ON PREDICTION TECHNIQUES FOR PALETTE CODING

Guoxin Jin[*,†] *, Ankur Saxena*[*] *and Felix Fernandes*[*]

gjin@u.northwestern.edu; {a.saxena1, felix.f}@samsung.com
[*]Samsung Information Systems America, 1301 E. Lookout Drive, Richardson, TX - 75082
[†]Electrical Engineering & Computer Science Dept., Northwestern University, Evanston, IL - 60201

ABSTRACT

Screen content video coding is becoming increasingly important in various applications, such as desktop sharing, video conferencing, and remote education. In January 2014, the ITU-T and ISO/IEC MPEG jointly issued call for proposals for screen content coding as an extension of HEVC. In general, compared to natural camera-captured video content, screen content has different characteristics, e.g., sharper edges, and fewer unique colors on a block-by-block basis. Palette coding for screen content has been recently proposed in the ongoing HEVC Range Extensions standardization. Palette coding utilizes the fact that there are few unique colors in screen content video blocks, and tries to send *palettes* of these unique colors. However, the size of these palettes can expand, especially in high-resolution video. Therefore, to reduce the overall bits for palette transmission, we present various palette prediction techniques in this paper. We first analyze the characteristics of palettes in screen content, and then propose various palette prediction techniques. Our experiments on HEVC test sequences in state-of-the-art HM range extensions software for HEVC show significant improvements over palette coding without any prediction, at almost negligible additional complexity.

Index Terms— Palette Coding, palette prediction, screen content coding, HEVC, Range Extensions.

1. INTRODUCTION

With the rapid development of communication network bandwidth, and the rise of light weight consumer multimedia devices, the amount of visual data being captured, stored, and transmitted is increasing substantially. In addition to natural camera captured data, screen content also takes a large portion of the network bandwidth for transmission. Screen content comprises of computer-generated data such as documents, graphics and web-pages. In addition, cloud video games and virtual reality applications are other examples of screen content. Nowadays, there are a lot of applications of using screen content, such as video conferencing, remote education, remote assistance and social networks. In theory, screen content can be compressed as the same way as natural camera-captured video. However, the statistical characteristics of screen content differ significantly from those of camera content, and significantly higher compression is achievable if compression tools are designed specifically for screen content. Consequently, the HEVC version 2 (Range Extensions) standardization has considered such compression tools, and the ITU-T and ISO-IEC MPEG have jointly issued a Call for Proposal in January 2014 to launch the standardization of screen content coding extension of HEVC.

There are various technologies which currently can improve the compression efficiency of screen content coding. Intra-Block Copy [1] predicts the current block from other previously coded blocks in intra frames, similar to motion estimation/compensation for inter frames, and works extremely well in regions with repeating patterns such as text, maps etc. Edge mode [2] tries to reduce the high bit cost of sharp edges in screen content by modeling six edge positions, and selecting one of them during encoding. In [3], transform skipping was proposed, where it was shown that coding without a transform is better for screen content, since the spatial residue is sparse.

Recently, another technique: palette coding [4, 5, 6] has been proposed for encoding screen content. Palette coding utilizes the fact that screen content primarily consists of only a few unique colors in a coding block. Thus, the pixels in each coding block are used to form a color palette (set of dominant colors). During encoding, all the pixels in the block can simply be quantified by an index each in the palette. However, encoding the actual palette can typically consume a large number of bits, and therefore prediction techniques to exploit the correlation between palettes of neighboring blocks should be utilized. In this paper, we first study the characteristics of palettes in screen content video sequence, and then present various palette prediction technique. The rest of the paper is organized as follows: palette coding is formally introduced in Sec. 2. Palette prediction framework, and prediction schemes are presented in Sec. 3. Simulation results are shown in Sec. 4, finally followed by conclusions in Sec. 5 .

