Partition Map-Based Fast Block Partitioning for VVC Inter Coding

Xinmin Feng, Zhuoyuan Li, *Graduate Student Member, IEEE*, Li Li, *Senior Member, IEEE*, Dong Liu, *Senior Member, IEEE*, and Feng Wu, *Fellow, IEEE*

***Abstract*—Among the new techniques of Versatile Video Coding (VVC), the quadtree with nested multi-type tree (MTT) block structure yields significant coding gains by providing more flexible block partitioning patterns. However, the recursive partition search in the VVC encoder increases the encoder complexity substantially. To address this issue, we propose a partition map-based algorithm to pursue fast block partitioning in inter coding. Based on our previous work on partition map-based methods for intra coding, we analyze the characteristics of VVC inter coding and improve the partition map by incorporating an MTT mask for early termination. Next, we develop a neural network that uses both spatial and temporal features to predict the partition map. It consists of several special designs, including stacked top-down and bottom-up processing, quantization parameter modulation layers, and partitioning-adaptive warping. Furthermore, we present a dual-threshold decision scheme to achieve a fine-grained trade-off between complexity reduction and rate-distortion performance loss. The experimental results demonstrate that the proposed method achieves an average 51.30% encoding time saving with a 2.12% Bjøntegaard-delta-bit-rate under the random access configuration.**

***Index Terms*—Block partitioning, convolutional neural network, inter coding, partition map, quadtree plus multi-type tree, versatile video coding.**

I. Introduction

With the development of display technology, the demand for compressing high-resolution videos has increased significantly. To address this requirement, the Joint Video Experts Team (JVET) developed Versatile Video Coding (VVC) [1]. Among all the various new technical aspects introduced by VVC, the quadtree plus multi-type tree (QT+MTT) block partitioning structure has been identified as one of the most significant changes compared with the High Efficiency Video Coding (HEVC) standard [2], [3]. Specifically, the JVET test software (JEM) adopted quadtree plus binary tree (QTBT) as the block partition structure to adapt to the characteristics of various texture patterns [4]. The VVC test model (VTM) further enhances the QTBT by introducing two ternary partition modes, enabling more flexible coding unit (CU) partition patterns. However, it significantly amplifies encoder complexity, as the optimal CU partition is determined through a brute-force rate-distortion optimization (RDO) search in the VTM. Specifically, VTM increases 617% encoding runtimes under RA condition compared to the HEVC model (HM) in default settings [5]. Therefore, it is necessary to accelerate the VVC encoder while preserving a desirable coding efficiency.

Considerable effort has been devoted to reduce the encoding complexity of VVC intra and inter coding, including handcrafted feature-based methods [6], [7], [8], [9], [10], [11], [12], [13], [14], [15], [16], [17] and neural network-based approaches [18], [19], [20], [21], [22], [23], [24], [25], [26], [27], [28]. By utilizing effective partition representations, neural network-based methods achieve higher acceleration ratios with lower compression loss in VVC intra coding compared to other approaches. However, in contrast to the progress in intra coding, the reduction of inter coding complexity still leaves room for improvement. Specifically, fast block partitioning algorithms for VVC inter coding can be enhanced at three aspects: (1) developing an effective representation of the QT+MTT partition structure that reflects the characteristics of inter coding, (2) designing a neural network capable of modeling the complex relationship between pixel-level features and block partitioning in inter coding, and (3) implementing a flexible post-processing algorithm to balance network predictions with the recursive partition search process in a fine-grained manner.

In this paper, we propose a partition map-based algorithm to pursue fast block partitioning in VVC inter coding. Building on our previous work in VVC intra coding [23], we extend this approach to VVC inter coding through three key aspects: representation, neural network architecture, and post-processing. For the **representation**, we analyze the characteristics of VVC inter coding and enhance the partition map by introducing an MTT mask to enable early termination of unnecessary MTT splits. Regarding the **neural network architecture**, we propose a coarse-to-fine prediction framework for predicting the improved partition map. To simulate the partition search process of inter coding, we design several novel modules, such as the slice quantization parameter (QP) modulation layer, which

Received 18 September 2024; revised 2 February 2025 and 25 April 2025; accepted 6 May 2025. Date of publication 14 November 2025; date of current version 8 January 2026. This work was supported in part by the Natural Science Foundation of China under Grant 62021001 and in part by the Graphical Processing Unit (GPU) cluster built by the Multimedia Computing and Communications (MCC) Lab of Information Science and Technology Institution. The associate editor coordinating the review of this article and approving it for publication was Dr. Nuno M.M. Rodrigues. *(Corresponding author: Dong Liu.)*

The authors are with the MOE Key Laboratory of Brain-Inspired Intelligent Perception and Cognition, University of Science and Technology of China, Hefei 230093, China (e-mail: xmfeng2000@mail.ustc.edu.cn; zhuoyuanli@mail.ustc.edu.cn; lil1@ustc.edu.cn; dongeliu@ustc.edu.cn; fengwu@ustc.edu.cn).

The source code is publicly available at https://github.com/ustcivclab/IPM.

This article has supplementary downloadable material available at https://doi.org/10.1109/TMM.2025.3632639, provided by the authors.

Digital Object Identifier 10.1109/TMM.2025.3632639

adapts to varied compression qualities across frames, and the partitioning-adaptive warping to simulate motion estimation. Unlike previous designs, we elevate the QT depth map prediction from the CTU level to the frame level, thereby leveraging global motion cues. However, ensuring the compliance of the prediction results with the partitioning rules remains an unresolved challenge, particularly for frame-level predictions. To address this, we employ a repeated bottom-up, top-down inference process with stacked hourglass blocks as the backbone. This approach enhances prediction accuracy and ensures that the predictions adhere to the partitioning rules, referred to as *local consistency*, through iterative refinement of intermediate predictions. For the **post-processing**, we introduce a dual-threshold decision scheme that balances complexity reduction with RD performance loss. Experimental results show that the proposed method reduces VVC inter coding time by 51.30%, with only a 2.12% increase in Bjøntegaard-delta-bit-rate (BDBR) based on JVET common test sequences [29], outperforming existing approaches. In summary, the main contributions of this paper are as follows:

- We enhance the partition map to efficiently represent the QT+MTT partition structure in VVC inter coding by introducing the MTT mask.
- Inspired by the partition search process in inter coding, we design a novel neural network that predicts the partition map in a coarse-to-fine manner. The network incorporates several new modules, including repeated top-down and bottom-up processing, a partitioning-adaptive warping module, and a QP modulation layer, among others.
- We present a dual-threshold decision scheme to achieve a fine-grained trade-off between complexity reduction and RD performance degradation, facilitated by the introduction of the MTT mask.

The rest of the paper is organized as follows: Section II briefly reviews fast block partitioning algorithms of the VVC and previous standards. Section III presents the improved partition map based on the statistics of block partitioning results for VVC inter coding. Section IV introduces the proposed network architecture. Section VI provides detailed experimental results, including ablation studies on key techniques. Lastly, Section VII concludes the paper and discusses limitations.

II. Related Work

In this section, we briefly review the previous fast block partitioning algorithms used in HEVC and VVC standards.

A. Fast Block Partitioning Algorithms for HEVC Standard

Before VVC, HEVC has been widely adopted in practical scenarios. Thus, numerous techniques are employed to reduce the complexity of the QT partition search of HEVC in both intra coding and inter coding. Regarding intra coding of HEVC, the relevant methods can be classified into two categories: heuristic methods [30], [31], [32], [33], [34] and neural network-based methods [35], [36], [37]. For heuristic methods, researchers [30], [31], [32], [33], [34] utilized the intermediate characteristics of the current CU and the spatial correlations with neighboring CUs to early terminate the unnecessary partition search. After this, researchers focused on the effective representation of partition search and predicted it with deep neural networks. For instance, Liu et al. [35] proposed a VLSI-friendly fast algorithm using convolutional neural networks (CNN). Xu et al. [37] proposed an early-terminated hierarchical CNN for learning to predict the hierarchical CU partition map. Tissier et al. [36] proposed a probability vector for each 64 $\times$ 64 CU to speed up block partitioning. Feng et al. [38] used the depth map to represent the block partition of a CTU and designed a CNN to predict the depth map.

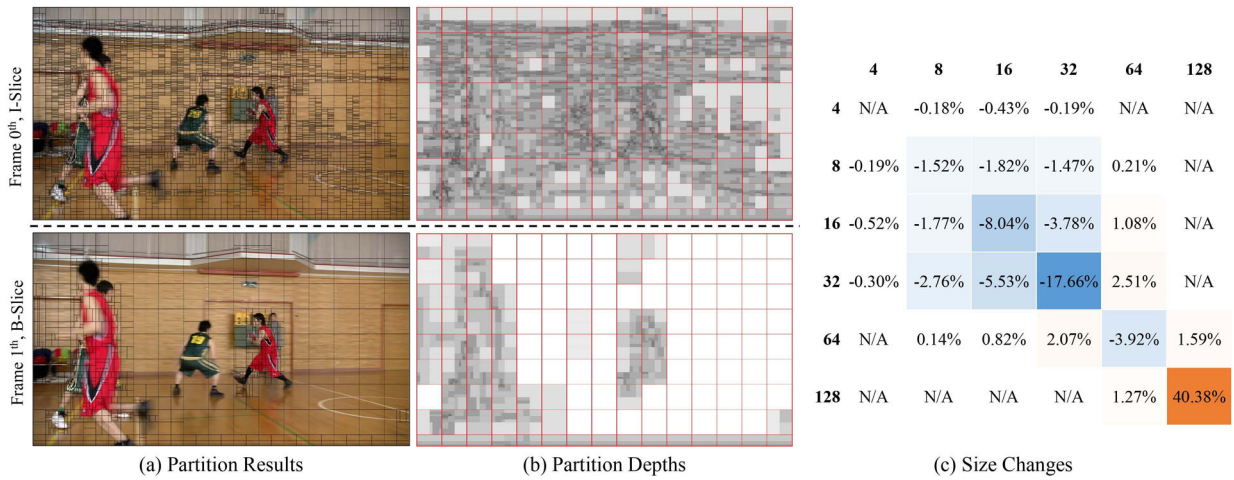
In HEVC inter coding, researchers explored temporal features to early-terminate the redundant partition searches. Correa et al. [39] extracted intermediate data and built three sets of decision trees. Zhang et al. [40] designed a three-output joint classifier considering nine features relevant to CU depth decision. Zhu et al. [41] proposed a binary and multi-class SVM algorithm to predict both the CU partition and PU mode with an offline-and-online machine learning mechanism. Xu et al. [37] proposed an early-terminated hierarchical long- and short-term memory network to learn the temporal correlation of the CU partition.

B. Fast Block Partitioning Algorithms for VVC Standard

With the introduction of the QT+MTT structure, VVC provides more flexible partitioning patterns compared to HEVC, but it also significantly increases encoding time. To address this, various approaches have been developed to speed up the VVC encoding process for both intra and inter coding.

For intra coding of VVC, earlier work focused on handcrafted feature engineering for fast block partitioning [6], [7], [8], [9], [10], [11], [12], [13], [8]. Notably, Saldanha et al. [13] proposed a configurable light gradient boosting machine that uses effective texture, coding, and context features to predict the best partition type. After this, researchers attempted to model the QT+MTT structure in effective representations and predict them using powerful neural networks. For instance, Galpin et al. [19] modeled the CU boundaries as a single vector, and designed a convolutional neural network to predict the vector in a bottom-up manner. Li et al. [20] categorized all possible CU sizes into six stages and designed an MSE-CNN to determine the CU partition. Wu et al. [21] devised a hierarchy grid map to represent the QT+MTT structure and proposed HG-FCN to predict it in a stage-wise top-down manner. Park et al. [28] proposed a lightweight neural network that decides whether to terminate the nested TT block structures following a quadtree, based on both explicit and derived VVC features. Tissier et al. [22] proposed a decision tree model to predict the probabilities at each block of the entire CTU. Feng et al. [23] formulated the QT+MTT structure as a partition map and designed a Down-Up-CNN to predict it.

Regarding inter coding of VVC, fast block partitioning algorithms can also be categorized into heuristic [14], [15], [16] and learning-based [17], [24], [25], [26], [27] methods. For heuristic methods, Wieckowski et al. [14] introduced some practical tools, such as split cost-based early-termination, content-based gradient speed-up, and residual-based TT split prohibition, etc.



