EURASIP Journal on Image and Video Processing

RESEARCH Open Access

Jointly probability-based fast intra prediction algorithm for spatial SHVC

Weian Li[1], Weihua Ou[1], Hongbing Wang[1*], Yang Wu[1] and Yunhao Zhong[1]

*Correspondence:
hbwang@gznu.edu.cn

[1] School of Big Data and Computer Science, Guizhou Normal University, Guiyang 550025, Guizhou, China

Abstract

Since Spatial Scalable High Efficiency Video Coding (SSHVC) utilizes multi-layer encoding and inter-layer prediction, it has extremely high coding complexity which has severely hindered its wide spread. In order to improve coding speed, we develop a fast intra coding algorithm to improve coding speed of SSHVC. First, we utilize the textural feature to obtain the probabilities of using Coding Units (CUs), so as to predict candidate CUs and exclude unlikely ones. Second, we jointly utilize the textural features, the neighboring Coding Modes (CMs) and Rate Distortion (RD) cost to obtain the probability of Intra Mode (IM), which is then combined with the probability of its collocated CU to exclude unlikely IMs. Finally, we investigate distribution of Direction Modes (DMs), which are then combined with the probability of their collocated CU to predict candidate DMs and exclude unlikely ones. Experimental results demonstrate that the proposed algorithm can significantly improve coding speed by 78.64% with negligible coding efficiency change by a 0.27% decrease in BDBR.

Keywords: SHVC, Coding units, Coding mode, Direction mode

1 Introduction

With the rapid development of science and technology, video applications have been deeply embedded in our daily life. Digital TV broadcasting, video conferencing, wireless video streaming, smart phone communications and so on, have gained widespread application. Correspondingly, the number of terminal devices with various spatial resolutions is continuously increasing. This poses the requirements for video streams to be adaptive to different spatial resolutions. Conventional video coding schemes cannot effectively meet these needs. As the scalable extension of HEVC, Spatial Scalable High Efficiency Video Coding (SSHVC) provides a practical solution for accommodating various device requirements. By employing a hierarchical encoding structure, SSHVC assigns each layer a specific spatial resolution. This design allows the system to efficiently adapt to devices with different resolution needs by selecting the corresponding layer.

The SSHVC architecture includes a Base Layer (BL) and one or more Enhancement Layers (ELs). The BL relies on the standard prediction methods of HEVC, while the ELs introduce inter-layer prediction to further optimize encoding efficiency. Since the coding process of HEVC is already very complex [1], the process of SSHVC is certainly

much more complex. This increased complexity limits its deployment in widespread applications, presenting a notable challenge for further adoption. Therefore, developing its fast-coding algorithms is very important. For this purpose, we propose a fast-coding algorithm for SSHVC in this paper. The major novelties of the proposed algorithm include:

1. Feeding the textural feature into a decision tree to obtain the probabilities of using Coding Units (CUs), so as to remove unlikely CUs.
2. We combine the probability of Intra Mode (IM) with probability of its collocated CU to early skip unlikely IM. More specifically, we first use a decision tree to obtain IM' s probability. Since IM is within its collocated CU, they certainly have strong relationship. Therefore, we can combine the probability of IM with probability of its collocated CU to early skip unlikely IM.
3. We combine Direction Modes (DMs) distribution with probability of their collocated CU to predict candidate DMs and exclude unlikely ones. More specifically, we first investigate DMs distribution. Since DMs are within their collocated CU, they certainly have strong relationship. Therefore, we combine DMs distribution with probability of their collocated CU to predict candidate DMs and exclude unlikely ones.

The remainder of this paper is organized as follows. Section 2 provides related work. Section 3 presents an overview of our proposed algorithm. Sections 4, 5 and 6 describe the proposed CU selection, Coding Mode (CM) selection and DM selection in details. Section 7 discusses and analyses the experimental results. Finally, Section 8 concludes this research and plans for future work.

2 Related work

In order to improve the coding speed, many algorithms have been proposed to improve coding speed of SSHVC, which are reviewed and discussed below:

2.1 CU prediction

Generally speaking, depths are closely related to textural complexity. CUs with simple textures usually use smaller coding depths, such as 64×64 and 32×32. On the contrary, CUs containing complex textures usually use larger coding depths, such as depth 16×16 and 8×8. Therefore, textural features are usually used to predict candidate CUs and skip unlikely CUs. Liu et al.[2] used convolutional neural network (CNN) to exclude unlikely CUs. Wang et al.[3] integrate depth probabilities with textural complexity to early skip unlikely CUs and early terminate CU selection. Xu et al.[4] use deep learning to decide whether to early terminate CU selection. These algorithms are developed based on texture features. Since a CU and its neighboring CUs are usually very similar, neighboring CUs can be utilized to predict candidate CUs and skip unlikely CUs. In [5], the temporal and spatial correlations and their correlation degrees were combined to predict candidate CUs and skip the unlikely CUs. In [6], both inter-layer and spatial correlations as well as residual coefficients are jointly used to predict candidate CUs of a CU. In [7], the inter-layer correlation and spatial correlation were combined to predict likely CUs and remove unlikely CUs. The above algorithms are developed based on correlation.

Residual coefficients are closely related to CU selections. Thus they can be used to early terminate CU selection process. In [8], a hypothesis test is conducted to assess whether the residuals exhibit significant differences so as to early terminate CU selection process. Common significance testing methods, such as t-test [5] and F-test [9], are applied to assess whether the expected values or variances of the residual coefficients are sufficiently similar, facilitating the early termination of CU selection, so as to accelerate coding speed.

2.2 CM prediction

Since the residual coefficients and Rate Distortion (RD) cost are strongly related to CM prediction, they can be used to predict the likely CMs. The research in [10] sorts each CM from small to large by the RD cost expectation, and then develops a threshold to early exclude unlikely CMs. Tohidypour et al. [11] uses the already coded neighboring CUs of the current CU to obtain its predicted RD cost, and if the actual RD cost of a CM is less than or equal to the predicted RD cost, CM selection can be early terminated. The work in [12] first predicts candidate CMs by using inter-layer and spatial correlations, subsequently early terminate CM selection process based on RD cost and residual coefficients. Zhao et al. [13] uses inter-layer and spatial correlations to sort the candidate CM, and then uses a constrained model with optimal stopping to terminate the coding process early. Wang et al. [14] first check Inter-layer Reference (ILR) CM, and then calculate its part-zero block to derive a threshold, so as to early terminate CM selection based on the threshold. Tohidypour et al. [9] use RD cost to determine whether the current CM is optimal so as to skip unlikely CM.

Generally speaking, inter-layer and spatial correlations are usually used to improve coding speed. Based on neighboring blocks' CMs in EL and its co-located block' s CM in BL, Tohidypour et al. [15] uses Bayesian method to predict candidate CMs and exclude unlikely CUs. Lu et al. [16] uses inter-layer and spatial correlations, and textural features to predict candidate CMs and early terminate the coding process. Cho et al. [17] uses both inter-layer and spatial correlations to only predict SKIP CMs in EL, so as to exclude other CMs. Textures are also related to CM selection. Wang et al. [5] integrates textural complexity with depth probabilities to skip unlikely CMs and early terminated CM selection.

