Sparse Coding of Intra Prediction Residuals for Screen Content Coding

Michael G. Schimpf
Computer Science and Engineering
Santa Clara University
Santa Clara, USA
mschimpf@scu.edu

Nam Ling
Computer Science and Engineering
Santa Clara University
Santa Clara, USA
nling@scu.edu

Yunhui Shi
Key Lab of Multimedia and Intelligent Software Technology
Beijing University of Technology
Beijing, China
syhzm@bjut.edu.cn

Ying Liu
Computer Science and Engineering
Santa Clara University
Santa Clara, USA
yliu15@scu.edu

2021 IEEE International Conference on Consumer Electronics (ICCE) | 978-1-7281-9766-1/20/$31.00 ©2021 IEEE | DOI: 10.1109/ICCE50685.2021.9427722

***Abstract*—High Efficiency Video Coding - Screen Content Coding (HEVC-SCC) is an extension to HEVC which adds sophisticated compression methods for computer generated content. A video frame is usually split into blocks that are predicted and subtracted from the original, which leaves a residual. These blocks are transformed by integer discrete sine transform (IntDST) or integer discrete cosine transform (IntDCT), quantized, and entropy coded into a bitstream. In contrast to camera captured content, screen content contains a lot of similar and repeated blocks. The HEVC-SCC tools utilize these similarities in various ways. After these tools are executed, the remaining signals are handled by IntDST/IntDCT which is designed to code camera-captured content. Fortunately, in sparse coding, the dictionary learning process which uses these residuals adapts much better and the outcome is significantly sparser than for camera captured content. This paper proposes a sparse coding scheme which takes advantage of the similar and repeated intra prediction residuals and targets low to mid frequency/energy blocks with a low sparsity setup. We also applied an approach which splits the common test conditions (CTC) sequences into categories for training and testing purposes. It is integrated as an alternate transform where the selection between traditional transform and our proposed method is based on a rate-distortion optimization (RDO) decision. It is integrated in HEVC-SCC test model (HM) HM-16.18+SCM-8.7. Experimental results show that the proposed method achieves a Bjontegaard rate difference (BD-rate) of up to 4.6% in an extreme computationally demanding setup for the "all intra" configuration compared with HM-16.18+SCM-8.7.**

Keywords— video coding, sparse coding, sparse representation, orthogonal matching pursuit, screen content coding, HEVC, residual coding, intra prediction, KSVD

I. INTRODUCTION

A significant part of HEVC video compression is Intra prediction. Intra signals are split into blocks, each intra block can chooses among 33 angular, DC and planar modes for a prediction of the original signal [1]. After an optimal prediction is found based on RDO, the predicted block is subtracted from the original block. The resulting residual represents the prediction error. The residual is then IntDST/IntDCT transformed, quantized, and entropy coded into a bitstream. HEVC-SCC, especially HM-16.18+SCM-8.7 [2] is an extension to HEVC. It shares the same coding architecture as HEVC along with several new elements, some inherited from HEVC version 1 and some from HEVC-REext. These include the adaptive color transform (ACT), adaptive motion vector resolution (AMVR), cross-component prediction (CCP), intra block copy (IBC), palette mode (PM), residual rotation (RR), residual differential pulse code modulation (RDPCM), and transform skip (TS). The rationale behind these tools is explained in [3]. Its purpose is to improve the compression of graphics, text, animation, and mixed content. Screen content has important properties such as no sensor noise, large uniform flat areas, many repeated patterns, highly saturated and limited colors, high contrast, discrete tones, sharp edges, and high frequencies in certain regions [3]. After applying all HEVC-SCC tools, a significant amount of similar and repeated signals remains in the residual. A reason for this is the IBC tool which integrates the inter motion vector concept into the intra domain. As a result, IBC replaces a lot of similar and repeated intra predictions but keeps the residual transform process as it is.

Sparse coding aims to find a sparse representation of the signal y in the form of a linear combination of basic elements. These elements are called atoms and they compose a dictionary. The dictionary can be found by the K-singular-value decomposition (K-SVD) algorithm [4]. An atom contains the index to the dictionary entry as also the coefficient which represents the magnitude to which the dictionary entry is later linearly added to reconstruct the signal. The proposed method uses Orthogonal Matching Pursuit (OMP) [5] to solve equation (1).

