|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| |  |  | | --- | --- | | **Joint Collaborative Team on Video Coding (JCT-VC)**  **of ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11**  15th Meeting: Geneva, CH, 23 Oct. – 1 Nov. 2013 | Document: JCTVC-Oxxxx | | Document: JCTVC-O0277 |  |

|  |  |  |  |
| --- | --- | --- | --- |
| *Title:* | **RExt: On intra block copy motion vector coding** | | |
| *Status:* | Input Document to JCT-VC | | |
| *Purpose:* | Proposal | | |
| *Author(s) or Contact(s):* | Guoxin Jin, Ankur Saxena and  Felix Fernandes | Email: Tel: | [guoxin.jin@sta.samsung.com](mailto:guoxin.jin@sta.samsung.com),  [asaxena@sta.samsung.com](mailto:asaxena@sta.samsung.com),  [ffernandes@sta.samsung.com](mailto:ffernandes@sta.samsung.com)  1-972-761-7761 |
| *Source:* | Samsung Electronics, Co., Ltd. | | |

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

# Abstract

Improved Intra Block Copy mode and 2-D Intra Block Copy mode were proposed in JCTVC-N0254 and JCTVC-N0256 respectively. Intra Block Copy mode can significantly improve the coding performance for screen content. For coding the motion vector differences in Intra Block copy, Exponential-Golomb code was used. However, Exponential-Golomb code is not optimal for coding such motion vector differences. In this contribution, a simplified fixed length code instead of Exponential-Golomb code is first presented. Next, a Huffman code to further improve the performance is also presented. Moreover, one of the possible reasons why in JCTVC-N0256 Exponential –Golomb code achieves a 1% coding gain is also explained. It is asserted that applying the fixed length code in lossless setting leads to average bit-rate savings of 0.3 % for All Intra, 0.2% for Random Access, and 0.2% for Low Delay settings for YUV444 screen content coding sequences. For lossy setting, the gains are 1% gains for All Intra Main-tier, 0.8% for Random Access Main-tier and 0.6 % for Low delay Main-tier of YUV444 screen content. For the Huffman coding based technique, the gain for All Intra Main-tier SC YUV444 is 1.5%.

# Introduction

Intra Block Copy [1, 2] is very helpful in screen content, since screen content is likely to have similar regions locally. After the motion vector is predicted, the motion vector difference for the Intra Block Copy must be encoded. Exponential-Golomb code proposed in [2] showed more than 1% BD-Rate improvement for screen content.

However, Exponential-Golomb code may not be ideal, as the statistics for the motion vector differences do not necessarily follow an exponential distribution on which Exponential- Golomb code can work efficiently. We plot the histogram of Intra Block Copy motion vector differences for all the test sequences in Fig. 1. The range of horizontal motion vector is between -120 to +56 and the range of vertical motion vector is between -56 to +56.



**(a) Horizontal MV value (b) Vertical MV value**

**Figure 1: Statistics (pdf) of 2-D Intra Block Copy Motion Vectors.**

Typical Exponential-Golomb code from HM12.0+RExt4.1 are next shown in Table 1.

**Table 1:** Typical IntraBC motion vector difference codes in HM12+RExt4.1

|  |  |
| --- | --- |
| Value | Binary Code |
| 0 | 0 |
| 1 | 100 |
| -1 | 101 |
| 2 | 11000 |
| -2 | 11001 |

From Table 1, in general, the Exponential-Golomb code will work efficiently if the value of motion vector difference is symmetric to 0, and the probability density decreasing exponentially. Comparing to Fig. 1, this is not the case in general. Motion vector differences are not symmetric, and for a period of 8, there is a strong peak compared to the neighboring values, which violates the decreasing assumption. As a result, Exponential-Golomb code won’t work well for coding the Intra Block Copy motion vector differences, especially for the Horizontal motion vectors.