2.3 DM prediction

DM prediction: etLiu et al. [18] use textural complexities and directional complexities to predict DMs and exclude unlikely DMs in rough mode decision procedure. Jiang et al. [19] use a Sobel operator to predict candidate DMs and skip unlikely ones, so as to reduce coding time. Zhang et al. [20] calculate the average gradients in the horizontal (AGH) and vertical (AGV) directions, and subsequently compute the ratio of AGH to AGV to forecast the potential DMs. Feeding the improved pixel data of CUs, adjacent blocks, the Sum of Absolute Hadamard Transformed Difference into a Decision Tree, M. Jamali et al. [21] predicts the DM candidate and skip some unlikely DMs to speed up the coding process.

Hadamard Costs (HCs) are closely associated with the selection of DMs and are commonly employed to predict potential DM candidates. Zhang et al. [22] developed

a HC-based progressive rough DM search that can check likely DMs and skip unlikely ones to increase the coding speed. The research in [5] combines textural characteristics with DMs and their associated HCs, thereby enabling the prediction of potential DMs and efficiently skipping unlikely ones. Wang et al. [8] combines relevant CUs with the relationships between the IMs and their corresponding HC values to predict candidate DMs and skip unlikely ones. Wang et al. [3] utilize the difference in HCs of typical DMs to predict potential DMs. Subsequently, percentages of gradient amplitudes and HCs are combined to predict likely DMs and early terminate DM selection process. In order to improve coding speed, M. Jamali et al. [21] developed a HC-based statistical model for RD cost to predict candidate DMs and remove unlikely DMs.

Although the above algorithms can improve the coding speed, they only predict CU's depth, CMs, and DMs independently. Due to both CMs and DMs are collocated in a CU, they certainly have closely relationship with a probability of their collocated CU. Therefore, independent prediction without considering their relationship may not achieve the optimal performance. In order to address this issue, we first feed texture features into a decision tree to obtain a CU's probability. Second, we use texture features and RD cost as eigenvalues, and use a decision tree to obtain the probability of IM, which is combined with probability of its collocated CU to skip unlikely IM. Finally, we combine DM distributions and probability of their collocated CU to predict likely DMs and exclude unlikely DMs. Since the probabilities of IM and DM distributions are combined with a probability of their collocated CU, more CUs can early exclude unlikely IM and DMs, and coding speed can be significantly improved accordingly.

3 Overview of the proposed algorithm framework

This paper proposes the following three strategies to improve coding speed and maintain the encoding efficiency of SSHVC: Textural Features-Based Coding Units Prediction (TFBCUP), Probability-Based IM Early Skip (PBIMES), Probability and Distribution-Based DM Selection (PDBDS). The overall structure of the proposed algorithm is shown in Fig. 1. First, we use TFBCUP to predict candidate CUs and exclude unlikely ones. Second, we use PBIMES to determine whether IM can be skipped. If PBIMES confirms the IM skippability condition, IM prediction can be early skipped. Otherwise, we need to use PDBDS to predict candidate DMs and exclude unlikely ones. In Fig. 1, the left side shows the three strategies, and the right side shows the process of the proposed algorithm.

To thoroughly investigate the characteristics of Intra coding within SSHVC, a series of comprehensive experiments were performed. Eight representative video sequences, each exhibiting varying levels of motion and texture complexity, were selected for training. These include Blue_sky, Ducks, Park_Joy, Pedestrian, Tractor, Town, Station2, and Average. Following the Common SHM Test Conditions (CSTC) [23], two spatial scalability ratios were considered: 1.5x and 2x, which correspond to the ratios of frame height and width in the EL relative to those in the BL, being 1.5 and 2 times, respectively.

For the BL, a single Quantization Parameter (QP) set of (22, 26, 30, 34) was applied across all configurations. In contrast, the EL was tested with two distinct QP sets: (22, 26, 30, 34) and (24, 28, 32, 36). Combining these QP settings with the two scalability ratios resulted in four experimental cases. Specifically, Case 1 applies a 1.5x ratio with

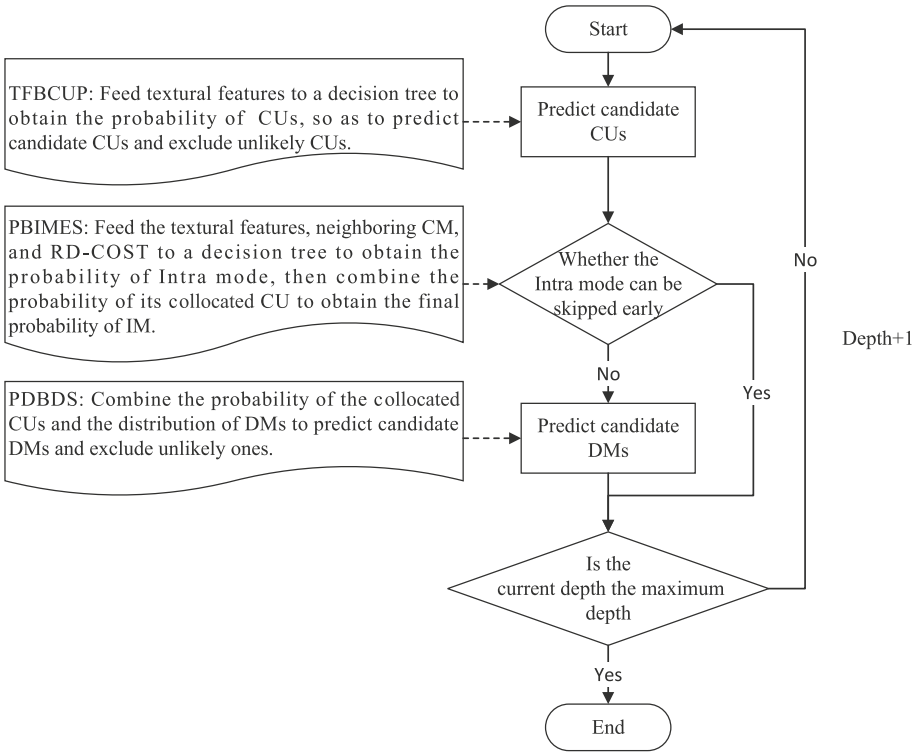

Fig. 1 Overview of the Proposed Algorithm

EL QPs of (22, 26, 30, 34); Case 2 also uses a 1.5x ratio but with EL QPs of (24, 28, 32, 36); Case 3 involves a 2x ratio with EL QPs of (22, 26, 30, 34) and Case 4 uses a 2x ratio combined with EL QPs of (24, 28, 32, 36). Among these, Case 4 presents the highest disparity in both scaling factor and QP values between EL and BL, implying a minimal inter-layer dependency.

It should also be emphasized that, in this paper, eight representative sequences were selected for training. These sequences were deliberately chosen to cover a broad spectrum of characteristics, including high versus low texture complexity, strong versus weak inter-layer dependency, and varying levels of motion activity. Therefore, although increasing the number of sequences would likely further stabilize the statistical values, the relative trends and conclusions drawn from our methods are expected to remain consistent. Moreover, the hyperparameters employed in our strategies (such as probability thresholds for CU, IM, and DM decisions) are derived from intrinsic coding properties rather than sequence-specific tuning. Consequently, expanding the dataset would mainly refine the numerical performance values but would not substantially affect either the hyperparameters or the overall conclusions of the proposed framework.

4 Textural features-based coding unit prediction

According to HEVC standard, each 64×64 coding tree unit (CTU) is recursively split into four equally sized sub-CUs until reach 8×8 CUs. Through the above process, the optimal division can be obtained, but its computational complexity is also significantly

increased. To enhance the coding speed, we propose a fast CU selection strategy which aims to skip unlikely CUs, thereby significantly accelerating the coding speed.

