|  |  |
| --- | --- |
| **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-S0038 |

|  |  |  |  |
| --- | --- | --- | --- |
| *Title:* | **CE6 subtest A.5: Contexts for run coding in palette mode** | | |
| *Status:* | Input Document to JCT-VC | | |
| *Purpose:* | Proposal | | |
| *Author(s) or Contact(s):* | Rajan Joshi, Marta Karczewicz, Wei Pu, 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

Additional contexts for coding of the runs in 'INDEX' mode are proposed. The context is dependent on the index value. BD-rate results for two additional contexts are presented. It is reported that the proposed approach provides BD-rates of −0.6% and −0.6% for All-Intra 1080p text and graphics RGB and YUV categories, respectively, over SCM-2.0 anchor. When context is determined using parsed index instead of decoded index, BD-rates of −0.5% and −0.6% are reported for the above two classes.

# Introduction

In SCM2.0, each sample in a palette block belongs to one of the two modes

1. COPY\_ABOVE mode: In this mode, the palette index for the current pixel is copied from the pixel located directly above in a block.
2. INDEX mode: In this mode, the index of the palette to be used to reconstruct the pixel is explicitly signalled. This also includes the 'escape' mode.

For ‘COPY\_ABOVE’ and ‘INDEX’ modes (except 'escape'), a run value is signalled as well. The run value specifies the number of subsequent samples that belong to the same mode. Currently for coding ‘greater than zero’ bin, one context is used for ‘INDEX’ runs and another context is used for ‘COPY\_ABOVE’ runs.

We propose the use of two additional contexts to code the ‘greater than zero’ bin for ‘INDEX’ runs. The context is chosen depending on the index.

|  |  |  |  |
| --- | --- | --- | --- |
| Index | 0 | [1, 2] | Greater than 2 |
| Run context | 0 | 1 | 2 |

In 'INDEX' mode, certain index value is impossible, so while encoding, some index values are adjusted downwards whereas on the decoder side, they have to be adjusted upwards. We provide two sets of simulation results:

1. Run context depends on the index after adjustment (decoded index)
2. Run context depends on the index before adjustment (parsed 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. Tables 1 and 2 show the BD-rate performance for the proposed method when using decoded index and parsed index, respectively, for lossy configuration. Tables 3 and 4 show the corresponding results for lossless configuration.



Table 1: BD-rate results for run contexts dependent on decoded index (lossy configuration)



Table 2: BD-rate results for run contexts dependent on parsed index (lossy configuration)



Table 3: BD-rate results for run contexts dependent on decoded index (lossless configuration)



Table 4: BD-rate results for run contexts dependent on parsed index (lossless configuration)

# Conclusions

Additional contexts for coding of the runs in 'INDEX' mode are proposed. The context is dependent on the index value. BD-rate results for two additional contexts are presented. It is reported that the proposed approach provides BD-rates of −0.6% and −0.6% for All-Intra 1080p text and graphics RGB and YUV categories, respectively, over SCM-2.0 anchor. When context is determined using parsed index instead of decoded index, BD-rates of −0.5% and −0.6% are reported for the above two classes.

The proposed method based on parsed index provides good gains with minimal increase in complexity (two additional contexts) and has no parsing dependency. We recommend its adoption to the next version of SCC test model and software.

# Draft text specification

| Table 9‑43 – Assignment of ctxInc to syntax elements with context coded bins | | | | | | |
| --- | --- | --- | --- | --- | --- | --- |
| **Syntax element** | **binIdx** | | | | | |
| **0** | **1** | **2** | **3** | **4** | **>= 5** | |
| ... |  |  |  |  |  |  | |
| palette\_run\_type\_flag | 0, 1 (subclause 9.3.4.2.2) | na | Na | na | Na | Na | |
| palette\_index | bypass | bypass | bypass | bypass | bypass | bypass | |
| palette\_run | 0, 1, 2, 3 (subclause 9.3.4.2.8) | 4, 5 (subclause 9.3.4.2.8) | 6, 7 (subclause 9.3.4.2.8) | bypass | bypass | bypass | |
| palette\_all\_zeros\_in\_group | bypass | na | na | na | na | na | |
| ... |  |  |  |  |  |  | |

**9.3.4.2.8 *Derivation process of ctxInc for the syntax element palette\_run***

Inputs to this process are binIdx, palette\_index and palette\_run\_type\_flag.

Output of this process is the variable ctxInc.

Table x: Specification of ctxInc for palette\_run

|  |  |  |  |
| --- | --- | --- | --- |
| binIdx | palette\_run\_type\_flag | palette\_index | ctxInc |
| 0 | 1 |  | 0 |
| 0 | 0 | 1 |
| 1,2 | 2 |
| greater than 2 | 3 |
| 1 | 1 |  | 4 |
| 0 |  | 5 |
| 2 | 1 |  | 6 |
| 0 |  | 7 |

# 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).**