2. PALETTE CODING: REVIEW AND SHORTCOMINGS

Palette coding [4, 5] has been recently proposed for encoding screen content. Fig. 1 demonstrates the palette coding process with a Coding Unit (CU) of video as input (a CU can be considered equivalent to a macro-block in H.264/AVC). In palette coding, the most frequent pixel values are first selected as major colors. After that, all the pixels in the current coding block are quantized to these palette-colors. If the total number of unique colors in the coding block is small, then the palette formed with the major colors can typically cover all the pixels in the block, and the residue vanishes. The palette colors, quantization indices and the quantization errors, when residuals are non-zero, are then encoded. High compression ratio can then be achieved by carefully encoding the quantization indices by using run-length coding, or DPCM techniques. Other techniques such as reordering palettes [6, 7], and escape colors [5, 6] can further be applied to improve the compression ratio.

For screen content video sequences such as documents and computer user interfaces, palette coding is very efficient as there are very few unique colors in a particular coding block of the video sequence. For example, Fig. 2 shows a 16x16 coding block of screen content

* This work was performed when G. Jin was an intern at Samsung.

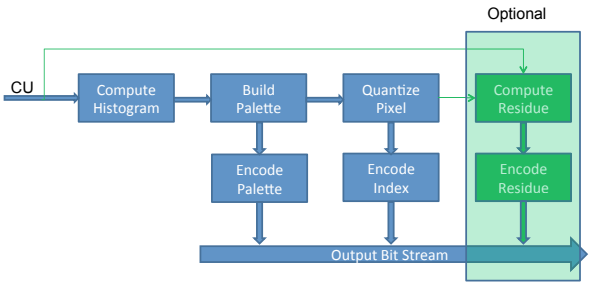


Fig. 1. Palette Coding framework.

(a) SC (b) NCC

Fig. 2. A zoomed version of 16x16 screen content block (left) and natural captured content (right). Figure best viewed in color.

taken from a document with text, and has only 9 unique colors (note that by a unique color, here we mean a unique vector in [Y,U,V], or [R,G,B] color space.). However, a natural captured content of the same size contains 256 unique colors (different vectors in YUV color space. Note that the possible number of different vectors of 8 bit video with 3 components is 2^{24}).

Typically, there are multiple unique colors in a palette, and a large number of bits will be utilized in encoding these different colors. For example, consider a 8-bit YUV video sequence. Each color is represented by 24 bits. Assuming there are 10 different colors in a 8x8 block, there would be 240 additional bits required to be transmitted for each 8x8 block being encoded by palette mode. Clearly, it is expensive if the palettes for each block in a video are coded separately. To reduce the overall bits of encoding palettes separately, [5] and [6] proposed palette prediction techniques. At a high level, in [5] and [6], prediction for palettes can happen from a CU on top or left, if they were also coded using palette mode. However, this is strictly restrictive. For example, consider a video frame being encoded by palette mode in Fig. 3. The red colored regions around the text 'int mai' are encoded by palette mode, while the other regions (e.g., in purple color) are encoded by a different modes (e.g., intra mode). Here, for the green-colored CU 1, a palette-coded CU on left is available, and palette prediction is possible. However, for CU 2, both the top, and left neighbors are not coded by palettes. Due to the restrictive requirement of a left, or top palette-coded CU to be available for prediction, the additional gain of palette prediction was only around 0.3% to 0.4 % in terms of BD-Rate [8] in [5]. We next present our palette prediction techniques in the next section, and show how to avoid these shortcomings.

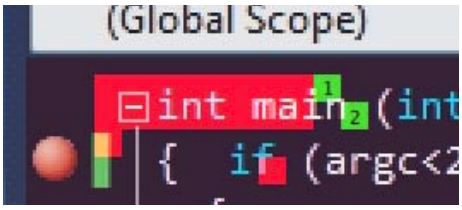


Fig. 3. Palette prediction example: Red blocks are palette coded blocks without any palette prediction. Green blocks represent possible blocks where palette prediction can be used. Figure best viewed in color.