There are different machine learning classifiers, such as Naive Bayes Classifier (NBC), Decision Tree (DT) and Support Vector Machines (SVM) [24]. NBC is a probabilistic machine learning model based on Bayes' theorem, which assumes conditional independence between features. NBC requires the statistical estimation of the probability of each feature first, which can be quite burdensome. SVM is primarily used for classification and regression analysis. In classification problems, SVM aims to find a hyperplane to maximize the spacing between different categories, which usually leads to complex training and prediction processes [25]. DT is a widely used binary classification approach known for its comprehensible flow-chart-like structure [26]. Each non-leaf node performs a test on a specific attribute, with branches indicating the test outcomes and leaf nodes holding class labels. Due to its simple prediction process and high prediction accuracy, we have chosen DT as the classifier in this paper.

Since feature selection critically influences DT performance, we conduct thorough analysis to identify optimal eigenvalues for CU partition prediction. Three key features are selected: 1. Textural complexity: Generally speaking, the more complex the textures of CUs, the greater their textural variances are. And the corresponding CUs are more likely to be further split, and vice versa. Obviously, variances of CUs can well reflect their textural complexity and have strong relationship with CU partition. 2. Directional complexities: Directional complexities are also usually selected to measure textural complexity [27]. Therefore, we apply the Sobel Operator to calculate directional complexities. 3. QP: QP directly affects CU partitioning decisions: at high QP values, the encoder tends to select larger CUs; at low QP values, the encoder tends to select smaller CUs.

Although we employed a DT classifier in this paper due to its efficiency and interpretability, the selected features (including textural complexity, directional complexities and QP) are intrinsic descriptors of CU partitioning and are not dependent on the specific classifier used. These features could, in principle, also be applied in other classification models (e.g., NBC, SVM, or neural networks), with the main difference being in the form of the decision boundaries. Thus, while the classifier type may change, the fundamental validity of the chosen features remains unaffected.

4.1 Selection of eigenvalues

When choosing a decision tree as a classifier, the selection of eigenvalues is very important, which is selected as follows:

4.1.1 Textural complexity

Variances of CUs are selected as eigenvalues to predict CU partition as shown in Fig. 2:

1. The more complex the texture of a CU, the greater the overall pixel variance is, the CU is more likely to be split, and vice versa (Fig. 2a). Therefore, the overall pixel variance is closely related to CU partition, which is selected as an eigenvalue and can be calculated as follows:

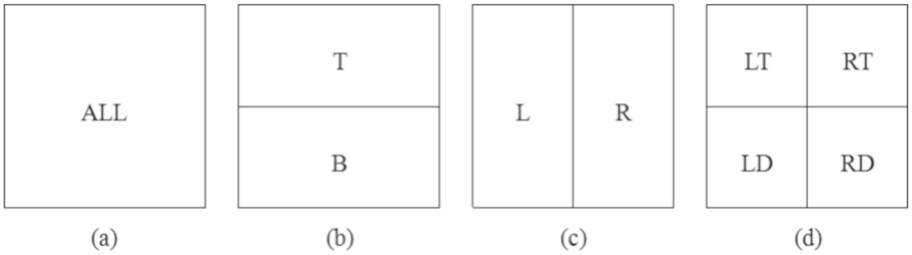

Fig. 2 The division of a CU

$$\sigma_{all}^2 = \frac{1}{w \times h} \sum_{i=0,j=0}^{w,h} \left(p_{ij} - \overline{p}\right)^2 \tag{1}$$

p_{ij} represents luminance value of a pixel located at (i, j) in a CU; $\overline{p}$ represents the average pixel value of the CU; w and h are width and height of the CU, respectively.

2. If a CU is equally divided into top and bottom parts (Fig. 2b), where T and B respectively refer to top and bottom parts of the CU. The more complex the texture of the two parts, the greater their average of the pixel variances, the current CU is likely to be split, and vice versa. Therefore, the average of the pixel variances of the two parts is closely related to CU partition, which is selected as an eigenvalue and can be calculated as follows:

$$\sigma_{\text{top-bottom}}^2 = \frac{1}{2}\left(\sigma_{\text{top}}^2 + \sigma_{\text{bottom}}^2\right) \tag{2}$$

σ_{top}^2 represents the pixel variance of the top part of the CU, σ_{bottom}^2 represents the pixel variance of the bottom part of the CU, and $\sigma_{top-bottom}^2$ is their average value of variances. Similarly, the average of pixel variance of the left and right parts is also closely related to CU partition. We can equally divide a CU into left and right parts (Fig. 2c), namely parts L and R, and select their average value of pixel variances as an eigenvalue, which is denoted as $\sigma_{left-right}^2$.

3. If we equally divided a CU into four parts (Fig. 2d), namely LT, LD, RT and RD. The more complex the texture of the four parts, the greater the average of the variances of the four parts, the corresponding CU will be more likely split, and vice versa. Therefore, the average of the variances of the four parts is also closely related to CU partition[27], which is selected as an eigenvalue and can be calculated as follows:

$$\begin{aligned}\sigma_{\text{four}}^2 = \frac{1}{4}\Big(&\sigma_{\text{left-top}}^2 + \sigma_{\text{right-top}}^2 \\ &+ \sigma_{\text{left-bottom}}^2 + \sigma_{\text{right-bottom}}^2\Big)\end{aligned} \tag{3}$$

where $\sigma_{left-top}^2$, $\sigma_{right-top}^2$,$\sigma_{left-bottom}^2$ and $\sigma_{right-bottom}^2$ represent the variance of the left-top part, the right-top part, the left-bottom part and the right-bottom part of a CU; σ_{four}^2 is their average value of pixel variances of the two parts.

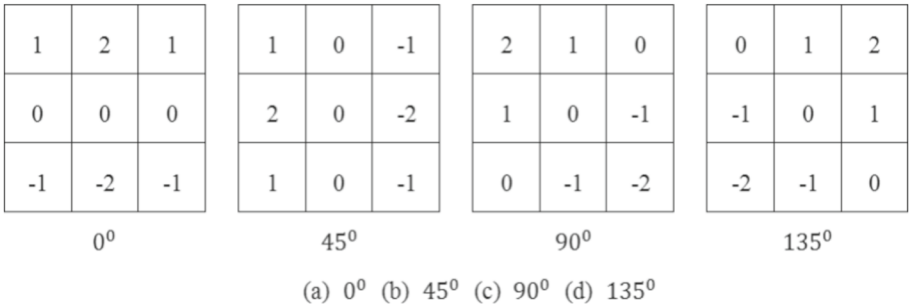

Fig. 3 Four angular Sobel operators for each pixel

Table 1 Average directional complexities under different angle degrees

Class	Sequence	Split Mode	$\overline{G_0^\circ}$	$\overline{G_{45}^\circ}$	$\overline{G_{90}^\circ}$	$\overline{G_{135}^\circ}$
B	Traffic	split	32	35	20	16
		No-split	12	17	22	9
C	Park	split	40	42	27	33
		No-split	22	15	12	26
D	Flowervase	split	31	33	35	27
		No-split	13	21	19	8
E	RaceHorses	split	21	27	15	24
		No-split	15	17	10	11

4.1.2 Directional complexity

More specially, we adopt four typical diagonal directional (0° 45° 90° 135°) Sobel Operator components to calculate their directional complexities of a CU, as shown in Fig. 3.

