2024 Asia Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC)

Forward Prediction-Guided Cross-Partition Targeted Pruning for VVenC

Jingyuan Tang* and Songlin Sun†
* Beijing University of Posts and Telecommunications, Beijing
E-mail: tangjingyuan2018@bupt.edu.cn
† Beijing University of Posts and Telecommunications, Beijing
E-mail: slsun@bupt.edu.cn

***Abstract*—With the rapid development of digital video technology, video coding plays a crucial role in multimedia communication, network streaming, and other fields. Efficient video encoding algorithms can significantly reduce data storage and transmission costs, and improve video quality. This article focuses on VVenC, an open-source encoding platform based on the latest video encoding standard H.266/VVC, and studies and optimizes its video encoding block partitioning strategy. In the encoding process, the division of encoding units and pattern prediction are interdependent, and the accuracy of division is transmitted to the accuracy of pattern prediction, and vice versa. In order to ensure encoding efficiency and reduce encoding time, we propose a pruning algorithm based forward prediction-guided and cross-partition (FPG-CTP). We perform targeted pruning on the horizontal and vertical block partition based on the prediction results of the forward prediction model. The experimental results show that the optimization strategy proposed in this paper significantly improves compression performance while keeping the encoding complexity basically unchanged. On various test sequences, compared to the original VVenC encoder, the average time savings reached 48.67%, and the average bit rate increased by 1.81%. In addition, the work of this article also provides reference for optimizing block partitioning in other video coding standards, and has a certain promoting effect on the further development of video coding technology.**

I. Introduction

The rapid development of the video industry has gradually made applications such as live streaming, video calls, panoramic videos, and short video socializing mainstream. These applications also drive the development of video coding with different levels of video quality requirements. The video encoding standard Versatile Video Coding (H.266/VVC) released by the Joint Video Exploration Team (JVET) in 2020 has improved encoding efficiency by 50% compared to the previous generation High Efficiency Video Coding (H.265/HEVC) [1]. The H.266/VVC testing model is mainly used as a technical testing platform for evaluating and validating proposals during the standardization process in the field of video coding. It is a universal reference implementation, but the official believes that it should not be used as an encoding and decoding platform in the industrial sector [2]. VVenC can provide a publicly available and efficient implementation of H.266/VVC encoder. VVenC has made further optimizations based on VTM, including assembly depth optimization, search algorithm improvement, and support for parallel encoding [3]. The VVenC encoder provides 5 preset modes: slower, slow, medium, fast, and faster, which represent different encoding speeds and compression quality offsets, respectively. The slowest preset has the slowest encoding speed but the best encoding quality, while the fastest preset has the fastest encoding speed but the worst encoding quality. In the fastest configuration, VVenC runs approximately 140 times faster than VTM, while still having a performance gain of about 10% compared to HM.

The VVenC encoder still adopts the mainstream hybrid encoding framework, which divides the input video frame into multiple encoding tree units (CTUs) using different partitioning modes to recursively divide into encoding units of different sizes (CU) for encoding. The method of block segmentation greatly determines the efficiency of subsequent tasks, and an appropriate partitioning method can quickly partition pixel groups with the same features into the same area. The latest generation of encoding H.266/VVC has made significant improvements in block partitioning compared to the previous generation, making partitioning more flexible. AVS3, independently developed in China, proposed a similar approach to extend Quad Tree (EQT) [4] partitioning. However, the increase in complexity of partitioning patterns means an increase in coding time, and achieving a balance between time and performance has always been a persistent issue in coding. In the H.266/VVC reference software VTM 3.0, in order to reduce the intra frame encoding complexity, some scholars have proposed a H.266/VVC encoding unit length pruning based on prospective prediction [5]. This method first determines the optimal intra prediction mode within a two partition region based on Satd's Mode Decision Making (SMD), and then calculates the rate distortion cost (RD Cost) [6] of the optimal prediction mode for each block to select a better partition direction.

Inspired by this, we propose a forward prediction-guided cross-partition targeted pruning algorithm. Our approach utilizes encoded features within the block for decision-making. Furthermore, in predicting partition directions, we incorporate the Hadamard transform coefficients of residual signals and block gradients, providing a more comprehensive and robust basis for partitioning decisions.

