|  |  |
| --- | --- |
| **Joint Collaborative Team on Video Coding (JCT-VC)**  **of ITU-T SG16 WP3 and ISO/IEC JTC1/SC29/WG11**  6th Meeting: Torino, 14-22 July, 2011 | Document: JCTVC-F242 |

|  |  |  |  |
| --- | --- | --- | --- |
| *Title:* | **Bilinear chroma interpolation for small block sizes** | | |
| *Status:* | Input Document to JCT-VC | | |
| *Purpose:* | Proposal | | |
| *Author(s) or Contact(s):* | Kemal Ugur and Jani Lainema Visiokatu 1, Tampere, Finland   Kenji Kondo and Teruhiko Suzuki  Gate City Osaki West 12F 1-11-1 Osaki, Shinagawa-ku Tokyo, 141-0032, JAPAN | Tel: Email:  Tel: Email: | +358 50 4860857 [kemal.ugur@nokia.com](mailto:kemal.ugur@nokia.com) [jani.lainema@nokia.com](mailto:jani.lainema@nokia.com)  +81-3-5435-3308 [kenji.kondo@jp.sony.com](mailto:kenji.kondo@jp.sony.com)  [teruhikos@jp.sony.com](mailto:teruhikos@jp.sony.com) |
| *Source:* | Nokia, Sony | | |

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

# Abstract

In Core Experiment-3, various tools related to luma interpolation filters are proposed to reduce the motion compensation memory bandwidth. Some of the proposed tools use shorter-tap length filters for bi-predicted blocks and small PU sizes. In this contribution, an extension of this idea is proposed for the chroma component as well. The proposed idea utilizes a bilinear filter for chroma blocks instead of the current 4-tap filter when the PU size is smaller than 8x8 (8x4, 4x8 and 4x4). It is reported that using a filter with fewer taps, the worst-case complexity of the current interpolation filter in HM is reduced. Experimental results show that the impact on coding efficiency is negligible.

# Introduction

Current HM3.0 defines a 4-tap separable filter to interpolate chroma samples. For small motion blocks, 4-tap filter brings a significant overhead in terms of memory bandwidth. For example if the PU size is 4x4, then a 2x2 chroma sample block needs to be interpolated. Assuming the MV points one of the sub-pixels that are not horizontally or vertically aligned with the integer samples, decoder need to load (2+3).(2 + 3) = 25 samples (which is around 6 times more than the prediction block).

In order to reduce the memory bandwidth overhead we propose the following:

* For bi-predicted blocks of PU size < 8x8, use bi-linear filter for chroma samples
* For other cases, use 4-tap chroma as defined in HM3.0.

The main reason to limit the usage of bilinear filter for small block sizes is that memory bandwidth is more problematic when the block size is small. For example, for a 16x16 block size, decoder needs to load a 19x19 block (which is 41% more of the prediction block), but for a 2x2 block size, decoder needs to load a 5x5 block (which is 625% more of the prediction block).

For a PU size of 4x4 (where a 2x2 chroma block needs to be interpolated), the benefit of proposed approach could be illustrated as follows:

(a)

(b)

Figure 1 Illustration of required memory bandwidth for a 4x4 PU and MV points between integer samples. 4-tap filter (a) needs to load 25 samples, whereas bilinear filter (b) needs to load 9 samples. Circles indicate sub-pixel samples to be interpolated and dots indicate integer reference samples.

It should be noted that some variations of the algorithm could also be thought and are presented in other contributions (e.g. using 2-tap filter for chroma, using smaller filter for both bi and uni prediction).

# Experimental Results

Two sets of experimental results are presented (i) Bilinear filter for chroma for PU<8x8 and bi-predicted and (ii) Bilinear filter for chroma for PU<8x8 bi and uni-predicted

**Bilinear filter for chroma for PU<8x8 and bi-predicted blocks**

This variation of the algorithm brings 0.01% change in coding efficiency on average.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  | Random access |  |  | Random access LoCo |  |
| Y BD-rate | U BD-rate | V BD-rate | Y BD-rate | U BD-rate | V BD-rate |
| Class A | 0.0 | -0.1 | -0.2 | 0.0 | 0.1 | 0.2 |
| Class B | 0.0 | 0.1 | 0.1 | 0.0 | 0.2 | 0.2 |
| Class C | 0.0 | 0.2 | 0.1 | 0.0 | 0.2 | 0.1 |
| Class D | 0.0 | 0.4 | 0.4 | 0.0 | 0.6 | 0.6 |
| Class E |  |  |  |  |  |  |
| All | 0.00 | 0.1 | 0.1 | 0.01 | 0.2 | 0.3 |
| Enc Time[%] | 96% | | | 100% | | |
| Dec Time[%] | 100% | | | 98% | | |
|  |  |  |  |  |  |  |
|  | Low delay | | | Low delay LoCo | | |
|  | Y BD-rate | U BD-rate | V BD-rate | Y BD-rate | U BD-rate | V BD-rate |
| Class A |  |  |  |  |  |  |
| Class B | 0.0 | 0.2 | 0.3 | 0.0 | 0.1 | -0.1 |
| Class C | 0.0 | 0.0 | -0.1 | 0.0 | 0.2 | 0.0 |
| Class D | 0.1 | 0.5 | 0.7 | 0.0 | 0.7 | 0.8 |
| Class E | 0.0 | -0.1 | -0.2 | 0.0 | -0.3 | -0.6 |
| All | 0.01 | 0.2 | 0.2 | 0.02 | 0.2 | 0.0 |
| Enc Time[%] | 104% | | | 98% | | |
| Dec Time[%] | 102% | | | 99% | | |

