Sparse Coding-based Intra Prediction in VVC

Jens Schneider, Dominik Mehlem, Maria Meyer, and Christian Rohlfing
Institut für Nachrichtentechnik, RWTH Aachen University, Germany
schneider@ient.rwth-aachen.de

***Abstract*—Intra prediction is crucial to video coding as it is the only option for prediction when motion compensation either fails or no reference frames are available. Hence, current state-of-the-art video coding standards utilize numerous intra prediction modes in order to predict different angled structures as well as smooth areas. This contribution introduces an additional mode for intra prediction which is based on the concepts of Dictionary Learning (DL), Sparse Coding (SC) [1] and adjusted Anchored Neighborhood Regression (ANR) [2] to be able to adapt to more arbitrary structures. The general idea is built on trained dictionaries, which sparsely represent the reference area of a block to be predicted. Alongside learning the dictionaries, linear projection matrices, projecting the reference areas to the corresponding blocks, are trained with ANR. For the actual intra prediction step, each given reference area is then projected onto the to-be-predicted block by multiple linear projections, which are blended according to the sparse codes representing the reference area.**

Experimentally, offering the proposed mode to the state-of-the-art video coding standard Versatile Video Coding (VVC) outperforms the traditional VVC modes: In particular, $-0.26\,\%$ BD-rate gains in comparison to the VVC reference software VTM-9.3 and a usage percentage of $12.83\,\%$ can be achieved on average for the All Intra (AI) coding configuration. Furthermore, a peak coding gain of $-0.6\,\%$ and a usage percentage of $26.66\,\%$ is observed for the same setup.

***Index Terms*—video compression, intra prediction, dictionary learning, versatile video coding**

I. Introduction

Modern video coding standards such as High Efficiency Video Coding (HEVC) [3] or VVC [4] utilize intra prediction in order to remove intra frame redundancies and improve the coding efficiency. Typically, the prediction signal is generated by extrapolation of the samples from a reference area, which is sourced from already coded regions of the picture. For the purpose of adaptability, usually multiple intra prediction modes are defined leading to smooth prediction signals in case of the DC and planar mode and a variety of angular prediction signals otherwise. Since the coding efficiency strongly depends on the number of modes defined for a codec, this number has increased over the last generations of video coding standards from 9 (Advanced Video Coding (AVC) [5]) to 67 (VVC). Furthermore, in VVC the Matrix-based Intra Prediction (MIP) [6] mode is defined, which allows for a linear prediction of the current block by multiplying the vector of reference samples by an applicable matrix. For complexity reasons this approach still relies on the principles of linear prediction and does not introduce any nonlinearities. An important aspect of MIP is that the prediction matrices are trained on natural image data introducing a data driven model to the intra prediction process in VVC. Moreover, nonlinear models such as Convolutional Neural Networks (CNNs) have been subject to recent research activities with respect to intra prediction. Typically, these approaches utilize an extended set of reference samples for the generation of a prediction signal, as visualized in Fig. 1. A network architecture based on fully connected layers was shown to outperform the intra prediction of HEVC [7]. Further, convolutional layers and recurrent architectures were introduced to neural network-based intra prediction [8] [9], and an approach based on convolutional layers and cross-component prediction showed promising results compared to HEVC [10]. During the development of MIP, also more complex network structures were considered [11] but eventually disregarded for VVC.

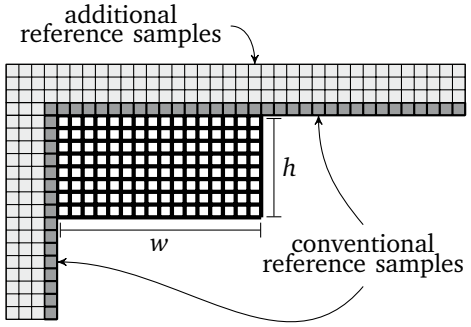

Fig. 1. Reference area and prediction block in the context of intra prediction. The dark gray area marks the conventional reference area and the light gray area an extended reference area by 4 rows or columns respectively.