The remainder of this paper is organized as follows: Section 2 offers a thorough review of contemporary strategies for optimizing block partitioning algorithms, with a focus on the

H.266/VVC and VVenC encoding platforms. This section also examines related research efforts aimed at achieving a balance between block partitioning complexity and encoding efficiency. In Section 3, we provide a detailed exposition of the proposed partition pruning algorithm, highlighting the innovative use of forward and down-sampling predictive modes. Section 4 presents and analyzes the experimental results, demonstrating the efficacy and advantages of our approach through various performance metrics and comparative analyses. Finally, Section 5 concludes the paper with a summary of our findings, implications for future research, and potential applications of the proposed algorithm in practical encoding scenarios.

II. Related Work

From the development trend of intra frame coding, there are currently two mainstream approaches for optimizing intra frame video coding block partitioning: One is to consider the potential features of blocks that have an impact on block partitioning and subsequent prediction processes. Based on the probability distribution obtained from mathematical statistics, appropriate empirical values are found through experiments to terminate the partitioning in advance. Under certain conditions, the subsequent RDO process is truncated to achieve time savings.Texture information is an important consideration factor in reducing the complexity of block partitioning decisions[7][8][9]. Cui Jing et al. [10] used texture information from four directions to terminate early through numerical statistics and experimental thresholds. Zhang Qiuwen et al. [11] used Bayesian algorithm to calculate the rough mode decision evaluation cost of the current CU, the texture distribution position and complexity in the CU, and thus skipped bad candidate partitioning methods. Liu Shishi et al. proposed a scoring mechanism based on a bottom tracing split pattern to predict the likelihood of being selected as the optimal partition type for the time-consuming search process of CU partitioning.

The second approach is to use machine learning or neural networks for training. Based on the first approach, neural networks are introduced to better simulate the termination of partitions and seek termination algorithms that are closer to the final partition results. Quan He [12] and others used random forests to predict the partitioning of CU. Divide CU into three categories: simple, complex, and fuzzy. The first two categories of random forest classifiers directly predict the optimal partition mode; For fuzzy CU, train another random forest to predict whether the partitioning process will terminate. Li Tianyi et al. established a large-scale database and applied a multi-stage exit CNN (MSE-CNN) model with an early exit mechanism to determine CU partitioning. Wang Fengqin [13] et al. divided the CU partitioning process into two stages based on FSVM and dag-SVM, and symmetrically extracted some features of standard deviation, directional complexity, and content difference complexity of the same CU, and judged whether to terminate partitioning early based on these features. In addition, lightweight neural networks have also been applied by Maraoui et al. in fast partitioning decision-making [14].

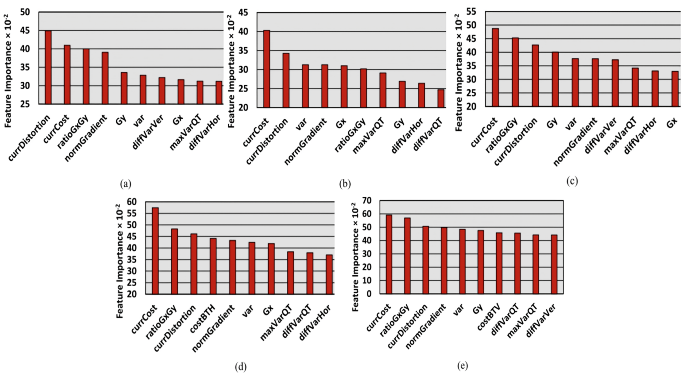


Fig. 1. Feature importance ranking of top ten features for (a) QT, (b) BTH, (c) BTV, (d) TTH, and (e) TTV classifiers [15].

The above two approaches are both aimed at the problem of increased encoding time caused by the complex RDO process that needs to be carried out in the video encoding process. The more accurate the simulation and early termination of the RDO process, the better the encoding performance can be provided. Overall, in the prediction process of CU partitioning, the information of CU itself and local regions play a crucial role, and gradients and textures play a decisive role in decision-making in most algorithms. In addition, there are also some local information that can potentially affect the partitioning method, including the current partitioning depth, current block QP, etc. Saldanha [15] and others conducted a large amount of statistical analysis on intra frame prediction and partitioning. By calculating the number of times different models use different features, they ranked the 10 features that are more important to the classifier, as shown in Fig.1.