Their corresponding directional complexity ($DC_{k(i,j)}$) can be calculated as follows:

$$DC_{k(i,j)} = S_k * A, (k = 0^\circ, 45^\circ, 90^\circ, 135^\circ) \tag{4}$$

Where i, j are horizontal and vertical coordinates of a pixel in a CU, k is angle degree, S_k represents Sobel Operator for angle degree k as shown in Fig. 3, A is a 3×3-pixel block. For a pixel located at (i, j), it's A block is:

$$A = \begin{pmatrix} p(i-1,j-1) & p(i-1,j) & p(i-1,j+1) \\ p(i,j-1) & p(i,j) & p(i,j+1) \\ p(i+1,j-1) & p(i+1,j) & p(i+1,j+1) \end{pmatrix} \tag{5}$$

In order to verify the relationship between directional complexities and CU partition, the average directional complexities for 32×32 CUs are listed in Table 1.

Split indicates a CU is further divided, and no-split indicates a CU is no longer divided. From Table 1, it can be observed that the average directional complexity of split CUs is significantly greater than that of the no-split CUs. Experiment results show that other CUs also obey this rule. Therefore, the directional complexity is selected as the eigenvalue.

4.1.3 QP

Generally speaking, QP plays an important role in CU partition. When QP rises, the quantity of the large CUs increases, while that of the small CUs reduces, and vice versa [27]. Therefore, we employ QP as an important eigenvalue to determine CU partition.

4.2 The process of probability based CU selection

Feeding above eigenvalues into the decision tree, we can obtain the probability of using a CU. Obviously, if the probability is greater than or equal to 0.9, the CU is very likely to be the optimal CU. Therefore, we can early terminate CU selection. Conversely, if the probability is less than or equal to 0.1, the CU is unlikely to be the optimal one. Thus, we can early skip this CU checking.

5 Probability-based IM early skip

In coding process of the EL, CUs will check both ILR mode (ILRM) and IM, and select the CM with smaller RD cost as the optimal one. In this way, the optimal CM can be obtained, but this process is quite complex. If we can early skip unlikely CMs, the coding speed can be improved. In order to improve the coding speed, we first investigate coding complexity and distribution of CMs, and then use a decision tree to predict likely CMs and early exclude unlikely ones.

5.1 The coding complexity and distribution of CMs

Generally speaking, if coding complexity of a CM is higher, skipping the CM can save much time, and vice versa. Obviously, coding complexity of a CM severely influences the coding speed improvement. Therefore, we need to investigate coding complexity of CMs. Using the same experimental conditions and sequences as mentioned above in testing, coding complexity of CMs is listed below in Table 2.

In Table 2, IM and ILRM refer to the proportion of their corresponding coding time. From Table 2, the average coding time of ILRM and IM are 19.18% and 80.82%, respectively. Obviously, the average coding time of the IM is much more than that of the ILRM. Therefore, coding complexity of ILRM is much lower than that of IM.

Generally speaking, if proportion of a CM is higher, the probability of using this CM as optimal CM is greater, and vice versa. Therefore, proportions of CMs are closely related to CM selection. We refer to proportions of CMs as CM distribution. We use the same

Table 2 Coding complexity of CMs

Category	Sequence	ILRM	IM
B	Sunflower	19.36%	80.74%
	Tractor	28.91%	81.09%
C	Flower vase	19.27%	80.73%
	Party Scene	19.10%	80.90%
D	Race Horses	18.66%	81.34%
	Bubbles	19.33%	80.67%
E	Park	18.65%	81.35%
	Town	28.24%	79.76%
Average		19.18%	80.82%

experimental conditions and sequences as mention above to obtain CM distribution as Table 3.

In Table 3, IM and ILRM represent their corresponding proportions. From Table 3, the average proportion of ILRM and IM are 73.75% and 26.25%, respectively. Obviously, the average proportion of ILRM is significantly higher than that of IM.

Based on the above analysis, we can draw conclusions: (1) The coding complexity of ILRM is much lower than that of IM. (2) The proportion of ILRM is higher than that of IM.

Based on above analysis, since the proportion of ILRM is very large, making it more likely to be selected as the optimal CM. In addition, the coding process of ILRM is very simple, even if it is not the optimal CM, checking ILRM will not waste much time. Therefore, we can first directly check ILRM, and then use a decision tree to obtain the probability of ILRM, denoted as P_{ILR}, to determine whether ILRM is the best one, so as to early skip unlikely IM. In order to accurately obtain the probability, the selection of eigenvalues is the key, which will be discussed below.

5.2 Selection of eigenvalues

Generally speaking, CM selection is closely related to texture, correlations, and RD cost. Therefore, we use them as eigenvalues as follows:

5.2.1 Textural complexity

Generally speaking, textural complexity of a CU may have relationship with selection of CM. The textural variance can well reflect the texture complexity of a CU. In order to investigate the relationship between textural variance and CM selection, we use same video sequences and conditions as above in testing. Taking 32×32 CUs for example, their textural variances of different CMs are listed in Table 4.

In Table 4, IM and ILRM indicate their corresponding textural variance. From Table 4, the average textural variance of ILRM and IM are 62.57 and 743.1, respectively. Obviously, textural variance of ILRM is much smaller than that of IM. In addition, experiment shows textural variances of CMs in different CUs also obey the same rule. Obviously, textural complexity has closely relationship with CM selection. Therefore, we can select textural variance as an eigenvalue.

Table 3 CM distribution

Category	Sequence	ILRM	IM
B	Sunflower	81.81%	18.19%
	Tractor	71.54%	28.46%
C	Flower vase	58.79%	41.21%
	Party Scene	77.99%	22.00%
D	Race Horses	73.75%	26.25%
	Bubbles	74.43%	25.57%
E	Park	88.63%	11.37%
	Town	63.05%	36.95%
Average		73.75%	26.25%

Table 4 Textural variances of CMs

Video	32x32	
Sequence	ILRM	IM
Sunflower	221.37	764.13
Tractor	49.65	1196.34
Flower vase	103.17	847.07
Party Scene	36.31	428.13
Race Horses	5.43	1229.72
Bubbles	43.95	658.13
Park	11.32	548.12
Town	29.35	273.19
Average	62.57	743.1

5.2.2 Other eigenvalues

Beyond textural complexity, we select three eigenvalues for CM selection: directional complexity, the neighboring CM, and RD Cost. These features jointly characterize texture properties, spatial dependencies, and compression efficiency.

Directional complexity is also an important texture feature [27]. Therefore, we also select it as an eigenvalue, which can be obtained by using the similar way as 5.2.1.

The neighboring CM information leverages the high spatial correlation between neighboring CUs. This exploits spatial consistency to infer optimal CM selection. Therefore, we can use CMs of neighboring CUs as an eigenvalue.

RD cost can effectively reflect performance of CM prediction. If RD cost of a CM is very small, it indicates that the corresponding CU is predicted very well by the CM, and vice versa. This deterministic relationship establishes RD cost as an essential eigenvalue for evaluating CM suitability.