	4	8	16	32	64	128
4	N/A	-0.18%	-0.43%	-0.19%	N/A	N/A
8	-0.19%	-1.52%	-1.82%	-1.47%	0.21%	N/A
16	-0.52%	-1.77%	-8.04%	-3.78%	1.08%	N/A
32	-0.30%	-2.76%	-5.53%	-17.66%	2.51%	N/A
64	N/A	0.14%	0.82%	2.07%	-3.92%	1.59%
128	N/A	N/A	N/A	N/A	1.27%	40.38%

Fig. 1. **(a) Partition results of the I-Slice and B-Slice. (b) Partition depths of the two slices.** The red solid lines are the boundaries of CTUs. The darker colors signify deeper partition depths. **(c) Changes in the Proportion of Pixels Associated with Different CU Sizes in VVC Inter-Slice Compared to Intra-Slice.** Each square indicates a certain size of CUs, with their height and width marked on the left and top. "N/A" indicates not available

Besides, Huang et al. [16] implemented a scheme that allowed for precisely accelerating the encoding process within one pass. For learning-based methods, most research has focused on exploiting temporal motion features with deep learning tools. Amestoy et al. [17] introduced the first complexity reduction for the VTM reference software in the inter coding configuration using random forest classifiers. Pan et al. [24] proposed a CNN scheme by fusing features extracted from the luma component, residue, and motion field. However, their approach could only handle a subset of CUs, which limited the acceleration ratio. Then, Tissier et al. [25] utilized MobileNetV2 [42] as the backbone, which took the current CTU and reference CTUs as input, and predicted a vector that represents all possible CTU partition structures. Furthermore, Peng et al. [26] modeled the structure as a partition homogeneity map (PHM), outperforming the previous work considering the trade-off between complexity reduction and RD performance loss. However, the PHM lacked a tunable complexity reduction mechanism for fast QT+MTT partitioning decisions, which limited the application scenarios. In conclusion, by utilizing effective partition representations, neural network-based methods achieved a higher acceleration ratio with lower compression loss compared to other approaches.

III. Representation of the Partition Structure

In this section, we explore the differences between the partition results of VVC intra and inter coding, and then enhance the partition map by integrating early-termination mechanisms based on the statistical analysis.

A. Observations on Partition Results

To investigate the differences in block partitioning between VVC intra and inter coding, we compress the JVET test sequences using VTM-20.0 at various quantization parameters and analyze the differences in CU size distribution between intra and inter coding. Taking *BasketballDrive_1920×1080* as an example, Fig. 1(a) shows the partition results for the first two frames, i.e., an I-Slice and a B-Slice. While the visual content is similar, CUs in B-Slices tend to be split more coarsely than those in I-Slices, especially since early-terminated CTUs are common in B-Slices but rare in I-Slices. Furthermore, we analyze the proportions of pixels associated with different CU sizes between VVC intra and inter coding, and present the changes in Fig. 1(b). Each square represents a certain size of CUs, with their height and width marked on the left and top. Negative values in blue indicate a decrease in the proportion of pixels belonging to CU sizes in the inter-slice compared to the intra-slice, whereas positive values in orange signify an increase in proportion. The statistics show that the proportion of pixels associated with 128 $\times$ 128 CTUs increases significantly in inter coding compared to that in intra coding.

B. The Improved Partition Map

The partition map is a complete representation of the QT+MTT structure for intra coding [23], as shown in Fig. 2. It comprises a QT depth map, three MTT depth maps, and three MTT direction maps. It can be mutually converted with the QT+MTT partition tree. Specifically, a single QT depth map represents the QT partition, where the depth value indicates the number of QT partitions [38]. MTT depth maps build upon the QT depth map by further adding depth values, corresponding to MTT nodes as child nodes of QT nodes or root nodes in the partition search. When a node is split using a Binary Tree (BT), the depth increases by 1, and for a Ternary Tree (TT), the depth of the middle child node increases by 1, while the depth of the two end nodes increases by 2, considering consistency with depth and granularity of partition. However, depth maps alone are not sufficient, thus, direction maps are designed to construct a complete representation. In particular, each layer of the MTT direction maps operates independently and aligns with the MTT depth map. When a CU is horizontally split, the corresponding value in the MTT direction map is set to 1. The value is set to -1 when a CU is vertically split. Otherwise, the value is set

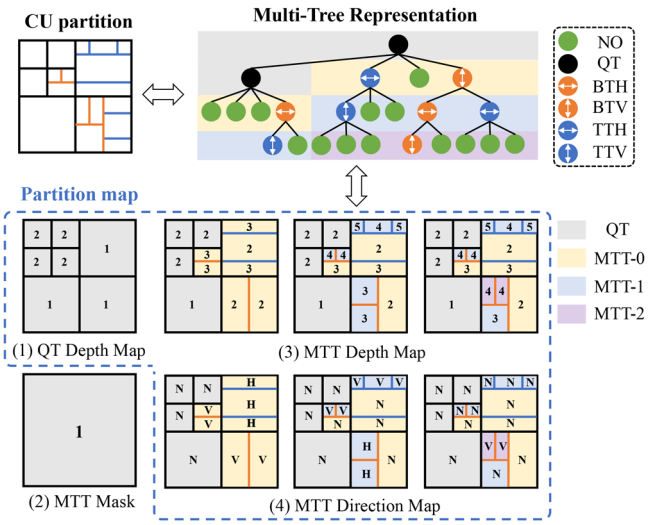


Fig. 2. **An example of the improved partition map**. The original partition map includes three components: QT depth map, MTT depth map, and MTT direction map. It can be converted back and forth with the multi-tree representation. To represent the coarse partition results of inter coding, we introduce an MTT mask that indicates whether or not early terminating MTT splits.

to 0. Thus, a partition map with four layers can be utilized to represent any QT+MTT partition tree. Note that multiple acceleration levels are achieved with the hierarchical representation, which is important in realistic scenarios.

Although the partition map provides a complete representation of the QT+MTT structure, its direct application in VVC inter coding may not be suitable. To adapt the representation to the coarse partition, we introduce a new trade-off flag called the MTT mask, which is associated with the QT depth map and indicates early terminating MTT splits for CTUs. Specifically, when the QT nodes or root nodes stop further splitting, the MTT mask is set to false; otherwise, it is set to true. Thus, the improved partition map consists of the QT depth map, MTT depth/direction map, and MTT mask. The new representation provides two benefits over the previous one, from both the neural network architecture and post-processing perspectives. On the one hand, it effectively models the coarse block partitioning in inter coding, thereby dynamically avoiding the redundant complexity of network inference. On the other hand, by utilizing the MTT mask, a more flexible trade-off between complexity and coding efficiency can be achieved through an appropriate RDO acceleration strategy.

IV. Neural Network Architecture

In this section, a neural network is developed to predict the partition map in a coarse-to-fine manner, where higher-resolution inputs correspond to the partitioning of smaller blocks. The detailed architecture is shown in Fig. 3 and includes the QT depth map, MTT mask, and MTT depth/direction map prediction.

Notations: Let I_c, I_{r+}, and I_{r-} denote the luma components of the current frame, the most recent forward reference frame, and the most recent backward reference frame, respectively. If the backward reference frame is missing, the forward reference frame is replicated as the backward reference frame. The superscripts $'$ and $''$ indicate bilinear downsampling by factors of two and four, respectively. Moreover, Q, M, MD, and $MDir$ represent the QT depth map, MTT mask, MTT depth map, and MTT direction map, respectively.

A. QT Depth Map Prediction

We extract the spatial-temporal features from the luma component $\{I''_c, I''_{r+}, I''_{r-}\}$, and then feed them into the QT Net to predict the QT depth map.

1) Spatial-Temporal Feature Extraction: Given that block partitioning is relevant to both the content of the current frame and the temporal relationship between the reference frames and the current frame, we separate the input frames into two data groups: $\{I''_c, I''_{r+}\}$ and $\{I''_c, I''_{r-}\}$, and then extract features separately. Taking the group $\{I''_c, I''_{r+}\}$ as an example, the optical flow from I''_c to I''_{r+}, denoted as $\tilde{V}''$, is fed into a feature extraction module to obtain the motion features. Additionally, $I''_c - I''_{r+}$ and I''_c are also fed into similar modules to obtain the residual and appearance features, respectively.

In this process, the optical flow is estimated by a lightweight optical flow network called SpyNet [44], which is fine-tuned on the motion vector field derived from the VTM encoder. Tang et al. [45] pointed out that this enhanced optical flow exhibits better visual recovery of sharp motion boundaries and regions with rich details compared to SpyNet pre-trained on synthetic data, e.g., Sintel Dataset. Moreover, we employ a widely used full convolutional network [43] as the feature extraction module. This network comprises several 2D convolutional and asymmetric convolutional layers, leveraging the directional characteristics captured by asymmetric kernels.

2) QT Net: After obtaining the motion, residual, and appearance features, we concatenate them along the channel dimension and input the concatenated features into the QT Net. The detailed structure of the QT Net is illustrated in Fig. 4. Specifically, it comprises three stacked compression-aware sub-networks with top-down and bottom-up processing, and outputs three intermediate QT depth maps $\{Q_0, Q_1, Q_2\}$. During training, we compute the L1 loss between the three maps and the ground truth map, while only the final prediction Q_2 is used during testing. Such repeated top-down and bottom-up processing enhances the *local consistency* of the results through progressive prediction, which refers to the adherence of the predicted results to the quadtree partitioning rules.

The structure of each QT sub-network is depicted in Fig. 4, which includes three key components: the hourglass block, the quantization parameter modulation layer, and the Guided CNN. Within each sub-network, the input feature, denoted as F_{in}, is initially processed by the hourglass block [46] to generate features F_h. The hourglass block efficiently integrates local and global cues through top-down and bottom-up processing [46], [47]. Next, a quantization parameter modulation mechanism is introduced to facilitate compression-aware prediction. Specifically, F_h undergoes channel-wise global average pooling (GAP) and is then combined with the embedding of the target

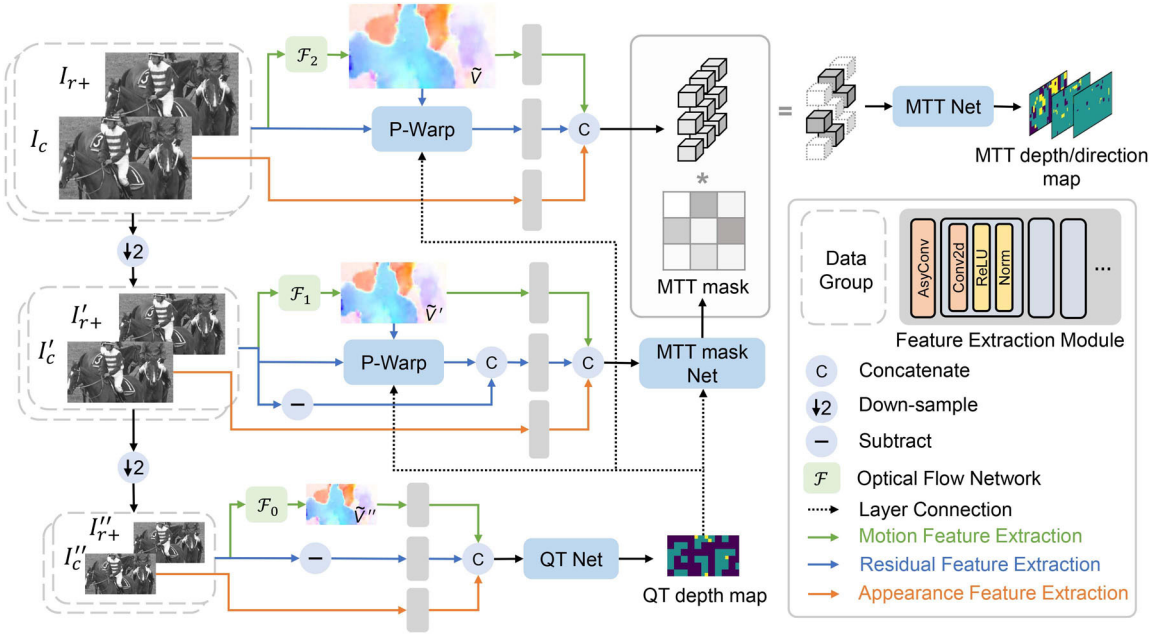


Fig. 3. **Structure of the proposed neural network.** It takes the luma component of the current frame I_c and the two nearest reference frames I_{r+} and I_{r-} as input, and outputs the predicted partition map. At each layer, the three input frames are divided into two data groups $\{I_c, I_r+\}$ and $\{I_c, I_r-\}$. These groups are then individually processed to extract motion features, residual features, and appearance features. The extracted features are concatenated and fed into neural network modules to predict partition maps in a coarse-to-fine manner. "AsyConv" refers to 2D asymmetric convolution [43]. For convenience, only data group $\{I_c, I_r+\}$ is shown in the figure.

