Analysis of Palette Mode on Versatile Video Coding

Yu-Chen Sun[1], Jian Lou[1], Yung-Hsuan Chao[2], Hongtao Wang[2], Vadim Seregin[2], Marta Karczewicz[2]
Alibaba Group (U.S.) Inc.[1]., Qualcomm Technologies Inc.[2]
{yuchen.s, j.lou}@alibaba-inc.com, {yunghsua, hongtaow, vseregin, martak}@qti.qualcomm.com

Abstract

The next generation video compression standard, Versatile Video Coding (VVC), is being developed by the ITU-T/ISO/IEC Joint Video Experts Team (JVET). Screen content coding is one of the target applications of the new standard. Palette mode has been investigated in the previous standard, High Efficient Video Coding Screen Content Coding Extension (HEVC-SCC), and can efficiently compress color clusters in screen content. This paper evaluates the palette mode on VVC. With the proposed modifications, the palette mode can achieve 33.1%, 16.6%, and 8.8% BD-rate saving for "YUV, text & graphics sequences" under all intra (AI), random access (RA), and low-delay B (LB) common test conditions, respectively.

Keywords: Palette Mode, Versatile Video Coding (VVC), Screen Content Coding

1. Introduction

The next generation video coding standardization, Versatile Video Coding (VVC), was initiated by ITU-T/ISO/IEC Joint Video Experts Team (JVET) in 2017 [1]. In [2], VVC shows 23% BD-rate saving under random access test conditions compared with High Efficient Video Coding (HEVC), and the performance is still being improved. Since screen content coding (SCC) has been widely used for many applications, such as remote desktop, web conferencing, and cloud computing, screen content coding is included in the target applications of VVC. Palette mode is the coding tool that improves the coding efficiency of SCC. It has been adopted in HEVC SCC [3] and is one of the major topics under discussion in the VVC standard committee [4][5].

Pixel values in screen contents concentrate on few color values, a fact that is central to the concept of the palette mode [6][7]. An example of the method can be seen in Figure 1. Here, the pixels in a coding unit (CU) are analyzed by the encoder, which determines the representative colors in this CU and creates a color mapping table between the representative color values

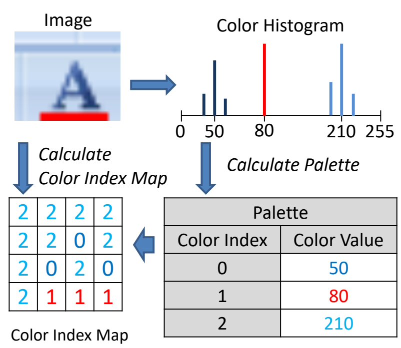


Figure 1. An example of palette mode

and the color indices. The table, or the palette for this CU, is subsequently transmitted to the decoder. Based on this palette, most pixels in the CU with values close to those of the palette colors are quantized by the encoder to those palette colors and represented by the corresponding color indices. The remaining pixels whose values are not similar to any palette colors are signaled directly. As they are not being quantized to a specific palette color value, these pixels are referred to as escape pixels and are represented by a special color index. Next, the encoder constructs a color index map from all of the color indices in the CU and transmits this map, along with the escape pixel values, to the decoder. Figure 1 shows a simplified illustration of this process, where a pixel or a color index represents only one value. It is important to note, however, that as indicated in the HEVC SCC Draft [3], a pixel or a color index can represent multiple color component values, such as YCbCr or GBR.

Since palette mode has demonstrated significant improvement in coding efficiency in the previous standard, it is worthwhile to investigate this method on top of VVC. Because the structure of HEVC and VVC is different, in order to introduce the palette mode to VVC, some required modifications are proposed in this paper. In the following sections, a brief introduction of the palette mode in HEVC and the proposed palette mode modifications for VVC are presented in Sections 2 and 3, respectively. In Section 4, the experimental results are