# Simplified Fixed length Coding For Intra Block Copy Motion Vectors

Based on the statistics in Fig. 1, and the implementation of Exponential-Golomb code in HM12.0+RExt4.1, the average length for the Exponential-Golomb code can be computed as 9.9 bits for the horizontal motion vector difference, and 6.0 bits for the vertical motion vector difference. However, the entropy for the distribution is around 6.23 bits/symbol and 4.5 bits/symbols for the horizontal and vertical motion vector differences respectively. Thus, there is a huge gap between the Exponential-Golomb code performance as compared to a code which can approach the entropy. Therefore, it is necessary to use a better coding technique.

Surprisingly a very simple fixed length coding scheme would help. From the distribution in Fig 1(a), the range of horizontal motion vector differences is between -120 to 56 (integers). Naively, 8 bits integer can cover all the values. However, the average length of Exponential-Golomb code used in HM12.0+RExt4.1 for horizontal motion vector difference is 9.9 bits. The contribution proposes the following fixed length encoding scheme.

The proposed coding scheme simplifies upon HM12.0+RExt4.1. In HM12.0+RExt4.1, the motion vector difference coder assumes value 0 happens more frequently than any other cases, so one bit is used to represent value 0. Moreover, magnitude 1 cases are also separately coded. In this contribution, instead of coding the magnitude of motion vector other than 0 and 1 by Exponential-Golomb code, this contribution uses L bit fixed length code for all the value other than 0. Table 2 shows the algorithm.

**Table 2:** IntraBC motion vector difference fixed length coding

|  |
| --- |
| Encoding  Given IntraBC motion vector difference x:  if x is 0  encode 0 and return  else  encode 1  use L bit fixed length code for abs(x)  encode sign |
| Decoding  read a bit  if bit is 0  return 0  else  value = read L bits and convert to unsigned integer  read a bit  if bit is 1 return (-value)  else return (value) |

Using algorithm of Table 2 and the distribution in Fig. 1, Table 3 shows the average code length for Intra Block Copy motion vectors.

**Table 3:** Average code word length for Intra BC MVd

|  |  |  |  |
| --- | --- | --- | --- |
| bits per Symbol | Proposed | Exp-Golomb | Entropy |
| Horizontal MVd | 8.04 (L=8) | 9.90 | 6.23 |
| Vertical MVd | 5.14 (L=7) | 5.95 | 4.50 |

# Reason why in JCTVC-N0256 Exponential-Golomb code gain vs. fixed length code

For the intra block copy motion vector differences in vertical direction, the probability to have value 0 is around 40% of entire distribution from Fig. 1(b). However, in Fig 1(a), 0 values are just about 10%. The distribution of vertical motion vector differences in Fig 1(b) is suitable to separate 0 and non-zero cases. This effect can be clearly seen in Table 4, where naïve 8 bit fixed length code is applied to horizontal motion vector differences and 7 bit fixed length code is applied to vertical motion vector differences. By separating the 0 value and non-zero case, average code word length for vertical motion vector differences drops 5.95 in Exponential-Golomb code. It is asserted that the gain in JCTVC-N0256 actually comes from separating 0 and nonzero, and not from the Exponential-Golomb code.

**Table 4: Average code word length for Intra BC motion vectors**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| bits per Symbol | Naïve fixed | proposed | Exp-Golomb | Entropy |
| Horizontal MV | 8 | 8.04 | 9.90 | 6.23 |
| Vertical MV | 7 | 5.14 | 5.95 | 4.50 |

# Extension using Huffman Coding

To further decrease the average code word length, we next propose Huffman coding. For simplicity, like in section 2, one can still separate 0 and the other non-zero case. So first, one bit is used to indicate 0 or not. Then for the value not equal to 0, Huffman code is used. Huffman codebook can be built by using the distribution of the magnitude of horizontal and vertical motion vector differences in IntraBC in Fig. 1. Finally, the sign is coded by 1 bit. The average code length for the proposed Huffman code is summarized in Table 5.

