|  |  |
| --- | --- |
| **Joint Collaborative Team on Video Coding (JCT-VC)**  **of ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11**  19th Meeting: Strasbourg, FR, 17–24 Oct. 2014 | Document: JCTVC-S0039 |

|  |  |  |  |
| --- | --- | --- | --- |
| *Title:* | **CE6 subtest A.6: Binarization for run coding in palette mode** | | |
| *Status:* | Input Document to JCT-VC | | |
| *Purpose:* | Proposal | | |
| *Author(s) or Contact(s):* | Rajan Joshi, Wei Pu, Marta Karczewicz, Feng Zou, Vadim Seregin, Joel Sole  5775 Morehouse Drive San Diego, CA 92121, USA | Tel: Email: | 1-858-658-4511 [rajanj@qti.qualcomm.com](mailto:rajanj@qti.qualcomm.com) |
| *Source:* | Qualcomm Incorporated | | |

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

# Abstract

A binarization for coding of the palette run values is proposed. The proposed binarization is a concatenation of unary code and exponential Golomb code of order 0 (switching point at 1). Both the prefix and suffix use truncated signalling based on the maximum possible run value. The first 3 bins of the prefix are context-coded. It is reported that the proposed method provides BD-rates of −0.8% and −1.1% for All-Intra 1080p text and graphics RGB and YUV categories, respectively, over SCM2.0 anchor. When the proposed approach is combined with CE6 test A.5 (run contexts dependent on index), BD-rates of −1.4% and −1.7% are reported for the above two classes. When parsed index instead of decoded index is used for context determination and up to the first 5 bins of the prefix are context-coded instead of 3, BD-rates of −1.5% and −1.8% are reported for the above two classes.

# Introduction

A binarization for coding the palette run values is proposed. The binarization is a concatenation of unary code and exponential Golomb code of order 0 (switching point at 1). This is shown in Table 1.

|  |  |  |  |
| --- | --- | --- | --- |
| Symbol | Prefix | Suffix | length |
| 0 | 1 |  | 1 |
| 1 | 01 |  | 2 |
| [2, 3] | 001 | X | 4 |
| [4, 7] | 0001 | XX | 6 |
| ... |  |  |  |

Table 1: Proposed binarization for coding of palette runs

Since the maximum possible value for the run is bounded by the end of the block, we propose using truncated version for the prefix as well as the suffix. For example, let the CU size be 8×8 and let the current index be 57. This means that the maximum run value can be 6. In this case, the prefix for the [4, 7] range is coded as 000 and if run value belongs to the interval [4, 7], run value minus 4 is coded using truncated binary coding with maximum symbol equal to 2.

The first 3 bins of the prefix are context-coded and the rest are bypass-coded. Thus in the worst case, the number of context-coded bins is the same as in the binarization of the run values in SCM2.0. We also present results of combining this binarization with CE6 test A.5 (run contexts dependent on index).

# Results

The proposed methods are implemented on top SCM 2.0 and simulated under common test conditions (JCTVC-R1015). The simulation platform is a homogenous LINUX cluster consisting of Intel(R) XEON CPUs. Table 2 shows the BD-rate performance for the proposed method for lossy configuration. Table 3 provides the BD-rate results for the combination of the proposed method with CE6 test A.5 for lossy configuration. Table 4 and Table 5 show the corresponding results for lossless configuration.



Table 2: BD-rate results for the proposed binarization of run values (lossy configuration)



Table 3: BD-rate results for the combination of proposed method with CE6 test A.5 (lossy configuration)



Table 4: BD-rate results for the proposed binarization of run values (lossless configuration)



Table 5: BD-rate results for the combination of proposed method with CE6 test A.5 (lossless configuration)

## Additional results

Table 6 shows BD-rate results for the proposed method in combination with CE6 test A.5 when parsed index instead of decoded index is used for context determination. Table 7 shows the BD-rate results for the same configuration when up to the first 5 bins of the prefix are context coded instead of 3. Corresponding results for lossless configurations can be found in the accompanying spreadsheets.



Table 6: BD-rate results for the combination of proposed method with CE6 test A.5 when parsed index is used for context determination (lossy configuration)



Table 7: BD-rate results for the combination of proposed method with CE6 test A.5 when parsed index is used for context determination and up to 5 bins of the prefix are context-coded instead of 3 (lossy configuration)

# Conclusions

A binarization for coding the palette run values is proposed. The binarization is a concatenation of unary code and exponential Golomb code of order 0 (switching point at 1). Both the prefix and suffix use truncated signalling based on the maximum possible run value. The first 3 bins of the prefix are context-coded. The proposed method provides BD-rates of −0.8% and −1.1% for All-Intra 1080p text and graphics RGB and YUV categories, respectively, over SCM2.0 anchor. When the proposed approach is combined with CE6 test A.5 (run contexts dependent on index), BD-rates of −1.4% and −1.7% are achieved for the above two classes. When parsed index instead of decoded index is used for context determination and up to the first 5 bins of the prefix are context-coded instead of 3, BD-rates of −1.5% and −1.8% are achieved for the above two classes.

The proposed method uses the same number of context-coded bins as the existing binarization and provides substantial BD-rate improvements. Using up to 5 context coded bins for the prefix further improves BD-rate performance without affecting the worst-case. We recommend adoption of the proposed method in combination with CE6 subtest A.5 to the next version of the SCC test model and software.

# Draft text specification

#### 9.3.3.13 *Binarization process for palette\_run*

Input to this process is a request for a binarization for the syntax element palette\_run, ~~a colour component index cIdx~~, scanPos, and nCbS \* nCbS.

Output of this process is the binarization of palette\_run as specified in Table 9‑46.

Table 46 – Binarization for palette\_run

|  |  |
| --- | --- |
| palette\_run | Codeword |
| 0 | '0' |
| >0 | Prefix = '1', Suffix = 0-th order TEGk code symbolVal = palette\_run – 1, cTEGParam = nCbS \* nCbS – 2 – scanPos |

#### 

#### 9.3.3.x k-th order Truncated Exp-Golomb (TEGk) binarization process

Inputs to this process is a request for an TEGk binarization of symbolVal and a parameter cTEGParam.

Output of this process is the TEGk binarization associating each value symbolVal with a corresponding bin string.

The bin string of the TEGk binarization process for each value symbolVal is specified as follows, where each call of the function put( X ), with X being equal to 0 or 1, adds the binary value X at the end of the bin string:

stopLoop = 0  
do  
 if(symbolVal >= ( 1 << k ) ) {  
 put( 1 )  
 symbolVal = symbolVal − ( 1 << k )   
 cTEGParam = cTEGParam − ( 1 << k )   
 k++  
 } else {  
 gapVal = cTEGParam − ( 1 << k )  
 if(gapVal >= 0) { (9-yy)  
 put( 0 )  
 while( k− − )  
 put( (symbolVal >> k) & 1 )  
 stopLoop = 1  
 } else {  
 9.3.3.6 TB binarization, synVal = symbolVal, cMax = gapVal  
 stopLoop = 1  
 }  
 }  
while( !stopLoop )

# Patent rights declaration(s)

**Qualcomm Incorporated 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).**