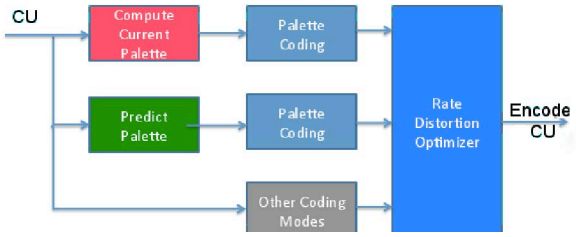


Fig. 4. Palette prediction framework.

3. PALETTE PREDICTION FOR PALETTE CODING

As we have demonstrated in the previous section, prediction of palettes in prior work was not optimal. We present a palette prediction framework in Fig. 4. The palette to be used for encoding the CU can either be computed by the current CU itself, or predicted from already palette-coded blocks. A Rate-Distortion optimization is performed at the encoder to select the best mode (using current palette, or previous palette, or other modes) for a coding block. The Rate-Distortion optimizer (RDO) at the encoder selects the mode based on the bits (R) used and distortion (D) as $J = R + \lambda D$, where the value of λ depends on the Quality Parameter (QP) of the codec [9].

Depending on the technique used for palette prediction, as we describe later in this section, the operations of RDO may be different. Note that, in HEVC, there are various other coding modes such as Intra-Block Copy and Angular Prediction [10] at the encoder, and RDO also compares the Rate-Distortion cost with these other modes. The mode which gives the smallest cost J will be used, and a flag is signaled to the decoder about the chosen mode at the encoder.

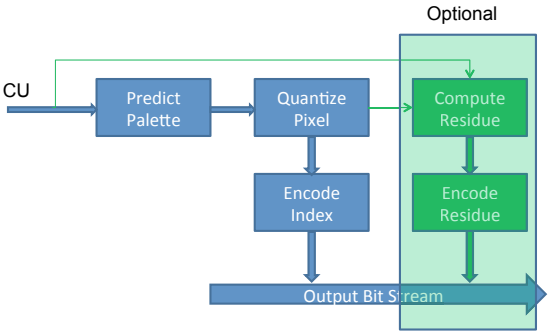


Fig. 5. Palette coding with palette prediction.

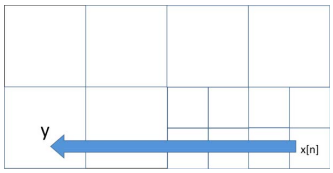


Fig. 6. Palette prediction without motion vector.

Table 1. Palette Prediction Mode Signaling.

Mode	Flag
Not a Palette Mode	0
Palette mode, palette computed from current block	10
Palette mode, and palette predicted.	11

When palette prediction is used, the coding process is demonstrated in Fig. 5. Note that unlike Fig. 1, there is no actual palette that needs to be encoded. We next present the two different variants of how to use a previous palette for prediction.

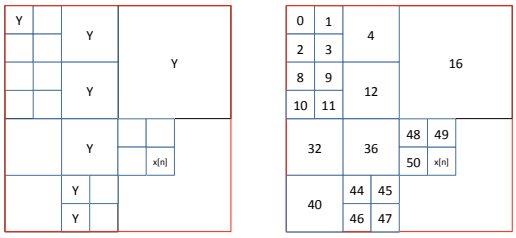
3.1. Scheme A: No motion vector

In this proposed scheme, we simply predict the palette from the last palette-coded CU on the left, which may not necessarily adjacent to current CU. The proposed scheme is shown in Fig. 6. The current CU $x(n)$ being coded uses a palette-coded CU Y on its left as predictor for its palette. The Rate-Distortion process picks one of the two options: (a) use palette from current block $x(n)$ itself; and (b) use the predicted palette from the last palette-coded CU Y on the left. The signaling mechanism for this variant is shown in Table 1. Note that the major advantage of this scheme is that no motion vector is required to be sent for palette prediction. At the decoder, if the received flag is 11, the decoder finds the predictor palette similar to the encoder. Also, if we are at the picture boundary, and there is no palette-coded CU on the left, palette prediction mode is not used.