$y \in \mathbb{R}^n$	$\mathbf{y}$ = signal, n = signal dimension
$\mathbf{D} \in \mathbb{R}^{n \times K}$	$\mathbf{D}$ = dictionary, K = # of atoms
$\mathbf{x} \in \mathbb{R}^K$	$\mathbf{x}$ = vector of coefficients

Approximated representation of y:

$$\min_{\mathbf{x}} \|\mathbf{y} - \mathbf{D}\mathbf{x}\|_2 \text{ s.t. } \|\mathbf{x}\|_0 \leq \varepsilon \quad (1)$$

$\|\cdot\|_2$ represents the L2 norm and $\|\cdot\|_0$ the L0 pseudo norm.

Existing sparse coding research covers the topic from different angles. However, sparse coding of similar and repeated intra predicted residuals with low to mid frequency/energy blocks and a low sparsity setup for screen content video compression has not been discussed. Most research applies sparse coding to camera captured inter prediction residuals like [6] which handles anisotropic correlation (high frequency, sharp edges) signals or [7] which puts DCT and sparse coding in a sequence for a two-layered approach (both in HEVC). Work [8] is an example of a video compression paper that used matching pursuit to code inter and intra prediction residuals. It was integrated into the H.264

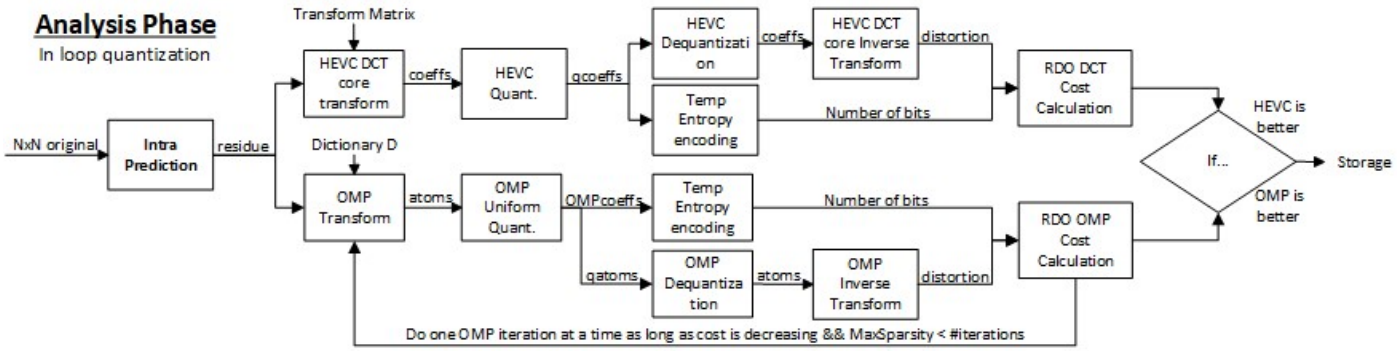

Figure 1. Block diagram – In-loop-quantization RDO Intra prediction residual sparse coding scheme (qcoeffs = IntDST/IntDCT quantized coefficients, qatoms = quantized sparse coefficients and sparse indices)

standard and replaced the core transforms through an adaptive mixed transform scheme. Sparse coding of intra prediction residuals is discussed most in papers which use it in image compression like [9].

In this work, we propose a method integrated into HEVC-SCC to compress intra prediction residuals with an alternate sparse coding transform and an offline dictionary training process that is adapted for similar and repeated low to mid frequency/energy blocks with a low sparsity setup. In general, HEVC-SCC checks the standard IntDST/IntDCT process and calculates the RDO cost, then the proposed method is executed. If the proposed method's RDO cost is lower than that of the standard process, the atoms of sparse vector x are encoded into a bitstream. The proposed sparse coding solution is set up for a low sparsity configuration which means maximum sparsity is set to 4.

The remainder of the paper is organized as follows: section 2 explains the proposed method, section 3 depicts implementation details, section 4 presents the experimental results, and section 5 gives the conclusion and future work.