5.3 IM early skip

After select the above eigenvalues, we feed them into a decision tree to obtain P_{ILR}. Since there are only two CMs, the probability of IM denoted as P_{Intra} can be derived by:

$$P_{Intra} = 1 - P_{ILR} \tag{6}$$

Since IM is within its collocated CU, IM selection is closely with the CU's probability. Thus, we can combine P_{Intra} and probability of its collocated CU denoted as P_{CU} to obtain the final IM probability denoted as P_{IM}, which can be obtained by:

$$P_{IM} = P_{Intra} \times P_{CU} \tag{7}$$

Obviously, if probability of IM is less than or equal to 0.1, it indicates that IM is unlikely to be selected as optimal one and can be directly skipped. Therefore, we select 0.1 as the threshold of the IM early skip.

6 Probability and distribution-based adaptive DM selection

In each CU, there are 35 DMs which are shown in the Fig. 4. All DMs will be checked to obtain the optimal DM, which lead to a complex coding process. If we can early skip unlikely DMs, coding speed will be improved.

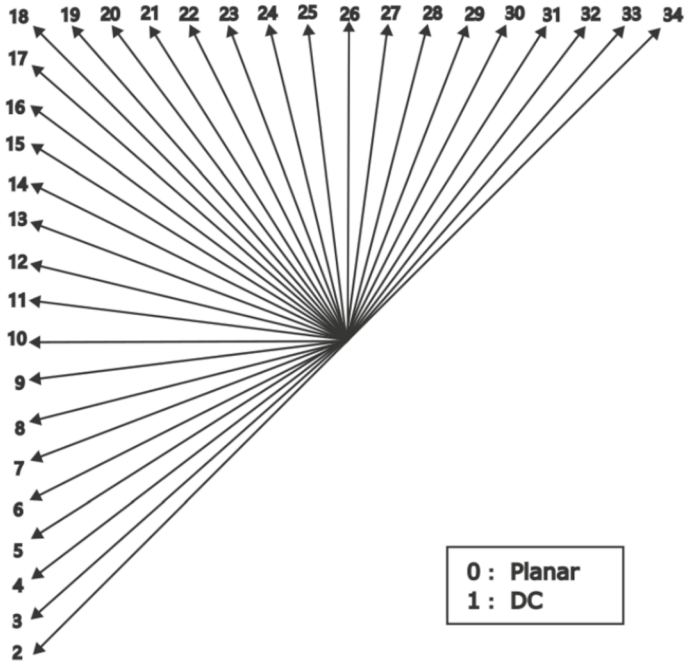

Fig. 4 All the DMs in SHVC [28]

Table 5 The DM distribution of 8×8 CUs

Index	DMs	Percentage	Index	DMs	Percentage
1	0	29.37%	11	27	1.73%
2	1	12.54%	12	6	1.72%
3	26	12.06%	13	24	1.50%
4	10	5.20%	14	13	1.39%
5	11	4.58%	15	23	1.10%
6	25	4.53%	16	2	1.06%
7	9	3.79%	17	18	0.98%
8	8	2.04%	18	34	0.97%
9	7	1.89%	19	5	0.96%
10	12	1.79%	20	14	0.96%

In order to improve the coding speed, we first investigate the DMs distribution, and then predict the candidate DMs based on distribution.

6.1 DM distribution

In order to obtain candidate DMs, we need to first investigate their distribution. Using the experimental parameters and sequences as above in testing, we can obtain the distribution of the DMs. Taking the DM distribution of 8×8 CUs as an example, the corresponding DM distribution is listed in Table 5.

In Table 5, the index refers to the order of the corresponding DM and the percentage refers to the ration of this DM is selected as the optimal one. From Table 5, we can observe that with the increase of the index, the percentage of the DMs decreases. As the

index reaches 10, its percentage decreases to be 1.79%. The sum of the percentage of the first ten DMs reaches 77.79%, which means most of the CUs eventually select the optimal DM in these 10 DMs. The remainder occupies 22.21%, which means few of CUs will eventually select the optimal DM in the remainder DMs. Therefore, we define the first ten DMs of the as the Basic Modes Set (BMS), and the rest DMs as the Rest Direction Modes Set (RDMS) When the index is greater than 20, the corresponding percentage of a DM is very small, so we have not list them in Table 5.

6.2 DMs selection based on probability

Since DMs are within their collocated CU, it is certain that they have a strong relationship. Therefore, we can combine DMs within their collocated CU to predict candidate DMs. When probability of the CU is very small, even its DM prediction is not very accurate, the corresponding influence on coding performance is not very significant. In this condition, in order to improve coding speed, we can only choose a part of DMs to check. As mentioned above, the DMs in BMS occupy 77.79% percentage, and there are only 10 DMs. If probabilities of CUs are not very high, we can only choose the DMs in BMS, thereby the coding speed can be significantly improved. The key is how to obtain the optimal probability of these CUs. In order to obtain the optimal probability, using the experimental parameters and sequences mentioned above, we select commonly test values, such as: 5%, 10%, 15%, 20%, 25%, ...and so on in testing, the corresponding coding performances for 8×8 CU are listed in Fig. 5:

In Fig. 5, the horizontal axis refers to probabilities of 8×8 CUs, the vertical axis refers to BDBR. As can be seen in Fig. 5: the value of BDBR stays approximately constant when probabilities are smaller than or equal to 0.25. Therefore, when probabilities are smaller than or equal to 0.25, we can only use the BMS. To further improve coding efficiency, for CUs with higher probability, we can continue to increase the number of the DMs in the BMS. Experimental results indicates that, when probabilities are smaller than or equal to 0.35, shifting 4 DMs with the highest probability from the RDMS to the BMS can obtain the optimal performance. When probabilities are smaller than or equal to 0.45, shifting 4 DMs with the highest probability from the RDMS to the BMS can also obtain the optimal performance. When probabilities are smaller than or equal to 0.55, shifting 6 DMs with the highest probability from the RDMS to the BMS can obtain the optimal

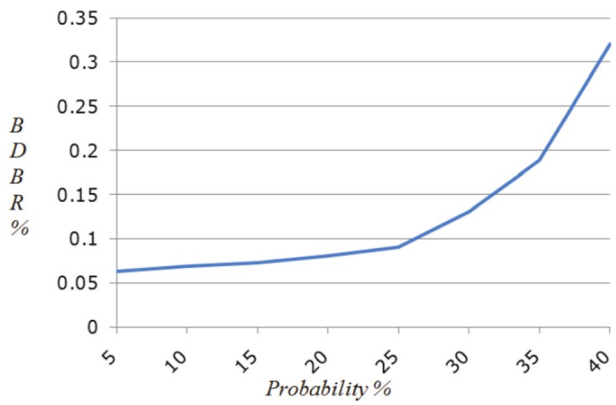

Fig. 5 Probabilities of 8×8 CUs and their BDBRs

Table 6 The number of DMs in the BMS under different probabilities of 8×8 CUs

Probability of CUs	DMs Number in the BMS
Less than or equal to 0.25	10
Reaches 0.35	14
Reaches 0.45	18
Reaches 0.55	24

Table 7 Performance comparisons among different proposed strategies

Type	Sequence	TFBCUP		PBIMES		PDBDS		Proposed	
		TS	BDBR	TS	BDBR	TS	BDBR	TS	BDBR
A	Traffic	43.12%	−0.05%	62.12%	−0.10%	26.80%	−0.17%	83.52%	0.3%
	PeopleOnstreet	40.67%	−0.30%	61.35%	−0.25%	27.72%	−0.11%	84.74%	0.6%
B	Kimono	44.87%	−0.20%	67.03%	−0.24%	32.50%	−0.19%	83.53%	0.0%
	ParkScene	43.26%	−0.10%	64.58%	−0.13%	30.49%	−0.12%	82.84%	−0.1%
	Cactus	40.35%	−0.02%	60.35%	0.32%	26.31%	−0.19%	70.27%	0.8%
	BasketballDrive	48.29%	−0.25%	54.02%	0.81%	23.45%	−0.21%	69.24%	1.5%
	BQTerrace	41.12%	0.02%	42.11%	0.94%	24.60%	−0.10%	68.05%	0.8%
Average		43.10%	−0.04%	58.79%	0.19%	27.41%	−0.16%	77.60%	0.56%

performance. The number of DMs in the BMs under different probabilities of 8×8 CUs is listed in Table 6. The results of the remaining depths of CUs are similar to those for 8×8 CUs. Through the above processes, the candidate DMs to be checked are remarkably reduced, thereby the coding speed can be significantly improved.