**Table 5:** Average code word length for Intra BC MVd

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| bits per Symbol | Modified Fix | Exp-Golomb | Huffman | Entropy |
| Horizontal MVd | 8.04 | 9.90 | 7.73 | 6.23 |
| Vertical MVd | 5.14 | 5.95 | 4.61 | 4.50 |

# Experimental results

In this section, coding results for fixed length code and Huffman code intra block copy motion vector differences are presented for the test conditions stipulated in RCE3 core experiment description [3] for lossless and lossy settings. The anchor is HM 12.0+RExt-4.1 in lossless and lossy settings. The following tables present the coding efficiency with the simplified fixed length coding, and Huffman coding:

**Table 6:** Fixed length IntraBC MVd coding in lossless setting

**Table 7**: Fixed length IntraBC MVd coding in lossy setting

**Table 8:** Huffman coding IntraBC MVd in lossless setting

**Table 9**: Huffman coding IntraBC MVd in lossy setting

**Table 6:** Fixed length IntraBC MV coding in lossless setting (Anchor is HM12.0-RExt-4.1)







**Table 7:** Fixed length IntraBC MV coding in lossy setting (Anchor is HM12.0-RExt-4.1)



**Table 8:** Huffman Coding IntraBC MV coding in lossless setting (Anchor is HM12.0-RExt-4.1)







**Table 9:** Huffman Coding IntraBC MV in lossy setting (Anchor is HM12.0-RExt-4.1)



# Complexity

At both encoder and decoder side, using fixed length IntraBC MVd coding reduces the complexity. No mathematical computations are required compares to Exp-Golomb code in HM. When the proposed Huffman coding will be used, both encoder and decoder side will require storage of a pre-computed Huffman codebook. If the table is stored as array, the encoder requires O(1) to find the code word and the decoder requires O(N) to find the code word where N is the size of codebook. If the table is stored as a tree, the decoder could reduce the decoding time to O(logN) which is the same as Exp-Golomb code.

# Conclusion

The proposed fixed length coding achieves 1 % gain for SC YUV 444 coding which is actually a simplification of the existing Exponential-Golomb coding scheme in HM. Next the Huffman coding scheme provides 1.5% BD-Rate gain for SC YUV 444 in Lossy setting. We therefore recommend adopting this proposal in committee draft of HEVC range extensions.

# References

1. C. Pang, J. Sole, L. Guo, M. Karczewicz, R. Joshi, “Non-RCE3: Pipeline Friendly Intra Motion Compensation”, JCTVC-N0254, Vienna, Austria, July 2013
2. C. Pang, J. Sole, L. Guo, M. Karczewicz, R. Joshi, “Non-RCE3: Intra Motion Compensation with 2-D MVs”, JCTVC-N0256, Vienna, Austria, July 2013techniques,” JCTVC-N1123, Vienna, Austria, July 2013.
3. A. Saxena, D. Kwon, M. Naccari and C. Pang, “HEVC Range Extensions Core Experiment 3 (RCE3): Intra Prediction techniques,” JCTVC-N1123, Vienna, Austria, July 2013.

