2-D Index Map Coding for HEVC Screen Content Compression

Yiling Xu[†], Wei Huang[†], Wei Wang[‡], Fanyi Duanmu[◇], and Zhan Ma[*]

[†] Shanghai Jiaotong University, Shanghai, China
[‡] FutureWei Technologies, Santa Clara, USA
[◇] New York University, New York, USA
[*] Nanjing University, Nanjing, China (*corresponding author*)

Abstract

This paper introduces a 2-D index map coding of the palette mode in screen content coding extension of the High-Efficiency Video Coding (HEVC SCC) standard to further improve the compression performance. In contrast to the current 1-D search using RUN to represent the length of matched string, we bring the block width and height to describe the arbitrary rectangle shape. We also use the block vector displacement to signal the matched block distance efficiently. By enlarging the search range from current coding tree unit (CTU) to a small neighbor CTU window (i.e., 3×5 CTUs), it provides the coding efficiency comparable to the case that full-frame intra block copy is used. It is more practical to use the local search window in real life considering the trade-off between the coding efficiency and implementation cost.

1 Introduction

Cloud based screen application, such as shared screen collaboration, virtual desktop interface, gaming, etc, have drawn more and more attention in practice. Allowing users to connect the super computer (locally or globally) via their light weight ultra book, or even a touch screen display to manage and process the daily work enables the pervasive computing as long as one has stable network access. It also provides more secure workplace since it is no longer required to distribute information to hundreds and thousands of individual clients. Instead, all information is cached inside super computers (or data center). Basically, the challenging problem is how to compress the screen efficiently to provide the high fidelity representation at client with the limited bandwidth of underlying access network. There are few successful proprietary solutions, such as Citrix HDX, Microsoft RemoteFX, Teradici PCoIP, etc. On the other hand, this promising wave also motivates the standardization of screen content compression on top of the well-designed HEVC [1]. Such standardized resolution could enable the fully interoperability for a massive market, given that video codec is already deployed into the billions of devices.

A computer screen picture mixes discontinuous-tone content such as text, icon, and graphics, and continuous-tone content such as video sequences and images with camera captured natural content [2]. Discontinuous-tone and continuous-tone content typically have distant statistical distributions which generally require quite different methods for efficiently representation. Current HEVC could efficiently deal with the

1068-0314/15 $31.00 © 2015 IEEE
DOI 10.1109/DCC.2015.62

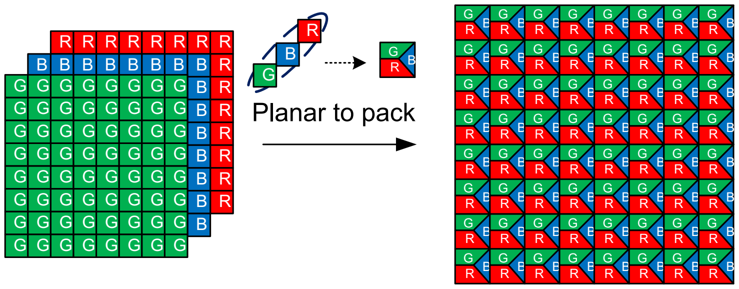

Figure 1: Interleaved planar color components in pack fashion for coding unit color table and index derivation. GBR is used as an example. YCbCr will follow the same fashion for processing.

camera captured content. To compress the discontinuous-tone samples, additional tools have been developed, such as intra block copy (IBC) [3], [4], [5], color palette [6], [7], [8], [9], [10], [11], [12], and etc.

This paper focuses on the improvement of color palette coding by introducing the 2-D index map coding scheme. Generally, on top of current 1-D search based palette coding method adopted in HEVC SCC draft 1 [13], we have developed another 2-D index map coding to signal the matched string block using its block vector displacement (i.e., bvx and bvy) and area (i.e., width and height). To greatly leverage the local neighbor correlation, we allow the 2-D index search using reconstructed neighbors, i.e., from left CTUs only to a $m \times n$ search window. Simulations have been carried out extensively to demonstrate the coding efficiency. With the support of a 3×5 local search windows for both intra block copy and proposed 2-D index map coding, it even provides the comparable coding efficiency in comparison to the full frame intra block copy. Apparently, it requires much less resource using the local search windows. 2-D index map coding could be further improved, i.e., mainly about the complexity reduction, using the ideas in conventional motion estimation to constrain the search range and points. But this is deferred as our future focus.

