A Fast CU Size Decision Optimal Algorithm based on Coding Bits for HEVC

Jianhua Wang (wangjianhua655@163.com)
South China Agricultural University College of Economics and Management
Zhihao Chen
South China Agricultural University
Feng Lin
Nanyang Technological University
Jing Zhao
South China Agricultural University
Yongbing Long
South China Agricultural University
Yubin Lan
South China Agricultural University

Research

Keywords: HEVC, CU partition, Intra prediction, Coding bits

Posted Date: August 31st, 2020

DOI: https://doi.org/10.21203/rs.3.rs-67775/v1

Abstract

HEVC (High Efficiency Video Codng) employs quadtree CTU (Coding Tree Unit) structure to improve its coding efficiency, but at the same time, it also requires a very high computational complexity due to its exhaustive search process for an optimal partition mode for the current CU(Coding Unit). Aiming to solve the problem, a fast CU size decision optimal algorithm based on coding bits is presented for HEVC in this paper. The contribution of this paper lies that we successfully use the coding bits technology to quickly determine the optimal partition mode for the current CU. Specially, in our scheme, firstly we carefully observe and statistically analyze the relationship among the texture complexity and partition size and coding bits in the CUs of video image; Secondly we find the correlation between coding bits and partition size based on the relationship above; Thirdly, we build the corresponding threshold of coding bits for partition size under different CU size and QP value based on the correlation above to reduce many unnecessary prediction and partition operations for the current CU. As a result, our proposed algorithm can effectively saving lots of computational complexity for HEVC. The simulation results show that our proposed fast CU size decision algorithm based on coding bits in this paper can save about 34.67% coding time, and only at a cost of 0.61% BDBR increase and 0.043db BDPSNR decline compared with the standard reference of HM16.1, thus improving the coding performance of HEVC.

Introduction

HEVC (High Efficiency Video Coding)[1] as the latest international video compression standard, has been jointly developed by the ISO/IEC Moving Picture Experts Group (MPEG) and ITU-T Video Coding Experts Group (VCEG) [2-3]. HEVC mainly users the new quadtree structure based on Coding Tree Unit (CTU) to achieve a more high-efficient video coding compression rate, especially in some high-resolution video contents, such as HD(High Definition) videos, UHD (Ultra HD) videos, and so on, compared to previous video coding standards, such as H.264/AVC. It is reported that the latest HEVC standard only requires about 50% bit-rate of the H.264/AVC at the same perceptual quality [4-5].

HEVC improves its coding efficiency by using the new structures technology, but at the same time, it also greatly increases its computational complexity because it needs to exhaustive execute the RDCost (Rate-Distortion Cost) process to obtain the optimal partition mode for each CU (Coding Unit) from all of their CU combinations[6]. However, the large computational complexity has become a very important problem in some embedded real-time devices, especially in some power constrained or storage constrained application devices. Therefore, it is very necessary to lower the computational complexity for HEVC by developing some efficient CU partition algorithms with negligible degradation [7].

In order to solve the problem above, many efficient methods have been taken to reduce the computational complexity in intra CU size decision for HEVC recently. For example, a fast CU size decision algorithm based on homogenous CUs termination [8], a fast CU size decision algorithm based on texture[9], a fast CU size determination algorithm based on adaptive discretization total variation threshold[10], a fast CU size partitioning algorithm based on data mining[11], and so on, have been proposed to reduce the computational complexity for HEVC. Although these schemes above are well designed, but there is still a need to further develop more efficient schemes to greatly reduce their high computational complexity for HEVC.

Different from these methods above, in this paper, a fast CU size decision optimal algorithm based on coding bits is proposed to reduce its total computational complexity for HEVC. The contribution of this paper lies in the following:

(1) Find the correlation between coding bits and partition size for a CU based on the relationship among the texture complexity and coding bits and partition size above after carefully observation and statistical analysis of the relationships among the texture complexity and partition size and coding bits of a CU in video image;

(2) The corresponding threshold of coding bits for CU partition under different size and QP value is exploited to quickly judge whether the current CU needs to be further divided based on the correlation above, which can reduce many unnecessary prediction and partition operations for the current CU;

(3) A series of experiments are performed to verify the effectiveness of our proposed method in this paper. And the kinds of simulation results show the effectiveness of our proposed method in this paper;

The rest of this paper are organized as follows. In section 2, the related knowledge on CU size decision schemes are introduced. The proposed method is presented in section 3. The simulation result and analysis of proposed scheme compared with existing methods are presented in section 4. In section 5, we give some my conclusion.

2. The Related knowledge on CU Size Decision Schemes

2.1 Standard process of CU size decision scheme

During the standard process of CU size decision in HEVC, HEVC needs to traverse all possible partitions modes to find the optimal size partition mode for the current CU, and the partition mode with minimum RDCost will be taken as the optimal size partition mode for the current CU. A CTU with 64×64 can be recursively divided into four sub CUs with the same size in the form of quadtree. Then, each sub CUs can be further divided into four sub CUs with the same size, and the process will end only when the smallest CU is reached. The depth change of CU size from 64×64 to 8×8 corresponds to the depth 0 to 3. Fig.1 is partitioning process for a LCU with 64×64 size, and its specific division process can be shown in following steps.

Step 1: Calculate the RDCost of LCU with 64×64 in depth 0 before partition, and take it as $RDCost_{unsplit}$;

Step 2: Calculate the RDCost of four sub CUs in depth 1 after partition respectively, then calculate their sum and take it as $RDCost_{split}$. It is necessary to recursively search down layer by layer until the maximum depth is reached;

Step 3: Compare the calculated $RDCost_{split}$ and $RDCost_{unsplit}$ before and after the partition. If $Cost_{split}$ is less than $RDCost_{unsplit}$, the optimal partition mode for the current LCU is the sub CUs in smaller level after the partition; otherwise, if $RDCost_{split} \geq RDCost_{unsplit}$, the best partition mode is the current LCU, and the current LCU does not need to be divided;

Step4: Traverse all CUs step by step according to the Z-scan sequence, and execute step 1 to step 3 until each CU of all levels in the LCU is calculated and judged, and the optimal partition mode of the whole LCU is finally determined;

After the four steps operation above, the optimal partition mode for the LCU with 64 × 64 can be determined, and higher coding efficiency can be realized by using the optimal partition mode above.

2.2 Complexity Analysis of CU size decision

As mentioned above, HEVC needs to use recursive quadtree to divide a LCU with 64×64 size, and finally find an optimal partition mode for the LCU. Since a 64×64 LCU can be divided into four sub CUs with 32×32, each sub 32×32 CUs can be divided into four sub 16×16 CUs, each sub 16×16 CUs can be further divided into four sub 8×8 CUs, therefore, a LCU with 64×64 can generate $4^0+4^1+4^2+4^3=85$ sub CUs. Each in 85 sub CUs needs to be predicted and encoded by dividing it into one or more PUs, whose size ranges from 64×64 to 4×4, so a LCU with 64×64 can produce $4^0+4^1+4^2+4^3+4^4=241$ PUs. Table 1 shows the corresponding number of sub PUs with different CU sizes for a LCD.