II. PROPOSED SPARSE CODING FOR SCC

In HEVC-SCC, all HEVC-SCC and HEVC tools, are executed and RDO calculated, and the best RDO candidate is selected to be encoded into the bitstream. Our proposed method is executed and RDO calculated after this process is done, and only for the IntDST/IntDCT candidates. The IntDST/IntDCT process predict a given block, and calculated the residual block, by subtracting the predicted from the original block, quantize the residual block and entropy code the remaining into the bitstream, as depicted in the upper branch in Figure 1. RDO delivers an optimal approximation by minimizing the R-D cost J(k). The HEVC RDO process is based on [2][10].

$$J(k) = D(k) + \lambda R(k) \quad (2)$$

$$\lambda = 0.57 \times 2^{\frac{QP-12}{3.0}} \quad (3)$$

where k is the number of nonzero coefficients, R(k) is the number of bits after entropy coding, D(k) is the mean-squared error (MSE) between the original and the reconstructed image block, λ represents the Lagrange multiplier in HEVC [2][10] and QP is the quantization parameter.

The proposed sparse coding solution, as depicted in the Figure 1 lower branch, is applied as an alternative transform. It specifically targets similar and repeated intra predicted residuals with a low to mid frequency/energy distribution and uses a low sparsity setup. The solution probes every residual block that has a low to medium absolute sum and uses only a maximum of 4 dictionary atoms. This low maximum sparsity setup is in many cases sufficient enough to be selected by the RDO process over the IntDST/IntDCT transform and saves execution time. Our proposed method uses an unusual in-loop-quantization-scheme which means that quantization is part of the RDO process. HEVC's rate-distortion optimized quantization (RDOQ) [10] is not used for quantizing sparse-coefficients because the process is not well adapted to the one-dimensional sparse coefficients, but the standard HEVC Qstep calculation is used for quantization and dequantization of sparse coefficients:

$$Qstep(QP) = \left(2^{\frac{1}{6}}\right)^{QP-4} \quad (4)$$

where QP represents the quantization parameter which is set in the common test conditions CTC [11] to 22, 27, 32, and 37. After every sparse iteration, the coefficients are quantized and dequantized based on (4), entropy encoded, RDO cost calculated (see formulas (2) and (3)) and finally compared with the previous iteration. For the first iteration, the cost is high because the distortion is high, but it decreases with every iteration and atom until it reaches a minimum. Then, it increases again because the number of bits rises and makes the cost higher even as the distortion decreases. When the minimum is reached, the proposed method stops. If the RDO cost is less than the IntDST/IntDCT RDO cost, the sparse atoms are stored into the HEVC data structures and later encoded into the bitstream.

As a second part of the proposed method, it is important that the offline K-SVD training process [4] is adapted to these similar and repeated low to mid frequency/energy blocks. To achieve this, a higher dictionary size k gives the K-SVD/OMP algorithm [4][5] more overall atoms to choose from plus K-SVD parameter optimizations, which is explained in the implementation details. This decreases the average sparsity. We adopt a category approach applied to the training process based on the common test conditions CTC [11] categories. This better utilizes the uniformity in a specific category because the number bases in these screen content categories are

similar. These categories are animation (A), mixed content (MC), and text and graphics with motion (TGM). Overall, the similarity in the residual blocks training data has a significant impact on the dictionary training process [4]. It better converges to this training setup and creates dictionaries which significantly improve and encourage the sparsity in contrast to camera captured content. It needs less dictionary atoms to come to the same results, which saves bits and makes sparse coding in the RDO process more suitable.

III. IMPLEMENTATION DETAILS

The HM16.18 HEVC CABAC engine is used to entropy code most of the proposed method side information. Every intra predicted residual block is extended by a flag which states whether our method or IntDST/IntDCT transforms are used. All sparse indices are coded in fixed length (CABAC bypass) code because of the uniform distribution. The sparse coefficient coding closely follows the standard "transform coefficient coding" [12] with minor optimizations and own context models. Every QP setting has the same maximum sparsity setting of 4 for every block size, which sets a low upper bound of iterations per sparse execution. This low upper bound saves RDO cost and computational time, as well as emphasizes low to mid frequency/energy blocks since only a limited amount of iterations are executed (maximum 4 iterations).