**Bilinear filter for chroma for PU<=8x8 and bi-predicted blocks**

This variation of the algorithm brings 0.06% change in coding efficiency on average and reduces the memory bandwidth significantly especially in higher levels.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  | Random access | |  | Random access LoCo | |
|  | Y BD-rate | U BD-rate | V BD-rate | Y BD-rate | U BD-rate | V BD-rate |
| Class A | 0.0 | 0.2 | 0.3 | 0.0 | 0.3 | 0.6 |
| Class B | 0.0 | 0.3 | 0.4 | 0.0 | 0.6 | 0.6 |
| Class C | 0.0 | 0.6 | 0.5 | 0.1 | 0.7 | 0.8 |
| Class D | 0.1 | 1.1 | 1.1 | 0.1 | 1.7 | 1.8 |
| Class E |  |  |  |  |  |  |
| All | 0.0 | 0.5 | 0.6 | 0.1 | 0.8 | 0.9 |
| Enc Time[%] | 97% |  |  | 97% |  |  |
| Dec Time[%] | 98% |  |  | 101% |  |  |
|  |  |  |  |  |  |  |
|  | Low delay |  |  | Low delay LoCo | |  |
|  | Y BD-rate | U BD-rate | V BD-rate | Y BD-rate | U BD-rate | V BD-rate |
| Class A |  |  |  |  |  |  |
| Class B | 0.0 | 0.4 | 0.2 | 0.1 | 0.4 | 0.3 |
| Class C | 0.0 | 0.0 | 0.2 | 0.1 | 0.6 | 0.5 |
| Class D | 0.1 | 2.0 | 1.9 | 0.2 | 2.3 | 2.2 |
| Class E | 0.0 | -0.2 | -0.2 | 0.0 | -1.2 | -1.0 |
| All | 0.0 | 0.6 | 0.6 | 0.1 | 0.6 | 0.6 |
| Enc Time[%] | 105% |  |  | 101% |  |  |
| Dec Time[%] | 101% |  |  | 105% |  |  |

**Bilinear filter for chroma for PU<8x8, bi and uni-predicted blocks**

This variation reduced the memory bandwidth more with 0.07% change in coding efficiency on average.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  | Random access |  |  | Random access LC |  |
| Y BD-rate | U BD-rate | V BD-rate | Y BD-rate | U BD-rate | V BD-rate |
| Class A | -0.1 | -0.3 | 0.0 | 0.0 | 0.0 | 0.3 |
| Class B | 0.0 | 0.2 | 0.2 | 0.0 | 0.5 | 0.5 |
| Class C | 0.0 | 0.4 | 0.3 | 0.0 | 0.5 | 0.5 |
| Class D | 0.1 | 0.9 | 0.7 | 0.1 | 1.4 | 1.5 |
| Class E |  |  |  |  |  |  |
| All | 0.01 | 0.3 | 0.3 | 0.04 | 0.6 | 0.7 |
| Enc Time[%] |  | | |  | | |
| Dec Time[%] |  | | |  | | |
|  |  |  |  |  |  |  |
|  | Low delay | | | Low delay LC | | |
|  | Y BD-rate | U BD-rate | V BD-rate | Y BD-rate | U BD-rate | V BD-rate |
| Class A |  |  |  |  |  |  |
| Class B | 0.0 | 0.4 | 0.4 | 0.1 | 0.2 | 0.1 |
| Class C | 0.0 | 0.3 | 0.5 | 0.1 | 0.7 | 0.3 |
| Class D | 0.1 | 1.3 | 1.5 | 0.2 | 1.6 | 1.6 |
| Class E | 0.1 | 0.2 | 0.1 | -0.1 | -0.3 | -0.2 |
| All | 0.06 | 0.6 | 0.6 | 0.09 | 0.6 | 0.5 |
| Enc Time[%] |  | | |  | | |
| Dec Time[%] |  | | |  | | |
|  |  |  |  |  |  |  |
|  | Low delay (P) | | | Low delay LC (P) | | |
|  | Y BD-rate | U BD-rate | V BD-rate | Y BD-rate | U BD-rate | V BD-rate |
| Class A |  |  |  |  |  |  |
| Class B | 0.1 | 0.7 | 0.4 | 0.1 | 0.3 | 0.6 |
| Class C | 0.1 | 0.6 | 0.5 | 0.1 | 0.7 | 0.4 |
| Class D | 0.2 | 2.4 | 2.4 | 0.3 | 2.3 | 2.0 |
| Class E | -0.1 | 0.0 | -0.9 | 0.0 | -0.9 | -0.4 |
| All | 0.07 | 1.0 | 0.7 | 0.12 | 0.7 | 0.7 |
| Enc Time[%] |  | | |  | | |
| Dec Time[%] |  | | |  | | |

# Memory bandwidth analysis

Below table compares the memory bandwidth reduction by using bilinear filter instead of 4-tap filter for small block sizes. It can be seen that the worst case memory bandwidth is significantly reduced by using a simpler filter for chroma when the block size is small.

|  |  |  |  |
| --- | --- | --- | --- |
| PU Size | 4-tap filter | Bilinear filter | Percent reduction in memory bandwidth |
| 4x4 | 9x9 = 81 | 5x5 = 25 | 69% |
| 4x8 | 9x13 = 117 | 5x9 = 45 | 61% |
| 8x4 | 13x9 = 117 | 9x5 = 45 | 61% |
| 8x8 | 13x13 = 169 | 9x9 = 81 | 52% |

# Patent rights declaration(s)

**Nokia may have IPR 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).**

**Sony Corporation may have IPR 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).**