An approach to intra prediction based on DL / SC in combination with linear prediction constitutes the main contribution of this paper. The rest of the paper is organized as follows: Sec. II describes the foundations of DL and SC in the context of image processing, followed by the explanation of its application to intra prediction in Sec. III. In Sec. IV, the simulation setup is outlined and results are presented. Finally, Sec. V concludes the paper with a summary and an outlook.

II. Dictionary Learning and Sparse Coding

A general assumption for the application of DL and SC to image processing tasks is that a vectorized natural image

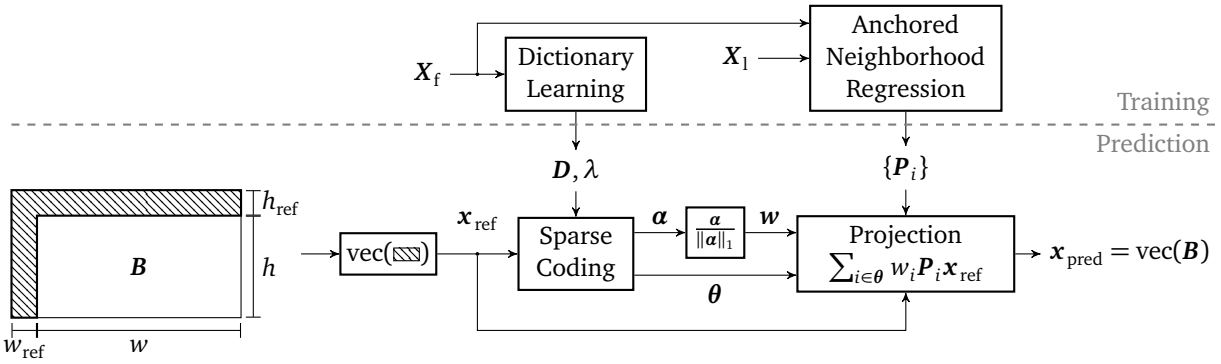

Fig. 2. Overview of the SCIP scheme. The upper part shows the training process in a high level overview. In the lower part the prediction process is depicted. The vectorized reference area is multiplied by several projection matrices, which are selected by a sparse coding approach.

patch $\boldsymbol{x} \in \mathbb{R}^{s_p^2}$ can be represented sparsely in a set of K exemplary elements $\boldsymbol{d}_i \in \mathbb{R}^{s_p^2}$, with $1 < i \leq K$, up to an error $\boldsymbol{\varepsilon}$ [1]. These patches typically possess a square shape of size $s_p \times s_p$. The exemplary elements $\boldsymbol{d}_i$ are also referred to as dictionary atoms in the context of DL, and the dictionary is defined as the concatenation of atoms such that $\boldsymbol{D} = [\boldsymbol{d}_1, \boldsymbol{d}_2, \ldots, \boldsymbol{d}_K]$. Hence, the representation reads

$$\boldsymbol{x} = \boldsymbol{D}\boldsymbol{\alpha} + \boldsymbol{\varepsilon} \tag{1}$$

with the sparse coefficient vector $\boldsymbol{\alpha} \in \mathbb{R}^K$. In the case of a known dictionary, a sparse representation can be found by e.g. solving the regularized minimization problem

$$\boldsymbol{\alpha} \leftarrow \arg\min_{\boldsymbol{\alpha}} \frac{1}{2}\|\boldsymbol{x} - \boldsymbol{D}\boldsymbol{\alpha}\|_2^2 + \lambda\|\boldsymbol{\alpha}\|_1. \tag{2}$$

Obviously, the solution is a tradeoff between the accuracy of the solution in an ℓ_2-norm error and its sparsity controlled by the Lagrangian multiplier λ. The process described in (2) is called Sparse Coding and its solution can be found by e.g. the LARS algorithm [1]. Typically, the patches are centered, i.e. their average intensity is subtracted and normalized to unit ℓ_2-norm, before processing them with SC-based methods.