The training data, which is basically all intra predicted residual blocks with an absolute sum > 0, is created by a separated HM-16.18+SCM-8.7 [2] run. All CTC [11] test sequences are split 20% for training and 80% for testing. These trained files are grouped and randomly shuffled into the mentioned categories (based on CTC [11]) which goes into an offline K-SVD training process [4]. The process itself is optimized for a low sparsity setup, which emphasizes similar and repeated low to mid frequency/energy blocks. The KSVD-hyperparameters are set as follows: dictionary K size is 2,048, the number of atoms for training is set to only 2, and 800 iterations per dictionary are executed. Another optimization is to take the training data from a higher quantization training run than used for the test execution. As a rule, higher quantization setup leads to greater similarity and repetition in the residual training data, which encourages a sparser representation. Overall, there are three dictionaries per category and block size. These offline learned static dictionaries are included in the encoder and decoder, so that only the sparse atoms and not the dictionaries themselves are encoded in a bitstream.

IV. EXPERIMENTAL RESULTS

The proposed solution is implemented on the HEVC reference software HM-16.18+SCM-8.7 [2]. The test setup follows the common test conditions (CTC) [11] which tests 13 screen content sequences in four different QP settings: 22, 27, 32 and 37. Our setup diverts at one point, it uses 20% of the frames of each test sequence for training and the remaining 80% for testing and this separation is strict, so that there are no training data included in the result tables. The proposed method is used in three configurations: a maximum BD setup, a medium BD setup, and a low BD setup. The maximum BD setup, which tries to get as much BD-rate savings as possible while ignoring the execution time, uses the following configuration: "all intra," QP settings 22, 27, 32, and 37, block sizes 4x4, 8x8, 16x16, and 32x32, and 4:2:0 color scheme. In general, for the 4:2:0 color scheme, HEVC-SCC uses only four of seven HEVC-SCC tools: transform skip (TS), residual differential pulse-code modulation (RDPCM), intra block copy (IBC), and palette mode (PM) [3]. In contrast to the maximum BD setup, the medium BD setup focusses more on a lower execution time and is not using block sizes 4x4 and 32x32 and is not sparse coding at the QP 22 setting. Finally, the low BD setup, which is encoding only 8x8 block sizes, skips QP22 and the chroma part.

A. Similarity comparison

We compared the intra residual data between camera captured common test sequences [13] (except ClassF) and screen content common test sequences [11] with the number of repeated residual blocks and the average cosine similarity. All residual blocks from all test sequences are sorted and the cosine similarity for each pair is calculated by formula (5):

$$s(\mathbf{a},\mathbf{b}) = \frac{\sum_n a_n b_n}{|\mathbf{a}||\mathbf{b}|} \ (n = \text{vector dimension}) \qquad (5)$$

where **a** and **b** are neighbors. Then, the average of all results is taken. For camera captured content repeated blocks are 0.2% and the cosine similarity is 0.2, for screen content repeated blocks are 34.4% and cosine similarity is 0.4. As expected, for camera captured content each intra residual block is unique from the others and differs at least slightly. However, it is remarkable that for screen content the repetition (identical) of intra residual blocks is very high, which means 1/3 of all intra residual screen content blocks are duplicates. The cosine similarity is up by 50% from 0.2 to 0.4 for screen content (0 means orthogonal, 1 means match).

B. Test results for maximum BD-rate savings setup

Table 1 – Test results for the maximum BD-rate savings setup of the proposed method for block sizes: 4x4, 8x8, 16x16, and 32x32 and QP settings: 22, 27, 32, and 37, compared to HM-16.18+SCM-8.7.

Category	Luma[1]	Cr[1]	Cb[1]	Enc Time
TGM720&1080	-3.0%	-1.6%	-1.8%	1,018
MixedContent	-9.0%	-5.3%	-5.0%	1,025
Animation	-4.4%	-0.5%	-1.7%	1,034
Average	-4.6%	-2.3%	-2.6%	1,022

Table 1 shows what the maximum upper bound for the proposed method is. This is the maximum setup without consideration for the computational complexity. It is obvious that this setup is not useable since the increase in execution time is more than ninefold higher and the decoder execution time is slightly even higher than the encoder execution time. Nonetheless, it gives an upper limit to what the proposed method is capable of, if complexity did not matter, even though in practice it does. The percentage of proposed method block selection over the block sizes 4x4, 8x8, 16x16, and 32x32 are 3.0%, 19.6% 27.8%, and 30.2%, which is overall 20.2%. The

[1] BD-rate savings (piecewise cubic)

low number for the 4x4 block size reflects that 4x4 only minimally contributes to the overall outcome. In general, bigger block sizes lead to bigger BD-rate savings, but at the same time the execution time exponentially increases.