3.2. Scheme B: with motion vector

In general, there can be a practical hardware limitation that the Largest Coding Units (LCU's) in a video frame need to be encoded/decoded in parallel. In such a case, there may be no palettes available for prediction at the left boundary of LCU. However, in such a case, there may be numerous previously palette-coded blocks in the current LCU which can serve as prediction. An example is shown in Fig. 7(a). Here current coding block is marked as $x(n)$ and all previously palette-coded blocks are marked as Y. In such a case, the encoder searches through all the available palettes for the coding block and uses the one which results in least Rate-Distortion cost.

For implementation purposes, we label the coding blocks in the LCU in zig-zag order as defined in HEVC [10], as in Figure 7(b). As a result, only one dimension motion vector is required. Given current coding block index n, and the best predicted palette index m, the motion vector is $k = n - m - 1$, since n is always greater than m. We use Exponential-Golomb coding with order 3 to encode k based on the empirical distribution of the motion vector k.



(a) Scheme B (b) Block zig-zag order of (a)

Fig. 7. Palette prediction with motion vector.

4. EXPERIMENTAL RESULTS

We encoded full length sequences (which had 200 to 600 frames) and various resolutions varying from 1280x720 to 1920x1080 at QP's 22, 27, 32, and 37 for the lossy scenario, and QP=0 for lossless scenario. For our simulations, we used HM12.1+RExt5.1 [11], the latest HEVC Range Extensions reference software. All the sequences used in our simulations are in YUV color space and 4:4:4 sampled. In the current ITU-T and ISO/IEC MPEG standardization for screen content tools, compression gains on YUV 444 sequences are considered important, and hence we show the results on YUV 444 sequences only. Note that the state-of-the-art tools, such as transform skip, sample adaptive prediction and intra block copy are already implemented in the reference software, and are enabled in our experimental tests.

We first present the results of palette coding without proposed prediction [4] in Table 2 for the following 3 settings: All Intra (AI), Random Access (RA), and Low-Delay (LB) settings as specified in [12] and [13]. In the AI setting, all the images were encoded as Intra, while RA setting had periodic Intra frames; and the LB settings had only the first frame as Intra. These video sequences are being tested as part of HEVC standardization. Full details about the GOP size, Intra period, coding structure of these video sequences etc. are available in [12] and [13]. Note that the gains of palette coding on some sequences are huge, and modest on some others as is generally the trend with many compression tools on various video sequences. The reason is that for the sequences with lower gains, such as WordEditing, and Programming, intra block copy already provides sufficient gain, and palette coding on top of intra block copy for these sequences has modest gains. Also, note that we do not present the gains of palette coding on natural camera captured sequences, since the gains are minimal as they have a richer set of colors. We refer the interested reader to [4] for detailed results on natural sequences.

Next, we present the results of our two proposed palette-prediction techniques. Tables 3 and 4 show the bit rate savings in lossless and lossy settings for our proposed palette prediction schemes on top of the palette coding results in Table 2. From these results, we find that coding gains of upto 10.9% for lossless, and 7.2 % for lossy settings respectively are obtained for All Intra setting of Scheme A. The corresponding gains for Scheme B are 3.5% and 3.2%, which is expected since Scheme B only has the current LCU as search space for the palette prediction. In addition, in Scheme B, additional bits for the motion vector need to be transmitted to the decoder. due to possible hardware-friendly implementation. Nevertheless, it should be noted that both prediction schemes A and B provide substantial gains. In fact, the additional gain of palette prediction on top of palette coding is almost 50 % more (e.g., compression gain for WebBrowsing scheme is 19.4% in All Intra

Table 2. BD-bitrate savings from [4] for Luma component (in %'s) for palette coding over HM12.1+RExt5.1 software . Note that negative BD-Rate means compression gain.