Generally, the dictionary used for SC can either be hand-crafted or trained from natural image data. A prominent example for a hand-crafted dictionary are the basis functions of the Discrete Cosine Transform (DCT). However, the DCT dictionary does not utilize sparse characteristics of natural image patches. A dictionary can be trained tied to the sparsity of the representation by solving

$$\boldsymbol{D} \leftarrow \arg\min_{\boldsymbol{D}} \sum_{j=1}^{N} \frac{1}{2}\|\boldsymbol{x}_j - \boldsymbol{D}\boldsymbol{\alpha}_j\|_2^2 + \lambda\|\boldsymbol{\alpha}_j\|_1, \tag{3}$$

where N is the number of training patches. The dictionary learning can be approached by e.g. the Online Dictionary Learning (ODL) algorithm, which alternates the optimization with respect to $\boldsymbol{D}$ and $\boldsymbol{\alpha}$ [12].

III. Sparse Coding-based intra prediction

A. Overview

The idea behind the proposed Sparse Coding-based Intra Prediction (SCIP) relies on the assumption that an extended reference area of the current block $\boldsymbol{B} \in \mathbb{R}^{h \times w}$ can be represented sparsely in a trained dictionary and that an applicable prediction signal can be generated by multiple linear projections of the reference area to the current block. This could be understood as an extended version of MIP, because a prediction signal is calculated by a matrix multiplication as in MIP. Additionally, multiple of those prediction signals are blended in order to obtain the final prediction signal. Fig. 2 depicts the overall prediction scheme in the lower part. Note that the training process is not visualized in the block diagram but detailed in Sec. III-B. The extended reference area (▨) is defined by $h_{ref} = 4$ rows to the top and $w_{ref} = 4$ columns to the left of the current block and its vectorized form $\boldsymbol{x}_{ref} \in \mathbb{R}^m$ with $m = w_{ref}(h + h_{ref}) + wh_{ref}$ entries is the input to the process. The reference area is encoded sparsely in a trained dictionary $\boldsymbol{D}$ with a further constraint on the coefficient vector $\boldsymbol{\alpha}$ forcing all entries in $\boldsymbol{\alpha}$ to either positive values or zero. Thus, the optimization problem reads

$$\begin{aligned} \boldsymbol{\alpha} \leftarrow \arg\min_{\boldsymbol{\alpha}} \frac{1}{2}\|\boldsymbol{D}\boldsymbol{\alpha} - \boldsymbol{x}_{ref}\|_2^2 + \lambda\|\boldsymbol{\alpha}\|_1 \\ \text{s.t. } \alpha_i \geq 0 \forall\, i \in \boldsymbol{\theta} \end{aligned} \tag{4}$$

with $\boldsymbol{\theta}$ being the active set of $\boldsymbol{\alpha}$, i.e. the set of indices referencing to nonzero values in $\boldsymbol{\alpha}$. The sparse coding penalty was fixed to $\lambda = 0.1$. For illustration purposes, the centering and normalization operation are not shown in the block diagram, but are performed as preprocessing operations of the SC block. Further, from the sparse codes $\boldsymbol{\alpha}$, a weight vector is calculated such that $\boldsymbol{w} = \frac{\boldsymbol{\alpha}}{\|\boldsymbol{\alpha}\|_1}$ with $\sum_{i\in\boldsymbol{\theta}} w_i = 1$. Finally, the prediction signal is obtained by the weighted average of multiple linear predictions

$$\boldsymbol{x}_{pred} = \mathrm{vec}(\boldsymbol{B}) = \sum_{i\in\boldsymbol{\theta}} w_i \boldsymbol{P}_i \boldsymbol{x}_{ref}, \tag{5}$$