978-1-7281-1198-8/19/$31.00 ©2019 IEEE
DOI 10.1109/MIPR.2019.00091

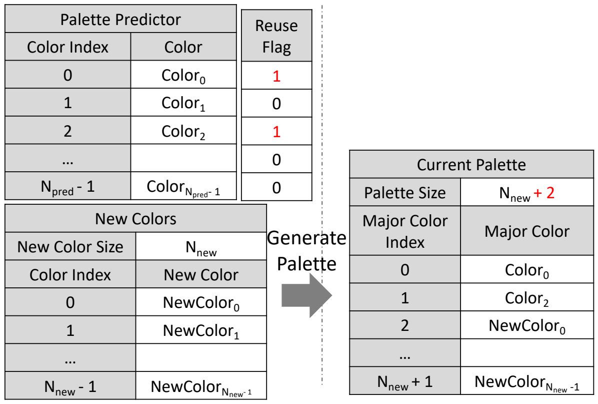
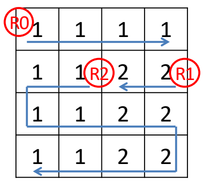


Figure 2. An example of palette coding.



(a)

Run Index	Run Mode	Colour Index (derived from palette_index)	Run of Pixels (derived from palette_run)
R0	Copy Index	1	4
R1	Copy Index	2	2
R2	Copy Above	N/A	10

(b)

Figure 3. An example of color index map coding. (a) an index map example. (b) the information to be encoded and the corresponding syntax in bit stream.

shown to demonstrate the coding gains of the modified palette mode on top of VVC. Finally, conclusions and future work are described in Section 5.

2. Palette mode in HEVC

In the palette mode, a flag is signaled at the CU level to indicate that the tool is being used. The pixels in a CU are categorized into two types: major colors and escape colors. While the most representative colors in the CU are defined by the major colors, those that differ from the major colors become the escape colors. Each color is comprised of three value components (e.g., YCbCr or GBR) as documented in the HEVC SCC palette mode.

Palette coding tools encode the major colors by describing them with a palette. Based on the palette, each pixel, which is composed of three color components (e.g., YCbCr or GBR) , is assigned a color index. A color index map is formed from the color indices associated with all of the pixels in a CU.

Using palette coding tools, a palette can be predictively coded. As shown in Figure 2, the derivation and coding process for the palette of a CU begins with the construction of a palette predictor based on previously coded palettes. From the palette predictor, prediction information of a palette, in the form of reuse flags, is signaled to dictate the derivation of the palette for the current CU. In the example demonstrated in Figure 2, the N_{pred} reuse flags associated with the colors in the predictor are coded after a palette predictor is constructed. The reuse flag identifies a color as being reused in the palette of the current CU. In Figure 2, these reused colors are $Color_0$ and $Color_2$. The decoder can allocate the reused colors to the beginning of the palette of the current CU. Next, the encoder signals N_{new} new colors, $NewColor_0$… $NewColor_{Nnew-1}$, which are placed after the reused colors by the decoder. In the illustration, the current palette comprises 2 reused and N_{new} newly generated colors.

Pixels in a CU are mapped to color indices associated with the colors in the encoded palette. An index map is built from these color indices that are run-based encoded in the horizontal or vertical traverse scan order. Note that figures presented in this paper assume the horizontal scan order.

A traverse scan, regardless of its direction, consists of two run modes, copy index mode and copy above mode. To indicate which run mode is utilized, a flag is signaled initially at the starting position of each run. In the copy index mode, the same color index is share among a run of pixels, and the palette index and run length are signaled by encoder. In the copy above mode, on the other hand, the index of the pixel is copied from their above pixels, and only the run length is signaled.

An example of the color index map coding can be found in Figure 3, in which an index map of 16 color indices is encoded by three runs with a horizontal traverse scan. The copy index mode is illustrated in the first two runs with a color index of (1, 4) and a run value of (2, 2). The copy above mode is shown in the third run with a run value of 10.

Using the decoded indices of the completed index map, the decoder can obtain the associated colors in the palette and reconstruct pixels based on these colors. In HEVC, the syntax is nearly identical for videos that are in non-4:4:4 chroma formats. A slight difference exists for escape samples whose sample position only contains a luma component. In such cases, instead of three quantized components, only one is included in the bitstream. Likewise, for the reconstruction of non-escape samples, the position of the sample dictates whether one or three components from the palette entry corresponding to the palette index is utilized.