# Patent rights declaration(s)

**Samsung Electronics Co., Ltd., 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).**

# Annex Huffman Code Book

Table A‑1 – Huffman Codebook for IntraBC Mvd

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Horizontal Mvd | Horizontal Mvd Huffman Code | Code Len. | Vertical Mvd | Vertical Mvd Huffman Code | Code Len. |
| -120 | 1010010001101 | 13 | -56 | 111011111 | 9 |
| -119 | 1011100001111 | 13 | -55 | 111011011111 | 12 |
| -118 | 1011100001101 | 13 | -54 | 10001011011 | 11 |
| -117 | 10010001101001 | 14 | -53 | 101110001111 | 12 |
| -116 | 1110100100111 | 13 | -52 | 10111000101 | 11 |
| -115 | 1110100100101 | 13 | -51 | 11101101101 | 11 |
| -114 | 1010010001111 | 13 | -50 | 101110001101 | 12 |
| -113 | 10010001101011 | 14 | -49 | 111011011101 | 12 |
| -112 | 11100100101 | 11 | -48 | 10011111 | 8 |
| -111 | 1000101000001 | 13 | -47 | 111011001 | 9 |
| -110 | 111010010001 | 12 | -46 | 10001011001 | 11 |
| -109 | 1001000110111 | 13 | -45 | 1011100001 | 10 |
| -108 | 100000000011 | 12 | -44 | 1000101011 | 10 |
| -107 | 1000101000011 | 13 | -43 | 11010010101 | 11 |
| -106 | 111001000011 | 12 | -42 | 1101001001 | 10 |
| -105 | 111001000001 | 12 | -41 | 11010010111 | 11 |
| -104 | 11101001011 | 11 | -40 | 10100001 | 8 |
| -103 | 101110000101 | 12 | -39 | 1000101111 | 10 |
| -102 | 100000000001 | 12 | -38 | 1011101101 | 10 |
| -101 | 101110000011 | 12 | -37 | 100111001 | 9 |
| -100 | 11100100011 | 11 | -36 | 101000111 | 9 |
| -99 | 101110000001 | 12 | -35 | 1110110101 | 10 |
| -98 | 10010001001 | 11 | -34 | 100111011 | 9 |
| -97 | 101001000101 | 12 | -33 | 1011101111 | 10 |
| -96 | 10010001011 | 11 | -32 | 1011011 | 7 |
| -95 | 11100100111 | 11 | -31 | 101110101 | 9 |
| -94 | 10011000001 | 11 | -30 | 11001011 | 8 |
| -93 | 100010100011 | 12 | -29 | 1000101001 | 10 |
| -92 | 10101010111 | 11 | -28 | 1011111 | 7 |
| -91 | 10100100101 | 11 | -27 | 111011101 | 9 |
| -90 | 10100100001 | 11 | -26 | 11001001 | 8 |
| -89 | 100100011001 | 12 | -25 | 110100111 | 9 |
| -88 | 10000000011 | 11 | -24 | 1010111 | 7 |
| -87 | 11011000011 | 11 | -23 | 11010001 | 8 |
| -86 | 1110100001 | 10 | -22 | 1110011 | 7 |
| -85 | 11011000001 | 11 | -21 | 101110011 | 9 |
| -84 | 1001011011 | 10 | -20 | 1001011 | 7 |
| -83 | 10100100111 | 11 | -19 | 101000101 | 9 |
| -82 | 10001010011 | 11 | -18 | 1101011 | 7 |
| -81 | 10101010101 | 11 | -17 | 10010001 | 8 |
| -80 | 1011100011 | 10 | -16 | 110111 | 6 |
| -79 | 10010001111 | 11 | -15 | 1100111 | 7 |
| -78 | 1101111001 | 10 | -14 | 1110001 | 7 |