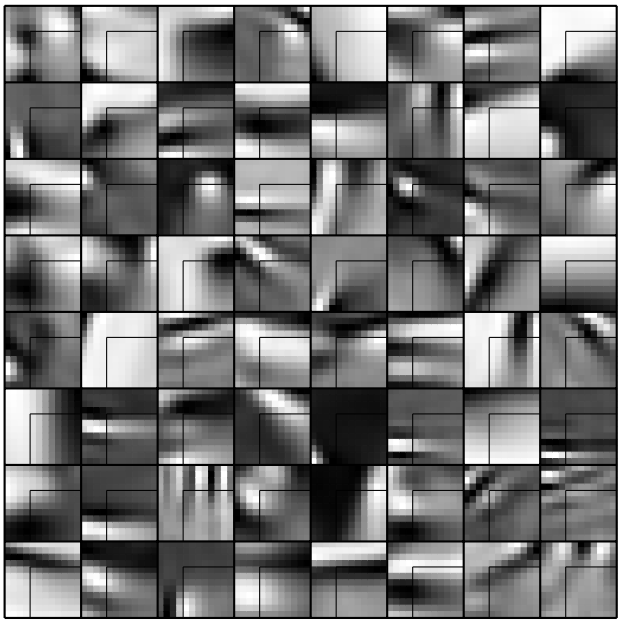

Fig. 3. An example dictionary with 64 L-shaped atoms and corresponding prediction signals for 8×8 blocks. The reference area covers $w_{\text{ref}} = h_{\text{ref}} = 4$ reference lines and columns respectively. The sparse coding penalty was chosen to $\lambda = 0.1$ for the dictionary learning task and for the optimization of the projections $\lambda_+ \approx 0.14$ was chosen. The neighborhood size was set to $N_{\text{n}} = 2048$. For visualization purposes all signals are scaled to the full dynamic range.

with the pretrained set of matrices $\{\boldsymbol{P}_i\}_{1<i\leq K}$ storing a projection matrix $\boldsymbol{P}_i \in \mathbb{R}^{wh\times m}$ for every dictionary atom of the corresponding dictionary (refer to Sec. III-B).

As the reference area varies in size with the size of the current block, different dictionaries need to be trained, and the dictionary corresponding to the current block size has to be selected for sparse coding. In total, 16 dictionaries were trained for the block sizes $s_{\text{b}} \in \{4\times4, 4\times8, \ldots, 4\times32, \ldots 32\times32\}$. For this reason, the reference samples to the bottom and to right of the block (comp. Fig. 1) are disregarded, as they are not always available and consequently the number of needed dictionaries would be even higher, if these samples would be taken into account. Fig. 3 shows an example dictionary with L-shaped atoms and their projections to the attached 8 × 8 block, i.e. $\boldsymbol{d}_{\text{pred},i} = \boldsymbol{P}_i \boldsymbol{d}_i$. It is observable that the dictionary atoms show edge-like structures, which are also continued in the prediction area. Furthermore, the prediction signals appear to be smoother in the bottom right corner of the block. This behavior is explainable, since calculating an accurate prediction becomes more difficult when the samples are farther away from the reference samples, and consequently a smoothed prediction signal is optimum in a least squares sense.

B. Dictionary and Projection Learning

This section details both, the training of the dictionary $\boldsymbol{D}$ with its parameters and the training of the projection matrices $\{\boldsymbol{P}_i\}$. Fig. 2 visualizes the training process as a high level block diagram in the upper part. The dictionaries for different block sizes are trained according to (3). The training set used for the entire training process was sourced from the Y-component of 4:2:0 sequences as proposed in [10]. All training data was extracted from the fifths frame of each sequence only, and scaled to the interval $[0, 1]$. Furthermore, the actual training data no longer consists of square image patches but of L-shaped patches matching the shape of the extended reference areas for intra prediction.

These L-shaped patches generate the training features and are vectorized and concatenated in order to form the data matrix $\boldsymbol{X}_{\text{f}} = \left[\boldsymbol{x}_{\text{f},1}, \boldsymbol{x}_{\text{f},2}, \ldots, \boldsymbol{x}_{\text{f},N}\right]$ with $\boldsymbol{x}_{\text{f},j} \in \mathbb{R}^m$. Only training features satisfying $0.003 < \text{var}(\boldsymbol{x}_{\text{f},j}) < 0.02$ were included in the training data, since data with too low or too high variance is assumed to belong to either very flat or textured areas, respectively. The corresponding training features were neglected since sparse image representations are known to focus rather on the structure in images than on flat regions or texture. Moreover, the training features are centered and normalized before the dictionary training as already explained above.