Table 2 Detailed BD-rate savings of Table 1 broken down by separate test sequences compared to HM-16.18+SCM-8.7.

Video Sequence	Category	Luma[1]	Cr[1]	Cb[1]
FlyingGraphics	TGM1080	0.3%	0.9%	0.9%
Desktop	TGM1080	-2.4%	-1.2%	-1.1%
Console	TGM1080	0.2%	0.6%	0.5%
ChineseEditing	TGM1080	-3.8%	-2.1%	-2.2%
WebBrowsing	TGM720	-6.1%	-5.0%	-5.3%
Map	TGM720	-8.1%	-5.8%	-6.2%
Programming	TGM720	-3.0%	0.3%	-0.5%
SlideShow	TGM720	-1.3%	-0.2%	-0.7%
BasketballScreen	MixedCo.	-15.3%	-9.2%	-8.5%
MissionControl2	MixedCo.	-4.0%	-2.1%	-1.8%
MissionControl3	MixedCo.	-7.8%	-4.7%	-4.9%
Robot	Animation	0.2%	2.9%	1.3%
ChinaSpeed	Animation	-9.0%	-3.9%	-4.8%
Average		-4.6%	-2.3%	-2.6%

Table 2 shows the details for every test sequence of Table 1. It is interesting that through all tests, there will be 3 to 4 outliers which are close to 0% BD-rate savings depending on the test configuration, but the overall relation between the percentages stays close for all setups only the absolute value varies.

Table 3 Number of sparse atoms in different block size.

Atoms	4x4	8x8	16x16	32x32
1	3,116,984	2,873,928	906,312	242,417
2	553,719	1,052,425	686,473	289,251
3	95,908	441,428	286,365	130,763
4	5,307	191,448	562,597	357,931

Table 3 shows the distribution for how many sparse atoms are used in which block size and for all QP settings 22, 27, 32 and 37 summed up. Interestingly, most sparse coded blocks use only one to two sparse atoms. The irregularity in the block size columns 16x16 and 32x32 between atoms 3 to 4 is unknown and could indicate an error or a statistical irregularity because it is not strictly decreasing. The average sparsity for 4x4, 8x8, 16x16, and 32x32 blocks are 1.2, 1.6, 2.2, and 2.6.

C. Test results for medium BD-rate savings setup

Based on RDO performance, we removed block size 4x4 because it does not contribute a lot, block size 32x32 because the sparse coding complexity is extremely high, and QP22 because the amount of side information increases significantly with higher QP [7][8], which makes sparse coded blocks less attractive in the RDO process and lowers the partial BD-rate savings contribution to around -0.1% to -0.2%.

Table 4 Test results on the medium BD-rate savings setup for the proposed method, compared to HM-16.18+SCM-8.7.

Category	Luma[2]	Cr[1]	Cb[1]	Enc Time	Dec Time
TGM720& TGM1080	-1.3%	-0.9%	-1.1%	201%	228%
Mixed Content	-5.1%	-3.5%	-3.1%	199%	164%
Animation	-3.5%	-2.8%	-3.1%	217%	286%
Average	-2.5%	-1.8%	-1.8%	203%	219%

Table 4 shows the results of a medium BD-rate savings setup which takes the time complexity a bit more into consideration. This setup has a BD-rate savings of 2.5%, but still exceeds the encoding time by 103% and the decoding time by 119%.

Table 5 Detailed BD-rate savings of Table 4 broken down by separate test sequences compared to HM-16.18+SCM-8.7.