Lossless	AI	RA	LB
WebBrowsing 1280x720	-19.4	-17.3	-13.7
PptDocXls 1920x1080	-27.0	-26.7	-24.6
Map 1280x720	-1.2	-2.4	-1.3
WordEditing 1280x720	-0.9	0.6	-0.1
Programming 1280x720	-0.5	0.0	0.0
Lossy	AI	RA	LB
WebBrowsing 1280x720	-5.0	-5.0	-2.2
PptDocXls 1920x1080	-9.3	-8.2	-8.5
Map 1280x720	-1.5	-1.7	-0.8
WordEditing 1280x720	-1.4	-0.9	-0.6
Programming 1280x720	-1.3	-1.1	-0.6

for palette coding, and palette prediction further provides 10.9% compression gain on top of it). Hence, it is obvious that palette prediction methods should be used with palette coding techniques. Which technique out of A or B, or a combination, or derivative to use depends on the software and hardware complexity, and compression gain trade-off, and will be possibly explored in screen content coding standardization in 2014/2015.

Table 3. Lossless: BD-bitrate savings for Luma component (in %'s) for the proposed palette prediction schemes over HM12.1+RExt5.1 + Palette coding as baseline. Note that negative BD-Rate means compression gain.

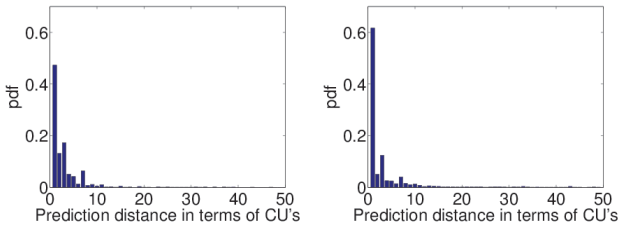
	Sequence	AI	RA	LB
Scheme A	WebBrowsing 1280x720	-10.9	-11	-10.7
	PptDocXls 1920x1080	-5.8	-5.8	-6.6
	Map 1280x720	-1.2	-0.5	-0.3
	WordEditing 1280x720	-1.3	-0.8	-0.3
	Programming 1280x720	-0.3	-0.4	0.0
Scheme B	WebBrowsing 1280x720	-3.5	-3.6	-3.7
	PptDocXls 1920x1080	-3.4	-3.4	-3.6
	Map 1280x720	-1.2	-0.5	-0.3
	WordEditing 1280x720	-0.4	-0.2	-0.2
	Programming 1280x720	-0.2	0.0	0.0

4.1. DISCUSSION

In our proposed Scheme A, the current CU palette can be predicted from previously encoded palettes on the left. Different to the palette reusing in [5, 6], our proposed palette prediction framework efficiently finds better palette predictions. We also plot the histogram of successful palette prediction in Fig. 8 for two sequences, averaged across all frames in AI, RA and LD settings in lossless scenario. The x-axis denotes the distance of the current CU to the (palette) predictor CU on the left. Note that 50-60 % of the predictor CU's are at distance 1, i.e., prediction is happening from the direct left CU. But there are significant number of CU's at other location on left from which prediction can be derived as shown in the figure.

Table 4. Lossy: BD-bitrate savings for Luma component (in %'s) for the proposed palette prediction schemes over HM12.1+RExt5.1 + Palette coding as baseline. Note that negative BD-Rate means compression gain.

	Sequence	AI	RA	LB
Scheme A	WebBrowsing 1280x720	-7.2	-8.0	-3.8
	PptDocXls 1920x1080	-6.6	-7.6	-5.2
	Map 1280x720	-1.6	-1.0	-0.4
	WordEditing 1280x720	-0.9	-0.8	-0.7
	Programming 1280x720	-0.9	-0.4	-0.4
Scheme B	WebBrowsing 1280x720	-1.4	-1.7	-0.8
	PptDocXls 1920x1080	-3.2	-4.0	-2.3
	Map 1280x720	-0.5	-0.5	-0.2
	WordEditing 1280x720	-0.2	0.0	-0.7
	Programming 1280x720	-0.4	-0.2	0.2



(a) WebBrowsing 1280x720 (b) PptDocXls 1920x1080

Fig. 8. Palette prediction distance in terms of CU's.

