|  |  |
| --- | --- |
| **Joint Collaborative Team on Video Coding (JCT-VC)**  **of ITU-T SG16 WP3 and ISO/IEC JTC1/SC29/WG11**  7th Meeting: Geneva, Switzerland, 21-30, Nov, 2011. | Document: JCTVC-G463 |

|  |  |  |  |
| --- | --- | --- | --- |
| *Title:* | **Block-based filter adaptation with intra prediction mode and CU depth information** | | |
| *Status:* | Input Document to JCT-VC | | |
| *Purpose:* | Proposal | | |
| *Author(s) or Contact(s):* | Shiqi Wang Siwei Ma Jie Jia Joonyoung Park | Tel: Email: | [sqwang@jdl.ac.cn](mailto:sqwang@jdl.ac.cn) [swma@jdl.ac.cn](mailto:swma@jdl.ac.cn) [jie.jia@lge.com](mailto:jie.jia@lge.com) [jy.park@lge.com](mailto:jy.park@lge.com) |
| *Source:* | Peking University, LG Electronics | | |

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

# Abstract

This contribution presents a simplification of block-based filter adaptation (BA) scheme for intra slice. For each 4x4 block, features are obtained from intra picture prediction mode and CU depth. So that filter adaptation is free from calculation of direction and Laplacian features. The proposed method reports an average 2% decoding time reduction, with 0.1% BD-rate loss for AI structures in comparison with the HM 4.0.

# Introduction

In the 5th JCT-VC meeting, block-based filter adaptation was adopted [1]. In the adopted method, each picture is divided into 4x4 non-overlapping blocks and each 4x4 block is classified based on directional and Laplacian features computed at every pixel. To reduce complexity, in [2], block-based features as in [1] are computed only at pixels subsampled by 2 in both vertical and horizontal directions. In this contribution, the block based features are directly obtained from the intra prediction direction, CU depth and PU partition mode, which further simplify the classification process in ALF. All the other aspects of ALF remain the same as in HM 4.0.

# 4x4 block classification

In HEVC, to classify a 4x4 block with pixels { *X*(*i,j*) | *i*=0,1,2,3 ; *j*=0,1,2,3}, the pixels are subsampled by 2 in both vertical and horizontal directions, as shown in Fig. 1. Then directional and Laplacian features *H*(*i,j*), *V*(*i,j*), and *L*(*i,j*) are only calculated at *i*=0,2 ; *j*=0,2. In this way, the computations are saved at both the encoder and the decoder.



**Figure. 1 Feature computation for block-based filter adaptation process [2].**

Although this method can significantly decrease the complexity for variance calculation, it still needs to calculate the directional and Laplacian features for the sub-sampled pixels. Generally, the intra prediction direction can reflect the direction information and the CU depth can reflect the smoothness of the input frame, which corresponds to the direction and Laplacian features. Therefore, in this contribution, we further simplify this process by incorporating the intra prediction direction, CU depth and PU partition mode.

Firstly, we classify all intra prediction directions into five groups as presented in Table 1 (logical prediction order in Table 1 is illustrated in Fig.1). This grouping approach is based on the assumption that DC and Planar (group 0), VER+1, …, and VER+8 (group 1), VER-8, … and VER (group 2), HOR-8, …, and HOR (group 3), HOR+1, …, and HOR+8 (group 4) respectively have similar directional feature.

**Table 1. Filter groups.**

|  |  |  |
| --- | --- | --- |
| Filter group number (Num) | Num\_Value | Logical prediction order |
| 0 | 0 | 0, 34 |
| 1 | 3 | 10~17 |
| 2 | 6 | 1~9 |
| 3 | 9 | 18~24 |
| 4 | 12 | 25~33 |

