

A Fast CU Decision Algorithm in Inter Coding Based on Residual Analysis

Lanfang Dong^(✉)  and Li Wang 

School of Computer Science and Technology,
University of Science and Technology of China, Hefei, China
lfdong@ustc.edu.cn, wanglia@mail.ustc.edu.cn

Abstract. Inter coding in HEVC can greatly improve video compression efficiency. However it also brings huge computational cost due to adaptive partition of Coding Tree Unit (CTU) with the quadtree technique. In this paper, a fast CU (Coding Unit) decision algorithm is proposed to alleviate the computational burden. This algorithm is fulfilled by analyzing the residual using mean and dispersion. And the optimal RQT (Residual Quad-tree Transform) depth is used innovatively to measure the residual dispersion. First, the optimal RQT depth and $avgdis$ (defined in 3.3) is obtained after inter coding in $2N \times 2N$ PU mode. Then the decision of CU partition is determined through comparing $avgdis$ with the corresponding threshold. Thresholds are predicted based on the distribution of CU partition in the encoded pictures and they can be adaptively changed as the video content changes. Compared to HM13.0 (HEVC test model), the improved algorithm could save about 56% of encoding time on average, with 0.2034% increase of bitrate and the influence on the quality of reconstructed videos is negligible.

Keywords: HEVC · Inter coding · CU partition · Residual · Dispersion

1 Introduction

More and more popular video applications and the increasing video resolution have brought the huge bandwidth consumption and the high storage cost. The existing video compression standard H.264/AVC has been unable to satisfy the high-definition and ultra-high-definition video. Joint Collaborative Team on Video Coding (JCT-VC) [1], established by ITU-T/VCEG and ISO-IEC/MPEG, released the video coding standard HEVC (High Efficiency Video Coding), and the standard is still evolving.

Compared to H.264/AVC, under the similar video quality, HEVC can save 50% of the video stream [2]. However, different from the fixed macroblock of H.264, HEVC uses the quadtree technique to realize adaptive partitioning of CTU which will traverse all possible combinations to determine the best partitioning, greatly increasing the computational complexity of video coding.

HEVC coding techniques can be divided into intra prediction and inter prediction. The compression rate of inter prediction several times that of intra prediction, the main reason of which lies in that inter prediction requires finding the most matching block of PU (Prediction Unit) in the reference frame. According to the relevant statistics, inter

prediction accounts for more than 60% of the overall coding time. Therefore, the acceleration of inter prediction is of great significance.

In this paper, a fast CU decision algorithm for inter prediction based on residual analysis is proposed. The residual is analyzed by mean and dispersion, and its optimal RQT depth is used innovatively to measure the residual dispersion. At first we statistically analyze the relationship between mean and dispersion and CU partition. Then the determination threshold of the current frame is predicted based on the distribution of CU partition of the threshold image. At last unnecessary CU calculations are terminated by threshold comparison. Experiments show that the proposed algorithm can significantly reduce the computational complexity of inter prediction without losing the video quality and compression ratio.

The remainder of this paper is organized as follows. Section 2 gives a brief introduction of background knowledge. Section 3 describes the proposed algorithm in detail. Experimental results are shown in Sect. 4. At last Sect. 5 concludes this paper.

2 Background Knowledge

2.1 Brief Introduction of Inter Coding in HEVC

Inter coding can eliminate the redundancy in time domain to achieve the purpose of video compression. For inter coding, current block would find its best matching block by motion estimation in the reference frame as the prediction block. The difference between the matching block and the current block is called the residual.

By using the quadtree technique, the CU with depth d can be divided into four sub-CUs in depth $d + 1$. And decision-making for partition is determined by the rate distortion of the CU and the sub-CUs. The definition of RD (Rate Distortion) can be seen in Formula (1), where D represents the distortion in the current prediction mode and R is the number of bits required to encode the prediction mode. Figure 1(a) is an example of CTU partition, in which the white indicates that CU continues to be divided and the black stands for stopping CU partition.

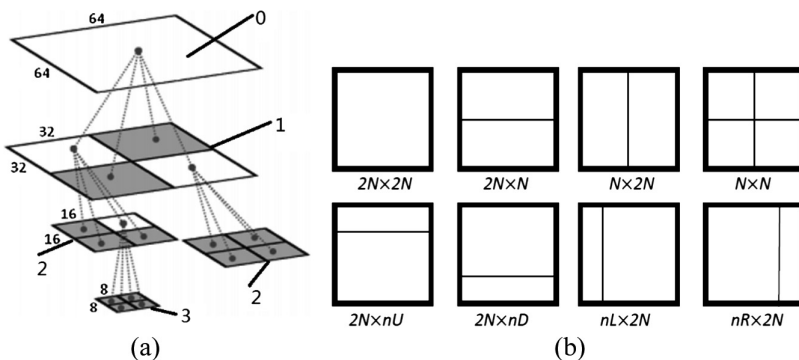


Fig. 1. (a) One example of CTU partition (b) PU partition modes

In HEVC the CU will be further divided into PUs which are basic units of inter prediction. There are eight types of PU partition, containing four types of symmetric partition and four types of asymmetric partition [3], as shown in Fig. 1(b). HEVC will test all the PU partition modes for CU, and ultimately select the optimal PU partition with the minimum RD cost.

$$\text{RDCost} = D + \lambda R \quad (1)$$

TU (Transform Unit) is the basic unit for transform and quantization. Similar to CU partition, TU partition can be flexibly fulfilled by quadtree technique in a recursive manner [4].