As for the training of the projection matrices $\{\boldsymbol{P}_i\}$ also the corresponding training labels $\boldsymbol{X}_{\text{l}} = \left[\boldsymbol{x}_{\text{l},1}, \boldsymbol{x}_{\text{l},2}, \ldots, \boldsymbol{x}_{\text{l},N}\right]$ are required, the labels $\boldsymbol{x}_{\text{f},j}$ consisting of the original blocks attached to the feature area are also gathered from the training set. The dictionary size was set to $K = 512$ and the sparse coding penalty in (3) was set to $\lambda = 0.1$, which is found to be reasonable for other SC-based applications [1].

The training process is aligned with the concept of adjusted ANR [2]. Conceptually, ANR is based on the assumption that a sparse encoding for the features can not only be found in the dictionary, but also in a local neighborhood of a certain dictionary atom. Typically, the applicable neighborhood is obtained by gathering those feature vectors from the training set which are most similar to the current atom. Consequently, the training set is searched for the N_{n} nearest neighbors of every dictionary atom $\boldsymbol{d}_i$. These nearest neighbors form the data matrix of the local neighborhood $\boldsymbol{N}_{\text{f},i} \in \mathbb{R}^{m\times N_{\text{n}}}$. The distance metric for the nearest neighbor search was chosen to be the Euclidean distance and the size of the local neighborhood was selected as $N_{\text{n}} = 2048$, which was found to be reasonable for other ANR-based applications [2], [13]. Mathematically, the assumption of an existing encoding of a feature vector $\boldsymbol{x}_{\text{f},j}$ in the neighborhood $\boldsymbol{N}_{\text{f},i}$ is given by

$$\boldsymbol{x}_{\text{f},j} = \boldsymbol{N}_{\text{f},i}\,\boldsymbol{\beta}_j + \boldsymbol{\varepsilon}_{\text{f},j}, \tag{6}$$

where $\boldsymbol{\varepsilon}_{\text{f}}$ models the approximation error. Conceptually, this approximation error is strongly dependent on the similarity of the current feature $\boldsymbol{x}_{\text{f},j}$ to the neighborhood $\boldsymbol{N}_{\text{f},i}$. Moreover, it is assumed that the labels $\boldsymbol{x}_{\text{l}}$ can be represented in a corresponding neighborhood by the same coefficient vector $\boldsymbol{\beta}_j$ such that

$$\boldsymbol{x}_{\text{l},j} = \boldsymbol{N}_{\text{l},i}\,\boldsymbol{\beta}_j + \boldsymbol{\varepsilon}_{\text{l},j}. \tag{7}$$

In particular, this means that once the coefficient vector $\boldsymbol{\beta}_j$ is calculated from a feature $\boldsymbol{x}_{\text{f},j}$ and a neighborhood $\boldsymbol{N}_{\text{f},i}$, an

approximation of the corresponding label $\boldsymbol{x}_{\mathrm{l},j}$ can be found. Obviously, calculating the coefficients in $\boldsymbol{\beta}_j$ during testing is not practical, as the neighborhoods would need to be stored alongside with the dictionary in that case. Moreover, this would require to store the neighborhoods, i.e. parts of the training data, and utilize it for inference, which is generally not desirable in machine learning-based applications. By concept of the adjusted ANR, the coefficients are calculated during training by ridge regression [14] as

$$\boldsymbol{\beta}_j \leftarrow \arg\min_{\boldsymbol{\beta}_j} \|\boldsymbol{N}_{\mathrm{f},i}\,\boldsymbol{\beta}_j - \boldsymbol{x}_{\mathrm{f},i}\|_2^2 + \lambda_+ \|\boldsymbol{\beta}_j\|_2^2 \tag{8}$$