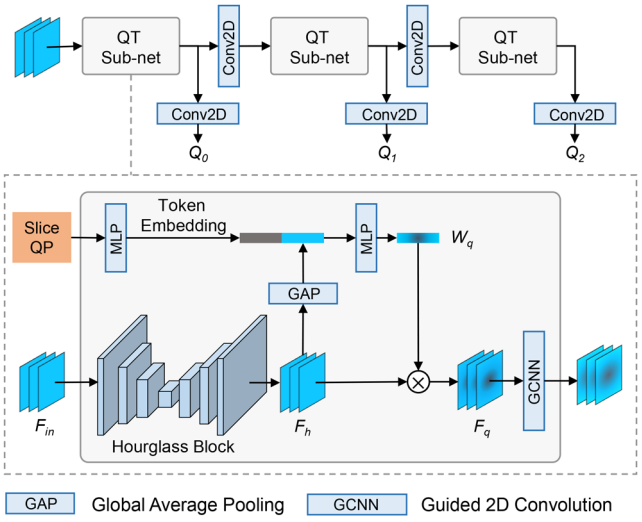


Fig. 4. **Structure of the QT Net.** It consists of three sub-networks, generating three intermediate QT depth maps. Each sub-network comprises an hourglass block with a QP modulation layer and a guided convolutional layer to integrate the targeted encoding quantization parameters and CTU boundary.

quantization parameter to generate channel attention weights, denoted as W_q. The product of F_h and W_q results in F_q, which is modulated with the quantization parameter to adapt to P/B frames with different Temporal IDs (TIDs). Finally, a guided convolutional layer [48] is employed, using a binarized 128×128 grid map as guidance to locate CTU boundaries. In the grid map, CTU boundaries are labeled as -1, while other regions are labeled as 1. Thus, the sub-network effectively incorporates both compression quality and CTU boundaries.

B. MTT Mask Prediction

Similar to QT depth map prediction, spatial-temporal features are first extracted from the frames $\{I'_c, I'_{r+}, I'_{r-}\}$ and then fed into the MTT mask net to predict the MTT mask probabilities. The MTT mask net consists of two stacked sub-networks, similar to the QT sub-networks, along with a softmax function. Drawing inspiration from the relationship between block-based motion estimation and block partitioning in VTM, we integrate the predicted QT partition into the motion feature extraction process to enhance MTT mask prediction.

During the inter coding of VVC, block-based motion estimation (ME) and motion compensation (MC) are generally performed on the assumption that *the motion of pixels within a block tends to be uniform* [49]. This assumption makes the motion field easy to represent with low complexity in a block-based coding framework. To align with this hypothesis, we introduce the partitioning-adaptive warping (P-warp). It enables partitioning-adaptive motion compensation using the predicted QT depth map, as shown in Fig. 5. This capability empowers the MTT mask Net to effectively handle the result of temporal alignment achieved by the QT Net.

The pipeline for the P-warp is shown in Fig. 5. Unlike the standard warping operation, it first transforms the original optical flow V into the partitioning-adaptive optical flow V_p using the predicted QT depth map Q, as follows:

$$V_p = \sum_{k=0}^{3} \left(\mathcal{U}_k \circ \mathcal{D}_k(V)\right) \odot \mathbb{1}\left([Q] = k\right), \tag{1}$$

where $\mathbb{1}(\cdot)$ denotes the element-wise indicator function that outputs 1 when the condition is met and 0 otherwise, $[\cdot]$ indicates a rounding operation, and $\mathcal{D}_k$ and $\mathcal{U}_k$ represent average

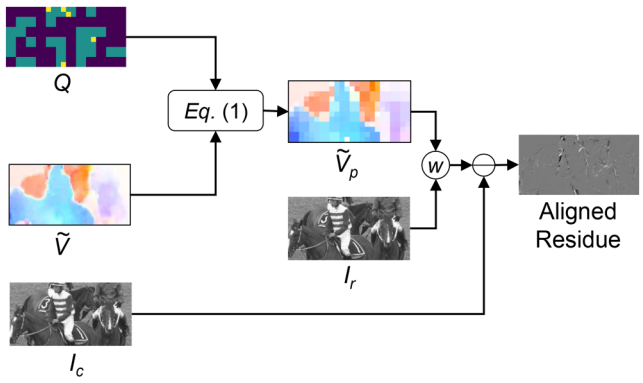


Fig. 5. **Pipeline of the proposed partitioning-adaptive warping (P-warp).** This operation estimates block-based motion compensation in inter coding through partitioning-adaptive optical flow V_p. Here, I_c, I_r, Q, V, and ⓦ represent the current frame, the reference frame, the predicted QT depth map, the predicted optical flow, and warping operations, respectively.

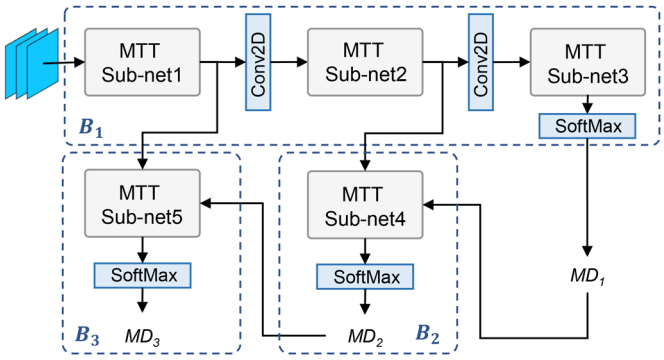


Fig. 6. **Structure of the MTT Net.** It employs a similar architecture to the Down-Up CNN [23], which is used for partition map prediction in VVC intra coding. The module takes a CTU-level feature map as input and progressively outputs the probabilities of MT depth/direction maps.

pooling and nearest up-sampling operations with a stride of k, respectively. The symbol $\odot$ denotes element-wise multiplication. Then, we use V_p to warp the reference frame I_r to the current frame I_c, and the aligned residuals are computed as the output of the P-warp. Compared to directly calculating the aligned residual using V, the aligned residual obtained from V_p reflects the ability of the predicted QT depth maps to reduce temporal redundancy, thereby promoting the subsequent network to recognize areas that require further partitioning.

Given that the predicted QT depth map consists of floating-point values, we use a linear relationship to estimate the soft version of (1) during the training process, as follows:

$$V_p = \sum_{k=0}^{2} \Big((Q - \lfloor Q \rfloor) \odot V^{(k+1)} + (\lceil Q \rceil - Q) \odot V^{(k)} \Big) \odot \mathbb{1}\,(k \le Q < k+1), \tag{2}$$

where $\lfloor\ \rfloor$ and $\lceil\ \rceil$ represent the floor and ceiling operations, respectively, and $V^{(k)} = \mathcal{U}_k \circ \mathcal{D}_k(V)$ denotes the partitioning-adaptive optical flow with a specific granularity. For instance, if the predicted QT depth value Q is 1.2, then during testing, V_p equals $V^{(1)}$, according to (1), which corresponds to the motion vector for a 64×64 CU granularity. During training, the estimated motion vector for that block is computed as $0.8 \times V^{(1)} + 0.2 \times V^{(2)}$ based on (2). Notably, the predicted QT values are close to integers due to the repeated top-down and bottom-up processing in QT Net, thereby reducing the gap between training and testing processes.

C. MTT Depth/Direction Map Prediction

Similar to MTT mask prediction, spatial-temporal features are first extracted from the frames $\{I_c, I_{r+}, I_{r-}\}$ and divided into patches that correspond to CTUs. The less informative CTU patches are discarded based on the predicted MTT mask, thereby reducing the inference cost of predicting MTT depth/direction maps during testing, as shown in Fig. 3. Lastly, the retained CTU-level features are processed by MTT Net, which employs a down-up architecture to predict the probabilities of MTT depth/direction maps.

The structure of the MTT Net is depicted in Fig. 6. It employs a similar architecture to the Down-Up CNN [23], which is used for partition map prediction in VVC intra coding. The network consists of two identical branches that separately predict the MTT depth map and direction map. Focusing on the depth map, the network employs three branches to progressively predict three layers of the MTT depth map, namely B_1, B_2, and B_3. Specifically, the B_1 branch predicts the first layer of the MTT depth map, MD_0, with three stacked sub-networks: MTT sub-net1, sub-net2, and sub-net3. The B_2 branch takes the outputs from MTT sub-net2 as input to predict the second layer of the MTT depth map, MD_1, where MD_0 functions as an attention map. Similarly, the B_3 branch takes the outputs from MTT sub-net1 as input to predict the final layer of the MTT depth map, MD_2, where MD_1 serves as an attention map. Each sub-network adopts a structure similar to the QT sub-network, as shown in Fig. 4. This Down-Up structure facilitates recursive partition search and enables the generation of deeper outputs from shallower features, corresponding to detailed features with finer granularity.

D. Loss Function

To train the proposed neural network, we use different loss functions for different parts of the network. For QT depth map prediction, we utilize the L1 loss with intermediate supervision, defined as:

$$\mathcal{L}_{\mathrm{QD}} = \frac{1}{3} \sum_{i=0}^{2} \mathcal{L}_1(Q_i - \widetilde{Q}), \tag{3}$$

where Q_i denotes the QT depth map predicted by the i-th QT sub-net, and $\widetilde{Q}$ represents the QT depth map labels. For predicting MTT mask, MTT depth map, and MTT direction map, we employ cross-entropy loss, expressed as:

$$\mathcal{L}_{\mathrm{Mask}} = \mathcal{L}_{CE}(p_{\mathrm{M}}, \widetilde{M}), \tag{4}$$

$$\mathcal{L}_{\text{MD}} = \frac{1}{3}\sum_{n=1}^{3}\mathcal{L}_{CE}\left(p_{\text{MD}}^{(n)}, \widetilde{\Delta MD}_n\right), \tag{5}$$

$$\mathcal{L}_{\text{MDir}} = \frac{1}{3}\sum_{n=1}^{3}\mathcal{L}_{\text{CE}}(p_{\text{MDir}}^{(n)}, \widetilde{MDir}_n), \tag{6}$$

where p_M, p_{MD}, and p_{MDir} represent the predicted probabilities of MTT mask, MTT depth map, and MTT direction map, respectively, while $\widetilde{M}$, $\widetilde{MD}$, and $\widetilde{MDir}$ denote their corresponding label values. Here, $\Delta\widetilde{MD}_n$ indicates the difference between the n-th layer of the MTT depth map and the $(n-1)$-th layer depth map. The MTT mask label $\widetilde{M}$ is set to false if every element of the predicted QT depth map Q is greater than or equal to the corresponding elements of MD_0, and true otherwise:

$$\widetilde{M} = \begin{cases} 0, & \text{if } Q \succeq MD_0, \\ 1, & \text{otherwise.} \end{cases} \tag{7}$$

Here, $\succeq$ denotes the element-wise "greater than or equal to every" operation between two matrices. The total loss function is the sum of all losses, as follows:

$$\mathcal{L} = \mathcal{L}_{\text{QD}} + \mathcal{L}_{\text{Mask}} + \mathcal{L}_{\text{MD}} + \mathcal{L}_{\text{MDir}}. \tag{8}$$

V. Post-Processing Algorithm

This section introduces a post-processing algorithm developed for the improved partition map. Similar to various existing methods that predict a CTU or large CU partition as a whole [21], [23], [38], [19], a post-processing algorithm is necessary to generate a partition structure in accordance with the VVC standards. Moreover, we propose a dual threshold decision scheme that enables a flexible trade-off between complexity reduction and coding efficiency.