These CU's remarkably improve the compression efficiency as we have seen from the experimental results.

The additional complexity of Scheme A at both the encoder and decoder in terms of run-time is negligible, since only one additional palette prediction mode is tested in the R-D optimization in addition to other modes, such as 35 intra prediction directions, inter etc. For Scheme B, there is additional R-D search cost at the encoder for finding the *best* palette in the LCU and this leads to around 10-15 % increase in encoder run-time, but the decoder has negligible complexity since it uses the motion vector to find from where to predict the palette.

5. CONCLUSION

In the recent ongoing version 2 of HEVC standardization for Range Extensions, and future screen content coding work for HEVC extensions in 2014/2015, palette coding has emerged as a promising compression tool which can provide remarkable gains on screen content video coding. In this paper, we present two palette prediction methods which can further increase palette coding gains significantly. Our first proposed scheme simply uses the palette of the last left palette coded CU as predictor; and attains upto 10.9% bitrate savings over palette coding. In the second proposed scheme for palette prediction, we restrict ourselves to search only in the current largest coding unit for a hardware-friendly implementation. The second scheme achieves around 5.0% compression gain. Overall, we show that efficient palette prediction schemes used in conjunction of palette coding can provide significant compression gains for screen content coding. Future work includes finding the optimal compression, and complexity point for these prediction schemes.

6. REFERENCES

[1] J. Sole, D. Kwon E. Alshina, and W. Peng, "Summary report on HEVC Range Extensions Core Experiment 3 (RCE3) on Intra block copy refinement," *JCTVC-P0034, San Jose, USA*, Jan 2014.

[2] S. Hu, R. Cohen, A. Vetro, and C. C. J. Kuo, "Screen content coding for HEVC using edge modes," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2013, pp. 1714–1718.

[3] M. Mrak and J. Xu, "Improving screen content coding in HEVC by transform skipping," in *Proceedings of the 20th European Signal Processing Conference (EUSIPCO)*, 2012, pp. 1209–1213.

[4] L. Guo, M. Karczewicz, J. Sole, and R. Joshi, "Evaluation of palette mode coding on HM-12.0+RExt-4.1," *JCTVC-O0218, Geneva, Switzerland*, October 2013.

[5] X. Guo, B. Li, J. Xu, Y. Lu, S. Li, and F. Wu, "AHG8: Major-color-based screen content coding," *JCTVC-O0182, Geneva, Switzerland*, October 2013.

[6] L. Guo, M. Karczewicz W. Pu, R. Joshi J. Sole, and F. Zou, "RCE4: Results of test 2 on palette mode for screen content coding," *JCTVC-P0198, San Jose, USA*, January 2014.

[7] Z. Li and A. Katsaggelos, "A color vector quantization based video coder," in *IEEE International Conference on Image Processing*, 2002, vol. 3, pp. 673–676.

[8] G. Bjontegard, "Calculation of average PSNR differences between rd-curves," *ITU-T VCEG-M33, Austin, TX*, April 2001.

[9] I. E. Richardson, *The H.264 Advanced Video Compression Standard*, Wiley, 2010.

[10] G. J. Sullivan, J. Ohm, W. Han, and T. Wiegand, "Overview of the High Efficiency Video Coding (HEVC) Standard," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 22, no. 12, pp. 1649–1668, Dec 2012.

[11] ITU-T and ISO-IEC: JCTVC, "HEVC range extensions test model (HM) http://hevc.kw.bbc.co.uk/git/w/jctvc-hm.git/commit/0edaa66758a088f53a2b8f5dd93bd0c34fcd7c75," 2013, Online.

[12] D. Flynn, K. Sharman, and C. Rosewarne, "Common test conditions and software reference configurations for HEVC range extensions," *JCTVC-N1006, Vienna, Austria*, July 2013.

[13] 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.