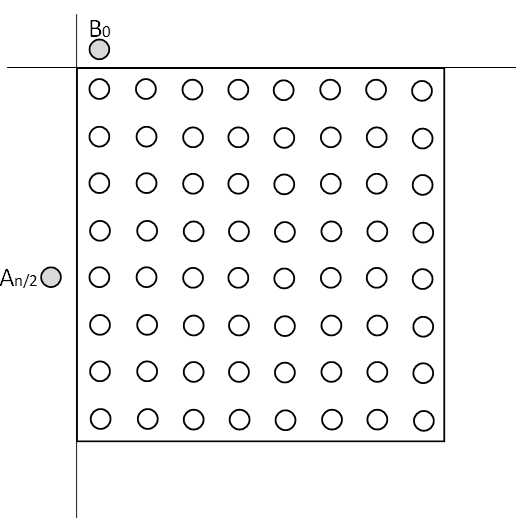
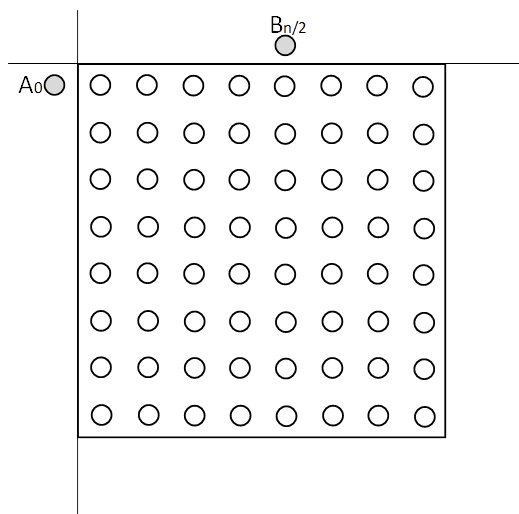
 

Figure 2: (a) Pattern T1 with candidate pair (An/2, B0) (b) Pattern T2 with candidate pair (A0, B n/2).

If the pattern pair is (An/2, B0), we will first check the availability of reference pixel An/2, if An/2 is available, it will be selected to fill the current CU; if An/2 is not available, we will first check the availability of reference pixel B0, if B0 is available, it will be selected to fill the current CU; if both An/2 and B0 are not available, we will use 128 (2bitDepth-1 when bitDepth=8) to fill the current CU. The similar process is also carried out for T2 pattern. In this way, we can derive the sample candidate without candidate list construction and pruning. Moreover, we reduce the worst case of position checking points from 5 points to 2 points. Last not the least, we make the reference pixels selection consistent with Segment-wise Depth Coding DC prediction since B-1 is no long used in our design.

**2.2. Method 2**

In Method 2, we replace reference pixel An/2 with An-1 in T1 and replace reference pixel Bn/2 with Bn-1 as shown in Figure 3. The candidate pair for T1 is (An-1, B0) while the candidate pair for T2 is (A0, B n-1).

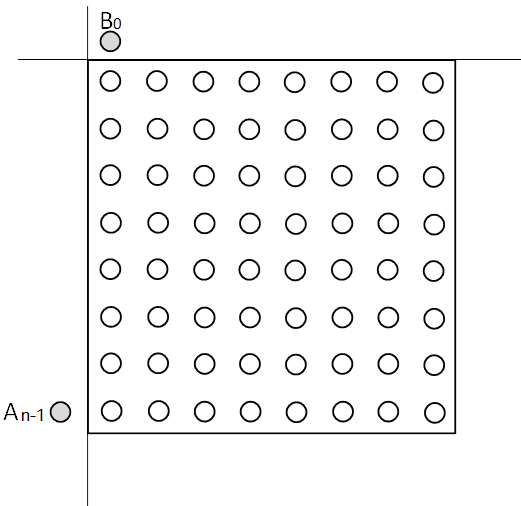
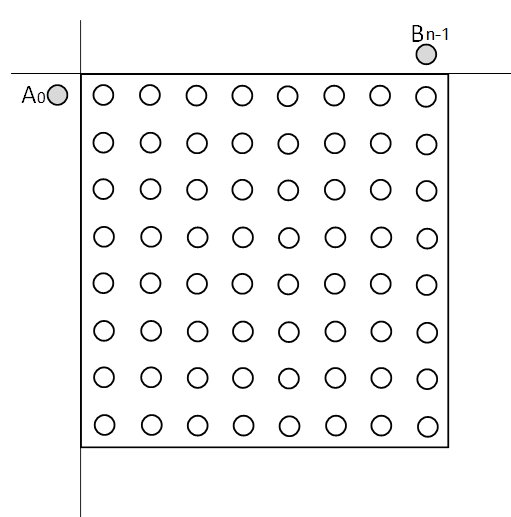
 

Figure 3: (a) Pattern T1 with candidate pair (An-1, B0) (b) Pattern T2 with candidate pair (A0, B n-1).