Video Sequence	Category	Luma[1]	Cr[1]	Cb[1]
FlyingGraphics	TGM1080	-0.1%	0.0%	0.0%
Desktop	TGM1080	-1.1%	-0.5%	-0.6%
Console	TGM1080	0.1%	-0.1%	-0.1%
ChineseEditing	TGM1080	-2.2%	-1.3%	-1.5%
WebBrowsing	TGM720	-1.8%	-1.8%	-1.8%
Map	TGM720	-2.9%	-2.3%	-2.5%
Programming	TGM720	-2.2%	-1.0%	-1.2%
SlideShow	TGM720	-0.3%	-0.3%	-0.6%
BasketballScreen	MixedCo.	-6.1%	-4.0%	-3.4%
MissionControl2	MixedCo.	-3.5%	-2.9%	-2.4%
MissionControl3	MixedCo.	-5.6%	-3.8%	-3.5%
Robot	Animation	-0.1%	0.0%	-0.3%
ChinaSpeed	Animation	-6.8%	-5.6%	-5.9%
Average		-2.5%	-1.8%	-1.8%

Table 5 shows the details for every test sequence of Table 4 and the percentages varies as expected.

D. Test results for low BD-rate savings setup

Table 6 Test results on the low BD-rate savings setup for the proposed method, compared to HM-16.18+SCM-8.7.

Category	Luma[3]	Enc Time	Dec Time
TGM720& TGM1080	-0.3%	123%	117%
Mixed Content	-1.2%	124%	109%
Animation	-1.6%	128%	120%
Average	-0.7%	124%	115%

Table 6 shows the results of the low BD rate setup, which has a BD-rate savings of 0.7%, and the encoding time is increased only by 24% and the decoding time by 15%.

[2] BD-rate savings (piecewise cubic)
[3] BD-rate savings (piecewise cubic)

Table 7 Detailed BD-rate savings of Table 6 broken down by separate test sequences compared to HM-16.18+SCM-8.7 without proposed method.

Video Sequence	Category	Luma[1]
FlyingGraphics	TGM1080	-0.1%
Desktop	TGM1080	-0.3%
Console	TGM1080	0.1%
ChineseEditing	TGM1080	-0.4%
WebBrowsing	TGM720	-0.4%
Map	TGM720	-0.7%
Programming	TGM720	-0.6%
SlideShow	TGM720	-0.3%
BasketballScreen	MixedCo.	-1.5%
MissionControl2	MixedCo.	-1.0%
MissionControl3	MixedCo.	-1.3%
Robot	Animation	-0.1%
ChinaSpeed	Animation	-3.0%
Average		-0.7%

Table 7 shows the details for every test sequence of Table 6 the percentages are as expected and the test sequences: FlyingGraphics, Console, and Robot are outliers and contributes only minimal.

E. Dictionary visualization

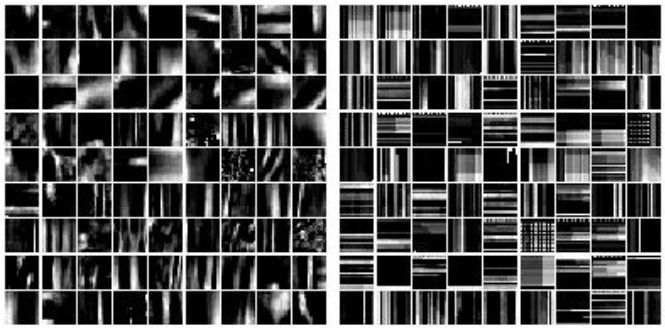

Figure 2. Visual dictionary comparison - animation (left) and TGM1080 (right).

Figure 2 shows two subsets of two dictionaries. As expected, each category develops a slightly different dictionary adapted to the number base. Animation on the left side contains a lot of vertical atoms as well as some unusual, round, and cloud-like patterns. On the right side is a subset of a TGM1080 dictionary which shows almost only straight horizontal and vertical atoms.

F. PSNR vs Bitrate

The proposed method is optimized to lower the bitrate and not mainly to improve PSNR, the reason is that for a higher PSNR, more sparse atoms are usually needed to achieve this, but our method focuses on low sparsity where the maximum sparsity is only 4, which is the reason why QP22 is not performing well.