The rest of this paper is organized as follows: a short review of color palette mode in HEVC SCC work draft (WD) is briefed in Section 2. Section 3 presents the motivation and resolution for the development of proposed 2-D index map coding followed by the performance evaluation and comparison in Section 4. Finally, conclusions are drawn in Section 5.

2 Color Palette Coding in HEVC SCC: A Review

Color table/palette method was studied almost two decades ago [14]. It was revisited and found to be another attractive technology for screen content coding [6], [15], [7], [8], [16], [12] which is under the observations that non-camera captured content typically contains a limited number of distinct colors, rather than the continuous color tone in natural videos. Neighboring colors in screen content generally have larger difference in terms of the intensity, hence conventional spatial intra prediction could not compress efficiently (since the residual is still very significant). By applying the

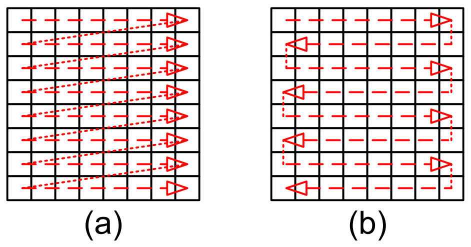

Figure 2: (a) Horizontal scan and (b) horizontal traverse scan for a 8×8 CU. Vertical scan or vertical traverse scan can be realized by transposing the input data.

color mapping using derived table or palette, original image block will be converted to a block of indices with reduced dynamic range that will be easier for prediction and compression. Many attempts have been made to improve the performance of color table coding, including color table differential coding, index map coding using run-length code, line based direct copy [9] and etc. After the joint efforts to resolve coding efficiency and complexity issues among several meeting cycles, a combined solution [12] is suggested to the HEVC screen content coding standardization committee and adopted into the HEVC SCC work draft revision 1 [13], and reference software SCM 2.0 [17].

Basically, color palette coding in HEVC SCC involves color table and index map processing. Interleaved color components of each pixel are used for 4:4:4 content shown in Fig. 1, resulting in one color table with the triplet of R, G, B or Y, U, V components and one color index map for any coding unit (CU). Note that such interleaved pattern is also applied in string search based dictionary coding [2], [5], [18].

More specifically, there are two color tables (or palettes) used in current palette mode. One is for current CU and the other one is the reference storing the historical colors often appeared. This reference table is also called palette predictor [12]. In current design, the maximum sizes of current palette and reference table are fixed to 31 and 64 respectively. Two types of color table prediction is developed to greatly leverage the reference table for implicit signaling, i.e.,

- Color Re-using is performed by comparing the color entry in both tables one-by-one;
- Color Sharing is re-using all colors from previous palette coded CU.

For those colors that could not be re-used (individually) or shared (completely), will be encoded explicitly. This happens often from one region to another in a screen where the colors for a specific region might be different from another one.

On the other hand, original CU in pixel domain is then translated to a corresponding index map through derived color table. In current design adopted in HEVC SCC work draft, a RUN-based index coding is applied. For each index, it is going to check whether it satisfies the condition of COPY_ABOVE, INDEX_MODE or ESCAPE, i.e.,