Table 1 Number of sub CU and PU blocks with different sizes

The size of CU blocks	The number of sub PUs blocks
4×4	256
8×8	64
16×16	16
32×32	4
64×64	1

There are 35 candidate modes for each PU partition, including 33 angular modes and DC and Planar modes in HEVC. If each in 341 PUs needs to independently execute 35 prediction and partition operations based on RDCost calculation, it will need to take 341×35=11935 times RDCost calculations for each CU in HEVC, which is very complex and time-consuming. Accordingly, it is very necessary to reduce the computational complexity for HEVC.

2.3 Related study for CU partition decision schemes

Recently, a number of CU size decision algorithms have been proposed to reduce the computational complexity for HEVC.

Zhang et.al [8] proposed a fast CU size decision algorithm based on homogenous CUs and linear support vector machines to reduce the computation complexity for HEVC, where homogenous CUs and two linear support vector machines based on depth difference and HAD cost ratio are used to complete the decisions of early CU split and early CU termination for some of the CUs. Ha et al [9] presented a fast CU size decision algorithm based on texture to reduce the computation complexity for HEVC. In this paper, the determined CU sizes, which are decided by the texture of the image, are only use. Zhou et al [10] suggested a fast CU size decision algorithm based on visual saliency detection to reduce the computation complexity for HEVC, which mainly use visual saliency detection to realize the adaptive and perceptual CU size decision, thus alleviating computation complexity for HEVC. Song et al [11] proposed a fast CU size determination algorithm based on adaptive discretization total variation (DTV) threshold to reduce the computation complexity for HEVC. In this paper, adaptive discretization total variation (DTV) threshold is used to skip some specific depth levels for HEVC. Ruiz et al [12] presented a fast CU partitioning algorithm based on offline-trained decision tree with three hierarchical nodes to reduce the computation complexity for HEVC, where they use the content texture properties of CUs as well as the inter-sub-CUs statistics of the same depth level to determine the decision rules computed in each node. Cen et al [13] proposed a fast CU depth decision mechanism based on adaptive CU depth range determination and comparison to reduce the encoding complexity for HEVC, where the RDCost calculations outside the adaptive CU depth range and at the current CU depth are skipped to alleviate computation complexity for HEVC. Zhang et al [14] proposed a fast and efficient CU size decision algorithm based on temporal and spatial correlation to reduce the computation complexity for HEVC. In this paper, they mainly used video temporal and spatial property of coding information in previous coded frames to predict the current treeblock prediction mode and to early skip unnecessary CU. In the paper [15], a fast CU size decision algorithm based on depth information of neighboring CUs is proposed to reduce its encoding complexity for HEVC. In their scheme, they mainly exploited the depth information of neighboring CUs to realize the split decision and pruning decision of early CU. In the reference [16], a fast CU size decision algorithm based on bayesian theorem detection is presented to reduce the encoding complexity for HEVC, where they mainly use CU property and bayesian theorem detection to reduce TU processing complexity. In the paper [17], a novel fast CU encoding scheme based on spatiotemporal encoding parameters is proposed to reduce the encoding complexity for HEVC. In their method, an improved early CU SKIP detection method and a fast CU split decision method are used in this paper. In this work [18], a fast CU size decision algorithm based on statistical analysis is developed to reduce its encoder complexity for HEVC. In the paper, three approaches based on the statistical analysis, SKIP mode decision (SMD), CU skip estimation (CUSE), and early CU termination (ECUT) are presented to reduce its computation complexity for HEVC. In the paper [19], a fast CU size selection based on the bayesian decision rule is presented to reduce its whole encoder complexity for HEVC, where they mainly collected relevant and computational-friendly features to assist decisions on CU splitting. Chen et al. [20] proposed a fast CU size decision algorithm based on online progressive bayesian classification. In this paper, they mainly used SATD (Sum of Absolute Transformed Differences) of the prediction

residual and neighboring CU depths as key features to decide the CU size. Liu et al.[21] presented a fast CU size decision algorithm based on dual-SVM, where they divided the complexity of video content as high, low, and middle, and used a four-dimensional features, textural variance, neighboring Mean Squared Error, directional complexity from Sobel operations and the variance difference of the four sub-CU blocks, to represent them. Liu et al. [22] proposed a fast CU size decision algorithm based on Convolution Neural Network (CNN) accelerator for hardwired INTRA encoder. In their scheme, they mainly used three trained CNNs as well as Very Large Scale Integration circuits to predict the depth of CU with 32×32, 16×16 and 8× 8. Xu et al. [23] proposed a fast CU size decision algorithm based on early terminated hierarchical CNN, where hierarchical CNN is mainly used to determine the partition or non- partition at each prediction layer. Zhang et al. [24] suggested a fast CU size decision algorithm based on two-stage classification. At the first stage of their scheme, they firstly used off-line learning to determine the split, non-split, and uncertain, and at the second stage of their scheme, they used on-line learning to refine the uncertain predictions.

Different from these methods above, in this paper, a fast CU size decision optimal algorithm based on coding bits is proposed to quickly determine its optimal CU partition mode for the current CU, which can reduce many unnecessary prediction and partition operations, thus saving much computational complexity for HEVC.

3. Proposed Scheme

3.1 Motivation of our proposed method

3.1.1 Observe and analyze the relationship between texture complexity and partition size

In HEVC, the optimal partition mode of a CU generally requires the least resource consuming and data transfer. A CU with complex texture information is usually divided into more sub CUs, and the sub CUs with complex texture can be further split into more sub CUs until the until the maximum depth is reached. On the contrary, a CU with simple texture information generally does not need be divided into more sub CUs because of its simple texture features. Smaller size of CU contains more texture information and needs complex partition computations, while the larger size of CU includes less texture information and requires less partition computations. There is closer relationship between texture complexity and its partition size in a CU of video image. Fig 2 is the partition number of sub CUs in different texture complexity of CU in a BasketballDrill test sequence.

Fig2 shows that texture complexity of CU in video image has an important influence on its partition size of CU. The CU1 with complex texture can be split into 85 sub CUs. The CU2 with less complex texture can be partitioned into 36 sub CUs. CU3 is not divided into some sub CUs due to it simple texture information in its CU. The other CUs in the video image have the similar phenomenon as CU1, CU2 and CU3 above.

3.1.2 Observe and analyze the relationship between texture complexity and coding bits

In HEVC, the texture complexity in a CU also has to do with its coding bits. Different texture complexity of CU generally requires different number of coding bits to encode. A CU with complex texture usually requires more coding bits to accurately describe its rich texture information when dealing with the CU. On the contrary, A CU with simple texture generally asks less coding bits to describe its simple texture information. Under the same condition, large number of coding bits can usually represents more texture information for a CU, while small number of coding bits can generally show less texture information for a CU. The number of coding bits can reflect the complexity of texture information of a CU in video image. Fig 3 is the number of coding bits in different texture complexity of CU in a BasketballDrill test sequence.