A. Map Tree-Based Post-Processing Algorithm

Given the predicted partition map, we apply the map tree-based post-processing algorithm proposed in [23] to generate a set of standard-compliant split mode decisions. The algorithm includes two stages. In the first stage, a depth-first search order is used to retrieve all possible partition structures starting from the root node, generating a collection of candidate partition maps. Thus, each node maintains a temporary partition map named the *map tree*. From the recorded map trees, multiple candidate partition paths can be derived. In the second stage, a predefined criterion is used to select the partition path with the least error from the candidates. Specifically, the corresponding criterion is as follows:

$$\begin{aligned} Error = &\|MD_t - MD_p[\text{curMTTdepth}]\|_1 \\ &+ \|MDir_t - MDir_p[\text{curMTTdepth}]\|_1 \end{aligned} \tag{9}$$

where MD_t and $MDir_t$ are the temporal partition layer of the MTT depth map and direction map, respectively, and $MD_p[\text{curMTTdepth}]$ and $MDir_p[\text{curMTTdepth}]$ are the current partition layer of the predicted MTT depth map and direction map. Although originally designed for CTUs with a shape of 64×64, this algorithm can be easily extended to partition maps corresponding to CTUs with a shape of 128×128 for VVC inter coding, thereby providing a set of standard-compliant partition mode decisions.

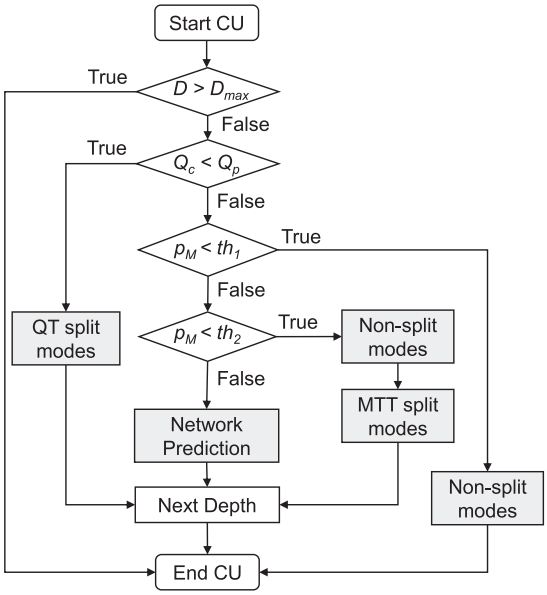


Fig. 7. **Flowchart of the dual threshold decision scheme.** When the current decision depth D is not larger than the maximum depth $D_{\max}$, and the current QT depth Q_c is not less than the predicted QT depth Q_p, the dual-threshold approach is performed to balance network prediction and RDO search. p_M denotes the predicted probabilities of the MTT mask. The "Next Depth" indicates the progression towards deeper levels of partitioning, given a recursive pipeline of the RDO. The modules marked in gray require mode checking.

B. Dual Threshold Decision Scheme

Although the four layers of the partition map enable four levels of acceleration, a fine-grained trade-off is required between complexity reduction and coding efficiency in practical scenarios. Additionally, since the neural network's predictions are not always accurate, incorrect decisions during the encoding process, particularly in context-sensitive inter coding, can result in a significant degradation in RD performance. To address these issues, based on the softmax value of the predicted MTT mask, we propose a dual threshold decision scheme that introduces a flexible trade-off between the neural network's predictions and the recursive RDO search, as shown in Fig. 7. Specifically, we set a lower threshold th_1 to early-terminate the MTT split mode for CTUs with softmax values below this threshold th_1. If the softmax value exceeds th_1, we treat the softmax value as a confidence score and introduce a higher threshold th_2. For softmax values lower than th_2, the CTUs perform the default RDO search; otherwise, the CTUs with higher softmax values follow the split decisions predicted by the neural network. By adjusting both thresholds, we can avoid too much RD performance degradation while maintaining high encoding time savings. Note that the scheme does not involve QT partition acceleration, as the RD loss resulting from accelerated QT partitions is negligible in practice.

VI. Experiments

In this section, we evaluate the effectiveness of our approach. Part VI-A presents dataset construction and configurations. The performance of the proposed methods is evaluated in VI-B. We then perform ablation studies on key techniques in VI-C. Moreover, we discuss adaptation to extended basic QPs and threshold selection in VI-D and VI-E, respectively. Lastly, we analyze the complexity of the proposed method in VI-F.

A. Configuration and Settings

Dataset Construction: We established a large-scale training dataset for the proposed neural network. Firstly, we collected 2,672 sequences from widely recognized 4 K sequence datasets, including the BVI-DVC dataset [50], the Tencent Video Dataset [51], and the UVG dataset [52]. These sequences were processed into short sequences of 32 frames across different resolutions, including 3840 × 2160, 1920 × 1080, 960 × 544, and 480 × 272. Next, we used VTM-10.0 to compress these sequences with the random access configuration defined by the *encoder_randomaccess_vtm_gop32.cfg* settings [53] at four basic QPs: {22, 27, 32, 37}, with an intraperiod of 32. Lastly, we obtained block partitioning results and converted them into partition maps. During the dataset construction, several fast tools defined in the configuration file were disabled to ensure an accurate block partitioning structure.

Training Configuration: We use the entire frame as input and train the neural network for a total of 1,200 epochs. The training process is divided into two stages. In the first stage, we progressively train the modules for QT depth map prediction, MTT mask prediction, and MTT depth/direction map prediction, requiring 500, 300, and 300 epochs, respectively. During this stage, the gradients of previously trained components are frozen while training subsequent modules. The batch sizes for 4 K sequences are 160, 32, and 8, corresponding to the three input scales. In the second stage, we jointly train the entire network for 100 epochs, with the batch size for 4 K sequences set to 8. During the training process, for sequences with resolutions of 1920×1080, 960×544, and 480×272, the batch sizes are 4, 16, and 64 times those used for the 4K sequences, respectively. The resolution of the input sequences is changed every 5 epochs.

All parameters are initialized using Xavier initialization [54]. The Adam optimizer [55] is used with an initial learning rate of 1×10^{-3} for the first stage and 1×10^{-4} for the second stage. The learning rate is reduced by a factor of 0.98 every 10 epochs. The entire training process takes two weeks to complete using eight 1080Ti GPUs. Consistent with previous work [23], we train models separately for four basic QPs: 22, 27, 32, and 37.

Evaluation Configuration: In our experiments, we integrate the proposed method into the VVC reference software VTM-10.0[1] to maintain compatibility with the state-of-the-art tunable encoding complexity reduction approach for VVC inter coding [25]. Our method is evaluated using the first 65 frames from 22 video sequences, based on the JVET common test conditions [29]. The evaluation uses the random access configuration defined by the *encoder_randomaccess_vtm.cfg* setting [53], with four basic QPs $\{22, 27, 32, 37\}$. Moreover, we assess the method with a broader slice QP range induced by a larger GOP size, as defined by the *encoder_randomaccess_vtm_gop32.cfg* setting. Note that the proposed fast algorithm does not process I-frames and only accelerates the encoding of B-frames. We measure encoding performance using the Bjøntegaard Delta Bit Rate (BD-rate/BDBR) metric [56], comparing the proposed method to the original VTM encoder. The reduction in complexity is evaluated using Encoding Time Saving (ETS), defined as:

$$\mathrm{ETS} = \frac{T_{\mathrm{anchor}} - T_{\mathrm{test}}}{T_{\mathrm{anchor}}} \times 100\%, \tag{10}$$

where T_{anchor} denotes the encoding time of the original VTM encoder, and T_{test} represents the total time of the accelerated encoder, including network inference, post-processing, and encoding time. We also use Encoding Time Acceleration (ETA) to represent complexity reduction at high efficiency settings, defined as:

$$\mathrm{ETA} = \frac{1}{1 - \mathrm{ETS}} = \frac{T_{\mathrm{anchor}}}{T_{\mathrm{test}}}, \tag{11}$$

which reflects the nonlinear relationship between the marginal savings in encoding time and encoding efficiency. Specifically, as encoding time decreases, further reductions become increasingly difficult without significant degradation in RD performance. For example, reducing encoding time from 40% to 45% may result in only a marginal increase in BDBR, whereas reducing encoding time from 80% to 85% with the same BDBR increase becomes progressively more challenging. Thus, at high complexity reduction settings, the marginal savings in encoding time become more pronounced, and ETA is more effective at capturing this than ETS.

All evaluation experiments are conducted on a computer with an Intel Xeon(R) CPU E5-2690@2.60 GHz and 256 GB of RAM, running Microsoft Windows Server 2012 R2. Hyperthreading is disabled to reduce variability in encoding time measurements. GPUs are disabled during the evaluation process by default.

B. Performance Evaluation

Notations for Acceleration Levels: The proposed method achieves tunable complexity reduction by either pruning the partition map or adjusting the thresholds of the post-processing algorithm. We introduce the notation used to represent different levels of acceleration. The coarse-grained acceleration level, denoted as L_n, refers to the acceleration of partition search using a pruned partition map, where n ranges from 0 to 3, corresponding to the four layers of the partition map. For example, L_1 converts both the QT depth map and the first layer of the MTT depth/direction map into split decisions, allowing for early termination of redundant partition modes. The fine-grained acceleration level involves a dual-threshold decision scheme, which can balance network prediction and RDO search by adjusting the thresholds th_1 and th_2. For instance, $L_1(th_1, th_2)$ indicates that CTUs with an MTT mask prediction probability below th_1

[1] https://vcgit.hhi.fraunhofer.de/jvet/VVCSoftware_VTM

TABLE I
COMPARISON WITH RELATED METHODS AT A LOW COMPLEXITY REDUCTION SETTING. A SMALLER BDBR (%) INDICATES LESS RD PERFORMANCE DEGRADATION, WHILE A LARGER ETS (%) REPRESENTS MORE ENCODING TIME SAVING

Class	Pan *et al.* [24]		Tissier *et al.* [25]		Peng *et al.* [26]		Lin *et al.* [27]		Ours $L_0(0,1)$		Ours $L_1(0.2,1)$	
	BDBR	**ETS**	**BDBR**	**ETS**	**BDBR**	**ETS**	**BDBR**	**ETS**	**BDBR**	**ETS**	**BDBR**	**ETS**
A1	2.98	40.97	1.81	51.10	2.01	48.10	2.36	49.75	1.29	48.32	1.47	52.78
A2	4.73	32.92	1.86	44.60	1.71	47.20	2.56	47.66	2.38	49.86	2.61	54.53
B	3.76	30.93	2.21	46.50	2.17	47.90	2.54	47.50	1.99	46.49	2.93	55.64
C	2.56	25.98	3.20	43.10	1.98	44.20	2.67	52.92	1.39	38.32	1.52	39.92
E	2.81	35.16	1.45	38.70	1.66	40.80	1.52	28.67	1.27	38.94	2.06	53.61
Average	**3.37**	**33.19**	**2.11**	**44.80**	**1.91**	**45.64**	**2.33**	**45.30**	**1.66**	**44.39**	**2.12**	**51.30**
D	2.35	22.05	3.02	36.80	2.09	38.50	2.50	41.42	1.78	27.37	2.04	37.66

skip MTT splits; those with a probability equal to or above th_2 follow the network decision; and the remaining CTUs undergo the default RDO search.

We compare our method with previous fast block partitioning algorithms for VVC inter coding [15], [16], [24], [25], [26], [27] under different complexity reduction settings. In the low complexity reduction setting, we present a detailed comparison with advanced methods [24], [25], [26], and [27] in Table I. Our method achieves an average reduction in encoding time of 44.39% to 51.30% at two acceleration levels, $L_0(0,1)$ and $L_1(0.2,1)$, with a BDBR degradation ranging from 1.66% to 2.12%. Compared to the tunable encoding complexity reduction method proposed by Tissier et al. [25], our approach achieves a 6.5% further reduction in encoding time, with comparable RD performance degradation. In comparison with the state-of-the-art method proposed by Peng et al. [26], which lacks scalability in complexity reduction, our approach not only achieves a 0.31% lower BDBR under similar complexity settings, but also supports a wider range of encoding complexity levels. Notably, our method provides a better trade-off between coding efficiency and complexity reduction at $L_1(0.2,1)$, benefiting from the introduction of the MTT mask and the design of the dual-threshold decision scheme. Furthermore, our method can be extended to higher complexity reduction settings. For example, it saves 63.21% of encoding time with a 5.31% increase in BDBR at $L_1(0.2,0.9)$, as shown in Fig. 8. However, few studies explore acceleration ratios beyond a 2.5x speedup while maintaining acceptable RD loss, highlighting the potential of partition map-based methods. Therefore, our approach not only outperforms other methods under regular complexity reduction settings but also enables higher potential complexity reduction trade-offs, which few previous studies achieve.