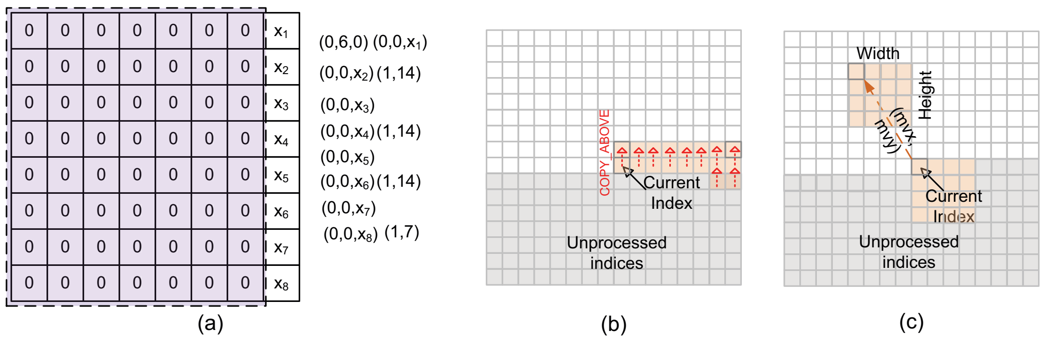

Figure 3: Examples of (a) a 8×8 index map and its encoded pairs after 1-D RUN-based string search: $x_i, i \in [1, 8]$ are other index values different from 0; (b) 1-D RUN-based string match (COPY_ABOVE and RUN); (c) 2-D flexible string match distance (mvx, mvy) and block width and height.

- COPY_ABOVE indicates that a string of indices is identical to the corresponding ones at above line starting from the current index position where the number of the matched indices is represented by the RUN.
- INDEX_MODE indicates that a string of indices is the same from the current index position where the number of the identical indices is also signaled by the `RUN`. This also covers the case that the index appears first time (with `RUN` $= 0$).
- ESCAPE is signaling the colors that have large intensity difference from those inside the color table. Its index value is assigned using actual maximum palette size of current CU, and actual quantized value is then signaled explicitly. Note that whenever ESCAPE color occurs, COPY_ABOVE or INDEX_MODE is stopped.

Also it is worth to mention that both horizontal and vertical traverse scanning are used to well explore the local redundancy within the 2-D index map shown in Figure 2. For a comparison, horizontal scan is illustrated as well.

Intuitively, such a RUN-string based 1-D index map coding is a special case of general string search based solutions described in [10] and [19], where COPY_ABOVE stands for the case that matched distance equals to the CU width and matched string length equals to the RUN, INDEX_MODE represents the matched string with n identical indices (n = RUN). We normally encode (RUN -1) rather RUN itself into the stream for bit saving. For INDEX_MODE, RUN = 1 is same as the unmatched index scenario studied in [20].

3 2-D Index Map Coding: Motivation and Solution

Fig. 3a shows an index map collected from the real test sequences, where encoded pairs after 1-D RUN-based string search are exemplified as well. There is still redundancy among match pairs for encoding from second row to the bottom, i.e., except the last position of each row using the INDEX_MODE (COPY_ABOVE = 0, RUN = 0, x_i)

$i \in [2, 8]$, other rows are encoded using matched pair (COPY_ABOVE = 1, RUN) with RUN depends on the exact number of indices matched following the pre-defined traverse scanning pattern. It will require less bits if we can signal the 8×7 zero block using a 2-D representation, i.e., (2DFlag = 1, `bvx`, `bvy`, `width`, `height`) rather than multiple 1-D matched pairs. Such case could be quite often if the content exhibit similar block pattern, and motivates the hybrid 1-D and 2-D string search design to further improve the coding efficiency. For the 1-D part, current RUN-string based method is kept without change. An additional `2DFlag` is introduced to signal whether a 2-D block or a 1-D string is used for matched string representation, i.e.,

- if `2DFlag` is false, syntax elements, such as COPY_ABOVE flag, RUN value and index value (if applicable)[1], are encoded;
- if `2DFlag` is true, syntax elements, such as `bvx`, `bvy`, `width` and `height`, are coded.

Both 1-D and 2-D string searches are performed for the indices in current CU. For the 1-D case, search engine records RUN value (number of matched indices) as shown in Fig. 3b. For the 2-D case, search engine records matched block width and height and distance as well, as illustrated in Fig. 3c, where the distance is interpreted as the 2-D block vector displacement (`bvx`, `bvy`).