| -77 | 1010001001 | 10 | -13 | 1111111 | 7 |
| -76 | 1100010111 | 10 | -12 | 1010011 | 7 |
| -75 | 1110100111 | 10 | -11 | 10010011 | 8 |
| -74 | 1101111011 | 10 | -10 | 1011001 | 7 |
| -73 | 10011000011 | 11 | -9 | 1111101 | 7 |
| -72 | 111000111 | 9 | -8 | 100001 | 6 |
| -71 | 1010101001 | 10 | -7 | 10001001 | 8 |
| -70 | 100101111 | 9 | -6 | 1000111 | 7 |
| -69 | 1101100011 | 10 | -5 | 1110101 | 7 |
| -68 | 1001110001 | 10 | -4 | 111101 | 6 |
| -67 | 1100010101 | 10 | -3 | 1001101 | 7 |
| -66 | 1001110011 | 10 | -2 | 110001 | 6 |
| -65 | 1110100011 | 10 | -1 | 1010101 | 7 |
| -64 | 101010111 | 9 | 0 | 0 | 1 |
| -63 | 110111111 | 9 | 1 | 1010100 | 7 |
| -62 | 1001011001 | 10 | 2 | 110000 | 6 |
| -61 | 1000010011 | 10 | 3 | 1001100 | 7 |
| -60 | 111001011 | 9 | 4 | 111100 | 6 |
| -59 | 1010001011 | 10 | 5 | 1110100 | 7 |
| -58 | 101100101 | 9 | 6 | 1000110 | 7 |
| -57 | 1000010001 | 10 | 7 | 10001000 | 8 |
| -56 | 11101111 | 8 | 8 | 100000 | 6 |
| -55 | 1001100011 | 10 | 9 | 1111100 | 7 |
| -54 | 100001011 | 9 | 10 | 1011000 | 7 |
| -53 | 1000101011 | 10 | 11 | 10010010 | 8 |
| -52 | 111000101 | 9 | 12 | 1010010 | 7 |
| -51 | 1000000011 | 10 | 13 | 1111110 | 7 |
| -50 | 110110011 | 9 | 14 | 1110000 | 7 |
| -49 | 101111001 | 9 | 15 | 1100110 | 7 |
| -48 | 11011101 | 8 | 16 | 110110 | 6 |
| -47 | 101100111 | 9 | 17 | 10010000 | 8 |
| -46 | 101110011 | 9 | 18 | 1101010 | 7 |
| -45 | 111010111 | 9 | 19 | 101000100 | 9 |
| -44 | 101100011 | 9 | 20 | 1001010 | 7 |
| -43 | 111010101 | 9 | 21 | 101110010 | 9 |
| -42 | 11100001 | 8 | 22 | 1110010 | 7 |
| -41 | 111001101 | 9 | 23 | 11010000 | 8 |
| -40 | 100101001 | 9 | 24 | 1010110 | 7 |
| -39 | 111001111 | 9 | 25 | 110100110 | 9 |
| -38 | 100111011 | 9 | 26 | 11001000 | 8 |
| -37 | 101111011 | 9 | 27 | 111011100 | 9 |
| -36 | 100110011 | 9 | 28 | 1011110 | 7 |
| -35 | 100000011 | 9 | 29 | 1000101000 | 10 |
| -34 | 101000111 | 9 | 30 | 11001010 | 8 |
| -33 | 110001001 | 9 | 31 | 101110100 | 9 |
| -32 | 10010011 | 8 | 32 | 1011010 | 7 |
| -31 | 101000011 | 9 | 33 | 1011101110 | 10 |
| -30 | 10111011 | 8 | 34 | 100111010 | 9 |
| -29 | 100101011 | 9 | 35 | 1110110100 | 10 |
| -28 | 10011111 | 8 | 36 | 101000110 | 9 |
| -27 | 101000001 | 9 | 37 | 100111000 | 9 |
| -26 | 100000111 | 9 | 38 | 1011101100 | 10 |
| -25 | 100010111 | 9 | 39 | 1000101110 | 10 |
| -24 | 1101001 | 7 | 40 | 10100000 | 8 |
| -23 | 11011011 | 8 | 41 | 11010010110 | 11 |
| -22 | 11000111 | 8 | 42 | 1101001000 | 10 |
| -21 | 10011011 | 8 | 43 | 11010010100 | 11 |
| -20 | 10111111 | 8 | 44 | 1000101010 | 10 |