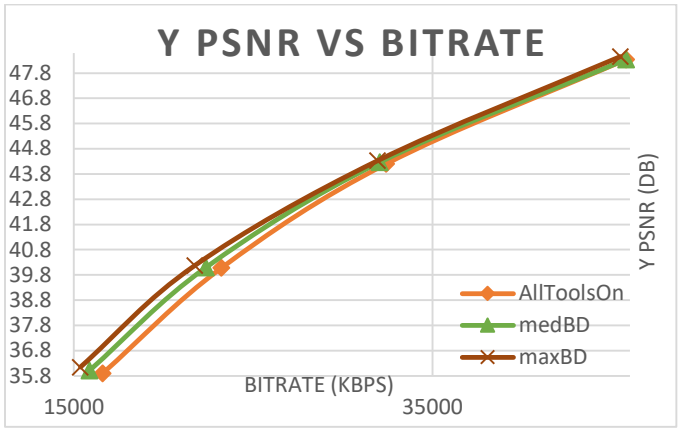

Figure 3. Average rate-distortion curve of all test sequences in comparison with the maximum BD setup (maxBD), medium BD setup (medBd) against HM-16.18+SCM-8.7.

Figure 3 shows an average RDO curve of all test sequences for the maximum and medium BD-rate setup, in comparison with a HEVC-SCC reference run (AllToolsOn). The PSNR/bitrate curves get closer together with higher PSNR/bitrate. Usually, the proposed method compresses well between QP27 and QP32. QP22 performs badly because the side information increases significantly but is limited by the already mentioned maximum sparsity setup, so that it is not often selected by the RDO process. Due to bitrate limitations, in QP37, usually only one sparse atom is optimal to represent the residual block and, in many cases, the RDO performance of IntDST/IntDCT is better. The major problem for QP22 and QP37 is that the sparse index is encoded in fixed length with the CABAC engine bypassed, due to its uniform distribution. In contrast, all side information of the IntDST/IntDCT process are CABAC encoded.

V. VISUALIZATION SPARSE CODED BLOCKS

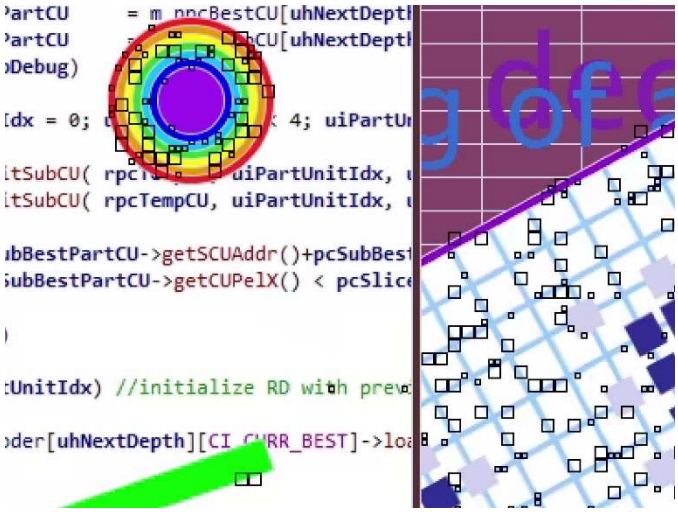

Figure 4 Coded blocks of proposed method for FlyingGraphics test sequence

The proposed method does not compress texts because this is usually done by the HEVC-SCC tools IBC and Palette Mode. Additionally, the maximum overall sparse limitation is set to only 4, which prevents encoding high frequency blocks and especially high frequency text blocks. This can be seen in Figure 4, where all text blocks are encoded with the standard HEVC-SCC tools and in Figure 5, where the Chinese characters at the bottom left are not encoded by the proposed method. Interestingly, in Figure 4 the proposed method is used for simple planar blocks and for blocks with straightforward lines which can also be seen in Figure 5.

Figure 5 Coded blocks of proposed method for ChinaSpeed test sequence.

Figure 5 represents a more general observation which shows, as mentioned before, blocks with planar characteristics and some straightforward lines where the direction of the lines does not matter.