Whether to choose 1-D RUN-string or 2-D block depends on the length of 1-D RUN-string and the area of 2-D matched block[2], i.e., if $\texttt{RUN} \geq \texttt{width}\times\texttt{height}$, 1-D string is selected by signaling (`2DFlag`=0, `COPY_ABOVE flag`, `RUN`, `INDEX`), otherwise, 2-D block is chosen by sending (`2DFlag`=1, `bvx`, `bvy`, `width`, `height`) to the decoder. In case that multiple 1-D or 2-D matches found for current index, the one providing the largest size will be chosen and encoded into the stream. In the current design, both block width and block height are larger than 4, i.e., the smallest matched 2-D string is a 4×4 block.

It often happens that a 1-D RUN-string has to cross the already matched 2-D string block when we perform the hybrid 1-D and 2-D string search. This is shown in Figure 4. We use a shadowed block to represent the matched 2-D string. Staring from the current index X, it continues the INDEX_MODE until reaching the last X as exemplified. Since X_i (i = 1, 2, 3, 4) are already covered by the previous 2-D matched string, they are ignored and the number is not counted into the RUN for this matched 1-D X string.

We have constrained the 1-D RUN-string search within current CU without requiring additional on-chip memory cache (except for color table storage). For hybrid 1-D/2-D string search, we extend the search range from current CTU to include previously coded CTUs. Note that different search range settings will have various impact on both coding efficiency and complexity (such as computational resource and memory bandwidth). Note that extended reference buffers are only used to the 2-D search, 1-D RUN-string search is still constrained within current CU.

[1]Escape colors are encoded as well, which is not discussed in details to emphasize the focus on

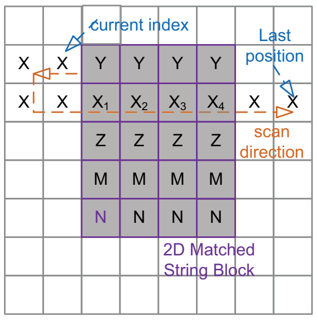

Figure 4: Hybrid 1-D and 2-D string search: indice are ignored if they are already covered by the previous match string.

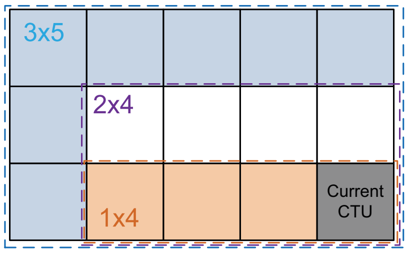

Figure 5: Illustration of a $m \times n$ search range for 2-D index map coding and intra block copy.

4 Performance Evaluation and Comparison

In this section, we have studied performance and complexity of the proposed 2-D index map coding method. We compared to the latest HEVC screen content coding reference software SCM 2.0 [17]. Different search ranges for intra block copy and proposed 2-D index map coding are evaluated, such as full frame intra block copy (FF-IBC), intra block copy using local search window ($m \times n$-IBC) and 2-D index map coding using local search window ($m \times n$-2D), respectively. Anchor is configured to use 1×4 search range for IBC. All other parameters are following the common test conditions [21]. Given that 2-D index map coding tool is generally developed for intra frame and screen content application is mainly for low-delay scenario, we perform the evaluation and comparison for All Intra and Low-Delay using B-picture scenarios.

Experiments are conducted using the screen content sequences selected by the experts in Joint Collaborative Team on Video Coding group [22]. These sequences are chosen to represent popular and typical screen content application scenarios. For instance, one category is for the text and graphics with motion representing the typical remote desktop applications, including “FlyGraphicsText”, “Desktop”, “Console”, “WebBrowsing”, “Map”, “Programming” and “SlideShow”. Another category is the mixed content for the most common screen in our daily life that contains texts, graphics as well as the camera-captured video, such as “MotionControlClip2”,

the 1-D and 2-D string based index map coding.