$$\boldsymbol{\beta}_j = \left(\boldsymbol{N}_{\mathrm{f},i}^\top \boldsymbol{N}_{\mathrm{f},i} + \lambda_+ \mathbf{I}\right)^{-1} \boldsymbol{N}_{\mathrm{f},i}^\top \boldsymbol{x}_{\mathrm{f},j}, \tag{9}$$

with the relaxation parameter λ_+. This parameter was also used to balance the two squared ℓ_2-norm terms in (8): Whilst the dimension of $\boldsymbol{\beta}_j$ is fixed to the neighborhood size N_n, the dimension of $\boldsymbol{x}_{\mathrm{f},i}$ varies with the block size. Therefore, λ_+ was chosen to be dependent on the block size as $\lambda_+ = 0.2\left(\frac{4(h+4)+4w}{48}\right)^2$ such that the balance between the squared ℓ_2-norms is approximately the same for all considered block sizes and $\lambda_+ = 0.2$ for 4×4 blocks. When $\boldsymbol{\beta}_j$ from (9) is inserted in (7), a label vector can be approximated by

$$\boldsymbol{x}_{\mathrm{l},j} \approx \underbrace{\boldsymbol{N}_{\mathrm{l},i}\left(\boldsymbol{N}_{\mathrm{f},i}^\top \boldsymbol{N}_{\mathrm{f},i} + \lambda_+ \mathbf{I}\right)^{-1} \boldsymbol{N}_{\mathrm{f},i}^\top}_{\boldsymbol{P}_i} \boldsymbol{x}_{\mathrm{f},j}, \tag{10}$$

which leads to the definition of the projection matrices

$$\boldsymbol{P}_i = \boldsymbol{N}_{\mathrm{l},i}\left(\boldsymbol{N}_{\mathrm{f},i}^\top \boldsymbol{N}_{\mathrm{f},i} + \lambda_+ \mathbf{I}\right)^{-1} \boldsymbol{N}_{\mathrm{f},i}^\top. \tag{11}$$

Note that data representing the local neighborhoods is not preprocessed. Therefore, the projection matrices define a direct relationship between the samples of the reference area and the prediction signal. Moreover, from (10) it can be observed that $\boldsymbol{\beta}_j$ has never to be calculated explicitly during inference, but the projection matrices can be stored alongside every dictionary atom and be used in the prediction process.

IV. Simulation Setup and Results

For evaluation purposes SCIP was integrated into the VVC reference software VTM-9.3 [15]. As SCIP was not assumed to outperform the available intra prediction modes for every block, an additional flag was signalled on a per-block basis indicating the usage of SCIP. This flag was coded into the bitstream right after the **pred_mode_flag**, and no further information such as the intra mode index is signalled, if SCIP is applied for a block. Moreover, one coding context was utilized in order to limit the overhead for the transmission of the flag in the case of local statistical dependencies of the flags value. Generally, SCIP was only applied to the Y-component of the video sequence, as the chroma channels were assumed to be efficiently predicted by the Cross Component Linear Model (CCLM) in VVC. Further, for implementation reasons, Intra Subpartitioning (ISP) was not allowed to be used in combination with SCIP. In summary, SCIP was implemented as a coding tool, which is fully RD-tested and compared against the available modes specified in VVC including MIP at the leafs of the coding tree.

TABLE I
BD rate gains and usage of SCIP anchored on VTM-9.3. All values are given in %, whereby the BD column indicates relative rate savings and the Usage column the percentage of pixels predicted by SCIP.