From Fig 3, we can see clearly that different texture information requires different number of coding bits to obtain the optimal partition mode for a CU. In Fig 2, the CU 1 has the most complex texture information, therefore it needs to 852 coding bits to describe its rich texture information. Since the CU 2 has the less texture information compared to CU 1, it only needs 414 coding bits to encode its texture information. The CU 3 only requires 91 coding bits to encode its texture information due to its simple and smooth texture information. The number of coding bits is closely related to its texture complexity in a CU. The other CUs in the video image has the similar phenomenon as that above.

3.2 Find the correlation between coding bits and partition size

Based on the observation and analysis above, we find an important correlation between coding bits and partition size for a CU. A CU with more encoding bits generally contains more texture complexity and it is possible to be divided into more sub CUs with smaller size until the optimal combination is reached. On the contrary, A CU with smaller encoding bits generally contains less texture information and it is possible not be divided into more sub CUs correspondingly. Therefore, in this paper, we can use the coding bits of a CU to quickly judge its partition mode, which can reduce many unnecessary prediction and partition operations for the CU, and then save much computational complexity for HEVC.

Besides the size of CU, in HEVC, the QP value can also affect the total coding bits and partition size in a CU. For the same size of CU, when the QP value is large, the total number of coding bits are small. On the contrary, when its QP value is small, the total number of coding bits is large. In order to obtain the accurate relationship between coding bits and partition size, we use 12 different test sequences to test the number of coding bits and number of occurrences for each coding bits under different CU size and QP value. The larger the number of occurrences for each coding bits, the higher the probability of optimal partition mode for the current CU at the coding bit. Fig 4 to Fig 9 are the related experimental results. In our experiment, the frame rate and frame number of 12 different test sequences are set 25 and 100 respectively. Three different CU size (CU64×64, CU32×32 and CU16×16) and four different QP values (QP=22, QP=27, QP=32 and QP=37) are used respectively.

From Fig 4 to Fig 9, we can see clearly that, firstly the number of coding bits is closely related to its occurrences number, With the increase of coding bits, the number of occurrences for each coding bits will reach a maximum value, which stands for the optimal partition mode for the current CU and can be used to

judge whether the current CU need to be further divided. Secondly, the size of CU and the value of QP have obvious effects on its coding bits and partition mode. Different CU size and QP value requires different threshold of coding bits for CU partition. Therefore, in this paper, we can make use of the close relationship between the number of coding bits and its occurrences number under different CU size and QP value to quickly judge whether the current CU needs to be further divided, which can reduce many unnecessary prediction and partition operations, and then save much computational complexity for HEVC.

3.3 Build the threshold of coding bits for CU partition under different size and QP value

Based on the analysis above, in this subsection, we get a fast CU size decision method which can quickly judge whether the current CU needs to be further divided by setting a certain threshold value for coding bits under different CU size and QP value based on the correlation between the number of coding bits and its occurrences number above. In our scheme, when the coding bits of the current CU are greater than its thresholds, the partition of the current CU will continue, otherwise, the partition will be terminated in advance, which can reduce many unnecessary prediction and partition operations, and then save much computational complexity for HEVC. The threshold of coding bits for CU partition can be expressed in the following formula.

$$splitflag = \begin{cases} unsplit, coding\ bits \le Coding\ Bits_{TH} \\ split, coding\ bits > Coding\ Bits \end{cases} \quad (1)$$

Where coding bits represents the number of coding bits for the current CU; Coding $Bits_{TH}$ stands for the partition thresholds of our proposed method in this paper, which are shown in Table 2. In our scheme, we can obtained the partition thresholds of coding bits for a CU partition under different size and QP value by making use of the correlation between the number of coding bits and its occurrences number and the methods of statistical analysis, such as the Fig 4 to Fig 9, which is described in Section 3.2.

Table 2 Partition thresholds of CU with different size and QP value.

CU size	QP=22	QP=27	QP=32	QP=37
64 × 64	850	560	220	130
32 × 32	430	240	130	50
16 × 16	140	90	50	30

In order to verify the accuracy rate of our proposed partition thresholds above, in this paper, we also use the 12 different test sequences above to test its accuracy rate of partition thresholds under different CU size and QP value. Table 3 show the experimental results with QP=22 and QP=32.

Table 3 The accuracy rate of proposed partition threshold

Picture Size	Sequences name	The accuracy rate of partition algorithm(%) with QP=22			The accuracy rate of partition algorithm(%)with QP=32		
		64×64	32×32	16×16	64×64	32×32	16×16
Class A	PeopleOnstreet	0.78	0.73	0.78	0.68	0.69	0.73
	Traffic	0.76	0.73	0.72	0.75	0.69	0.67
Class B	BasketballDrive	0.68	0.76	0.75	0.72	0.76	0.72
	Cactus	0.64	0.69	0.65	0.69	0.67	0.62
Class C	RaceHorses	0.61	0.63	0.64	0.66	0.62	0.58
	BQMall	0.64	0.68	0.66	0.67	0.71	0.69
Class D	BasketballPass	0.71	0.69	0.71	0.72	0.69	0.74
	BlowingBubbles	0.88	0.84	0.71	0.81	0.74	0.75
Class E	Fourpeople	0.76	0.62	0.68	0.78	0.71.	0.68
	Johnny	0.78	0.72	0.68	0.81	0.67	0.61
Class F	BasketballDrillText	0.75	0.76	0.68	0.67	0.66	0.72
	Slideshow	0.79	0.73	0.71	0.76	0.73	0.67
Average		0.74	0.72	0.69	0.73	0.68	0.68

From table 2, we can clearly observer that the average accuracy rate of partition threshold in our proposed method is very high, reaching to about 70% under different CU size and QP value, which proves the validity of our partition threshold value in this paper. Therefore in this paper, we can make use of the partition

threshold value to quick divide a CU under different CU size and QP value, which can reduce many unnecessary prediction and partition operations, and then reduce lots of computational complexity for HEVC.

3.4 Realizing process of our proposed method

The whole realizing principle for our proposed scheme is that we use the coding bits of the current CU as the main way to quickly judge the partition process for the current CU. When the number of coding bits of the current CU are greater than its selected thresholds, the partition of the current CU will continue, otherwise, the partition will be terminated in advance, which can reduce many unnecessary prediction and partition operations, and then save much computational complexity for HEVC.

The basic realizing process for our proposed scheme is that, we firstly carefully observe and statistically analyze the relationship among texture complexity, coding bits and partition modes of CU in video image; Secondly we find the correlation between coding bits and partition mode for the current CU based on the relationship above; Thirdly we build the corresponding thresholds between coding bit and partition mode under different size and QP value based on the correlation between coding bits and partition mode above; At last, we use the partition thresholds above to quickly determine the partition process for the current CU by reducing many unnecessary partition operations. As a result, our proposed algorithm can effectively saving lots of computational complexity for HEVC.

Fig.10 is the realizing process of our proposed scheme, which includes five main realizing contents: obtain coding bits of current CU, select threshold value, compare difference value, execute corresponding process and obtain the optimal partition mode.