[2]Apparently, rate-distortion based 1-D or 2-D decision could be developed to improve the performance compared with current heuristic scheme. It is our future focus of study.

Table 1: Performance Evaluation and Comparison (BD-Rate) of 2-D Index Map Coding and Intra Block Copy

		RGB		YUV 444	
		text/graphics	mixed	text/graphics	mixed
AI	FF-IBC	-14.6%	-12.1%	-15.6%	-12.7%
	1x4-IBC+1x4-2D	-3.0%	-0.3%	-2.4%	-0.4%
	2x4-IBC	-9.0%	-5.8%	-9.7%	-6.9%
	2x4-IBC+2x4-2D	-11.8%	-6.5%	-11.8%	-7.5%
	3x5-IBC	-12.0%	-9.0%	-13.0%	-10.1%
	3x5-IBC+3x5-2D	-14.7%	-9.7%	-14.9%	-10.7%
LB	FF-IBC	-4.7%	-3.4%	-4.5%	-4.0%
	1x4-IBC+1x4-2D	-1.3%	-0.4%	-1.4%	-0.6%
	2x4-IBC	-2.5%	-1.9%	-2.2%	-2.4%
	2x4-IBC+2x4-2D	-4.9%	-2.3%	-4.5%	-2.8%
	3x5-IBC	-3.6%	-2.5%	-3.3%	-2.9%
	3x5-IBC+3x5-2D	-6.5%	-3.1%	-6.0%	-3.6%

"MotionControlClip3", and "BasketballScreen". All sequences have both 8-bit RGB and YCbCr (or YUV) version with full chroma sampling resolution. There are another two categories, i.e., animation and camera-captured content. However, they do not benefit too much. We do not present here to save space. Simulation results are mainly averaged for text/graphics and mixed content with both RGB and YUV format, regardless of the test video resolutions.

4.1 Performance Evaluation and Comparison

Given that proposed 2-D index map coding utilized the reconstructed neighbors, we have experimented several options as follows:

- 1x4-2D: only 3 left CTUs of current CTU are used as reference;
- 2x4-2D: 3 left CTUs plus 4 upper CTUs are used as reference;
- 3x5-2D: 4 left CTUs, total 10 upper CTUs are used as reference.

Figure 5 pictures the $m \times n$ search range for both 2-D and IBC. Note that we will use 1x4 IBC as the anchor to illustrate the coding efficiency improvement from the 2-D index map coding. IBC will share the same search range as 2-D. For instance, if we use 3x5 search window for 2-D, IBC will choose the same search window. It is also worth to mention that left CTU neighbors are all using full search rather than hash based fast search. For meaningful comparison, we also give the results of FF-IBC, 2x4-IBC and 3x5-IBC.

Table 1 presents the averaged BD-Rate improvement over the anchor. Meanwhile, we also visualize the data in Figure 6. According to the results, with the local search for both intra block copy and proposed 2-D index map coding, it could provide comparable performance as the full frame intra block copy. Apparently, full frame intra