VI. CONCLUSION AND FUTURE WORK

In this work, we successfully adapted an unusual sparse coding scheme to HEVC-SCC which targets similar and repeated low to mid frequency/energy intra predicted residuals with a low sparsity setup. Experimental results show an improved BD-rate savings of up to 4.6% for a maximum BD-rate savings configuration, but it comes with a ninefold execution time increase. A medium BD rate savings configuration where the proposed method is not used for QP22, and for block sizes 4x4 and 32x32, resulted in 2.5% BD-rate savings and an execution time increase by 103%, and a low BD-rate savings configuration for the 8x8 block size only without QP22, reaches 0.7% BD-rate savings and a 24% increase in execution time. Overall, the increased execution time can be limited because the current implementation is a C++ translation of a generic OMP MATLAB algorithm with no optimizations at all, but sparse coding is computationally demanding so there will always be an increase in execution time. The reason why the proposed method works successfully is that the standard HEVC-SCC prediction process is not optimal enough to include tiny differences that sum up as prediction errors and increases the residual data which needs to be CABAC encoded. The approach to split up the test sequences in categories and train dedicated dictionaries is successful and without it the BD-rate savings decrease by half. However, it also shows that the current category selection is not perfect because the test sequences FlyingGraphics, Console, SlideShow and Robot are outliers, and their BD-rate savings are around 0. The approach also suggests that no single dictionary alone can handle all the different applications, block sizes, and QP settings. In future research, we will further improve the proposed scheme with an online learning K-SVD which should adapt better to various content and applications.

REFERENCES

[1] Jani Lainema, Frank Bossen, Woo-Jin Han, Junghye Min, and Kemal Ugur, "Intra Coding of the HEVC Standard," IEEE Transactions on Circuits and Systems for Video Technology, Volume 22, Number 12, December 2012.

[2] HEVC Reference Software HM-16.18+SCM-8.7 (2018 January) [Online]. Available: https://hevc.hhi.fraunhofer.de/svn/svn_HEVCSoftware/tags/HM-16.18+SCM-8.7

[3] Jizheng Xu, Rajan Joshi, and Robert A. Cohen, "Overview of the Emerging HEVC Screen Content Coding Extension", IEEE Transactions on Circuits and Systems for Video Technology, Volume 26, Issue 1, Jan. 2016.

[4] M. Aharon, M. Elad and A. M. Bruckstein, "K-SVD: An Algorithm for Designing Overcomplete Dictionaries for Sparse Representation," IEEE Transactions on Signal Processing, Volume 54, Issue 11, pp. 4311–4322, November 2006.

[5] Y. Pati, R. Rezaiifar, and P. S. Krishnaprasad, "Orthogonal Matching Pursuit: Recursive Function Approximation with Applications to Wavelet Decomposition," in Proceedings of 27th Asilomar Conference on Signals, Systems and Computers, pp. 40–44 vol.1, November 1993.

[6] Rui Song, Cuiling Lan, Houqiang Li, Jizheng Xu and Feng Wu, "OMP-based transform for inter coding in HEVC," 2016 IEEE International Symposium on Circuits and Systems (ISCAS), pp 798-801, August 2016.

[7] Je-Won Kang, Moncef Gabbouj and C.-C. Jay Kuo, "Sparse/DCT (S/DCT) Two-Layered Representation of Prediction Residuals for Video Coding", IEEE Transactions on Image Processing, Volume 22 Issue 7, pp.2711–2722, April 2013

[8] Je-Won Kang and C.-C. Jay Kuo, "Efficient Dictionary Based Video Coding with Reduced Side Information," 2011 IEEE International Symposium of Circuits and Systems (ISCAS), July 2011.

[9] Madhusudan Kalluri, Minqiang Jiang, Nam Ling, Jianhua Zheng and Philipp Zhang, "Adaptive RD Optimal Sparse Coding with Quantization for Image Compression," IEEE Transactions on Multimedia, Volume 21, Issue 1, pp 39-50, June 2019.

[10] L. Limin and T. Alexis, "Rate distortion optimized quantization in the JM reference software." JVT-AA027, 2008.

[11] Haoping Yu, "Common test conditions for screen content coding", JCTVC-Z1015, 2017.

[12] Joel Sole, Rajan Joshi, Nguyen Nguyen, Tianying Ji, Marta Karczewicz, Gordon Clare, Félix Henry and Alberto Duenas, "Transform Coefficient Coding in HEVC," IEEE Transactions on Circuits and Systems for Video Technology, Volume 22, Number 12, December 2012.

[13] Karsten Suehring, "JVET common test conditions and software reference configurations", JVET-B1010, 2016.