Class	Sequence	AI		RA	
		BD-Y	Usage	BD-Y	Usage
A1	Campfire	−0.21	11.0	−0.09	2.81
A1	FoodMarket4	−0.6	21.74	−0.16	0.92
A1	Tango2	−0.47	17.8	−0.18	1.38
A2	CatRobot1	−0.36	14.2	−0.05	0.62
A2	DaylightRoad2	−0.36	13.13	−0.08	0.73
A2	ParkRunning3	−0.34	20.99	−0.12	1.66
A	Average	−0.39	16.48	−0.11	1.35
B	BasketballDrive	−0.28	12.27	−0.13	1.31
B	BQTerrace	−0.18	8.77	−0.11	0.26
B	Cactus	−0.26	15.31	−0.13	1.04
B	MarketPlace	−0.41	26.66	−0.15	1.36
B	RitualDance	−0.51	20.23	−0.18	2.31
B	Average	−0.33	16.65	−0.14	1.26
C	BasketballDrill	−0.25	5.23	−0.1	0.9
C	BQMall	−0.24	12.59	−0.15	0.57
C	PartyScene	−0.1	11.18	−0.07	0.73
C	RaceHorsesL	−0.21	17.73	0.04	2.09
C	Average	−0.2	11.68	−0.07	1.07
D	BasketballPass	−0.28	15.73	−0.07	1.9
D	BlowingBubbles	−0.1	12.0	−0.1	0.55
D	BQSquare	0.03	5.65	0.04	0.1
D	RaceHorsesM	−0.19	18.65	0.02	1.87
D	Average	−0.14	13.0	−0.03	1.11
E	FourPeople	−0.38	12.99	−0.3	0.37
E	Johnny	−0.31	7.05	−0.35	0.13
E	KristenAndSara	−0.2	9.69	−0.14	0.3
E	Average	−0.30	9.91	−0.26	0.27
F	ArenaOfValor	−0.25	13.24	−0.09	0.53
F	BasketballDrillText	−0.18	4.92	−0.01	0.86
F	SlideEditing	0.04	2.28	−0.04	0.07
F	SlideShow	−0.11	2.48	−0.01	0.18
F	Average	−0.13	5.73	−0.04	0.41
Avg		−0.26	12.83	−0.1	0.98

For the assessment of the tool, simulations according to the Common Testing Conditions (CTC) for VVC [16] were performed for an AI and Random Access (RA) coding scenario and Bjøntegaard Delta (BD) rate savings [17] were measured with respect to the anchor of VTM-9.3. Further, the usage of SCIP was measured in terms of percentages of pixels predicted by SCIP. Tab. I shows the BD-rate savings and the usage percentage for the test sequences from the JVET testset. Clearly, SCIP can be assessed to outperform the intra prediction modes available in VVC even if the average relative rate savings are limited to −0.26 % for the AI coding scenario and −0.1 % in the RA case. The average usage percentage lies at 12.83 % and 0.98 %, respectively. The large difference in usage percentage between AI and RA

is explained by the fact that most blocks are predicted by motion compensated prediction in RA coding, which is not allowed in AI. The maximum rate savings are achieved for the FoodMarket4 sequence in the AI scenario with −0.6 % and the highest usage percentage was measured for the MarketPlace sequence with 26.66 % of the pixels predicted by SCIP. From these observations it can be stated that SCIP goes beyond the capabilities of the VVC toolset including MIP, since its usage percentage is remarkable. Generally, it is observed that on average the tool performs best on the class A and class B sequences, i.e. higher resolution content. Therefore, it can be concluded that SCIP particularly addresses intra coding of high resolution images, which might also be beneficial for still image coding, where resolution can be even higher than for the class A sequences.

V. Summary and Conclusion

A novel method for intra prediction, Sparse Coding-based Intra Prediction (SCIP), based on the concepts of dictionary learning, sparse coding and adjusted anchored neighborhood regression was presented and shown to improve the performance of VVC in this paper. Even though VVC introduces another trained approach to intra prediction which is MIP, this work indicates that there is still room for improvement from the signal modelling perspective. Especially, on higher resolution video sequences, SCIP leads to bit rate savings with respect to the VTM-9.3 reference software, and it is used for predicting approximately 16.5 % of the pixels for these sequences. For further research also the reference samples to the top right and bottom left of the current block could be introduced to the prediction scheme, which is likely to result in even higher coding gains, since more information is used for the calculation of the prediction signal. Further, the complexity required for the implementation of SCIP was not addressed in this contribution. In conclusion, it was shown that the presented approach yields overall rate savings while complexity reduction is an open topic for further research activities.