Due to the high memory consumption in the calculation process of deep learning methods[16], in order to meet the common terminal encoding requirements, we optimize intra frame encoding based on traditional methods. At present, the optimization of encoders in VTM mainly focuses on the VTM platform, while there is less optimization for VVenC [17]. Although these solutions have excellent performance in H.266/VVC, they cannot be directly applied to VVenC. As a publicly available and efficient implementation of H.266/VVC encoder, further optimization of VVenC has practical significance in promoting the application of next-generation encoders.

III. Method

In most early termination algorithms for CU partitioning, the pattern prediction process and partitioning are often considered separately. However, in the actual encoding process, the division of encoding units and pattern prediction are interdependent, and the accuracy of division will be transmitted to the accuracy of pattern prediction. In order to ensure encoding efficiency and reduce encoding time, we propose a partition pruning algorithm based on forward prediction mode. By comparing the distortion values of horizontal and vertical

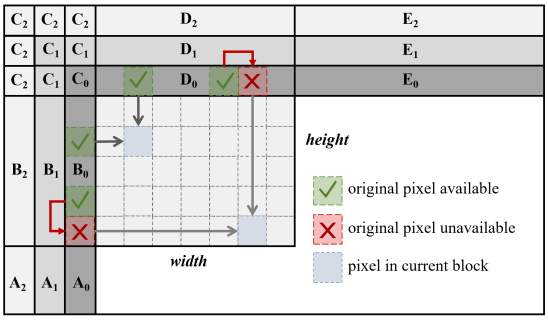


Fig. 2. Original pixel filling illustration.

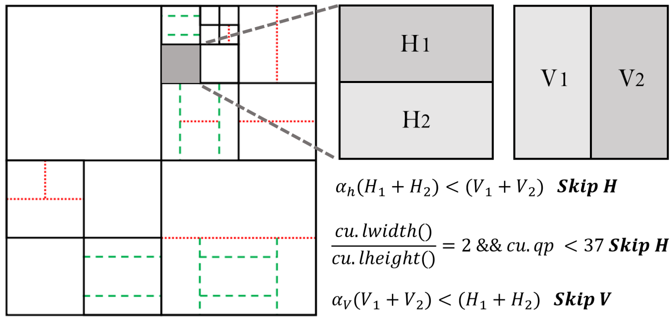


Fig. 3. Preliminary partitioning calculation illustration.

blocks, we determine whether certain prediction modes can be skipped.

The proposed methodology is encapsulated within a three-phase framework, encompassing forward prediction downsampling, original pixel filling, and preliminary partitioning computation. The following will provide a detailed introduction to the three steps.

- Forward Prediction Downsample
  Before performing intra frame mode prediction, downsample the predicted mode. Due to the expansion of angle prediction modes from 33 to 65 used in H.265/HEVC in the latest generation of video coding standards, complex mode selection is also increasing encoding time. We obtain the parent block prediction mode while retaining Planar, DC, and angle prediction modes 2, 4, 16, 32, 48, and 64. The above models have been retained from the earliest angle prediction models to the present day, which can represent as many directions as possible with the least number of angle models. The directions corresponding to these 6 angle modes include approximate directions of horizontal, vertical, diagonal, and transitional directions in the middle. Therefore, skipping the sub blocks in both horizontal and vertical directions in these modes is more representative. For the above prediction mode, its subsequent division is pruned through local features.
- Original Pixel Filling
  When the parent block operates in the specified mode, the reference pixels of the current prediction block are segmented and computed using either original pixel values or DC values to fill the reference pixels. If the reference pixel area is partially available, the algorithm first checks the availability of the reference pixel in the bottom-left corner. If available, all accessible reference pixels are filled with their original values, proceeding sequentially from bottom to top and left to right. Unavailable pixels are substituted with the nearest available pixel's original value. If the bottom-left reference pixel is unavailable, the algorithm traverses from bottom to top and left to right until the first available reference pixel is identified as shown in Fig.2. Since the reference pixels initially filled with reconstructed values are replaced with their original pixel values, streamlining the process of accessing reconstructed values.
- Preliminary Partitioning Calculation
  For each block that undergoes intra prediction, we calculate the SATD values for its four partitions. In video encoding, SATD (Sum of Absolute Transformed Differences) is a method of measuring the differences between two blocks. It is usually used in the block matching process to select the best matching block, thereby improving encoding efficiency. SATD calculates the difference between two blocks (usually the current block and the reference block). Firstly, the block difference is subjected to a transformation (such as Hadamard transformation), and then the absolute sum of the transformed coefficients is calculated. Assuming that $B1_{current}\ and B1_{ref}$ represent the pixel values of the current block and the reference block, respectively, and T represents the Hadamard transform, the calculation steps for SATD are as follows:

  $$SATD = \sum |T(B_{curr} - B_{ref})| \tag{1}$$

  SATD represents the sum of absolute values of transformed coefficients. During the partitioning process, the algorithm selects the more effective partitioning direction in both horizontal and vertical directions based on the predicted SATD value, and decides whether to skip, thereby effectively saving encoding time as shown in Fig.3. If the weighted distortion value in the vertical direction is less than the distortion value in the horizontal direction, horizontal segmentation is skipped, and vice versa. By comparing the distortion values in the horizontal and vertical directions, determine whether it is possible to skip the prediction in the corresponding direction.

In addition, we perform partition pruning based on forward prediction mode sampling based on the aspect ratio of important feature blocks and block level QP during the block partitioning process. We further prune small blocks with aspect ratio 2 and lower QP values. The specific algorithm steps are represented by pseudocode, as shown in algorithm 1.

This pseudocode demonstrates the main steps of intra prediction mode selection and segmentation pruning algorithms, and demonstrates how to optimize based on calculated SATD values and local features of blocks to save encoding time.

Algorithm 1 Intra Prediction Mode Selection and Partition Pruning Algorithm

Input: Current block B_{curr}, Reference block B_{ref}, Transform T, Weighting factors α_V, α_H
Output: Partition decision and SATD values
Initialization:
Define prediction modes: Planar, DC, Angular modes $\{2, 4, 16, 32, 48, 64\}$
SATD values: $minSadHad_{H1}$, $minSadHad_{H2}$, $minSadHad_{V1}$, $minSadHad_{V2}$
for each block B_{curr} **do**
  Downsample the prediction modes
  Use original pixel values instead of reconstructed values for reference pixels
  for each prediction mode in $\{Planar, DC, 2, 4, 16, 32, 48, 64\}$ **do**
    Calculate SATD for horizontal and vertical partitions:
    $SATD = \sum |T(B_{curr} - B_{ref})|$
    Compute SATD values: $minSATD_{H1}$, $minSATD_{H2}$, $minSATD_{V1}$, $minSadHad_{V2}$
  end for
  Evaluate partition efficiency:
  if $\alpha_V \times (minSATD_{V1} + minSATD_{V2}) < minSATD_{H1} + minSATD_{H2}$ **then**
    Skip horizontal partition
  else if $\alpha_H \times (minSATD_{H1} + minSATD_{H2}) < minSATD_{V1} + minSATD_{V2}$ **then**
    Skip vertical partition
  end if
  Prune partitions based on block characteristics:
  if block width/height ratio is 2 and QP is low **then**
    Skip horizontal partition
  end if
end for

IV. Experiments and Results

This experiment was conducted on a high-performance computer equipped with an Intel i9-4900K processor. The system is running a 64 bit Ubuntu 20.04 LTS operating system, equipped with 128GB memory and 1TB hard drive. All experiments were conducted on this unified platform to ensure consistency and comparability of results. We conducted experiments using the open-source video encoder VVenC (version 1.70). In order to study the performance of encoders under different configurations, we have made detailed settings and adjustments to the following aspects:

- Resolution
  We have chosen multiple different resolutions, including 720p, 1080p, and 4K, to evaluate the performance of the encoder under different video quality.
- Quantization parameter
  Four common quantization parameter, 22, 27, 32, and 37 were selected to test the performance of the encoder at different quantization levels.

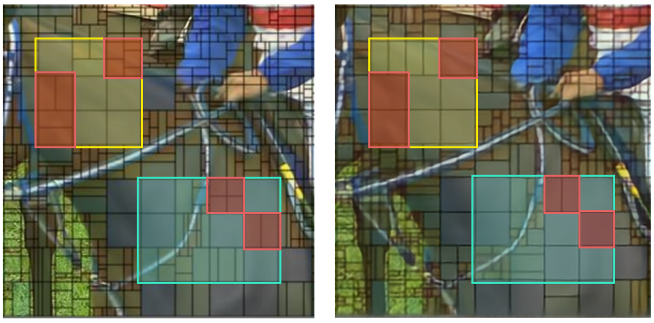
The test sequences are provided by JVET, divided into six categories: A (3840x2160), B (1920x1080), C (832x480), D (416x240), E (1280x720), and F. Class A is further divided into two subcategories: A1 and A2. These videos use our proposed algorithm embed in VVenC v1.7.0 into their full function encoder (vencffapp) and encode them using slow presets in their All Intra encoding mode. In addition, we use VTM 12.0 reference software for corresponding decoding to ensure that the algorithm corresponds on the encoding and decoding end. To test the results and performance of our solution, we used Bjontegaard Delta PSNR (BDPSNR) and time savings to measure the effectiveness of the algorithm. BDPSNR is used to measure the quality of encoding, while time savings are used to measure the speed of encoding. The calculation Equation for saving time ΔET measured by encoding time is:

$$\Delta ET = \frac{1}{4} \sum_{QP_i \in \{22,27,32,37\}} \frac{T_R(QP_i) - T_P(QP_i)}{T_R(QP_i)} \quad (2)$$

T_R and T_P respectively represent the encoding time spent by the VVenC reference encoder and the VVenC recommendation encoder. We tested each video sequence using full frame mode for 2 seconds to ensure sufficient test frame rates. The Table I shows the test results. Through the above experimental methods, we systematically evaluated the performance of the VVenC encoder. The experimental results demonstrate the

TABLE I
Algorithm Performance Test Results

Class/Sequences		Y(%)	U(%)	V(%)	TS
A1	Campfire	2.11	1.77	1.65	0.38
	FoodMarket4	2.55	1.94	1.67	0.48
	Tango2	1.96	1.53	1.48	0.41
A2	CatRobot	2.23	1.61	1.44	0.47
	DaylightRoad2	2.17	1.69	1.56	0.43
	ParkRunning3	2.46	1.87	1.83	0.52
B	BasketballDrive	2.34	1.37	1.36	0.42
	BQTerrace	2.48	1.83	1.62	0.47
	MarketPlace	1.12	0.97	0.94	0.36
	RitualDance	2.57	1.84	1.91	0.46
	Cactus	2.27	1.48	1.46	0.45
C	BasketballDrill	2.78	2.11	2.12	0.56
	BQMall	2.45	1.60	1.48	0.55
	PartyScene	2.89	1.80	1.80	0.62
	RaceHorsesC	2.22	1.34	1.29	0.57
D	BasketballPass	2.97	2.43	2.25	0.59
	BQSquare	2.46	1.99	1.96	0.57
	BlowingBubbles	1.96	1.23	0.98	0.55
	RaceHorses	1.88	1.21	1.03	0.49
E	FourPeople	1.85	1.29	1.20	0.44
	Johnny	1.92	1.36	1.27	0.39
	KristenAndSara	2.05	1.82	1.80	0.47
F	BasketballDrilltext	2.34	1.92	1.77	0.50
	SlideEditing	1.79	1.25	1.17	0.42
	SlideShow	2.13	1.76	1.72	0.53



Original *Using FPG-CTP Algorithm*

Fig. 4. Comparison of the results of original partitioning and block partitioning using algorithms.

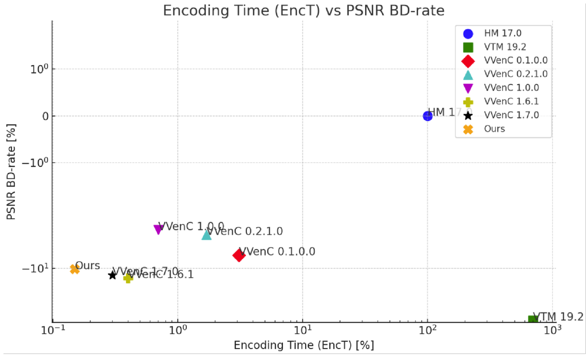


Fig. 5. Comparison of PSNR BD-rate performance corresponding to encoding time of different encoders.

encoding speed, compression efficiency, and video quality of the encoder at different QP value, sequence class, luminance and chromaticity components.

Table I presents a comprehensive comparison of the encoding performance between the algorithm proposed in this paper and the VVenC 1.7.0 anchor encoder, using the slow preset as a reference. The empirical evidence demonstrates that our solution markedly reduces the complexity associated with Coding Unit (CU) partitioning, with only a marginal loss in encoding efficiency. Specifically, the proposed method achieves an average time savings of 48.67%, while the Bjøntegaard Delta Bit Rate (BDBR) increases by a modest 1.81%.