Where obtain coding bits of current CU mainly executes the calculation operations of coding bits for the current CU. Select threshold value mainly executes the selection operations for partition threshold value for the current CU according to different CU size and QP value from Table 2. Compare difference value mainly executes the comparing operations between the coding bits of the current CU and selected threshold value from Table 2 above. Execute corresponding process mainly select the different partition process to execute according to the difference value above. When the number of coding bits of the current CU are greater than its selected thresholds, it will continue to execute the partition operation for the current CU, otherwise, it will terminated the partition process in advance. Obtain the optimal partition mainly execute the determining operation of optimal partition mode for the current CU.

3.5 Realizing steps of our scheme

Fig.11.is realizing flow of our proposed scheme in this paper. The detailed realizing steps for our proposed scheme can be summarized up into the following steps:

Step 1: Encode a CU and read its depth;

Step 2: Judge its depth of the CU is less than 3. If it is less than 3, jump Step 3 to execute; otherwise, jump Step 5 to execute;

Step 3: Calculate the coding bits of the current CU;

Step 4: Select the corresponding threshold value by Table 2 according to its CU size and QP value of the current CU;

Step 5: Compare the difference value between coding bits of the current CU and threshold value selected from Table 2. If the coding bits of the current CU is less than threshold value, jump Step6 to execute; otherwise, the depth of the current CU plus 1, and jump Step 2 to execute;

Step6: Obtain the optimal partition mode for the current CU;

Step7: Complete the optimal partition for the LCU;

Experimental Results And Analysis

In order to verify the performance of o4ur proposed method above, some experiments are taken in this section. In our experiment, our designed experiments mainly includes the following two parts: build experimental environment and test experimental results under different testing conditions.

4.1 Build experimental environment

Our simulation environment was conducted on Inter(R) Core(TM) i7–5500 CPU @2.4. GHz and 4GB RAM, Windows 7 64-bit operating system. The test conditions and configurations in our experiment are built based on the JCT-VC[25]. Eighteen different test sequences are used to evaluate the performance of our proposed method in our experiment. These selected eighteen test sequences come from six different class with different resolutions: Class A(2560 × 1600), Class B (1920 × 1080), Class C (832 × 480), Class D(416 × 240), Class E(1280 × 720) and Class F(416 × 240). The frame rate and frame number of these test sequences are set to 25 and 100 respectively. Table 4 is the parameters of 18 different test sequences. Other some general configuration parameters are listed in Table 5.

Table 4 Parameters of 18 different test sequences in our experiment

Picture Size	Test sequences	Resolution	Frame rate	Frame number
Class A	PeopleOnstreet	2560×1600	25	100
	Traffic	2560×1600	25	100
	BQTerrace	2560×1600	25	100
Class B	BasketballDrive	1920×1080	25	100
	Cactus	1920×1080	25	100
	ParkScene	1920×1080	25	100
Class C	RaceHorses	1280×720	25	100
	BQMall	1280×720	25	100
	PartyScene	1280×720	25	100
Class D	BasketballPass	1024×768	25	100
	BlowingBubbles	1024×768	25	100
	BQSquare	1024×768	25	100
Class E	Fourpeople	832×480	25	100
	Johnny	832×480	25	100
	KristenAndSara	832×480	25	100
Class F	BasketballDrillText	416×240	25	100
	Slideshow	416×240	25	100
	ChinaSpeed	416×240	25	100

Table 5 Other some configuration parameters

Profile	Main
Gop Structure	Low Delay Random access
Max CU size	64×64
Max CU depth	4
Motion Search Mode	TZsearch
Motion Search Range	64
QP	22, 27, 32,37
Encoder.cfg	encoder_intra_rnain.cfg
Video frame	I frame
Reference Software Profile[25]	HM16.1

In our experiment, three general test indicators, BDBR (Bjontegaard Delta Bit Rate), BDPSNR (Bjontegaard Delta Peak Signal-to-Noise Ratio) and TS (Coding Time Saving), are used to evaluate the performance for our proposed scheme. They can be expressed as follows.

$$\mathrm{BDBR} = \frac{\text{The processed or transmitted data}}{\text{a unit time}} \quad (2)$$

$$\mathrm{BDPSNR} = 10 * \log_{10}\left\{\frac{(2^n-1)^2}{MSE}\right\} \quad (3)$$

$$\mathrm{TS} = \left|\frac{(\text{coding time}_{our} - \text{coding time}_{HM16.1})}{\text{coding time}_{HM16.1}}\right| \times 100\% \quad (4)$$

Four mode CU partition decision methods, Ha's scheme[9], Huade's scheme[26], Chen's scheme[27] and our proposed scheme are used to compared with the standard HM16.1 algorithm[25] in our experiment. Where Ha's scheme mainly used texture similarity of neighborhood CUs to quickly determine the optimal

partition mode for the current CU; Huade's mainly used adaptive depth selection to quickly locate the optimal partition mode for the current CU; Chen's scheme mainly used depth range prediction and mode reduction to quickly determine the optimal partition mode for the current CU; while our proposed method mainly used coding bits technologies to quickly determine the optimal partition mode for the current CU. The experimental results are shown as follows.

4.2 Test experimental results

In this subsection, we mainly test the performance of BDBR, BDPSNR and TS in different testing sequences compared with the standard reference HM16.1 scheme. The related testing results are shown in table 6.

Table 6 Experimental results of BDPSNR, BDBR, and TS in the proposed sub-algorithms (%, db, %)