# Experimental results

The proposed method is implemented on top of HTM-12.0, and simulations were performed under both “common test condition” [3] and All Intra configurations.

**3.1. Method 1**

The performance of Method 1 is summarized in Table 1 and Table 2.

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



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



**3.1. Method 2**

The performance of Method 2 is summarized in Table 3 and Table 4.

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



Table 4: BD rate results for 3-view case under All Intra



# References

1. Y.-W. Chen, J.-L. Lin, Y.-W. Huang, S. Lei, “3D-CE2: Single depth intra mode for 3D-HEVC,” JCT3V-I0095, Sapporo Japan, 3 – 9 July. 2014
2. Y. Chen, G. Tech, K. Wegner, S. Yea “Test Model 9 of 3D-HEVC and MV-HEVC” in JCT3V-I1003, Sapporo Japan, 3 – 9 July. 2014
3. K. Müller, A. Vetro, “Common Test Conditions of 3DV Core Experiments,” JCT3V-G1100, San Jose, USA, January 2014.

# Change in WD

I8.4.4.2.9 Specification of intra prediction mode INTRA\_SINGLE\_SAMPLE

Inputs to this process are:

* a sample location ( xTb, yTb ) specifying the top-left sample of the current block relative to the top‑left sample of the current picture,
* the neighbouring samples p[ x ][ y ], with x = −1, y = −1..nTbS \* 2 − 1 and x = 0..nTbS \* 2 − 1, y = −1,
* a variable nTbS specifying the transform block size.
* a single sample index single\_sample\_idx specifies the single sample candidate index

Output of this process is:

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

The lists predSample[ i ] and availableFlag[ i ] are derived as specified in the following:

* Set testSample equals 1  <<  ( BitDepthY − 1 )
* If single\_sample\_idx equals 0, set nRange as 0 to 1, otherwise (single\_sample\_idx equals 1), set nRange as 2 to 3.

For i in the nRange, inclusive, the following applies:

* + - * + Depending on i, the luma location ( xN, yN ) is specified in Table I‑6.
        + When p[ xN, yN ] is marked as "available for intra prediction", set testSample equals p[ xN, yN ] and break the loop.

The values of the prediction samples predSamples[ x ][ y ], with x, y = 0..nTbS − 1 are derived as follows:

predSamples[ x ][ y ] = testSample (I‑XX)

* ~~For i in the range of 0 to 4, inclusive, the following applies:~~
  + ~~Depending on i, the luma location ( xN, yN ) is specified in Table I‑9.~~
  + ~~The variable availableFlag[ i ] is derived as specified in the following:~~
    - ~~If p[ xN, yN ] is marked as "available for intra prediction", availableFlag[ i ] is set equal to 1.~~
    - ~~Otherwise (p[ xN, yN ] is marked as "not available for intra prediction"), availableFlag[ i ] is set equal to 0.~~
  + ~~The variable predSample[ i ] is set equal to p[ xN, yN ].~~

Table I‑6 – Specification of xN and yN depending on i

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **i** | ~~0~~ 2 | ~~1~~ 0 | ~~2~~ 1 | 3 | 4 |
| **xN** | nTbS >> 1 (Method 1)  nTbS - 1 (Method 2) | 0 | 0 | −1 | ~~−1~~ |
| **yN** | 0 | nTbS >> 1 (Method 1)  nTbS - 1 (Method 2) | −1 | 0 | ~~−1~~ |

~~The list availableFlag[ i ] is modified as specified in the following:~~

* ~~For i in the range of 1 to 4, inclusive, the following applies:~~
  + ~~For j in the range of 0 to ( i − 1 ), inclusive, the following applies:~~
    - 1. ~~availableFlag[ i ] = availableFlag[ i ] &&  
          !( availableFlag[ j ] && ( predSample[ j ] = = predSample[ i ] ) ) (~~I~~‑59)~~

~~The list sampleCandList is derived as specified in the following:~~

~~numCand = 0  
 for( i = 0; i < 5; i++ )  
 if ( availableFlag[ i ] && numCand < 2 )   
 sampleCandList[ numCand++ ] = predSample[ i ]  
 if( numCand = = 0 )  
 sampleCandList[ numCand++ ] = ( 1  <<  ( BitDepth~~~~Y~~~~− 1 ) )  
 if( numCand = = 1 )  
 sampleCandList[ numCand++ ] = sampleCandList[ 0 ] + 1~~

~~The values of the prediction samples predSamples[ x ][ y ], with x, y = 0..nTbS − 1 are derived as follows:~~

* 1. ~~predSamples[ x ][ y ] = sampleCandList[ single\_sample\_idx ] (~~I~~‑60)~~

# 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).**