The extent of complexity reduction varies across different video sequences. For instance, the complexity reduction for the Campfire video sequence is 38.12%, while for the PartyScene video sequence, it reaches a maximum of 62.03%. Correspondingly, the BDBR increases are 1.84% and 2.16%, respectively.

A. *Performance Evaluation*

Moreover, we have visually compared the partitioning results of the RaceHorseC sequence as shown in Fig.4. The left image represents the partitioning results of the original video encoding platform, while the right image shows the partitioning results after applying our algorithm.

The color blocks of identical colors in the Fig.4 illustrate the comparison of partitioning results within the same regions, exemplified by the yellow and green areas. The red area indicates the blocks that skip the binary tree partitioning process. From the comparative analysis, it is evident that our algorithm not only preserves a partitioning scheme similar to the original anchor framework but also minimizes the occurrence of multiple binary partitions. Consequently, this leads to a reduction in the overall complexity of video encoding.

B. *Discussion*

We conducted a comparative analysis of encoding time and Bjøntegaard Delta (BD) rate test results across multiple mainstream video encoding reference software. Using the reference software HM 17.0 of H.265/HEVC as the baseline, we evaluated the encoding performance of the reference software VTM 19.2 of H.266/VVC alongside various versions of VVenc, as illustrated in Fig.5. The scatter plot presented above delineates the relationship between Encoding Time (EncT) and Peak Signal-to-Noise Ratio (PSNR) BD-rate for the different reference software. The x-axis employs a logarithmic scale to more effectively represent disparities in encoding times. A shorter encoding time denotes higher encoding efficiency, while a higher BD-rate signifies superior encoding quality. Therefore, data points situated in the lower left corner of the plot indicate an optimal balance between encoding time and quality.

The baseline HM 17.0 is positioned at 100% encoding time (EncT) and 0% PSNR BD-rate, while VTM 19.2 shows a significantly higher EncT (700%) but offers the most substantial reduction in PSNR BD-rate (-41.50%), indicating a trade-off between encoding complexity and compression efficiency. Various versions of VVenC exhibit a range of trade-offs: VVenC 0.1.0.0 and VVenC 0.2.1.0 offer moderate encoding times (3.10% and 1.70%) with corresponding reductions in PSNR BD-rate (-7% and -4%), whereas later versions (VVenC 1.0.0, 1.6.1, and 1.7.0) further reduce encoding times but show varying impacts on PSNR BD-rate. Compare to HM 17.0, our proposed algorithm stands out with the lowest encoding time and a substantial reduction in PSNR BD-rate, highlighting its efficiency in achieving high compression performance with minimal encoding time.

These results underscore the efficiency of our algorithm in significantly decreasing the encoding time required for the CU partitioning process, thereby enhancing overall encoding efficiency. The slight trade-off in bit rate is justified by the substantial computational savings achieved, making our algorithm a valuable contribution to the field of video encoding.

V. Conclusion

This paper presents a novel partition pruning algorithm designed to enhance the efficiency of video encoding in the H.266/VVC standard, using the open-source VVenC platform. Our proposed algorithm, distinguished by its use of forward downsampling prediction mode to targeted pruning partition process. Additionally, we integrates the block inner parameters during the encoding process for decision-making. Moreover, Hadamard transform coefficients of residual signals and block gradients are employed for predicting partition directions, thereby offering a more robust basis for partitioning decisions.

The experimental results indicate that our algorithm significantly improves compression performance while maintaining encoding complexity. Specifically, our optimization strategy achieves an average time savings of 48.67% and a 1.81% increase in bit rate compared to the original VVenC encoder. In this study, our proposed algorithm still has the potential for further optimization, but it can provide a reference for optimizing block partitioning in other video coding standards and promoting the development of video coding technology.

Acknowledgment

This work is funded by the National Key Research and Development Program of China for the research on the comprehensive plan for the extensive audiovisual environment of "Future TV", with Grant Number 2023YFF0904601.

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, 2012. DOI: 10.1109/TCSVT.2012.2221191.

[2] A. Wieckowski, J. Brandenburg, T. Hinz, *et al.*, "Vvenc: An open and optimized vvc encoder implementation," in *2021 IEEE International Conference on Multimedia Expo Workshops (ICMEW)*, 2021, pp. 1–2. DOI: 10.1109/ICMEW53276.2021.9455944.