Given that the proposed method does not accelerate I-frames, it is important to investigate its combination with existing fast algorithms for VVC intra coding. Thus, we apply the fast algorithm [23] to speed up the encoding of I-frames, while using the proposed method to accelerate the encoding of B-frames in RA mode, as shown by "Ours$^+$" in Fig. 8. The results indicate that accelerating the encoding of I-frames does *not necessarily* result in a better trade-off between coding efficiency and complexity reduction. At lower complexity reduction levels, fast algorithms tend to exhibit a greater sensitivity to RD performance degradation than to time savings. In contrast, at higher complexity reduction levels, the impact of complexity

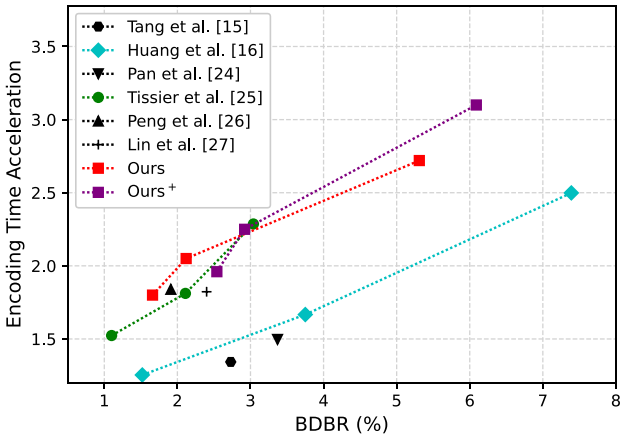


Fig. 8. **The trade-off between complexity and coding efficiency compared to other methods.** The horizontal axis represents the RD performance loss, while the vertical axis represents the encoding time acceleration ratio. The dashed line in the figure connects different trade-off points and does not indicate attainable values along the line. We present our approach at three acceleration levels: $L_0(0,1)$, $L_0(0.2,1)$, and $L_1(0.2,0.9)$. "Ours$^+$" refers to applying our method to accelerate B-frames and using the fast block partitioning method for VVC intra coding [23] to speed up I-frames.

reduction becomes more pronounced. For instance, at the acceleration level $L_0(0,1)$, the encoding time reduction achieved by accelerating I-frames does not sufficiently offset the associated RD performance degradation, as compared to $L_0(0.2,1)$. However, at $L_1(0.2,0.9)$, where encoding time is more critical, accelerating the encoding of I-frames significantly reduces encoding time with modest RD degradation, leading to a better trade-off.

Both methods exhibit some degree of overlap in RD performance degradation. From Table II, we observe that the actual increase in BDBR when accelerating both I-frames and B-frames is lower than the sum of the increases when each frame type is accelerated separately. Specifically, the BDBR increase when accelerating I-frames alone is 0.97%, when accelerating B-frames alone is 2.12%, and when accelerating both I-frames and B-frames is 2.84%, which is lower than the expected additive value of 0.97 + 2.12 = 3.09%. This discrepancy arises because accelerating I-frames not only affects the I-frames but also affects B-frames through inter-frame dependencies. The degradation of I-frames leads to low reference quality for subsequent B-frames, resulting in overlapping RD degradation.

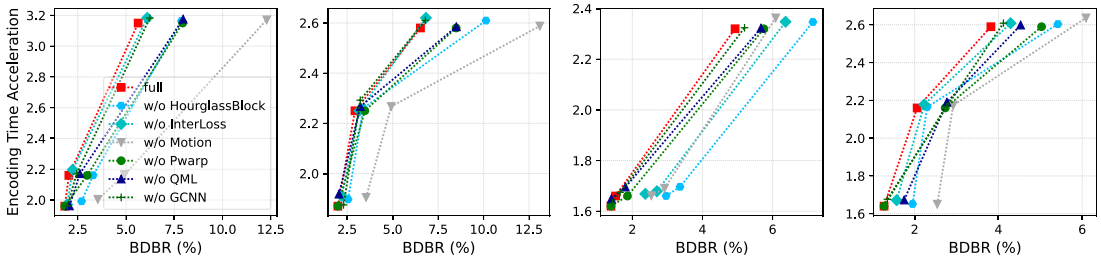


Fig. 9. **Ablation Experiments of Different Techniques.** From left to right, the results correspond to Class A, B, C, and E, respectively.

TABLE II
COMBINATION OF FAST ALGORITHMS FOR INTRA AND INTER CODING. "INTRA" REFERS TO ACCELERATING I-FRAMES USING [23], WHILE "INTER" REFERS TO ACCELERATING B-FRAMES USING THE PROPOSED METHOD AT ACCELERATION LEVEL $L_0(0.2, 1)$.

Class	Intra		Intra + Inter	
	BDBR	**ETS**	**BDBR**	**ETS**
A1	0.50	3.65	1.85	56.31
A2	0.71	3.92	3.17	58.40
B	0.69	3.75	3.53	59.69
C	0.87	2.93	2.14	42.78
E	2.06	8.40	3.89	61.02
Average	**0.97**	**4.53**	**2.84**	**55.64**

We evaluate the proposed method using VTM-10.0 and VTM-23.0, with a GOP size of 32 at three different acceleration levels, as detailed in Table III. The results indicate that the encoding time savings of our method remain consistent despite a larger GOP size or a newer software version. In contrast, increasing the GOP size from 16 to 32 results in a BDBR increase ranging from 0.05% to 0.18% across all acceleration levels, attributed to a broader range of slice QPs and the more complex inter-frame dependencies. In the case of VTM-23.0, a BDBR increase of 0.43% to 0.85% is observed, primarily due to the model being trained on the dataset generated by VTM-10.0. The generalization of our proposed scheme across different versions is robust, due to the similarity in core coding tools after version 10. Specifically, VTM-10 defined the core bitstream structure for VVC, and subsequent versions differ primarily in terms of encoder-side optimizations. In general, the proposed method works well with longer GOP sizes and across software versions.

We present the accuracy of both the network output and the post-processed partition map in Table IV. Post-processing, which converts the partition map into a set of standard-compliant partition mode decisions, may lead to a reduction in accuracy. On one hand, the accuracy of QT depth map predictions remained stable before and after post-processing, contributing to the remarkable local consistency of the predicted depth map. On the other hand, as the partition depth increases, the accuracy of the depth maps decreases while the accuracy of the direction maps rises, which may be attributed to the higher proportion of zero values in the direction maps.

C. Ablation Studies

We conduct a series of ablation experiments to assess the effectiveness of key components, categorized into three groups: (1) techniques related to repeated top-down and bottom-up processing (e.g., hourglass blocks and intermediate supervision), (2) motion-related techniques (e.g., motion feature extraction and partitioning-adaptive warping), and (3) other techniques. The impacts on coding efficiency and complexity reduction when selectively removing these techniques are shown in Fig. 9.

Ablation Study on Repeated Top-Down and Bottom-Up Processing: To improve the accuracy and local consistency of the predicted QT depth map, we adopt a repeated top-down and bottom-up architecture comprising stacked hourglass blocks and intermediate supervision. Specifically, our approach employs three stacked QT sub-networks built on hourglass blocks to integrate global and local contexts, with intermediate supervision applied to the QT sub-networks, labeled as full setting. In the ablation study, we replace the hourglass blocks with dense blocks [57] of comparable complexity, labeled as *w/o HourglassBlock*. We also remove the intermediate losses in (3) and compute the loss function based only on the output of the final QT sub-network, labeled as *w/o InterLoss*. The prediction accuracy and inconsistency error for these three settings are detailed in Fig. 11. Here, the inconsistency error indicates the proportion of inconsistent values before and after post-processing, reflecting how well the model's predictions align with quadtree partitioning rules. Furthermore, we present the progressive prediction results corresponding to three settings, using the second frame of *Marketplace* as an example in Fig. 11. Compared to the full setting, both *w/o HourglassBlock* and *w/o InterLoss* exhibit lower prediction accuracy. The former emphasizes the significance of hourglass blocks in capturing both global and local contexts. The latter demonstrates similar accuracy to the first QT sub-network in the full setting, but the subsequent QT sub-networks cannot further improve prediction accuracy in the absence of intermediate supervision. In addition to prediction accuracy, our approach ensures that the final predictions adhere to quadtree partitioning rules, thus reducing the burden on post-processing algorithms. Intermediate supervision is crucial for enhancing local consistency, and through interactive top-down and bottom-up processing, the model can maintain precise local information while reconsidering the overall coherence of the predicted frame-level QT depth map, thereby solving such a structured prediction problem.

Ablation Study on Motion-Related Techniques: Given the correlation between motion features and block partitioning in inter coding, we evaluate the effects of motion-related techniques, including motion feature extraction and partitioning-adaptive warping. Specifically, we define two settings as follows:

TABLE III
PERFORMANCE EVALUATION ON VTM-10.0 AND VTM-23.0 WITH A GOP SIZE OF 32

Class	VTM-10.0						VTM-23.0					
	$L_0(0,1)$		$L_0(0.2,1)$		$L_1(0.2,0.9)$		$L_0(0,1)$		$L_0(0.2,1)$		$L_1(0.2,0.9)$	
	BDBR	**ETS**	**BDBR**	**ETS**	**BDBR**	**ETS**	**BDBR**	**ETS**	**BDBR**	**ETS**	**BDBR**	**ETS**
A1	1.24	46.64	1.52	51.92	4.83	70.86	2.04	43.92	2.30	50.73	5.26	68.36
A2	2.52	47.70	2.91	53.98	6.49	66.71	3.21	45.48	3.61	53.08	7.37	71.44
B	2.09	45.30	3.02	54.31	6.76	70.53	3.10	45.06	3.97	54.35	7.03	69.13
C	1.42	35.81	1.61	39.93	5.03	57.97	2.49	33.46	2.60	37.70	5.19	55.91
E	1.30	37.37	2.25	52.07	4.32	60.90	1.50	34.65	2.37	50.17	3.85	57.61
Average	**1.71**	**42.56**	**2.26**	**50.44**	**5.49**	**65.39**	**2.47**	**40.52**	**2.97**	**49.21**	**5.74**	**64.49**
D	1.74	24.07	2.19	28.59	4.78	36.64	4.60	21.78	4.97	23.19	5.49	35.74

TABLE IV
ACCURACY OF NETWORK OUTPUT AND POST-PROCESSED PARTITION MAPS (%)

	QP	QD	MTT mask	MD_1	$MDir_1$	MD_2	$MDir_2$	MD_3	$MDir_3$	Average
Network output	22	63.64	84.49	63.64	56.10	45.25	63.22	46.41	73.71	61.93
	27	71.25	84.03	68.11	61.46	56.28	76.81	56.70	87.48	70.05
	32	76.36	84.08	69.23	64.89	62.09	81.71	62.64	91.16	73.83
	37	80.07	86.19	71.44	70.73	68.88	82.79	69.17	84.83	76.53
	Average	72.83	84.70	68.10	63.30	58.13	76.13	58.73	84.29	70.58
Post-processed	22	63.66	84.49	61.37	54.87	45.03	58.44	46.70	77.15	61.34
	27	71.11	84.03	65.40	59.66	56.06	73.24	56.62	87.28	68.95
	32	76.41	84.08	67.21	62.87	61.78	75.59	62.63	90.31	72.43
	37	80.03	86.19	68.82	66.19	68.51	83.90	69.18	95.26	77.02
	Average	72.80	84.70	65.70	60.90	57.85	72.79	58.78	87.50	69.94