With the probabilities increase, the number of the DMs in the BMS also continuously increase. When probabilities are greater than or equal to 0.55, we need to continue to add more DMs, which will lead to coding speed cannot be improved efficiently. To solve this problem, we use a Sobel operator to obtain the gradient for a pixel and sum up the amplitudes of similar gradients to predict potential DMs and efficiently skip unlikely ones [5].

7 Experimental results

In order to evaluate the performance of the proposed algorithm for SSHVC, we use the reference software (SHM 11.0) and test the proposed algorithm on a server with Intel (R) 2.0 GHz CPU and 30 GB memory. We evaluate the performances of the proposed algorithms in terms of coding efficiency and coding speed. Coding efficiency is measured using the BDBR, coding speed improvement is denoted as TS, which is the percentage of coding time saving only in EL.

As mentioned above, we have provided three strategies, including: TFBCUP, PBIMES and PDBDS. Since all the above processes use the parameter configuration in Case 4, we also use this parameter configuration. The corresponding coding performance is listed in Table 7.

As shown in Table 7, the average coding speed improvements of "TFBCUP", "PBIMES" and "PDBDS" are 43.10%, 58.79% and 27.41% respectively. Their corresponding average

BDBR changes are −0.04%, 0.19% and −0.16% respectively. These three strategies all significantly improve the encoding speed, while the loss of coding efficiency is small. The coding speed improvement of the PBIMES is the most significant, but its BDBR also increases most significantly accordingly. This is because in this process, the IM is very complex but its proportion is very small, skipping IM can significantly improve coding speed. In addition, IM is within its collocated CU, they should have a strong relationship. Combining probabilities of IM with probabilities of its collocated CU, majority of CUs can early skip unnecessary IM checking. Therefore, PBIMES can improve coding speed most significantly. Since each CU contain all CMs, TFBCUP can exclude many unlikely CUs, thus it also can significantly improve coding speed. Due to the strict early termination thresholds for depth in the TFBCUP, compared to PBIMES, it has a relatively small improvement in coding speed and the change of BDBR is also negligible. Since PDBDS only predict candidate DMs and exclude unlikely DMs in rough mode decision, whose own proportion of coding time is also not very high, which also lead to coding speed improvement of PDBDS not very significantly. TS and BDBR of the proposed algorithm with case 1 which uses "TFBCUP", "PBIMES" and "PDBDS" are listed in Table 7 also. From the result, the proposed algorithm significantly improves the encoding speed comparing to use one strategy only, while the loss of coding efficiency is acceptable.

To illustrate the overall performance of the proposed algorithm, which integrates all proposed strategies, we compare the performance of our algorithm with PBFIP[3] and HSEIP[29]. To the best of our knowledge, these two algorithms are the state-of-the-art fast Intra coding algorithms for SSHVC. To ensure a fair comparison, all algorithms are tested on the same computing platform. According to the two scalability ratios and two QP settings mentioned above, we group the combinations into four cases, their corresponding performance comparisons are listed in Tables 8, 9, 10 and 11 respectively.

In Table 8 (Case 1), the coding speed of the proposed algorithm, PBFIP and HSEIP are improved by 77.60%, 57.59% and 72.17%, respectively. Compared with PBFIP and HSEIP, the proposed algorithm is faster by 20.01% and 5.43%, respectively. The BDBR of the proposed algorithm, PBFIP and HSEIP are 0.56%, 0.27% and 0.67%, respectively. The BDBR of the proposed algorithm is greater than PBFIP by 0.29 but smaller than HSEIP by 0.11. In Table 9 (Case 2), the coding speed of the proposed algorithm, PBFIP and HSEIP are improved by 77.77%, 60.25% and 73.86%, respectively. Compared with PBFIP and HSEIP, the proposed algorithm is faster by 17.52% and 3.91%,

Table 8 Overall performance comparisons with case 1

Type	Sequence	Proposed		PBFIP		HSEIP	
		TS	BDBR	TS	BDBR	TS	BDBR
A	Traffic	83.52%	0.3%	55.89%	0.01%	74.45%	−0.03%
	PeopleOnstreet	84.74%	0.6%	54.51%	0.08%	75.02%	0.28%
B	Kimono	83.53%	0.0%	66.04%	0.19%	74.91%	−0.23%
	ParkScene	82.84%	−0.1%	57.28%	0.10%	73.26%	−0.16%
	Cactus	70.27%	0.8%	56.12%	0.68%	71.15%	1.02%
	BasketballDrive	69.24%	1.5%	60.47%	1.12%	66.23%	1.46%
	BQTerrace	68.05%	0.8%	53.52%	0.49%	70.14%	2.32%
Average		77.60%	0.56%	57.59%	0.27%	72.17%	0.67%

Table 9 Overall performance comparisons with case 2

Type	Sequence	Proposed		PBFIP		HSEIP	
		TS	BDBR	TS	BDBR	TS	BDBR
A	Traffic	84.20%	0.1%	60.20%	−0.15%	75.73%	−0.17%
	PeopleOnstreet	83.94%	−0.1%	56.63%	−0.20%	76.21%	−0.29%
B	Kimono	84.98%	0.0%	67.25%	−0.39%	75.43%	−0.3%
	ParkScene	83.88%	0.1%	60.65%	−0.10%	74.35%	−0.15%
	Cactus	71.18%	0.6%	59.11%	0.54%	73.08%	0.68%
	BasketballDrive	67.85%	1.4%	63.14%	1.08%	69.97%	1.47%
	BQTerrace	68.38%	0.9%	54.79%	0.47%	72.24%	2.15%
Average		77.77%	0.43%	60.25%	0.18%	73.86%	0.48%

Table 10 Overall performance comparisons with case 3

Type	Sequence	Proposed		PBFIP		HSEIP	
		TS	BDBR	TS	BDBR	TS	BDBR
B	Kimono	82.97%	−0.3%	68.02%	−0.42%	75.14%	−0.33%
	ParkScene	82.05%	−0.2%	61.03%	−0.22%	74.31%	−0.21%
	Cactus	77.87%	0.0%	59.61%	−0.12%	74.87%	−0.13%
	BasketballDrive	76.35%	0.8%	63.69%	0.06%	72.98%	0.12%
	BQTerrace	77.08%	0.2%	56.81%	−0.02%	73.95%	0.16%
Average		79.26%	0.10%	61.83%	−0.14%	74.25%	−0.08%

Table 11 Overall performance comparisons with case 4

Type	Sequence	Proposed		PBFIP		HSEIP	
		TS	BDBR	TS	BDBR	TS	BDBR
B	Kimono	84.27%	−0.2%	69.65%	−0.68%	76.83%	−0.21%
	ParkScene	83.81%	−0.1%	63.97%	−0.68%	76.18%	−0.13%
	Cactus	78.30%	0.0%	62.29%	−0.38%	76.45%	−0.31%
	BasketballDrive	76.10%	0.2%	66.04%	−0.23%	75.03%	−0.31%
	BQTerrace	77.13%	0.1%	59.75%	−0.12%	75.27%	−0.13%
Average		79.92%	0.0%	64.34%	−0.42%	75.95%	−0.22%

respectively. The BDBR of this algorithm is 0.43%, and the BDBR of PBFIP and HSEIP are 0.18% and 0.48%, respectively. The BDBR of the proposed algorithm is greater than PBFIP by 0.25 but smaller than HSEIP by 0.05.