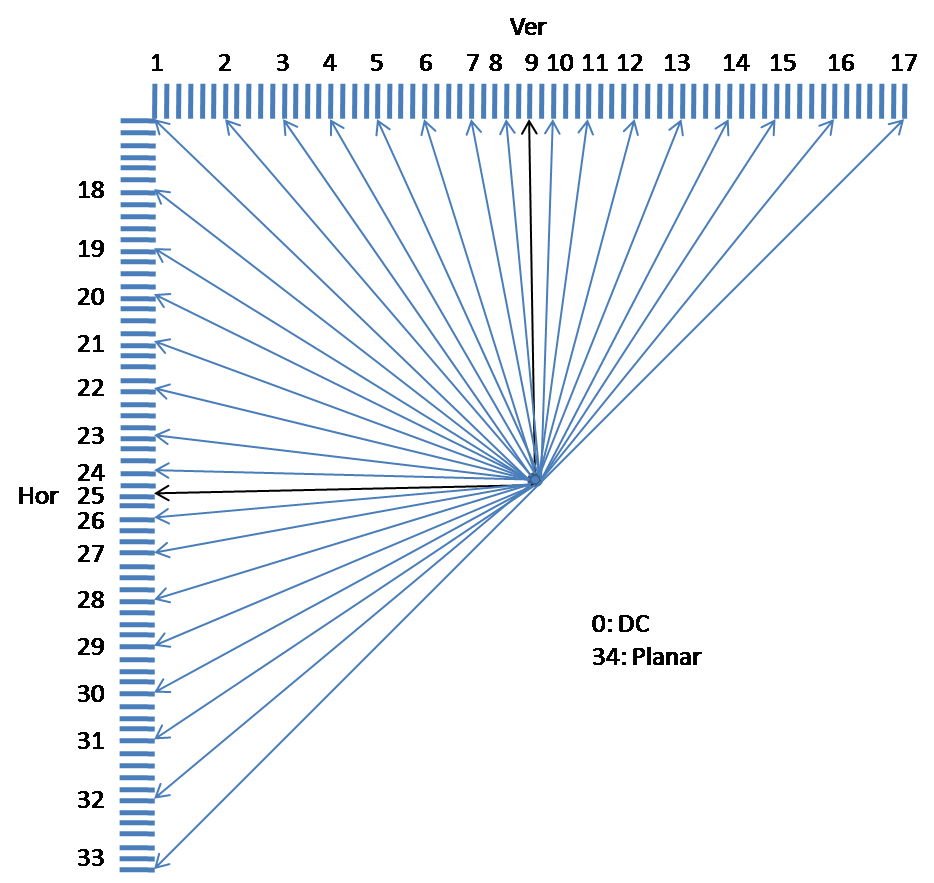


Fig.1. Logical order of all prediction directions.

The Laplacian feature is estimated from the CU depth and the PU partition mode. Suppose CU depth of the current 4 x 4 block is *depth*, the Laplacian feature is calculated as:

. (1)

We further divide *L* into three groups:

. (2)

Since adjacent filter types should have similar properties for further filter grouping, the classified type for the current 4x4 block is finally obtained by

, (3)

where *Num* and *Num\_Value* are defined in Table 1.

In HM 4.0, region-adaptive (RA) ALF require fewer computations than BA ALF, especially for the decoder, therefore it is preferred by HM4.0 to some extent. Consequently, in the selection between RA ALF and BA ALF, the RD (rate distortion) cost of the former first subtracts a positive value *adExtraCostReduction* and then is compared with the RD cost of the latter, where

. (4)

In this contribution, considering that computation of RA ALF is significantly reduced at the decoder, we set *adExtraCostReduction*=0 to allow a RD optimized selection between RA and BA.

# Analysis on complexity reduction

The computation analysis of the proposed algorithm is shown in Table 2. As can be seen, 3 operations are needed for each 4x4 block. If two or more adjacent 4 x 4 blocks share the same intra prediction direction, CU depth and PU partition mode, the computation can be further reduced. This is significantly reduced from 48 operations (i.e., reduce the operations by 93%) per 4x4 block as in HM 4.0 [2] shown in Table 3.

**Table 2. Computation analysis of the proposed algorithm.**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  | Grouping | *L* | Class(*L*) | Filt\_Num | Total |
| operations | Add |  |  |  | 1 | 1 |
| comparison |  | 1 |  | 1 | 2 |
| LUT access |  | 1 (size: 35) | 1 (size:5) | 1 (size:5) | 1 (size:5) | 4 |

**Table 3. Computation analysis of [2].**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  | *H*(*i,j*), *V*(*i,j*) | *H’B*, *V’B* | *L’B* | Classification | Total |
| operations | Adds | 16 | 6 | 1 | 3 | 26 |
| Abs | 8 |  |  |  | 8 |
| comparison |  |  |  | 4 | 4 |
| shift | 4 |  | 1 | 5 | 10 |
| LUT access |  |  |  |  | 1 (size: 16) | 1 |

Furthermore, from the view of memory access, accessing reconstructed pixel values is not required by the proposed method. In our method, for each 4 x 4 block, we only need to access the intra prediction direction, CU depth and PU partition mode.

# Experimental results

Experiments are conducted on HM4.0. The testing environment is: Intel (R) Xeon (R) CPU X5670@2.93GHz, 6cores, Memory 12GB，Windows 7, 32 bit compiler. Since the proposed method is designed for intra picture coding, experimental results for all intra cases are presented, as seen in Table 4. Simulation results report that the proposed method achieves 2% reduction on decoding time with 0.1% BD-rate loss. Regarding the encoding process, block classification only takes tiny part in the total computation of ALF process which includes time consuming filter coefficients training and filter selection. Therefore, the proposed method does not obviously reduce computation of ALF process. This is reason why reduction in encoding time is not clearly observed from the simulation results.

**Table 4. Comparison with HM4.0**

|  |  |  |  |
| --- | --- | --- | --- |
|  | All Intra HE | | |
| Y BD-rate | U BD-rate | V BD-rate |
| Class A | 0.26% | 0.03% | 0.03% |
| Class B | 0.10% | 0.00% | 0.00% |
| Class C | 0.07% | 0.03% | 0.04% |
| Class D | 0.05% | -0.02% | -0.02% |
| Class E | 0.13% | -0.01% | -0.02% |
| All | 0.12% | 0.01% | 0.01% |
| Enc (%) | 100% | | |
| Dec (%) | 98% | | |

# Changes in Working Draft Text

Changes of working draft are in subsection 8.6.3.3. Please see the red highlighted part in the following.