3. Palette mode modification for VVC

When transferred to VVC, a more intricate coding structure is adopted. In addition to the quaternary-tree structure in HEVC, VVC is able to divide a coding tree units (CTU) into several non-square CUs by a multi-type tree structure. Furthermore, while the luma coding tree block (CTB) and the chroma CTB in a CTU exhibit the same coding structure in HEVC, their block tree structures can differ from each other in VVC. More specifically, in VVC, with the exception of inter slices (P

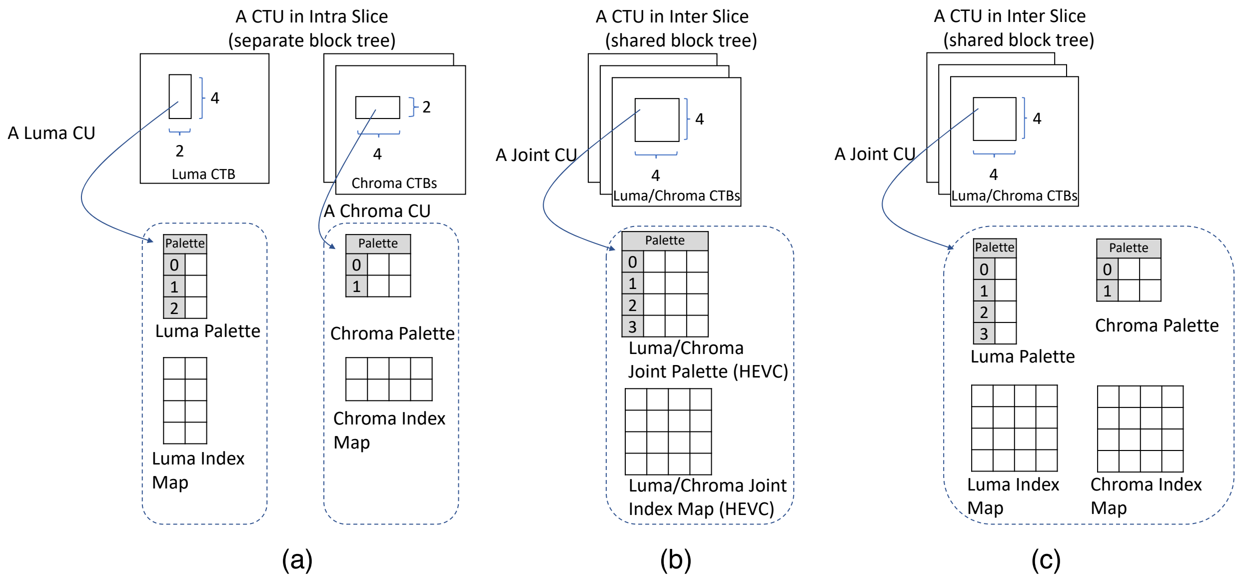


Figure 4. An example of palette mode modification for VVC. (a) For a separate coding tree (in intra slices of VVC), separate palette is applied. (b) For a shared coding tree (in inter slices of VVC), Method 1: joint palette is applied. (c) For a shared coding tree (in inter slices of VVC), Method 2: separate palette is applied.

and B slices), the luma and chroma CTBs in one CTU can have separate block tree structures. When the separate block tree mode is enabled, the partitioning of the luma CTB into luma CUs, and the chroma CTBs, into chroma CUs, are proceeded by independent coding tree structures. This would result in a CU of an I slice to have either a coding block of the luma component, or a coding block of two chroma components. Meanwhile, a CU in a P or B slice, with the exclusion of monochromatic videos, would always comprise coding blocks of all three color components.

The new coding structure disagrees with the palette mode design in HEVC. A CU in HEVC consists of both luma and chroma CBs, such that a palette always contains three color components, and the luma and chroma samples share the same color index map. Yet, in VVC, when the separate tree structure is applied, a CU may be composed of only luma or only chroma CB. In such cases, the HEVC palette mode is not applicable. To solve this problem, specific palette mode modifications are proposed in this paper.