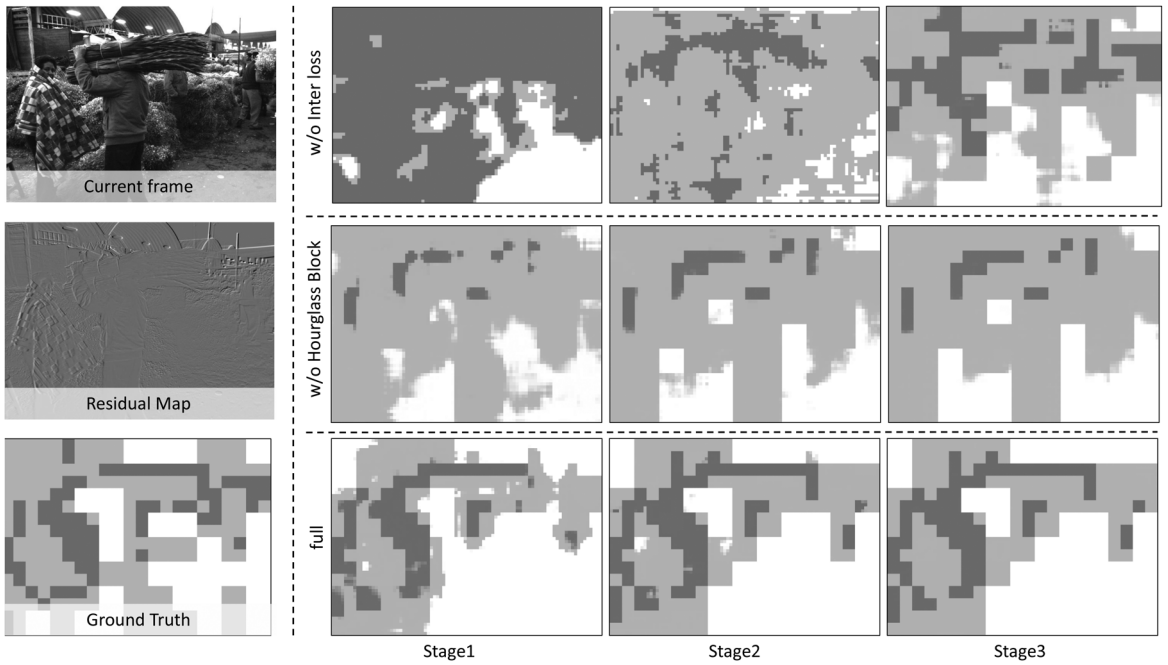


Fig. 10. **Visualization of the predicted QT depth map.** On the left, from top to bottom, are the luma components of the current frame, the residuals between the current and reference frames, and the QT depth map for the current frame, where darker colors indicate deeper partitioning. On the right are the predictions from three QT sub-networks under different settings. The term "full" refers to the default settings, while *w/o HourglassBlock* and *w/o InterLoss* refer to replacing the hourglass blocks with dense blocks and removing intermediate supervision, respectively.

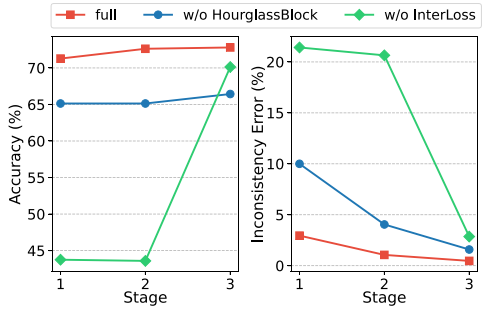


Fig. 11. **The accuracy and inconsistency error across different stages of QT-Net.** The inconsistency error reflects how well the prediction results adhere to the quadtree partitioning rules. Given the absence of intermediate loss functions to train the first two sub-networks, we use the prediction layer from the final stage to obtain the intermediate results for the first two layers.

- *w/o Motion* involves the removal of motion feature extraction by setting the features corresponding to optical flow and residual to constant values.
- *w/o Pwarp* refers to replacing partitioning-adaptive warping with a standard warping operation.

As shown in Fig. 9, removing the motion branch significantly impacts the trade-off between coding efficiency and encoding complexity reduction, highlighting the importance of motion features in our method. Similarly, replacing partitioning-adaptive warping with standard warping results in an obvious RD performance loss at acceleration levels $L_0(0.2, 1)$ and $L_1(0.2, 0.9)$, since the standard warping operation does not incorporate prior partitioning results into motion feature extraction.

Ablation Study on Other Techniques: We conduct experiments to evaluate the impact of other techniques within our approach. The cases under examination, each removing a specific component, are defined below:

- *w/o QML* indicates the model that sets the input of the QP modulation layer to a constant value.
- *w/o GCNN* denotes a model that excludes the Guided CNN in each sub-module of QT-Net.

The ablation study results are also shown in Fig. 9. It is evident that the RD performance is affected to different degrees when either the QP modulation layer or GCNN is removed. This highlights the effectiveness of compression-aware design in the QT sub-networks.

D. Quantization Parameters Adaptation Analysis

Although our model is trained on four basic QPs, this does not necessitate training a separate model for each basic QP, which would be impractical. Our method supports deployment across other basic QPs, as the training dataset for each basic QP includes frames corresponding to various slice QPs, where the slice QP is fed into the QP modulation layers. In the supplementary material, we present the prediction results when the input slice QP changes, demonstrating that the QP embedding can adjust the granularity of the predicted block partition structure. Building on this, we extend the model trained for each basic QP, denoted as $\mathrm{QP}_{\mathrm{basic}}$, to cover the adjacent QPs $\{\mathrm{QP}_{\mathrm{basic}} \pm 1, \mathrm{QP}_{\mathrm{basic}} \pm 2\}$.

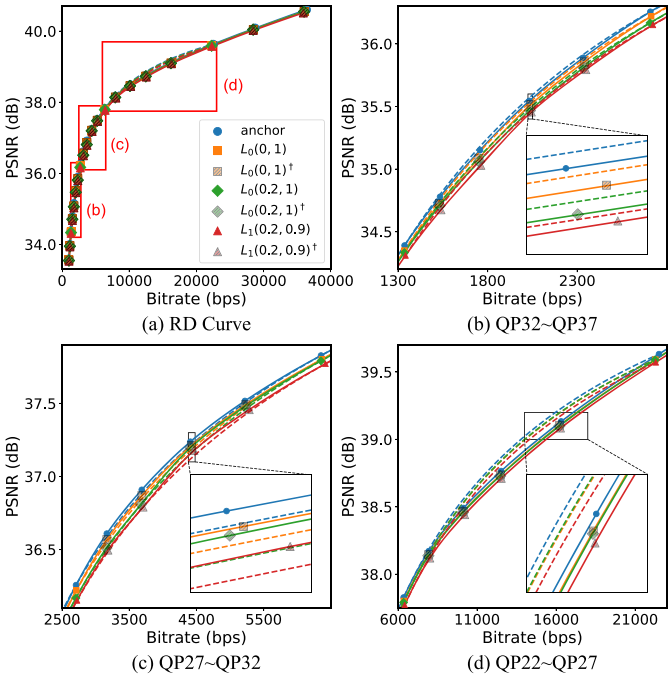


Fig. 12. **Rate-Distortion Curve with Extended Basic QPs.** Using the class B sequences as an example, we plot the RD points for basic QPs ranging from 20 to 39. The term "anchor" refers to the original VTM encoder, while the superscript † denotes the model applied to the extended QPs.

As a result, the four models can cover all QPs ranging from 20 to 39. Fig. 12 presents the RD curves for these QPs on the Class B sequence, where "anchor" represents the original VTM encoder, and the superscript † indicates the model applied to the extended QPs. The solid and dashed lines correspond to piecewise cubic interpolation for all QPs and the four basic QPs, respectively, which form the basis for calculating BDBR. In Fig. 12(b), (c), and (d), we show zoomed-in RD curves for three specific QP ranges. The results indicate that our approach achieves comparable RD performance on extended QPs to that on the four basic QPs across all three acceleration levels.

Furthermore, we investigate quantitative changes in BDBR and ETS by evaluating our approach across twenty QPs, denoted as $\mathcal{Q}_{\mathrm{total}} = \{20, 21, 22, \ldots, 39\}$, in comparison to the four basic QPs, denoted as $\mathcal{Q}_{\mathrm{basic}} = \{22, 27, 32, 37\}$. We use the percentage deviations ΔBDBR and ΔETS to represent the relative differences in RD performance and complexity reduction, respectively, as defined:

$$\Delta\mathrm{ETS} = \left(\frac{\sum_{i \in \mathcal{Q}_{\mathrm{total}}} \mathrm{ETS}(i)}{5 \times \sum_{i \in \mathcal{Q}_{\mathrm{basic}}} \mathrm{ETS}(i)} - 1 \right) \times 100\%, \tag{12}$$

$$\Delta\mathrm{BDBR} = \left(\frac{\mathrm{BDBR}_{\mathrm{total}}}{\mathrm{BDBR}_{\mathrm{basic}}} - 1 \right) \times 100\%. \tag{13}$$

Here, $\mathrm{ETS}(i)$ represents the encoding time savings for QP i, and $\mathrm{BDBR}_{\mathrm{total}}$ and $\mathrm{BDBR}_{\mathrm{basic}}$ denote the BDBR for $\mathcal{Q}_{\mathrm{total}}$ and $\mathcal{Q}_{\mathrm{basic}}$, respectively. A negative value of ΔBDBR indicates the coding gain achieved by the extended QPs, while a positive value of ΔETS reflects improved encoding time savings. The

TABLE V
PERCENTAGE DEVIATION OF BDBR AND ETS WHEN EVALUATING OUR APPROACH ACROSS A TOTAL OF TWENTY QPS, COMPARED TO THE FOUR BASIC QPS. A NEGATIVE VALUE OF ΔBDBR INDICATES THE CODING GAIN ACHIEVED BY THE EXTENDED QPS, WHILE A POSITIVE VALUE OF ΔETS REFLECTS THE GAIN IN ENCODING TIME SAVINGS.

	$L_0(0,1)$	$L_0(0.2,1)$	$L_1(0.2,0.9)$
ΔBDBR↓	-8.78%	0.37%	-13.50%
ΔETS↑	0.36%	1.08%	0.24%

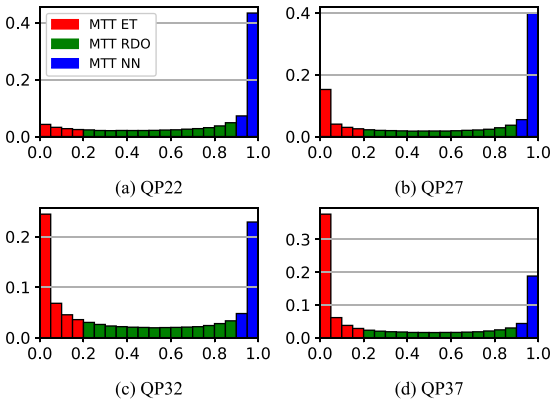


Fig. 13. **Distribution of Predicted MTT Mask Values.** The horizontal axis represents the predicted MTT mask values, while the vertical axis denotes the distribution frequency. We set th_1 and th_2 to 0.2 and 0.9, respectively, and present three types of CTUs. "MTT ET" refers to early-terminating MTT splits, "MTT RDO" represents performing the default RDO search, and "MTT NN" denotes using the prediction results obtained from the MTT Net.

results are presented in Table V. Our approach achieves better RD performance and encoding time savings on $\mathcal{Q}_{\text{total}}$ than on $\mathcal{Q}_{\text{basic}}$. Specifically, at the acceleration levels $L_0(0,1)$ and $L_1(0.2,0.9)$, relative coding gains of 8.79% and 13.50% are achieved on $\mathcal{Q}_{\text{total}}$ compared to $\mathcal{Q}_{\text{basic}}$. These results suggest that our method demonstrates robust QP adaptation, effectively generalizing to other QPs and alleviating concerns regarding the need to train separate models for each basic QP.

E. Analysis of Dual-Threshold Selection

Given the significant impact of the dual-threshold decision scheme on tunable complexity reduction, it is crucial to discuss how to select the optimal threshold values. The scheme uses predicted MTT mask values, denoted as p_M, which range from 0 to 1. The closer p_M is to 1, the more likely the corresponding CTU will be split with MTT splits. In fact, the distribution of p_M follows a bimodal pattern, as shown in Fig. 13. The overall CTUs can be classified into three categories—"MTT ET", "MTT RDO", and "MTT NN"—based on two thresholds th_1 and th_2. When $p_M < th_1$, MTT splits are skipped, and when $p_M \geq th_2$, MTT splits predicted by the MTT Net are applied. For the remaining CTUs, the neural network cannot confidently decide whether to split the blocks, and the default RDO search in VTM is executed. In our implementation, th_1 and th_2 are set to 0.2 and 0.9, respectively, ensuring that only a small fraction of CTUs undergo the exhaustive RDO search across various

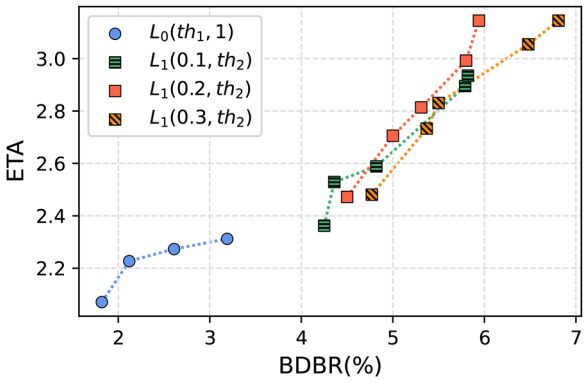


Fig. 14. **Trade-off between RD performance and encoding complexity reduction with different threshold values.** For $L_0(th_1, 1)$, the threshold value th_1 is sequentially set to 0.1, 0.2, 0.3, and 0.4 from left to right. For $L_1(0.1, th_2)$, $L_1(0.2, th_2)$, and $L_1(0.3, th_2)$, the threshold value th_2 is sequentially set to 0.98, 0.95, 0.9, 0.8, and 0.7 from left to right.