In Table 10 (Case 3), the coding speed of the proposed algorithm, PBFIP and HSEIP are improved by 79.26%, 61.83% and 74.25%, respectively. Compared with PBFIP and HSEIP, the proposed algorithm is faster by 17.43% and 5.01%, respectively. The BDBR of the proposed algorithm, PBFIP and HSEIP are 0.10%, −0.14% and −0.08%, respectively. Compared with PBFIP and HSEIP, the BDBR of the proposed algorithm is greater by 0.24 and 0.18, respectively. In Table 11 (Case 4), the coding speed of the proposed algorithm, PBFIP and HSEIP are improved by 79.92%, 64.34% and 75.95%, respectively. Compared with PBFIP and HSEIP, the proposed algorithm is faster by

15.58% and 3.97%, respectively. The BDBR of this algorithm is 0.0%, and the BDBR of PBFIP and HSEIP are −0.42% and −0.22%, respectively. Compared with PBFIP and HSEIP, the BDBR of the proposed algorithm is greater by 0.42 and 0.22, respectively.

In order to further demonstrate the overall performance of the proposed algorithm for all four cases, Table 12 provides their overall average performance comparisons among these three algorithms.

From Table 12 , the overall coding speed of the proposed algorithm, PBFIP and HSEIP are improved by 78.64%, 61.00% and 74.06%, respectively. Compared with PBFIP and HSEIP, the proposed algorithm is faster by 17.64% and 4.58%, respectively. The BDBR of the proposed algorithm, PBFIP and HSEIP are 0.27%, −0.03% and 0.21%, respectively. Compared with PBFIP and HSEIP, the BDBR of the proposed algorithm is greater by 0.30 and 0.06, respectively. Although there is a certain BDBR changes of this algorithm compared with the other two algorithms, considering the remarkable improvement in coding speed, these BDBR decreases are acceptable.

Here are the main reasons why the proposed algorithm can effectively improve coding speed. First, in CU selection, since the CU selection has a strong relationship with texture complexity, we can predict candidate CUs and early skip unlikely CUs based on texture complexity. Second, we observe that coding complexity of IM is much higher than that of ILRM, and its proportion is much smaller than that of ILRM. Therefore, skipping unlikely IM checking can significantly improve coding speed. Moreover, IM is within its collocated CU, they should have a strong relationship. We combine probabilities of IM with probabilities of its collocated CU to exclude unlikely IM of CUs. Since we simultaneously utilize both probabilities of IM and probabilities of their collocated CUs, relevant coding information is more fully used, more CUs can early skip unnecessary IM checking. Third, since DMs are within their collocated CU, they also should have a strong relationship. Therefore, we combine DMs distribution with the probability of their collocated CU to adaptively predict candidate DMs and exclude unlikely ones. More specially, for CUs with smaller probabilities, we can select less DMs to improve coding speed. On the contrary, for CUs with lager probabilities, we can select more DMs to maintain coding efficiency. Since we fully use DMs distribution and the probability of their collocated CU, more DMs can be early skipped. In a word, we not directly predict candidate CUs, IM and DMs, but combine CUs with IM and DMs to predict candidate CUs, IM and DMs, so more unlikely CUs, IM and DMs can be early skipped. Therefore, the proposed algorithm can effectively improve coding speed.

Table 12 Overall average performance comparison of all four cases

Test	Proposed		PBFIP		HSEIP	
	TS	BDBR	TS	BDBR	TS	BDBR
Table 8	77.60%	0.56%	57.59%	0.27%	72.17%	0.67%
Table 9	77.77%	0.43%	60.25%	0.18%	73.86%	0.48%
Table 10	79.26%	0.10%	61.83%	−0.14%	74.25%	−0.08%
Table 11	79.92%	0.00%	64.34%	−0.42%	75.95%	−0.22%
Average	78.64%	0.27%	61.00%	−0.03%	74.06%	0.21%

It is worth noting that BDBR decreases come from Case 1 and Case 2 (scalability ratio 1.5x) in Tables 8 and 9 are significantly greater than those from Case 3 and Case 4 (scalability ratio 2x) in Tables 10 and 11. The main reason is that the ILR prediction is not very accurate when the scalability ratio is 2x. Due to the large resolution difference between BL and EL, the interpolation effect is not very good, so the prediction of the ILR mode is not very accurate either. Therefore, the corresponding BDBR change increases significantly when the scalability ratio is 2x.

The performance differences observed in our proposed methods across video sequences are the result of three interdependent factors that collectively influence the effectiveness of the algorithm. Primarily, texture complexity differences play a pivotal role, where sequences with complex textures (e.g., "ParkScene") demonstrate superior gains from our texture-based CU prediction (TFBCUP), achieving 43.26% TS improvement in Table 7 compared to 41.12% for homogeneous sequences like "BQTerrace", as intricate spatial features enable more accurate probability estimations. Simultaneously, inter-layer dependency strength critically affects consistency. Sequences with strong BL-EL correlations (e.g., "Kimono") achieve stable improvements (84.27% TS in Table 11) as our algorithm effectively utilizes collocated CU information. However, sequences with weak dependencies (e.g., "BasketballDrive") show less improvements (76.10% TS in Table 11) , because their independent layer characteristics reduce the prediction reliability. Furthermore, motion complexity introduces additional dimensionality, high-motion sequences like "BasketballDrive" exhibit greater performance variability (1.5% BDBR in Case 1 vs 0.2% in Case 4) because rapid movement affects both inter-layer prediction accuracy and directional mode selection. This aligns with our DM distribution analysis showing that motion-compromised sequences require more exhaustive mode checks (Table 5). Texture complexity differences, layer interdependence, and motion complexity collectively determine the performance differences of our proposed methods across different video sequences.

8 Conclusion

In this paper, we have proposed a new and effective fast coding algorithm for SSHVC. First, we obtain a CU's probability based on textural features so as to predict candidate CUs and early skip unlikely CUs. Next, we combine both probability of IM and its collocated CU to early skip unlikely CMs. Then, we integrate DM distributions and probability of their collocated CU to predict candidate DMs and remove unlikely DMs. Since we jointly utilize probability of IM, DM distributions and probabilities of their collocated CUs, more CUs can early skip unlikely IM and DMs. Experimental results also demonstrate that the proposed algorithm can improve the coding speed significantly. Since Deep learning has been widely used and achieves excellent performance, we plan to use deep learning to further improve coding speed in our future work.

Author Contributions

W.L. conceptualized the study and wrote the main manuscript text. W.L. and Y.W. designed and performed the experiments. W.L. and Y.Z performed the editing. All authors reviewed and approved the manuscript.