Picture	Test sequences	Our scheme vs HM			Ha's scheme vs HM			Huade 's scheme vs HM			Chen's scheme vs HM		
Class		BDBR	BDPSNR	TS	BDBR	BDPSNR	TS	BDBR	BDPSNR	TS	BDBR	BDPSNR	TS
Class A	PeopleOnstreet	+0.64	-0.036	-34.16	+1.14	-0.054	-18.94	+1.62.	-0.058	-31.26	+0.51	-0.048	-42.11
	Traffic	+0.56	-0.048	-38.27	+0.96	-0.051	-18.47	+1.54	-0.046	-33.67	+0.48	-0.054	-41.54
	BQTerrace	+0.87	-0.042	-32.54	+0.93	-0.065	-18.12	+1.40	-0.054	-30.13	+0.76	-0.067	-40.77
Class B	BasketballDrive	+0.58	-0.050	-36.86	+1.07	-0.042	-23.34	+1.91	-0.057	-37.46	+0.63	-0.061	-40.22
	Cactus	+0.52	-0.041	-33.18	+1.01	-0.038	-24.55	+1.11	-0.042	-35.42	+0.61	-0.065	-42.87
	ParkScene	+0.61	-0.054	-34.34	+0.96	-0.074	-22.37	+1.08	-0.057	-27.05	+0.82	-0.074	-40.64
Class C	RaceHorses	+0.64	-0.046	-39.08	+0.87	-0.035	-21.67	+1.72	-0.038	-36.42	+0.69	-0.071	-44.23
	BQMall	+0.59	-0.044	-32.35	+0.85	-0.042	-20.83	+0.89	-0.032	-28.28	+0.72	-0.062	-41.63
	PartyScene	+0.41	-0.035	-30.16	+0.88	-0.056	-23.28	+0.22	-0.021	-25.52	+0.78	-0.078	-40.35
Class D	BasketballPass	+0.53	-0.048	-37.55	+0.78	-0.054	-17.22	+1.23	-0.041	-34.13	+0.63	-0.064	-40.51
	BlowingBubbles	+0.42	-0.037	-32.56	+1.03	-0.062	-20.47	+0.34	-0.021	-26.57	+0.78	-0.065	-41.05
	BQSquare	+0.57	-0.039	-32.47	+0.86	-0.048	-20.54	+0.76	-0.030	-28.62	-0.71	-0.073	-43.62
Class E	Fourpeople	+0.62	-0.056	-40.04	+1.24	-0.046	-16.65	+1.67	-0.054	-38.63	+0.72	-0.071	-42.23
	Johnny	+0.79	-0.035	-36.75	+0.89	-0.039	-19.48	+1.84	-0.032	-37.25	+0.65	-0.058	-40.36
	KristenAndSara	+0.69	-0.041	-31.78	+0.91	-0.057	-20.06	+1.58	-0.043	-31.39	+0.86	-0.061	-41.21
Class F	BasketballDrillText	+0.52	-0.037	-35.80	+1.21	-0.043	-21.82	+1.46	-0.035	-35.63	+0.57	-0.052	-42.87
	Slideshow	+0.71	-0.046	-33.87	+1.32	-0.053	-20.34	+1.22	-0.048	-33.95	+0.51	-0.054	-41.52
	ChinaSpeed	+0.68	-0.041	-32.35	+1.18	-0.061	-19.78	+1.37	-0.036	-30.10	+0.91	-0.069	-40.93
Average		+0.61	-0.043	-34.67	+1.01	+0.511	-20.44	+1.25	-0.041	-32.31	+0.69	-0.070	-41.59

From Table 6, we can clearly see that our proposed method in this paper can save about 34.67% coding time only at a cost of 0.61% BDBR increase and 0.043db BDPSNR decline compared with the standard reference HM16.1algorithm. Chen's scheme can save about 41.59% coding time saving, but with the increase of 0.69% BDBR and the decline of 0.070db BDPSNR respectively. Huade's scheme can achieve about 32.31% coding time saving with a cost of 1.25% BDBR increase and 0.041db BDPSNR decline. Ha's scheme show the worst experimental performance, only saving about 20.44% coding time, but it increases about 1.01% BDBR and lower 0.051db BDPSNR.

Fig.12 to Fig 13 is the testing results on BDBR in different QP value and test sequences respectively. Fig 14 to Fig.15 is the experimental results on BDPSNR in different QP value and test sequences respectively. Fig.16 to Fig 17 are the testing results on TS in different QP value and test sequences respectively. From Fig.12 to Fig.13, we can clearly observe that our proposed method can achieve the least increase of BDBR under different QP value and test sequence compared with standard reference HM16.1, only increasing about 0.61% BDBR. Chen's scheme follows, which increase about 0.69% BDBR. Ha's scheme and Huade's increases the more BDBR performance, reaching about 1.01% BDBR and 1.25% BDBR for HEVC respectively. Fig.14 and Fig.15 present that Huade's scheme can achieve the least reduction of BDPSNR in four methods compared with standard reference HM16.1 algorithm, only decrease about 0.041db BDPSNR. Our proposed method and Ha's scheme follows, which cut down about 0.043db BDPSNR and 0.051db BDPSNR for HEVC respectively. Chen's scheme shows the worst BDPSNR performance, reducing about 0.070 db BDPSNR for HEVC. Fig.16 to Fig.17 show that Chen's method can achieve the best saving performance in TS in four methods under the same testing conditions compared with standard reference HM16.1algorithm, reaching about 41.59% coding time saving. Our proposed scheme and Huade's scheme follows, which can save about 34.67% coding time and 32.31% coding time saving for HEVC respectively. Ha's scheme shows the worst saving performance of coding time, only saving about 20.44% coding time for HEVC.

The main reasons for them are that, in our proposed method, coding bits of current CU are used to reduce many unnecessary prediction and calculation operations in RDO process for HEVC, therefore saving about 34.67% computational complexity for HEVC with a cost of 0.61% BDBR increase and 0.043db BDPSNR decline compared with the standard reference HM16.1 method. Chen's scheme can save about 41.59% coding time due to the use of depth range

prediction and mode reduction in its scheme, but it needs to cost about 0.69% BDBR and 0.070 db BDPSNR. Huade's scheme uses the adaptive depth selection technology to reduce the unnecessary calculation operations in

RDO process, thus saving about 32.31% coding time saving rate for HEVC, with a cost of 1.25% BDBR increase and 0.041db BDPSNR decline. Ha's scheme can eliminate many partition modes for rate distortion optimization (RDO) in its scheme by the use of texture similarity of neighborhood CUs in its scheme, which only can save about 20.44% coding time for HEVC, but it increases about 1.01% BDBR and decreases about 0.0511db BDPSNR.

From Fig.12 through 17 above, we can also observe the BDBR, BDPSNR and TS in four algorithm presents to slightly rise with the increase of QP value, which shows that four methods need to cost more BDPSNR and BDPSNR and less TS in high lower QP value compared to lower QP value under the same testing conditions. The main reason for them are that, one the one hand, the four CU size decision algorithms above have a different impact on different QP value, thus having different BDBR, BDPSNR and TS performance compared to the standard reference HM16.1 algorithm; on the other hand, the four different methods has different processes to the texture information in different test sequence, thus lead to different BDBR, BDPSNR and TS performance compared to the standard reference HM16.1 algorithm.

Discussion

Although this method can reduce lots of computational complexity with less BDBR increase and BDPSNR decline for HEVC compared with the standard reference of HM16.1, it is not suitable for some high real-time embedded devices. This is because these devices require lower power consumption. However, as a fast CU size decision optimal algorithm, our proposed uses the thresholds of coding bit to quickly determine the optimal partition mode for the current CU, which can reduce many unnecessary partition and prediction operations, thereby saving much computational complexity for HEVC and suiting for some general real-time embedded devices.

Conclusion And Future Work

In this paper, a fast CU size decision optimal algorithm based on coding bits is proposed for HEVC intra prediction. In our scheme, we firstly set the corresponding relationship between coding bits and partition size for the current CU based on the relationship among content complexity and its partition size and coding bits; We then use the thresholds of coding bit to quickly determine the optimal partition mode for the current CU, which can reduce many unnecessary partition and prediction operations, thereby saving much computational complexity for HEVC. The simulation results show that our proposed fast CU size decision algorithm in this paper can save about 34.67% computational complexity only at a cost of 0.61% BDBR increase and 0.043db BDPSNR decline for HEVC compared with the standard reference of HM16.1. We plan to continue our future research in the following two directions. Firstly, we will extend our proposed method to suit the processing for inter CU size prediction. Secondly, we will incorporate Bayes' theory and machine learning in our proposed to improve its whole performance for HEVC.