QPs. This yields a substantial reduction in encoding time with minimal RD efficiency loss.

Furthermore, we validate the rationality of these empirical threshold values. First, we adjust th_1 from 0.1 to 0.4 at acceleration level $L_0(th_1, 1)$, and present the trade-off between RD performance and encoding time reduction in Fig. 14. As th_1 increases, the encoding acceleration ratio improves as more CTUs skip MTT splits, while the BDBR gradually increases. Specifically, when th_1 increases from 0.1 to 0.2, the BDBR increases by 0.3%, and the ETA rises from 1.95 to 2.09. However, the rate of increase of ETA slows down once th_1 exceeds 0.2. Thus, we select $L_0(0.2, 1)$ as a trade-off point, where an average of 34.72% of CTUs are skipped for MTT splits, as detailed in Table VII, thereby alleviating the burden of MTT Net inference overhead. Subsequently, we explore different threshold combinations for $L_1(th_1, th_2)$, with th_1 ranging from 0.1 to 0.3 and th_2 ranging from 0.7 to 0.98, as shown in Fig. 14. The results indicate that setting th_1 to 0.2 achieves a higher acceleration ratio compared to other threshold values. Meanwhile, setting th_2 to 0.9 yields an ideal configuration with acceptable RD efficiency loss.

F. Complexity Overhead Analysis

We analyze the complexity of the proposed method, including the inference cost of the neural network and the overhead caused by the fast algorithm.

Inference Cost of Neural Networks: To assess the inference cost of neural networks, we divide the proposed network into two parts: the pre-trained optical flow neural network and the remaining components. The optical flow network serves as a plug-and-play module. For the remaining components, QT depth map prediction requires 5.07 GFLOPs and 1.11 M parameters for a single 1080p frame. When combined with MTT mask prediction, the total rises to 7.37 GFLOPs and 3.64 M parameters. These correspond to the low and high complexity reduction settings, respectively. Compared to the state-of-the-art method [25], which uses MobileNetV2 as the backbone and requires 3.4 M

TABLE VI
BREAKDOWN OF TIME CONSUMPTION AT DIFFERENT ACCELERATION LEVELS. THE TERMS T_{ENC}, T_{NET}, AND T_{POST} REFER TO THE TIME TAKEN BY THE VTM ENCODER, NETWORK INFERENCE, AND POST-PROCESSING, RESPECTIVELY, MEASURED IN SECONDS PER FRAME. ρ INDICATES THE PROPORTION OF THE TOTAL TIME SPENT ON BOTH NETWORK INFERENCE AND POST-PROCESSING. GPU IS DISABLED BY DEFAULT DURING THE TESTING PROCESS, EXCEPT WHEN $^+$ DENOTES GPU-BASED ACCELERATION USING A SINGLE NVIDIA 1080TI.

Class	$L_0(0,1)$				$L_0(0.2,1)$				$L_1(0.2,0.9)$			
	T_{ENC}	T_{NET}	T_{POST}	ρ (%)	T_{ENC}	T_{NET}	T_{POST}	ρ (%)	T_{ENC}	$T_{\text{NET}}/T_{\text{NET}}^{+}$	T_{POST}	ρ/ρ^{+} (%)
A	1264.51	2.81	0.28	0.24	1105.43	8.06	0.29	0.75	760.78	30.22/2.46	2.09	4.07/0.59
B	294.37	0.87	0.08	0.32	244.93	1.86	0.08	0.78	174.77	9.61/0.74	0.62	5.53/0.77
C	96.51	0.28	0.02	0.31	80.25	0.40	0.02	0.51	59.78	1.50/0.25	0.18	2.73/0.71
E	48.87	0.44	0.03	0.96	36.58	0.82	0.03	2.26	32.42	3.21/0.37	0.13	9.34/1.52
Average	426.07	1.10	0.10	0.46	366.80	2.79	0.10	1.08	256.94	11.14/0.96	0.76	5.42/0.90
D	25.72	0.28	0.01	1.11	23.01	0.37	0.01	1.62	20.37	0.67/0.19	0.03	3.32/1.07

TABLE VII
THE RATIO OF CTUS EARLY TERMINATED BY MTT MASK (%)

Class	QP				Average
	22	27	32	37	
A	12.29	23.70	36.18	49.33	30.38
B	9.35	20.43	48.47	52.32	32.64
C	1.80	8.25	11.21	16.35	9.40
E	39.24	68.46	76.60	81.57	66.47
Average	15.67	30.21	43.11	49.89	34.72
D	0.00	0.10	10.24	9.23	4.89

parameters, our method requires fewer parameters (1.11 M) under the low complexity setting, and a comparable number under the high complexity setting.

Time Consumption: We decompose the total time consumption into three components: encoder time, network inference time, and post-processing time, denoted as T_{ENC}, T_{net}, and T_{post}, respectively. These are evaluated at three acceleration levels: $L_0(0,1)$, $L_0(0.2,1)$, and $L_1(0.2,0.9)$, as shown in Table VI. The results are averaged over all frames of the JVET test sequences across various resolutions, without GPU acceleration. We define ρ to quantify method overhead, as follows:

$$\rho = \frac{T_{\text{net}} + T_{\text{post}}}{T_{\text{ENC}} + T_{\text{net}} + T_{\text{post}}} \times 100\%. \tag{14}$$

At acceleration level $L_0(0,1)$, the average time required for network inference and post-processing is approximately 1 s, accounting for 0.46% of the total evaluation time. For 4 K resolution videos, the inference and post-processing time remains around 3 seconds per frame. This performance can be attributed to the fact that only the QT Net is inferred, the input image is downsampled by a factor of four, and the post-processing algorithm only handles the QT depth map. At the acceleration level $L_0(0.2,1)$, only the QT Net and MTT Mask Net are used, and the additional inference time from post-processing the MTT mask is negligible. The method's overhead at this level is 1.08%, demonstrating that the proposed fast algorithm introduces minimal overhead compared to the VTM encoding process. At the highest acceleration level $L_1(0.2,0.9)$, the entire neural network is invoked, and the method's overhead increases to 5.42%. This increase is partly due to the reduction in VTM encoder time and partly due to the higher computational cost resulting from optical flow estimation and feature extraction from raw pixels. To further accelerate network inference, a single NVIDIA 1080Ti GPU is employed, denoted by the superscript $^+$. Compared to CPU inference, the overhead of the method is reduced from 5.42% to 0.90%. For 4 K sequences, this corresponds to 2.46 seconds per frame, highlighting the necessity of GPU-based parallel computing for significant encoding time reduction.

VII. CONCLUSION

This paper presents a partition map-based fast block partitioning approach for VVC inter coding. The proposed method is explored from three main perspectives: representation, neural network architecture, and post-processing algorithms. We introduce an MTT mask and design a novel neural network that predicts the partition map in a coarse-to-fine manner. A dual-threshold decision scheme is also proposed to balance network prediction and recursive RDO search. Moreover, we incorporate several novel modules to enhance the neural network's performance by simulating the partition search of inter coding. Experimental results show that our method outperforms previous approaches significantly in terms of complexity reduction and RD performance.

This study has two limitations. First, due to the high computational overhead of optical flow neural networks, applying the proposed method to resource-constrained devices is challenging. Second, since we adopt an out-of-loop implementation, the fast algorithm does not account for the encoding context, limiting the model's ability to adapt to various encoding environments.

Future work will focus on reducing computational complexity, integrating the algorithm into the encoding loop, and extending the approach to other inter modes such as LDP and LDB.

REFERENCES

[1] B. Bross et al., "Overview of the versatile video coding (VVC) standard and its applications," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 31, no. 10, pp. 3736–3764, Oct. 2021.
[2] G. J. Sullivan, J.-R. Ohm, W.-J. Han, and T. Wiegand, "Overview of the high efficiency video coding (HEVC) standard," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 22, no. 12, pp. 1649–1668, Dec. 2012.
[3] Y.-W. Huang et al., "Block partitioning structure in the VVC standard," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 31, no. 10, pp. 3818–3833, Oct. 2021.

[4] JVET, Joint exploration model (JEM) reference software 7.0. Accessed: Jan. 2025. [Online]. Available: https://jvet.hhi.fraunhofer.de/svn/svn_HMJEMSoftware/branches/HM-16.6-JEM-7.0-dev/

[5] F. Bossen, K. Sühring, A. Wieckowski, and S. Liu, "VVC complexity and software implementation analysis," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 31, no. 10, pp. 3765–3778, Oct. 2021.

[6] X. Dong, L. Shen, M. Yu, and H. Yang, "Fast intra mode decision algorithm for Versatile Video Coding," *IEEE Trans. Multimedia*, vol. 24, pp. 400–414, 2022.

[7] T. Amestoy, A. Mercat, W. Hamidouche, C. Bergeron, and D. Menard, "Random forest oriented fast QTBT frame partitioning," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process.*, 2019, pp. 1837–1841.

[8] H. Yang et al., "Low-complexity CTU partition structure decision and fast intra mode decision for Versatile Video Coding," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 30, no. 6, pp. 1668–1682, Jun. 2020.

[9] M. Lei, F. Luo, X. Zhang, S. Wang, and S. Ma, "Look-ahead prediction based coding unit size pruning for VVC intra coding," in *Proc. IEEE Int. Conf. Image Process.*, 2019, pp. 4120–4124.

[10] G. Wu, Y. Huang, C. Zhu, L. Song, and W. Zhang, "SVM based fast CU partitioning algorithm for VVC intra coding," in *Proc. IEEE Int. Symp. Circuits Syst.*, 2021, pp. 1–5.

[11] J. Cui, T. Zhang, C. Gu, X. Zhang, and S. Ma, "Gradient-based early termination of CU partition in VVC intra coding," in *Proc. Data Compression Conf.*, 2020, pp. 103–112.

[12] J. Chen, H. Sun, J. Katto, X. Zeng, and Y. Fan, "Fast QTMT partition decision algorithm in VVC intra coding based on variance and gradient," in *Proc. IEEE Vis. Commun. Image Process.*, 2019, pp. 1–4.

[13] M. Saldanha, G. Sanchez, C. Marcon, and L. Agostini, "Configurable fast block partitioning for VVC intra coding using light gradient boosting machine," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 32, no. 6, pp. 3947–3960, Jun. 2022.

[14] A. Wieckowski, J. Ma, H. Schwarz, D. Marpe, and T. Wiegand, "Fast partitioning decision strategies for the upcoming Versatile Video Coding (VVC) standard," in *Proc. IEEE Int. Conf. Image Process.*, 2019, pp. 4130–4134.

[15] N. Tang et al., "Fast CTU partition decision algorithm for VVC intra and inter coding," in *Proc. IEEE Asia Pacific Conf. Circuits Syst.*, 2019, pp. 361–364.

[16] Y. Huang, J. Xu, C. Zhu, L. Song, and W. Zhang, "Precise encoding complexity control for Versatile Video Coding," *IEEE Trans. Broadcast.*, vol. 69, no. 1, pp. 33–48, Mar. 2023.

[17] T. Amestoy, A. Mercat, W. Hamidouche, D. Menard, and C. Bergeron, "Tunable VVC frame partitioning based on lightweight machine learning," *IEEE Trans. Image Process.*, vol. 29, pp. 1313–1328, 2020.

[18] Z. Jin, P. An, L. Shen, and C. Yang, "CNN oriented fast QTBT partition algorithm for JVET intra coding," in *Proc. IEEE Vis. Commun. Image Process.*, 2017, pp. 1–4.

[19] F. Galpin et al., "CNN-based driving of block partitioning for intra slices encoding," in *Proc. Data Compression Conf.*, 2019, pp. 162–171.