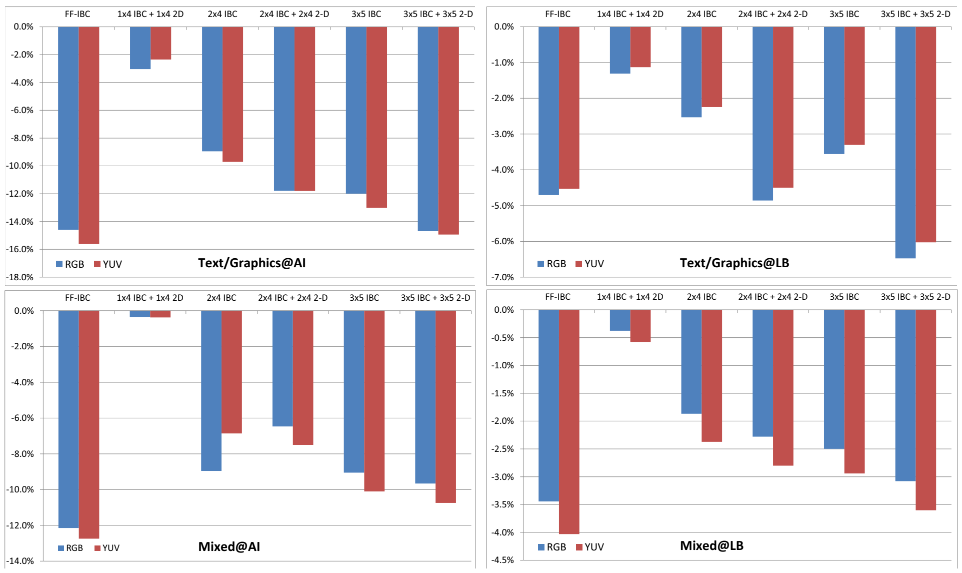

Figure 6: Performance comparison of FF-IBC, 1x4-IBC + 1x4-2D, 2x4-IBC, 2x4-IBC+2x4-2D, 3x5-IBC, 3x5-IBC+3x5-2D for AI and LB configuration

block copy will require much more resource. Also, it has to use local search window for practical implementation. Almost 2% relative loss is reported for 3x5-IBC+3x5-2D over FF-IBC with AI coded mixed content, but much less loss is observed for AI coded text/graphics videos. For LB, relative loss is less than 0.5% for mixed content, however combined IBC and 2-D with local 3$\times$5 search window gives better performance for text/graphics content.

4.2 Complexity Discussion

The complexity is measured using the encoder running time. All simulations are performed using uniform computing nodes. On top of the anchor where 1x4 IBC is used, it requires 1.5% more running time when we add one more reconstructed CTU as reference for 2-D index map coding. For instance, 1x4-IBC+1x4-2D requires 6% more running time, 3x5-IBC+3x5-2D demands another 22% running time overhead approximately, both against to the 1x4-IBC anchor. It is worth to mentioned that there are ways to optimize the 2-D search which are not studied yet in this paper, for example, leveraging the motion clustering idea in traditional motion search to constrain the search area and the search points.

5 Concluding Remarks

This paper introduces a 2-D index map coding of the palette mode in screen content coding extension of the High-Efficiency Video Coding (HEVC SCC) standard to further improve the coding efficiency. On top of current RUN-based 1-D search method, we bring the 2-D index block match by signaling its block vector displacement (i.e., bvx and bvy) and block area (i.e., width and height). We allow the 2-D index search to use neighbor CTUs after reconstruction and extensive simulations demonstrate that the coding efficiency could be close to the full frame block copy with the local search for both intra block copy and proposed 2-D index map coding. In comparison to full frame intra block copy, combined IBC and 2-D with a local 3×5 search window offers better performance for text/graphics content, but slightly loss for mixed content (i.e., less than 0.5% relatively). It suggests that the combined intra block copy and 2-D index map coding could offer the decent trade-off between coding efficiency and implementation cost by using the local search window, and apparently this is a competitive solution for practical product.

Acknowledgement

The work was supported by the Natural Science Foundation of the Jiangsu Province for Young Scholars (No. SBK2014041701).

References

[1] G.-J. Sullivan, J.-R. Ohm, W.-J. 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.

[2] T. Lin, P. Zhang, S. Wang, K. Zhou, and X. Chen, "Mixed chroma sampling-rate high efficiency video coding for full-chroma screen content," *IEEE Trans. Circuits and Systems for Video Technology*, vol. 23, no. 1, pp. 173–185, January 2013.

[3] D.-K. Kwon and M. Budagavi, *RCE3: Results of test 3.3 on Intra motion compensation*, Doc. JCTVC-N0205, July 2013.

[4] C. Pang, J. Sole, L. Guo, M. Karczewicz, and R. Joshi, *Non-RCE3: Intra Motion Compensation with 2-D MVs*, Doc. JCTVC-N0256, July 2013.