Acknowledgment

This work was funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) – under grant 407254021

References

[1] M. Elad, *Sparse and Redundant Representations: From Theory to Applications in Signal and Image Processing*, 1st ed. Springer Publishing Company, Incorporated, 2010.

[2] R. Timofte, V. De Smet, and L. Van Gool, "A+: Adjusted anchored neighborhood regression for fast super-resolution," in *Computer Vision – ACCV 2014*, D. Cremers, I. Reid, H. Saito, and M.-H. Yang, Eds. Cham: Springer International Publishing, 2015, pp. 111–126.

[3] *High efficiency video coding*, Recommendation ITU-T H.265, ITU-T Std. Recommendation ITU-T H.265.

[4] *Versatile video coding*, Recommendation ITU-T H.266, ITU-T Std. Recommendation ITU-T H.266.

[5] *Advanced video coding for generic audiovisual services*, Recommendation ITU-T H.264, ITU-T Std. Recommendation ITU-T H.264.

[6] M. Schäfer, B. Stallenberger, J. Pfaff, P. Helle, H. Schwarz, D. Marpe, and T. Wiegand, "An Affine-Linear Intra Prediction With Complexity Constraints," in *2019 IEEE International Conference on Image Processing (ICIP)*, Sep. 2019, pp. 1089–1093.

[7] J. Li, B. Li, J. Xu, R. Xiong, and W. Gao, "Fully connected network-based intra prediction for image coding," *IEEE Transactions on Image Processing*, vol. 27, no. 7, pp. 3236–3247, 2018.

[8] T. Dumas, A. Roumy, and C. Guillemot, "Context-adaptive neural network-based prediction for image compression," *IEEE Transactions on Image Processing*, vol. 29, pp. 679–693, 2020.

[9] Y. Hu, W. Yang, M. Li, and J. Liu, "Progressive spatial recurrent neural network for intra prediction," *IEEE Transactions on Multimedia*, vol. 21, no. 12, pp. 3024–3037, 2019.

[10] M. Meyer, J. Wiesner, J. Schneider, and C. Rohlfing, "Convolutional neural networks for video intra prediction using cross-component adaptation," in *Proc. of IEEE International Conference on Acoustics, Speech and Signal Processing ICASSP '19*. IEEE, Piscataway, May 2019, pp. 1607–1611.

[11] J. Pfaff, P. Helle, D. Maniry, S. Kaltenstadler, W. Samek, H. Schwarz, D. Marpe, and T. Wiegand, "Neural network based intra prediction for video coding," in *Applications of Digital Image Processing XLI*, A. G. Tescher, Ed., vol. 10752, International Society for Optics and Photonics. SPIE, 2018, pp. 359 – 365. [Online]. Available: https://doi.org/10.1117/12.2321273

[12] J. Mairal, F. Bach, J. Ponce, and G. Sapiro, "Online dictionary learning for sparse coding," in *Proceedings of the 26th annual international conference on machine learning*. ACM, 2009, pp. 689–696.

[13] R. Timofte, V. De Smet, and L. Van Gool, "Anchored neighborhood regression for fast example-based super-resolution," in *2013 IEEE International Conference on Computer Vision*, 2013, pp. 1920–1927.

[14] A. E. Hoerl and R. W. Kennard, "Ridge regression: Biased estimation for nonorthogonal problems," vol. 12, no. 1, pp. 55–67.

[15] J. Chen, Y. Ye, and S. H. Kim, "Algorithm description for versatile video coding and test model 9 (vtm 9)," JVET, Tech. Rep. JVET-R2002.

[16] F. Bossen, J. Boyce, K. Suehring, X. Li, and V. Seregin, "JVET common test conditions and software reference configurations for SDR video," JVET, Tech. Rep. JVET-N1010.

[17] G. Bjontegaard, "Calculation of average PSNR differences between RD-curves," ITU-T SG16/Q6 VCEG, Austin, USA, Tech. Rep. Doc. VCEG-M33, 2001.