[20] T. Li, M. Xu, R. Tang, Y. Chen, and Q. Xing, "DeepQTMT: A deep learning approach for fast QTMT-based CU partition of intra-mode VVC," *IEEE Trans. Image Process.*, vol. 30, pp. 5377–5390, 2021.

[21] S. Wu, J. Shi, and Z. Chen, "HG-FCN: Hierarchical grid fully convolutional network for fast VVC intra coding," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 32, no. 8, pp. 5638–5649, Aug. 2022.

[22] A. Tissier et al., "Machine learning based efficient QT-MTT partitioning scheme for VVC intra encoders," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 33, no. 8, pp. 4279–4293, Aug. 2023.

[23] A. Feng, K. Liu, D. Liu, L. Li, and F. Wu, "Partition map prediction for fast block partitioning in VVC intra-frame coding," *IEEE Trans. Image Process.*, vol. 32, pp. 2237–2251, 2023.

[24] Z. Pan, P. Zhang, B. Peng, N. Ling, and J. Lei, "A CNN-based fast inter coding method for VVC," *IEEE Signal Process. Lett.*, vol. 28, pp. 1260–1264, 2021.

[25] A. Tissier, W. Hamidouche, J. Vanne, and D. Menard, "Machine learning based efficient QT-MTT partitioning for VVC inter coding," in *Proc. 2022 IEEE Int. Conf. Image Process.*, 2022, pp. 1401–1405.

[26] Z. Peng and L. Shen, "A classification-prediction joint framework to accelerate QTMT-based CU partition of inter-mode VVC," *Electron. Lett.*, vol. 59, no. 7, 2023, Art. no. e12770.

[27] J. Lin, H. Lin, Z. Zhang, and Y. Xu, "Efficient inter partitioning of versatile video coding based on supervised contrastive learning," *Knowl.-Based Syst.*, vol. 296, 2024, Art. no. 111902.

[28] S.-h. Park and J.-W. Kang, "Fast multi-type tree partitioning for versatile video coding using a lightweight neural network," *IEEE Trans. Multimedia*, vol. 23, pp. 4388–4399, 2021.

[29] F. Bossen, J. Boyce, X. Li, V. Seregin, and K. Suehring, "VTM common test conditions and software reference configurations for SDR video," JVET, Tech. Rep. JVET-T2010, 2020.

[30] S. Cho and M. Kim, "Fast CU splitting and pruning for suboptimal CU partitioning in HEVC intra coding," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 23, no. 9, pp. 1555–1564, Sep. 2013.

[31] N. Kim et al., "Adaptive keypoint-based CU depth decision for HEVC intra coding," in *Proc. IEEE Int. Symp. Broadband Multimedia Syst. Broadcast.*, 2016, pp. 1–3.

[32] J.-C. Chiang, K.-K. Peng, C.-C. Wu, C.-Y. Deng, and W.-N. Lie, "Fast intra mode decision and fast CU size decision for depth video coding in 3D-HEVC," *Signal Process.: Image Commun.*, vol. 71, pp. 13–23, 2019. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0923596518309962

[33] H.-C. Fang, H.-C. Chen, and T.-S. Chang, "Fast intra prediction algorithm and design for high efficiency video coding," in *Proc. IEEE Int. Symp. Circuits Syst.*, 2016, pp. 1770–1773.

[34] Y. Zhang, N. Li, S. Kwong, G. Jiang, and H. Zeng, "Statistical early termination and early skip models for fast mode decision in HEVC intra coding," *ACM Trans. Multimedia Comput. Commun. Appl.*, vol. 15, no. 3, Jul. 2019, Art. no. 70, doi: 10.1145/3321510.

[35] Z. Liu et al., "CU partition mode decision for HEVC hardwired intra encoder using convolution neural network," *IEEE Trans. Image Process.*, vol. 25, no. 11, pp. 5088–5103, Nov. 2016.

[36] A. Tissier, W. Hamidouche, J. Vanne, F. Galpin, and D. Menard, "CNN oriented complexity reduction of VVC intra encoder," in *Proc. IEEE Int. Conf. Image Process.*, 2020, pp. 3139–3143.

[37] M. Xu et al., "Reducing complexity of HEVC: A deep learning approach," *IEEE Trans. Image Process.*, vol. 27, no. 10, pp. 5044–5059, Oct. 2018.

[38] A. Feng, C. Gao, L. Li, D. Liu, and F. Wu, "CNN-based depth map prediction for fast block partitioning in HEVC intra coding," in *Proc. 2021 IEEE Int. Conf. Multimedia Expo*, 2021, pp. 1–6.

[39] G. Correa, P. A. Assuncao, L. V. Agostini, L. A. da, and S. Cruz, "Fast HEVC encoding decisions using data mining," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 25, no. 4, pp. 660–673, Apr. 2015.

[40] Y. Zhang et al., "Machine learning-based coding unit depth decisions for flexible complexity allocation in High Efficiency Video Coding," *IEEE Trans. Image Process.*, vol. 24, no. 7, pp. 2225–2238, Jul. 2015.

[41] L. Zhu et al., "Binary and multi-class learning based low complexity optimization for HEVC encoding," *IEEE Trans. Broadcast.*, vol. 63, no. 3, pp. 547–561, Sep. 2017.

[42] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "Mobilenetv2: Inverted residuals and linear bottlenecks," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 4510–4520.

[43] J. Shi, C. Gao, and Z. Chen, "Asymmetric-kernel CNN based fast CTU partition for HEVC intra coding," in *Proc. IEEE Int. Symp. Circuits Syst.*, 2019, pp. 1–5.

[44] A. Ranjan and M. J. Black, "Optical flow estimation using a spatial pyramid network," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jul. 2017, pp. 4161–4170.

[45] C. Tang et al., "Offline and online optical flow enhancement for deep video compression," in *Proc. AAAI Conf. Artif. Intell.*, vol. 38, no. 6, 2024, pp. 5118–5126.

[46] A. Newell, K. Yang, and J. Deng, "Stacked hourglass networks for human pose estimation," in *Proc. 14th Eur. Conf. Comput. Vis.*, Amsterdam, The Netherlands, 2016, pp. 483–499.

[47] K. Duan et al., "CenterNet: Keypoint triplets for object detection," in *Proc. 2019 IEEE/CVF Int. Conf. Comput. Vis.*, 2019, pp. 6568–6577.

[48] T. Li et al., "A deep learning approach for multi-frame in-loop filter of HEVC," *IEEE Trans. Image Process.*, vol. 28, no. 11, pp. 5663–5678, Nov. 2019.

[49] Z. Li et al., "Object segmentation-assisted inter prediction for Versatile Video Coding," *IEEE Trans. Broadcast.*, vol. 70, no. 4, pp. 1236–1253, Dec. 2024.

[50] D. Ma, F. Zhang, and D. R. Bull, "BVI-DVC: A training database for deep video compression," *IEEE Trans. Multimedia*, vol. 24, pp. 3847–3858, 2022.

[51] X. Xu, S. Liu, and Z. Li, "A video dataset for learning-based visual data compression and analysis," in *Proc. Int. Conf. Vis. Commun. Image Process.*, 2021, pp. 1–4.

[52] A. Mercat, M. Viitanen, and J. Vanne, "UVG dataset: 50/120fps 4 K sequences for video codec analysis and development," in *Proc. 11th ACM Multimedia Syst. Conf.*, New York, NY, USA, 2020, pp. 297–302, doi: 10.1145/3339825.3394937.
[53] F. Bossen, "Common HM test conditions and software reference configurations," JCT-VC, Tech. Rep. JCTVC-L1100, 2013.
[54] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *Proc. 13th Int. Conf. Artif. Intell. Statist.*, 2010, pp. 249–256.
[55] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2014, *arXiv:1412.6980*.
[56] G. Bjontegaard, "Calcuation of average PSNR differences between RD-curves," VCEG, Tech. Rep. VCEG-M33, 2001.
[57] F. Iandola et al., "DenseNet: Implementing efficient convnet descriptor pyramids," 2014, *arXiv:1404.1869*.

Xinmin Feng received the B.S. degree in electrical engineering from Xidian University, Xi'an, China, in 2022. He is currently working toward the Ph.D. degree with the Department of Electronic Engineering and Information Science at the University of Science and Technology of China, Hefei, China. His research interests include low-latency and low-bandwidth video communications.

Zhuoyuan Li (Graduate Student Member, IEEE) received the B.S. degree in communication engineering from Southwest Jiaotong University, Chengdu, China, in 2020. He is currently working toward the Ph.D. degree with the Department of Electronic Engineering and Information Science, University of Science and Technology of China, Hefei, China. His research interests include video coding and processing.

Li Li (Senior Member, IEEE) received the B.S. and Ph.D. degrees in electronic engineering from University of Science and Technology of China (USTC), Hefei, Anhui, China, in 2011 and 2016, respectively. From 2016 to 2020, he was a Visiting Assistant Professor with the University of Missouri-Kansas City, Kansas City, MO, USA. In 2020, he joined the Department of Electronic Engineering and Information Science, USTC,as a Research Fellow and became a Professor in 2022. He has authored or coauthored more than 80 papers in international journals and conferences, and also has more than 20 granted patents. His research interests include image/video/point cloud coding and processing. He was the recipient of the Multimedia Rising Star 2023, Best 10% Paper Award at the 2016 IEEE Visual Communications and Image Processing (VCIP), and 2019 IEEE International Conference on Image Processing (ICIP). He also has several technique proposals adopted by Standardization Groups. From 2024 to 2025, he was an Associate Editor for IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS FOR VIDEO TECHNOLOGY.

Dong Liu (Senior Member, IEEE) received the B.S. and Ph.D. degrees in electrical engineering from the University of Science and Technology of China (USTC), Hefei, China, in 2004 and 2009, respectively. From 2009 to 2012, he was a Member of Research Staff with Nokia Research Center, Beijing, China. Since 2012, he has been a Faculty Member with USTC and currently holds the position of Full Professor. He has authored or coauthored more than 200 papers in international journals and conferences, which were cited more than 20,000 times according to Google Scholar, and also has more than 30 granted patents. His research interests include image and video processing, coding, and analysis. Dr. Liu was the recipient of the 2009 IEEE TCSVT Best Paper Award, VCIP 2016 Best 10% Paper Award, and ISCAS 2025 Grand Challenge Top Creativity Paper Award. He and his students were the recipient of several technical challenges held in ICIP 2024, ISCAS 2023, and ICCV 2019. He also has several technique proposals adopted by Standardization Groups. He is or was the Chair of IEEE 1857.11 Standard Working Subgroup (also known as Future Video Coding Study Group), an Associate Editor for IEEE TRANSACTIONS ON IMAGE PROCESSING, a Guest Editor for IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS FOR VIDEO TECHNOLOGY, an Organizing Committee Member for ICME 2026, ChinaMM 2024, VCIP 2022, and ICME 2021. He is also a Senior Member of CCF and CSIG and an Elected Member of IVMSP-TC of IEEE SP Society, MSA-TC of IEEE CAS Society, and Multimedia TC of CSIG.

Feng Wu (Fellow, IEEE) received the B.S. degree in electrical engineering from Xidian University, Xi'an, China, in 1992, and the M.S. and Ph.D. degrees in computer science from the Harbin Institute of Technology, Harbin, China, in 1996 and 1999, respectively. He was a Principle Researcher and Research Manager with Microsoft Research Asia, Beijing, China. He is currently a Professor and Vice President with the University of Science and Technology of China, Hefei, China. He has authored or coauthored more than 150 journal papers (including several dozens of IEEE Transactions papers) and top conference papers on MOBICOM, SIGIR, CVPR and ACM MM, and also more than 100 granted U.S. patents, and his 15 techniques have been adopted into international video coding standards. His research interests include various aspects of video technology and artificial intelligence. He is or was the Editor-in-Chief of IEEE TRANSACTIONS ON CIRCUITS AND SYSTEM FOR VIDEO TECHNOLOGY and as an Associate Editor for IEEE TRANSACTIONS ON IMAGE PROCESSING and IEEE TRANSACTIONS ON MULTIMEDIA. He is also the General Chair in ICME 2019, TPC Chair in MMSP 2011, VCIP 2010 and PCM 2009. He was the recipient of the Mac Van Valkerburg Award from the IEEE CAS Society in 2021, Best Paper awards in IEEE TCSVT 2009, VCIP 2016, PCM 2008, and VCIP 2007, and Best Associate Editor Award of IEEE Transactions on Image Processing in 2018.