Funding

This work is supported by Guizhou Provincial Basic Research Program (Natural Science) Qiankehejichu-ZK[2023] Yiban252, High-level Innovative Talents in Guizhou Province (No. GCC[2023]033), Guizhou Provincial Basic Research Program (Natural Science) Qiankehejichu-JC[2024]Zhongdian013, Natural Science Research Project of Guizhou Provincial

Department of Education (No. QJJ[2024]009), National Natural Science Foundation of China (U22A2026) and QIANKEHE PLATFORM TALENT BQW[2024]015/GZNU[2024]01.

Data Availability
Not applicable.

Declarations

Competing interests
The authors declare no Competing interests.

Received: 24 April 2025 Accepted: 2 October 2025
Published online: 04 November 2025

References

1. B. Li, G.J. Sullivan, J. Xu, Comparison of compression performance of hevc working draft 5 with avc high profile, in *JCTVC-H0360, JCT-VC Meeting, San Jose (February 2012)*, (2012)
2. Z. Feng, P. Liu, K. Jia, K. Duan, Fast intra ctu depth decision for hevc. IEEE Access **6**, 45262–45269 (2018)
3. D. Wang, Y. Sun, J. Liu, F. Dufaux, X. Lu, B. Hang, Probability-based fast intra prediction algorithm for spatial shvc. IEEE Trans. Broadcast. **68**(1), 83–96 (2021)
4. X. Mai, T. Li, Z. Wang, X. Deng, R. Yang, Z. Guan, Reducing complexity of hevc: a deep learning approach. IEEE Trans. Image Process. **27**(10), 5044–5059 (2018)
5. D. Wang, Y. Sun, C. Zhu, W. Li, F. Dufaux, J. Luo, Fast depth and mode decision in intra prediction for quality shvc. IEEE Trans. Image Process. **29**, 6136–6150 (2020)
6. D. Wang, Y. Sun, W. Li, C. Zhu, F. Dufaux, Fast inter mode predictions for shvc, in *2019 IEEE International Conference on Multimedia and Expo (ICME)*, pp 1696–1701. IEEE, (2019)
7. J. Zuo, Q. Li, Fast algorithm for intra prediction in quality shvc, in *2019 IEEE 2nd International Conference on Information Communication and Signal Processing (ICICSP)*, pp. 332–337. IEEE, (2019)
8. D. Wang, Yu. Ce Zhu, F.D. Sun, Y. Huang, Efficient multi-strategy intra prediction for quality scalable high efficiency video coding. IEEE Trans. Image Process. **28**(4), 2063–2074 (2017)
9. H.R. Tohidypour, M.T. Pourazad, P. Nasiopoulos, An encoder complexity reduction scheme for quality/fidelity scalable hevc. IEEE Trans. Broadcast. **62**(3), 664–674 (2016)
10. C.-S. Park, B.-K. Dan, H. Choi, S.-J. Ko, A statistical approach for fast mode decision in scalable video coding. IEEE Trans. Circuits Syst. Video Technol. **19**(12), 1915–1920 (2009)
11. H.R. Tohidypour, M.T. Pourazad, P. Nasiopoulos, Content adaptive complexity reduction scheme for quality/fidelity scalable hevc, in *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 1744–1748. IEEE, (2013)
12. D. Wang, C. Yuan, Y. Sun, J. Zhang, X. Jin, A fast mode decision algorithm applied to coarse-grain quality scalable video coding. J. Vis. Commun. Image Represent. **25**(7), 1631–1639 (2014)
13. T. Zhao, S. Kwong, H. Wang, C.-C. Jay Kuo, H. 264/svc mode decision based on optimal stopping theory. IEEE Trans. Image Process. **21**(5), 2607–2618 (2012)
14. D. Wang, X. Lu, F. Dufaux, Q. Wang, W. Li, B. Hang, C. Zhu, A probability-based all-zero block early termination algorithm for qshvc, in *2023 IEEE International Conference on Image Processing (ICIP)*, pp. 2295–2299. IEEE, (2023)
15. H.R. Tohidypour, H. Bashashati, M.T. Pourazad, P. Nasiopoulos, Fast mode assignment for quality scalable extension of the high efficiency video coding (hevc) standard: A bayesian approach, in *Proceedings of the 6th Balkan Conference in Informatics*, pp 61–65, (2013)
16. L. Xin, Yu. Chang, G.R. Martin, Fast intra-and inter-coding algorithms for the spatially scalable extension of h. 265/hevc. Multimedia Tools Appl. **79**(35), 26447–26465 (2020)
17. S. Cho, M. Kim, Fast cu splitting and pruning for suboptimal cu partitioning in hevc intra coding. IEEE Trans. Circuits Syst. Video Technol. **23**(9), 1555–1564 (2013)
18. X. Liu, Y. Liu, P. Wang, C.-F. Lai, H.-C. Chao, An adaptive mode decision algorithm based on video texture characteristics for hevc intra prediction. IEEE Trans. Circuits Syst. Video Technol. **27**(8), 1737–1748 (2016)
19. W. Jiang, H. Ma, Y. Chen, Gradient based fast mode decision algorithm for intra prediction in hevc, in *2012 2nd international conference on consumer electronics, communications and networks (CECNet)*, pp. 1836–1840. IEEE, (2012)
20. T. Zhang, M.-T. Sun, D. Zhao, W. Gao, Fast intra-mode and cu size decision for hevc. IEEE Trans. Circuits Syst. Video Technol. **27**(8), 1714–1726 (2016)
21. M. Jamali, S. Coulombe, F. Caron, Fast hevc intra mode decision based on edge detection and satd costs classification., in *2015 data compression conference*, pp 43–52. IEEE, (2015)
22. H. Zhang, Z. Ma, Fast intra mode decision for high efficiency video coding (hevc). IEEE Trans. Circuits Syst. Video Technol. **24**(4), 660–668 (2013)
23. SHM Common, Test conditions and software reference configurations, doc. jctvc-q1009, itu-t sg 16 wp 3 and iso/iec jtc1/sc 29/wg 11, mar. 2014. shm 12.1 software package
24. D. Wang, J. Yu, X. Lu, F. Dufaux, B. Hang, H. Guo, C. Zhu, Fast mode and cu splitting decision for intra prediction in vvc scc. IEEE Trans. Broadcast. **70**(3), 872–83 (2024)
25. R. Kusumawati, A. D'arofah, P.A. Pramana, Comparison performance of naive bayes classifier and support vector machine algorithm for twitter's classification of tokopedia services, in *Journal of Physics: Conference Series*, pp. 012016. IOP Publishing, (2019)

26. M. Bala, R.K. Agrawal, Evaluation of decision tree svm framework using different statistical measures, in *2009 International Conference on Advances in Recent Technologies in Communication and Computing*, pp. 341–345. IEEE, (2009)
27. X. Liu, Y. Li, D. Liu, P. Wang, L.T. Yang, An adaptive cu size decision algorithm for hevc intra prediction based on complexity classification using machine learning. IEEE Trans. Circuits Syst. Video Technol. **29**(1), 144–155 (2017)
28. V. Sze, M. Budagavi, G.J. Sullivan, High efficiency video coding (hevc): Algorithms and architectures, no. 9783319068947, (2014)
29. D. Wang, Y. Sun, W. Li, L. Xie, X. Lu, F. Dufaux, C. Zhu, Hybrid strategies for efficient intra prediction in spatial shvc. IEEE Trans. Broadcast. **69**(2), 455–468 (2022)

Publisher's Note