First, application of a different palette design is proposed for intra slices when the separate tree structure is implemented. As demonstrated in Figure 4(a), the luma CU has its own palette and index map. The palette, known to consists of three color values in HEVC, now contains colors represented by only one luma value. The chroma CU also has its own palette and index map; the colors in its palette are represented by two chroma values.

In the case of inter slices, two palette implementations are possible. The first method is the application of the HEVC palette mode without any modifications, since the shared tree coding structure is used for inter slices in both HEVC and VVC, as illustrated in Figure 4(b). Alternatively, as a second approach, the separate palette method can still be utilized even though the CU is a joint CU, as shown in Figure 4(c). The BD-rate difference between method 1 and method 2 is presented in Table 1. Eight screen content sequences are tested following [8]. Details of the experiment setting are described in Section 4. As demonstrated in Table 1, method 2 delivers an overall higher gain in BD-rate. Furthermore, when both inter and intra slices need to be processed, method 2 utilizes the separate palette approach regardless of the nature of the coding tree, while method 1 must use two different palette methods to treat the slices. Consequently, the design of method 2 is simpler than that of method 1.

In addition to the abovementioned modifications, an encoder fast algorithm is also proposed in this paper. In this method, when the encoder compresses a CU in an inter slice, it examines the RD cost of the inter prediction modes, palette mode, and intra prediction modes in order. The encoder is made to skip the remaining tests for the intra prediction modes if it determines the palette mode to be the best method, thereby reducing the encoding time. By the proposed fast algorithm, decreases in encoding time by 20% are achieved with negligible loss from skipping the intra prediction mode tests.

4. Experimental Results

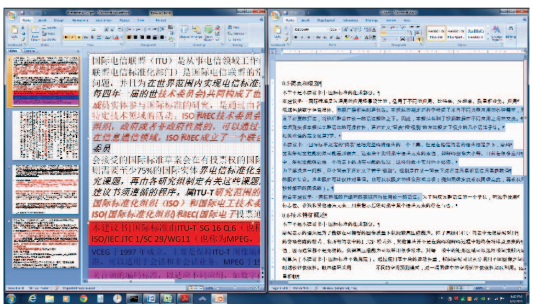
The palette mode has been integrated into VVC reference software, VTM-3.0 [9]. We evaluate the coding efficiency of the palette mode on top on VTM-3.0. The experiments are conducted under the test

Table 1. BD-rate comparison of Method 1 and Method 2 for palette implementations. The anchor is Method 1. The test is Method 2. Negative values indicate bit rate decreases (coding gains) and reflect the benefits of Method 2.

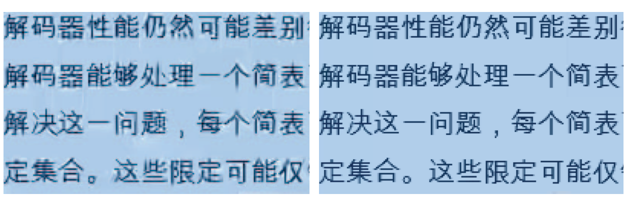
	Y	U	V	Y	U	V
	Random Access			**Low delay B**		
Class F	-0.16%	0.06%	-0.01%	-1.32%	-0.61%	-0.78%
TGM	-1.18%	-0.79%	-0.71%	-1.97%	-1.91%	-1.52%

Table 2. The results of the proposed palette mode. The anchor is VTM-3.0. The test is VTM-3.0 with the palette mode. Negative values indicate bit rate decreases (coding gains) and reflect the benefits of the palette mode.