| -19 | 11001101 | 8 | 45 | 1011100000 | 10 |
| -18 | 10101001 | 8 | 46 | 10001011000 | 11 |
| -17 | 10101111 | 8 | 47 | 111011000 | 9 |
| -16 | 1000111 | 7 | 48 | 10011110 | 8 |
| -15 | 10101101 | 8 | 49 | 111011011100 | 12 |
| -14 | 1101011 | 7 | 50 | 101110001100 | 12 |
| -13 | 10100111 | 8 | 51 | 11101101100 | 11 |
| -12 | 1011011 | 7 | 52 | 10111000100 | 11 |
| -11 | 10001001 | 8 | 53 | 101110001110 | 12 |
| -10 | 1100001 | 7 | 54 | 10001011010 | 11 |
| -9 | 1100101 | 7 | 55 | 111011011110 | 12 |
| -8 | 11111 | 5 | 56 | 111011110 | 9 |
| -7 | 100000101 | 9 |  |  |  |
| -6 | 10000111 | 8 |  |  |  |
| -5 | 101100001 | 9 |  |  |  |
| -4 | 11101101 | 8 |  |  |  |
| -3 | 100100001 | 9 |  |  |  |
| -2 | 11001111 | 8 |  |  |  |
| -1 | 101001011 | 9 |  |  |  |
| 0 | 0 | 1 |  |  |  |
| 1 | 101001010 | 9 |  |  |  |
| 2 | 11001110 | 8 |  |  |  |
| 3 | 100100000 | 9 |  |  |  |
| 4 | 11101100 | 8 |  |  |  |
| 5 | 101100000 | 9 |  |  |  |
| 6 | 10000110 | 8 |  |  |  |
| 7 | 100000100 | 9 |  |  |  |
| 8 | 11110 | 5 |  |  |  |
| 9 | 1100100 | 7 |  |  |  |
| 10 | 1100000 | 7 |  |  |  |
| 11 | 10001000 | 8 |  |  |  |
| 12 | 1011010 | 7 |  |  |  |
| 13 | 10100110 | 8 |  |  |  |
| 14 | 1101010 | 7 |  |  |  |
| 15 | 10101100 | 8 |  |  |  |
| 16 | 1000110 | 7 |  |  |  |
| 17 | 10101110 | 8 |  |  |  |
| 18 | 10101000 | 8 |  |  |  |
| 19 | 11001100 | 8 |  |  |  |
| 20 | 10111110 | 8 |  |  |  |
| 21 | 10011010 | 8 |  |  |  |
| 22 | 11000110 | 8 |  |  |  |
| 23 | 11011010 | 8 |  |  |  |
| 24 | 1101000 | 7 |  |  |  |
| 25 | 100010110 | 9 |  |  |  |
| 26 | 100000110 | 9 |  |  |  |
| 27 | 101000000 | 9 |  |  |  |
| 28 | 10011110 | 8 |  |  |  |
| 29 | 100101010 | 9 |  |  |  |
| 30 | 10111010 | 8 |  |  |  |
| 31 | 101000010 | 9 |  |  |  |
| 32 | 10010010 | 8 |  |  |  |
| 33 | 110001000 | 9 |  |  |  |
| 34 | 101000110 | 9 |  |  |  |
| 35 | 100000010 | 9 |  |  |  |
| 36 | 100110010 | 9 |  |  |  |
| 37 | 101111010 | 9 |  |  |  |
| 38 | 100111010 | 9 |  |  |  |
| 39 | 111001110 | 9 |  |  |  |
| 40 | 100101000 | 9 |  |  |  |
| 41 | 111001100 | 9 |  |  |  |
| 42 | 11100000 | 8 |  |  |  |
| 43 | 111010100 | 9 |  |  |  |
| 44 | 101100010 | 9 |  |  |  |
| 45 | 111010110 | 9 |  |  |  |
| 46 | 101110010 | 9 |  |  |  |
| 47 | 101100110 | 9 |  |  |  |
| 48 | 11011100 | 8 |  |  |  |
| 49 | 101111000 | 9 |  |  |  |
| 50 | 110110010 | 9 |  |  |  |
| 51 | 1000000010 | 10 |  |  |  |
| 52 | 111000100 | 9 |  |  |  |
| 53 | 1000101010 | 10 |  |  |  |
| 54 | 100001010 | 9 |  |  |  |
| 55 | 1001100010 | 10 |  |  |  |
| 56 | 11101110 | 8 |  |  |  |