Declarations

Acknowledgment

The work was supported by the National Natural Science Foundation of China(No.61602187) and (No.61601189), the National Key Research and Development Plan (No.2016YFD0200700); the science and technology projects in Guangdong Province (No.2016A020209007) and (No.2016A020210088) and(No.2015A010103014), the project of Natural Science Foundation of Guangdong Province (No. 2016A030310453)

Authors’contributions

Jianhua Wang, Zhihao Chen and Feng Lin found the idea and wrote this article, Jing Zhao, Yongbing Long, Yubin Lan put forward some constructive suggestions for revision. The authors read and approved the final manuscript.

Availability of data and materials

Not applicable.

Ethics approval and consent to participate

Not applicable.

Consent for publication

Not applicable.

Competing interests

The authors declare that they have no competing interests.

Abbreviations

HEVC: High Efficiency Video Coding; CTU :Coding Tree Unit; CU: Coding Unit; LCU: Large Coding Unit; QP: Quantitative Parameter; MPEG: Moving Picture Experts Group; VCEG: Video Coding Experts Group; HD: High Definition; UHD : Ultra High Definition; RDCost :Rate-Distortion Cost; CU :Coding Unit; DC: Direct

Current ; DTV: Discretization Total Variation; SATD :Sum of Absolute Transformed Differences; CNN :Convolution Neural Network; BDBR: Bjontegaard Delta Bit Rate; BDPSNR :Bjontegaard Delta Peak Signal-to-Noise Ratio; TS :Coding Time Saving;

References

1. J. Sullivan, J. Ohm, W.-J. Han, and T. Wiegand, Overview of the High Efficiency Video Coding (HEVC) standard, IEEE Trans. Circuits Syst. Video Technol, 22 , 1649–1668, (2012)
2. Han, J. Min, I. Kim, E. Alshina, A. Alshin et al., Improved VideoCompression Efficiency through Flexible Unit Representation and Corresponding Extension of Coding Tools, IEEE Trans CircuitSystem for Video Technology, 20(12),1709-1720, (2010)
3. J. Sullivan and J. R. Ohm, Recent developments in standardization of high efficiency video coding (HEVC), SPIE, 7798 77980V, (2010)
4. Vanne J, Viitanen M, Hämäläinen T D. Efficient mode decision schemes for HEVC inter prediction. IEEE Transactions on Circuits and Systems for Video Technology, 24(9), 1579-1593. (2014)
5. Lu Y, Zhang Q, Wei B. Real-time CPU based H. 265/HEVC encoding solution with x86 platform technology. IEEE 2015 International Conference on Computing, Networking and Communications, pp.418-421, (2015)
6. T. Pourazad, C. Doutre, M. Azimi, and P. Nasiopoulos, HEVC: The new gold standard for video compression: How does HEVC compare with H.264/AVC?. IEEE Consum. Electron. Mag, 1(3), 36–46, (2012)
7. Architectural Outline of Proposed High Efficiency Video Coding Design Elements, document JCTVC-A202, ITU-T SG16 WP3 and ISO/IEC JTC1/SC29/WG11, in Proc of 1st Meeting, Dresden, Germany, Apr. 2010.
8. T Zhang, M T Sun, D Zhao, et al. Fast Intra Mode and CU Size Decision for HEVC. IEEE Transactions on Circuits and Systems for Video Technology, 27 (8), 1714-1726, (2017).
9. J M Ha, J H Bae, M H Sunwoo. Texture-based fast CU size decision algorithm for HEVC intra coding. In proc of 2016 IEEE Asia Pacific Conference on Circuits and Systems, pp.702-705. (2016).
10. X Zhou, G Shi, W Zhou. Perceptual CU Size Decision and Fast Prediction Mode Decision Algorithm for HEVC Intra Coding. In proc of 2016 IEEE International Symposium on Multimedia, 375-378, (2016)
11. Y Song, Y Zeng, X Li, et al. Fast CU size decision and mode decision algorithm for intra prediction in HEVC. Multimedia Tools and Applications, 76, 2001-2017, (2017).
12. D Ruiz, G Fernández-Escribano, V Adzic, et al. Fast CU partitioning algorithm for HEVC intra coding using data mining. Multimedia Tools and Applications, 76 ,861-894, (2017).
13. Y F Cen, W L Wang, X W Yao. A fast CU depth decision mechanism for HEVC. Information Processing Letters, 115(9), 719-724, (2015)
14. Zhang Q, Zhao J, Huang X, et al. A fast and efficient coding unit size decision algorithm based on temporal and spatial correlation. Optik-International Journal for Light and Electron Optics, 126(21) ,2793-2798, (2015).
15. X Shang, G Wang, T Fan, et al. Fast CU size decision and PU mode decision algorithm in HEVC intra coding. Proc. 2015 IEEE International Conference on Image Processing. pp.1593-1597, 2015.
16. L Shen, Z Zhang, X Zhang, et al. Fast TU size decision algorithm for HEVC encoders using Bayesian theorem detection. Signal Processing: Image Communication, 32 ,121-128, (2015).
17. S Ahn, B Lee, M Kim. A novel fast CU encoding scheme based on spatiotemporal encoding parameters for HEVC inter coding. IEEE Transactions on Circuits and Systems for Video Technology, 25 ,422-435, (2015)
18. J Lee, S Kim, K Lim, et al. A fast CU size decision algorithm for HEVC. IEEE Transactions on Circuits and Systems for Video Technology, 25, 411-421, (2015).
19. Shen, L. Yu, and J. Chen, Fast coding unit size selection for HEVC based on Bayesian decision rule, in proc of Picture Coding Symp, pp.453–456,(2012)
20. Chen and L. Yu. Effective HEVC intra coding unit size decision based on online progressive Bayesian classification. In Proceedings of the IEEE International Conference on Multimedia and Expo. (ICME'16), pp.1–6, (2016)
21. Liu, Y. Li, D. Liu, P. Wang, and L. T. Yang. 2019. An adaptive CU size decision algorithm for HEVC intra prediction based on complexity classification using machine learning. IEEE Trans. Circ. Syst. Video Technol. 29(1),144–155, (2019).
22. Liu, X. Yu, Y. Gao S. Chen, X. Ji, and D.Wang. 2016. CU partition mode decision for HEVC hardwired intra encoder using convolution neural network. IEEE Trans. Image Proc. 25(11) ,5088–5103, (2016).
23. Xu, T. Li, Z.Wang, X. Deng, R. Yang, and Z. Guan.F. Bossen, Common Test Conditions and Software Reference Configurations, document JCTVC-G1200, Geneva, Switzerland, Feb. 2011.
24. [Y. Zhang, Z. Pan, N. Li, X. Wang, G. Jiang, and S. Kwong, 2018. Effective data driven coding unit size decision approaches for HEVC intra coding. IEEE Trans. Circ. Syst. Video Technol. 28 (11) ,3208–3222, (2018).
25. JCT-VC. Subversion Repository for the HEVC Test Model Version HM16.1. [Online]. Available: https://hevc.hhi.fraunhofer.de/svn/svn_HEVCSoftware/tags/HM-16.1/. (2016).
26. Huade S, Fan L, Huanbang C. A fast CU size decision algorithm based on adaptive depth selection for HEVC encoder. 2014 International Conference on Audio, Language and Image Processing. IEEE, pp.143-146, (2014)
27. Chen F, Jin D, Peng Z, et al. Fast intra coding algorithm for HEVC based on depth range prediction and mode reduction. Multimedia Tools and Applications, 77(21) 28375-28394, (2018)