The filter index array fIdx is specified in the follows:

* When alf\_region\_adaptation\_flag is equal to 1, the following ordered steps apply.
  + … …
* Otherwise (alf\_region\_flag is equal to 0), the following ordered steps apply.
  + If slice\_type is not equal to I

1. The variables varTempH[ x ][ y ], varTempV[ x ][ y ] and varTemp1[ x ][ y ] with x, y = 0, 2, 4,...(nS) is derived as

varTempH[ x ][ y ] = | ( s’’[ xC+x, yC+y ] << 1 ) – s’’[ xC+x-1, yC+y ] – s’’[ xC+x+1, yC+y ] | (8‑474)  
 varTempV[ x ][ y ] = | ( s’’[ xC+x, yC+y ] << 1 ) – s’’[ xC+x, yC+y-1 ] – s’’[ xC+x, yC+y+1 ] | (8‑474)

1. The variables varTempH1[ x, y ], varTempV1[ x, y ], and varTemp3[ x, y ] with x, y = 0..( (nS) – 1 )>>2 are derived as

varTempH1[ x ][ y ] = ΣiΣj varTempH[ (x << 2 ) + i ][ (y << 2) + j ] with i, j = 0, 2 (8‑474)  
 varTempV1[ x ][ y ] = ΣiΣj varTempV[ (x << 2 ) + i ][ (y << 2) + j ] with i, j = 0, 2 (8‑474)  
 varTemp3[ x ][ y ] = ( varTempH1[ x ][ y ] + varTempV1[ x ][ y ] ) >> 2 (8‑474)

1. The variable direction is derived as
   * + If varTempV1[ x >> 2 ][ y >> 2 ] is greater than varTempH1[ x >> 2 ][ y >> 2 ] << 1,

direction = 1

* + - Otherwise, if varTempH1[ x >> 2 ][ y >> 2 ] is greater than varTempV1[ x >> 2 ][ y >> 2 ] << 1,

direction = 2

* + - Otherwise,

direction = 0

1. The variable avgvar is derived as

varTab[16] = { 0, 1, 2, 2, 2, 3, 3, 3, 3, 3, 4, 4, 4, 4, 4, 4 } (8‑475)  
 avgVar = Clip3( 0, 15, (varTemp3[ x >> 2 ][ y >> 2 ] \* 1024 ) >> (3 + BitDepthY) ) (8‑475)

1. The filter index fIdx[ x, y ] with x, y = 0..(nS)-1 is derived as

fIdx[ x ][ y ] = Clip3( 0, 4, varTab[avgVar] ) + 5 \* direction (8‑504)

* + Otherwise

1. The variable IntraPredictionGroup is derived as

IntraModeTable[35] = {0,6,12,0,6,6,3,3,9,12,12,6,6,3,3,9,9,12,12,6,6,6,6,3,3,3,3,9,9,9,9,12,12,12,12}

IntraPredictionGroupValue = IntraModeTable[intraPredMode] (8‑xxx)

1. The variables VarTemp isderived as
   * + If PartMode is equal to PART\_NxN   
        VarTemp = cuDepth+1 (8‑xxx)
     + OtherWise

VarTemp = cuDepth (8‑xxx)

1. The variable VarTempGroup is derived as
   * + If VarTemp is equal to 0, 1   
        VarTempGroup = 0
     + Else if VarTemp is equal to 2, 3

VarTempGroup = 1

* + - OtherWise

VarTempGroup = 2

1. The filter index fIdx[x][y] with x,y= 0..(nS)-1 is derived as
   * + If IntraPredictionGroupValue is equal to 0, 6, 12

fIdx[ x ][ y ] = IntraPredictionGroupValue + VarTempGroup (8‑xxx)

* + - OtherWise

fIdx[ x ][ y ] = IntraPredictionGroupValue + 2 – VarTempGroup (8‑xxx)

# Conclusions

This contribution proposes a simplified block-based filter adaption method which reduces operations of the block classification from 48 to 3 for intra slice. The proposed method reports 2% decoding time reduction with 0.1% BD-rate loss for All intra HE case compared with HM 4.0. Meanwhile, no syntax change is required. It is recommended to incorporate proposed tools into HM test model.

# References

1. “CE8 Subtest 2: Block based adaptive loop filter”, JCTVC-E323, Geneva, Switzerland, Mar. 2011
2. “CE8 Subtest 1: Block-based filter adaptation with features on subset of pixels”, JCTVC-F301, Torino, Italy, 14-22, July. 2011

**Patent rights declaration(s)**

**LG Electronics / LG Electronics (china) R&D Center and Peking University** **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).**