[3] V. George, J. Brandenburg, G. Hege, *et al.*, "Efficient multi-threading strategies in vvenc, an open and optimized vvc encoder implementation," in *2022 IEEE International Symposium on Multimedia (ISM)*, 2022, pp. 18–25. DOI: 10.1109/ISM55400.2022.00009.

[4] J. Zhang, C. Jia, M. Lei, S. Wang, S. Ma, and W. Gao, "Recent development of avs video coding standard: Avs3," in *2019 Picture Coding Symposium (PCS)*, 2019, pp. 1–5. DOI: 10.1109/PCS48520.2019.8954503.

[5] M. Lei, F. Luo, X. Zhang, S. Wang, and S. Ma, "Look-ahead prediction based coding unit size pruning for vvc intra coding," in *2019 IEEE International Conference on Image Processing (ICIP)*, 2019, pp. 4120–4124. DOI: 10.1109/ICIP.2019.8803421.

[6] G. Sullivan and T. Wiegand, "Rate-distortion optimization for video compression," *IEEE Signal Processing Magazine*, vol. 15, no. 6, pp. 74–90, 1998. DOI: 10.1109/79.733497.

[7] Z. Liu, M. Dong, X. H. Guan, M. Zhang, and R. Wang, "Fast isp coding mode optimization algorithm based on cu texture complexity for vvc," *EURASIP Journal on Image and Video Processing*, vol. 2021, pp. 1–14, 2021.

[8] Y. Li, G. Yang, Y. Song, H. Zhang, X. Ding, and D. Zhang, "Early intra cu size decision for versatile video coding based on a tunable decision model," *IEEE transactions on broadcasting*, vol. 67, no. 3, pp. 710–720, 2021.

[9] Q. Zhang, Y. Zhao, B. Jiang, L. Huang, and T. Wei, "Fast cu partition decision method based on texture characteristics for h. 266/vvc," *IEEE Access*, vol. 8, pp. 203 516–203 524, 2020.

[10] J. Cui, T. Zhang, C. Gu, X. Zhang, and S. Ma, "Gradient-based early termination of cu partition in vvc intra coding," in *2020 Data Compression Conference (DCC)*, 2020, pp. 103–112. DOI: 10.1109/DCC47342.2020.00018.

[11] Q. Zhang, Y. Zhao, B. Jiang, and Q. Wu, "Fast cu partition decision method based on bayes and improved de-blocking filter for h.266/vvc," *IEEE Access*, vol. 9, pp. 70 382–70 391, 2021. DOI: 10.1109/ACCESS.2021.3079350.

[12] Q. He, W. Wu, L. Luo, C. Zhu, and H. Guo, "Random forest based fast cu partition for vvc intra coding," in *2021 IEEE International Symposium on Broadband Multimedia Systems and Broadcasting (BMSB)*, 2021, pp. 1–4. DOI: 10.1109/BMSB53066.2021.9547117.

[13] F. Wang, Z. Wang, and Q. Zhang, "Fsvm- and dag-svm-based fast cu-partitioning algorithm for vvc intra-coding," *Symmetry*, vol. 15, no. 5, 2023, ISSN: 2073-8994. DOI: 10.3390/sym15051078. [Online]. Available: https://www.mdpi.com/2073-8994/15/5/1078.

[14] M. Amna, W. Imen, and S. F. Ezahra, "Fast multi-type tree partitioning for versatile video coding using machine learning," *Signal, Image and Video Processing*, vol. 17, pp. 67–74, 2022. [Online]. Available: https://api.semanticscholar.org/CorpusID:248350588.

[15] M. Saldanha, G. Sanchez, C. Marcon, and L. Agostini, *Versatile Video Coding (VVC): Machine Learning and Heuristics*. Jan. 2022, ISBN: 978-3-031-11639-1. DOI: 10.1007/978-3-031-11640-7.

[16] K. Yang, D. Liu, Z. Chen, F. Wu, and W. Li, "Spatiotemporal generative adversarial network-based dynamic texture synthesis for surveillance video coding," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 32, no. 1, pp. 359–373, 2021.

[17] I. Taabane, D. Menard, A. Mansouri, and A. Ahaitouf, "A fast cu partition algorithm for vvenc encoder in intra configuration," in *2022 IEEE Workshop on Signal Processing Systems (SiPS)*, 2022, pp. 1–6. DOI: 10.1109/SiPS55645.2022.9919252.