Figures

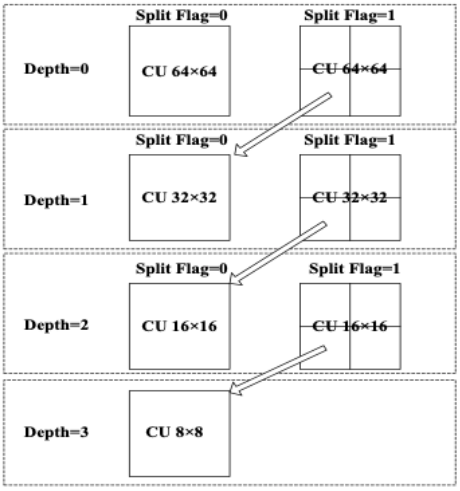

Figure 1

Partitioning process for a LCU with 64×64 size

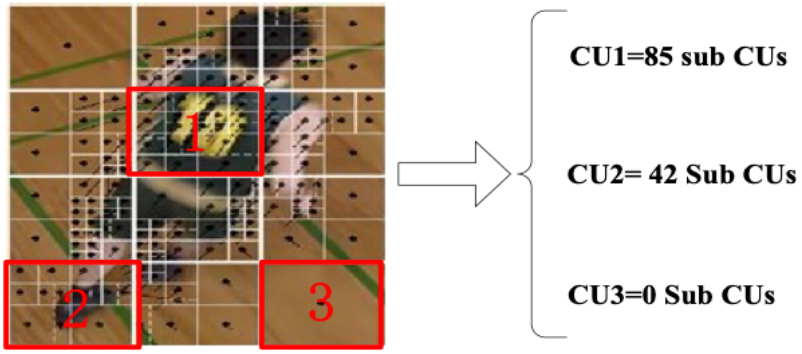

Figure 2

Partition number of sub CUs in different texture complexity of CU in BasketballDrill test sequence

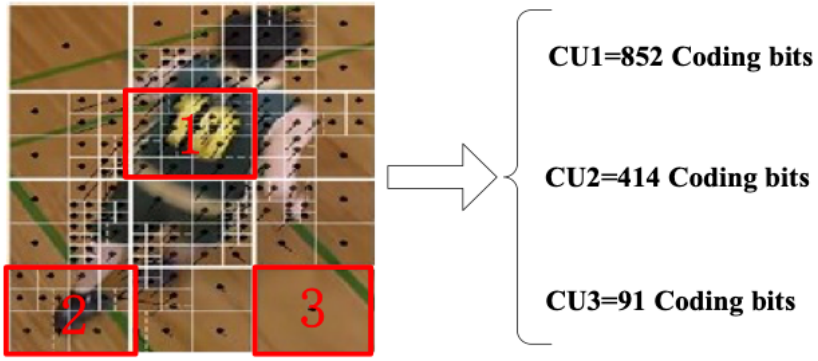

Figure 3

Number of coding bits in different texture complexity of CU in a BasketballDrill test sequence

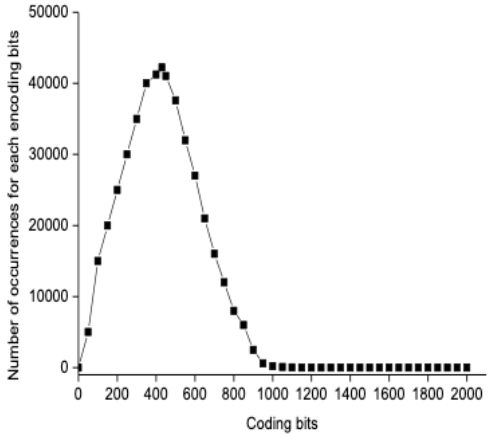

Figure 4

Coding bits and number of its occurrences (CU32×32, QP =22)

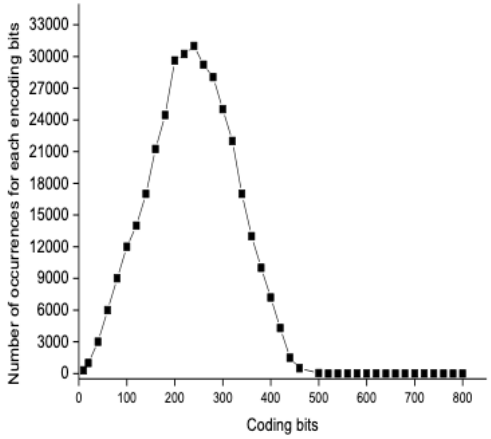

Figure 5

Coding bits and number of its occurrences (CU32×32, QP =27)

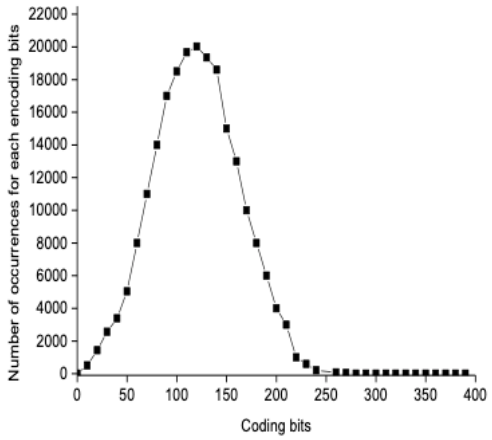

Figure 6

Coding bits and number of its occurrences (CU16×16 QP =32)

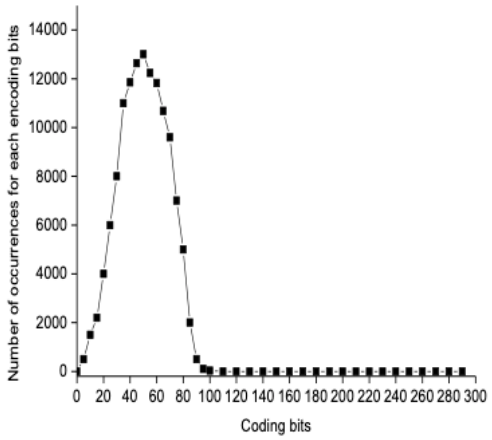

Figure 7

Coding bits and number of its occurrences (CU8×8 QP =37)