	Y	U	V	EncT	DecT
			All Intra		
Class F	-11.43%	-8.94%	-9.04%	121%	94%
TGM	-33.06%	-27.59%	-27.70%	135%	69%
			Random Access		
Class F	-8.65%	-7.95%	-8.32%	110%	102%
TGM	-16.64%	-15.24%	-15.02%	108%	94%
			Low delay B		
Class F	-5.07%	-4.98%	-5.52%	110%	99%
TGM	-8.81%	-8.43%	-7.74%	106%	94%



(a) The first frame in ChineseEditing



(b) without palette mode (c) with palette mode

Figure 5. Visual comparisons between VTM-3.0 and VTM-3.0 with the palette at first frame in ChineseEditing.

conditions suggested by [8]. Eight non-camera captured video sequences in different resolutions and frame rates are assessed, including Class F: { BasketballDrillText, ArenaOfValor, SlideEditing, SlideShow} and Class text and graphics with motion (TGM): { FlyingGraphics, Desktop, Console, ChineseEditing}. The coding performance is measured by BD-rate.

As shown in Table 2, the palette mode decreases luma BD-rate by 33.06%, 16.64%, and 8.81% for sequences in class TGM under AI, RA, and LB, respectively, in addition to providing significant subjective benefits. For instance, Figure 5(b) and 5(c) show the decoded frame crops in the first frame of ChineseEditing sequence encoded by VTM-3.0 with and without the palette mode, respectively. Without the palette mode, encoding the first frame consumes 707,096 bits while the first frame consumption with the palette mode is 699112 bits. The results demonstrate that the palette mode not only consumes fewer bits, it also generates much sharper decoded texture.

5. Conclusion

This paper evaluates the palette mode with the proposed modifications on top of VVC. The palette mode achieves 33.1%, 16.6%, and 8.8% BD-rate saving for "YUV, text & graphics sequences" under all intra (AI), random access (RA), and low-delay B (LB) common test conditions, respectively. Our modifications enable the palette mode to deliver enhanced performance and highlight its usability on VVC.

6. References

[1] A. Segall, V. Baroncini, J. Boyce, J. Chen, T. Suzuki, "*Joint Call for Proposals on Video Compression with Capability beyond HEVC*," in Document of JVET, JVET-H1002, 8th Meeting, Macao, CN, 2017.

[2] F. Bossen, X. Li, K. Sühring, "*JVET AHG report: Test model software development (AHG3)*" in Document of JVET, JVET-L0003, 12th Meeting, Macao, CN, 2018.

[3] R. Joshi, S. Liu, G. J. Sullivan, J. Xu, Y. Ye, "HEVC Screen Content Coding Draft Text 4," in Document of JCTVC, JCTVC-U1005, 21st Meeting, Warsaw, PL, 2015.

[4] Y.-C. Sun, J. An, J. Lou, Y.-H. Chao, H. Wang, V. Seregin, M. Karczewicz, "*CE15-1: Palette mode*" in Document of JVET, JVET-L0344, 12th Meeting, Macao, CN, 2018.

[5] Y.-H. Chao, H. Wang, V. Seregin, M. Karczewicz, Y.-C. Sun, J. An, J. Lou, "*CE15-2: Palette mode of HEVC SCC*" in Document of JVET, JVET-L0336, 12th Meeting, Macao, CN, 2018.

[6] Y.-C. Sun, T.-D. Chuang, P. Lai, Y.-W. Chen, S. Liu, Y.-W. Huang S. Lei, "*Palette Mode - a New Coding Tool in Screen Content Coding Extensions of HEVC*," in Proc. IEEE Int. Conf. Image Process (ICIP), Sep. 2015.

[7] W. Pu, M. Karczewicz, R. Joshi, V. Seregin, F. Zou, J. Sole, Y.-C. Sun, T.-D. Chuang, P. Lai, S. Liu, S.-T. Hsiang, J. Ye, and Y.-W. Huang, "*Palette Mode Coding in HEVC Screen Content Coding Extension*," IEEE Journal on Emerging and Selected Topics in Circuits and Systems, Vol.6, Issue 4, p.p.420-432, 2016.

[8] X. Xu, Y.-C. Chao, Y.-C. Sun, J. Xu, "*Description of Core Experiment 8: Screen Content Coding Tools*", in Document of JVET, JVET-L1028, 12th Meeting, Macao, CN, Oct. 2018.

[9] VTM, https://vcgit.hhi.fraunhofer.de/jvet/VVCSoftware_VTM/tree/VTM-3.0