[5] J. Chen, Y. Chen, T. Hsieh, R. Joshi, M. Karczewicz, W.-S. Kim, X. Li, C. Pang, W. Pu, K. Rapaka, J. Sole, L. Zhang, and F. Zou, *Description of screen content coding technology proposal by Qualcomm*, Doc. JCTVC-Q0031, Valencia, ES, April 2014.

[6] L. Guo, M. Karczewicz, and J. Sole, *RCE3: Results of Test 3.1 on Palette Mode for Screen Content Coding*, Doc. JCTVC-N0247, July 2013.

[7] W. Zhu, J. Xu, and W. Ding, *RCE3 Test 2: Multi-stage Base Color and Index Map*, Doc. JCTVC-N0287, July 2013.

[8] Z. Ma, W. Wang, M. Xu, X. Wang, and H. Yu, *Description of screen content coding technology proposal by Huawei Technologies (USA)*, Doc. JCTVC-Q0034, Valencia, ES, April 2014.

[9] W. Pu, X. Guo, P. Onno, P. Lai, and J. Xu, *AHG10: Suggested Software for Palette Coding based on RExt6.0*, Doc. JCTVC-Q0094, Valencia, ES, April 2014.

[10] Z. Ma, W. Wang, M. Xu, and H. Yu, *SCCE3 Test C.3: Combined Color Table and Index Map Coding of A.4 and B.8*, Doc. JCTVC-R0144, Sapporo, JP, July 2014.

[11] W. Wang, Z. Ma, M. Xu, F. Duanmu, and H. Yu, *Non-SCCE3: Results of SCCE3 with support of 2x4 CTUs as reference buffer*, Doc. JCTVC-R0268, Sapporo, JP, July 2014.

[12] P. Onno, X. Xiu, Y.-W. Huang, and R. Joshi, *Suggested combined software and text for run-based palette mode*, Doc. JCTVC-Q0348, Sapporo, JP, July 2014.

[13] R. Joshi and J. Xu, *HEVC Screen Content Coding Draft Text 1*, Doc. JCTVC-R1005, Sapporo, JP, July 2014.

[14] A. Zaccarin and B. Liu, "A novel approach for coding color quantized images," *IEEE Trans. on Image Processing*, vol. 2, no. 4, pp. 442 – 453, Oct. 1993.

[15] L. Guo, M. Karczewicz, J. Soel, and R. Joshi, *Non-RCE3: Modified Palette Mode for Screen Content Coding*, Doc. JCTVC-N0249, July 2013.

[16] W. Wang, Z. Ma, M. Xu, X. Wang, and H. Yu, *AHG8: String match in coding of screen content*, Doc. JCTVC-Q0176, Valencia, ES, April 2014.

[17] R. Joshi, J. Xu, R. Cohen, S. Liu, Z. Ma, and Y. Ye, *Screen Content Coding Test Model 2 Encoder Description (SCM 2)*, Doc. JCTVC-R1014, Sapporo, JP, July 2014.

[18] J. Ye, S. Liu, S. Lei, X. Chen, L. Zhao, and T. Lin, *Improvements on 1D dictionary coding*, Doc. JCTVC-Q0124, April 2014.

[19] Z. Ma, W. Wang, M. Xu, and H. Yu, "Advanced screen content coding using color table and index map," *IEEE Trans. on Image Processing*, vol. 23, no. 10, pp. 4399 – 4412, Oct. 2014.

[20] Z. Ma, W. Wang, M. Xu, F. Duanmu, and H. Yu, *Non-SCCE3: Improvements for SCCE3 Test C3*, Doc. JCTVC-R0304, Sapporo, JP, July 2014.

[21] H. Yu, R. Cohen, K. Rapaka, and J. Xu, *Common Test Conditions for Screen Content Coding* , Doc. JCTVC-R1015, Sapporo, JP, July 2014.

[22] M. N14715, *Joint Call for Proposals for Coding of Screen Content*, Jan. 2014.