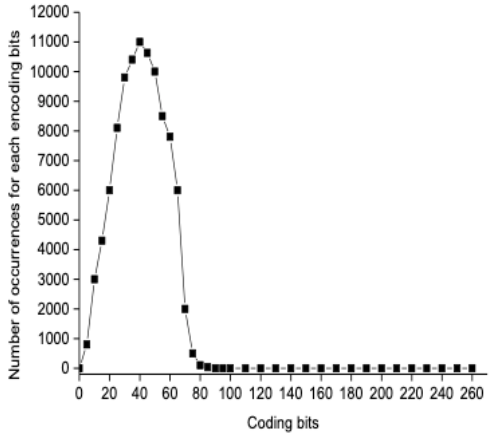

Figure 8

Coding bits and number of its occurrences (CU16×16 QP =32)

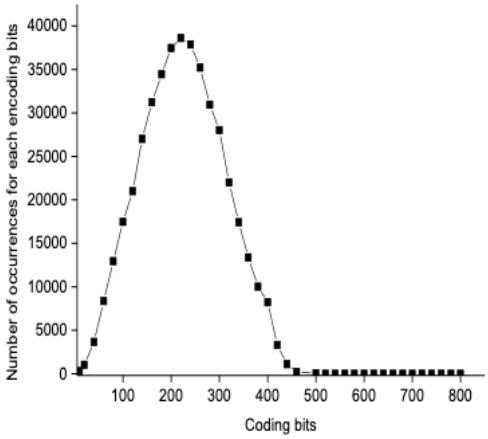

Figure 9

Coding bits and number of its occurrences (CU64×64 QP =32)

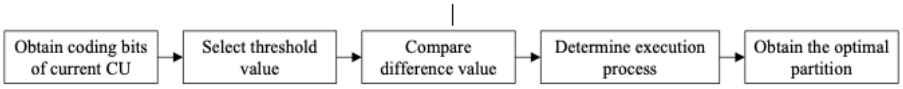

Figure 10

The realizing process of our proposed scheme

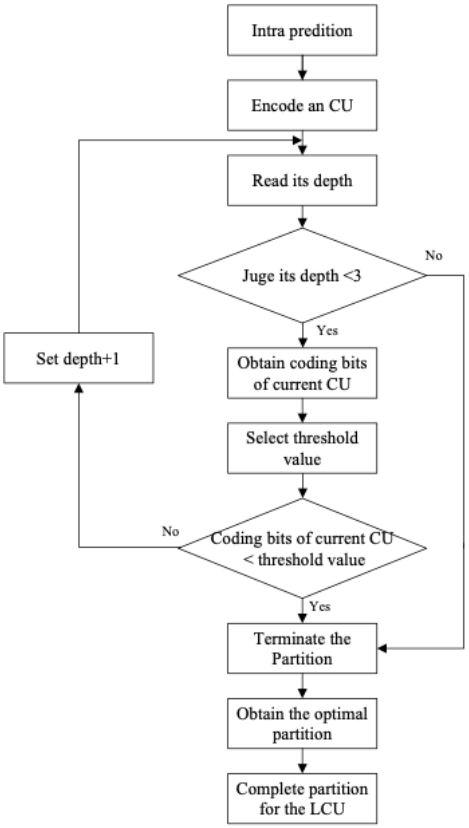

Figure 11

Realizing flow of our proposed scheme

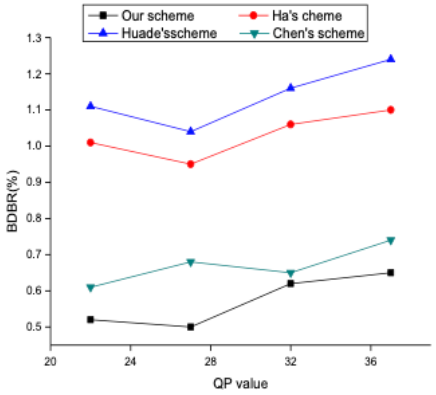

Figure 12

BDBR in different QP value for PeopleOnstreet sequence

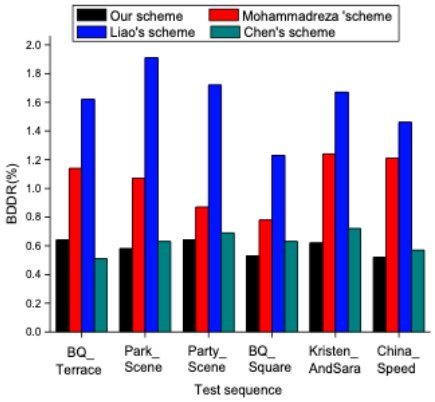

Figure 13

BDBR in six different test sequence

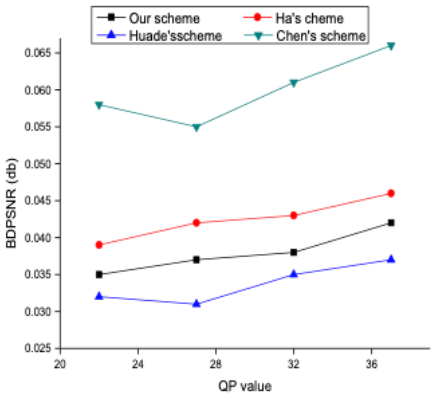

Figure 14

BDPSNR in different QP value for Johnny sequence

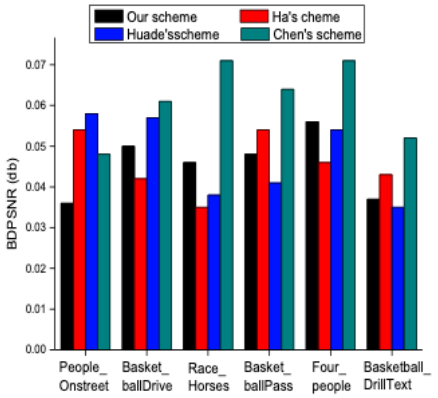

Figure 15

BDPSNR in six different test sequence

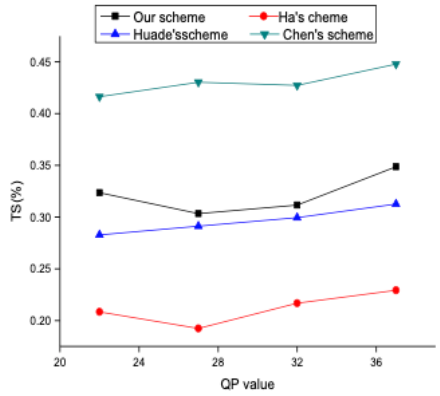

Figure 16

TS in different QP value for BQMall sequence

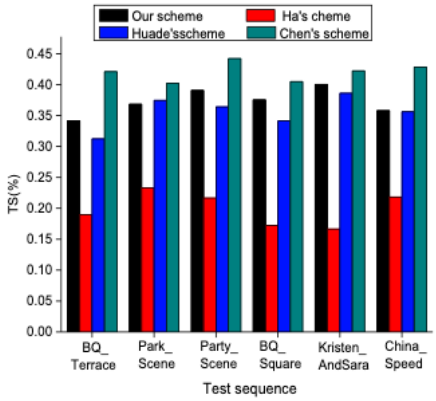

Figure 17

TS in six different test sequence