2.2 Related Work

HEVC inter coding can be accelerated in three aspects: CU level, PU level and TU level.

In the CU level, accelerating can be realized by skipping unnecessary CU partition tests or terminating CU partition earlier. The paper [5] first got the depth range of CU partition based on visual saliency map detection, and then CU partition was determined by the comparison of coding bits and the specific threshold. A RD cost based accelerative algorithm was proposed by [6], where the decision of CU partition was made by comparing RD cost of the current CU with the specific threshold. Paper [7] proposed a fast algorithm determining the depth range of CU partition, which made a comprehensive utilization of zero CU detection, standard deviation of statistic spatiotemporal depth information and edge gradient of CTU. Paper [8] introduced an accelerating algorithm of inter encoding aiming at surveillance videos. Inter-frame difference was utilized to segment out moving objects encoding in CU of small size from background encoding in CU of large size. Wu et al. presented a fast inter encoding algorithm aiming at H.264 in [9] which determined the size of the coding block according to spatial homogeneity based on edge information and temporal stationary characteristics judged by MB differencing. The paper [10] constructed a feature vector for each CU consisting of RD cost and motion category, according to which, the decision of CU partition was determined.

The acceleration algorithm in TU level can predict TU partition by some means. The paper [11] could adaptively determine the optimal RQT depth according to spatial neighboring blocks and the co-located block at the previously coded frame. The experimental results showed that the improved algorithm could save 60% to 80% of encoding time. A fast decision algorithm in TU level based on Bayesian model was introduced in [12] which took the depths of current TU, upper TU, left TU and co-located TU as reference features. TU partition could be terminated earlier in [13] by taking advantage of the feature of all coefficients of TU being zero. The paper [14] created three decision trees for CU, PU and TU respectively obtained through data mining techniques. Decision trees for CU and TU decided whether to terminate the partition of CU and TU and the decision tree of PU determined whether to test the rest of PU partition modes.

In paper [15], an improved algorithm in PU level for HEVC inter prediction was proposed whose core idea was to reduce motion estimation calls. The CU was divided into four blocks of the same size, and then the PU partition was determined by comparing the motion vectors of the four PU blocks. Gao et al. in [16] calculated a quadtree probability model based on a quantization parameter and a group of pictures and then a new quadtree was constructed by skipping low probability tree nodes.

Algorithms above have achieved a good acceleration effect with minimal impact on video quality and the increment of bitrate on average is also in a reasonable range. However, the increment of bitrate between different videos has a big difference in this paper, we propose a fast CU decision algorithm based on residual analysis in which no extra calculation is required and intermediate results are fully utilized. CU splitting decision is fulfilled by thresholds comparing and the thresholds can adaptively be adjusted as the video content changes. Experimental results prove that the improved method can significantly reduce encoding time.

3 Fast CU Decision Algorithm Based on Residual Analysis

3.1 Residual Analysis

Generally speaking, for the same image region, compared to encoding in large-size CU, small-size CU can achieve better matching effect with smaller prediction error but larger coding bits. If the matching performance of current CU in $2N \times 2N$ PU mode is in good condition, coding bits become larger and prediction error reduces slightly after CU is divided, leading to larger RD cost. So current CU is not appropriate to be divided into smaller CUs. On the contrary, if the matching performance is too bad, current CU has significant reduction of prediction error after being divided, bringing smaller RD cost. Further CU partition should be performed.

The residual is the result after motion compensation which directly reflects the matching performance of the current block. Therefore the decision of CU partition can be made by analyzing the residual.

In this paper, dispersion and mean of residuals are utilized to analyze residuals. Dispersion refers to the degree of difference between values in residuals. And mean is the average of all data items in residuals.

3.2 How to Measure the Dispersion

There are two commonly used methods for measuring the dispersion:

- Standard deviation: Calculate standard deviation of data items in the residual.
- Edge detection: Edge detection such as Sobel operator or Canny operator is operated on the residual block to extract edges. And then calculate the number of nonzero values in the edge map.

Both of the two methods require additional computation, and they also need to set thresholds between uniform and non-uniform artificially. Here we make use of the optimal RQT depth of current CU to measure dispersion of residuals creatively.

After inter prediction, the residual block is imputed into the transform module for further compression. In the transform module, DCT (Discrete Cosine Transform) is performed on the residual block, which can remove the correlation between data items in the residual block.

If the residual block is in uniform distribution, DCT coefficients that have been quantized remain a small amount of larger value with most of them being zero, resulting in lower coding bits and smaller RD cost. Then current TU will not be divided into smaller TUs, as shown in Fig. 2(a). On the other hand, if the residual has rich texture, there are many larger values and a small amount of smaller values, leading to more coding bits and larger RD cost. Therefore current TU can get more zero DCT coefficients by further TU partition, as shown in Fig. 2(b). According to the local variation characteristics of the predicted residuals, TU can adaptively select the optimal TU partition.

2	6	5	2	2	-1	-2	-1	137	125	35	-17	3	1	11	-7
2	8	5	1	0	-5	-1	-1	136	133	34	-43	-52	15	22	-6
5	1	4	2	1	-1	2	1	135	136	43	-38	-57	23	25	2
7	2	5	3	1	8	5	0	135	135	47	-17	-39	54	19	4
4	9	6	7	6	5	4	4	135	142	59	7	4	49	16	-11
1	11	5	7	7	5	1	2	135	139	58	5	-3	31	54	-14
-5	8	6	2	3	5	1	1	135	139	63	2	11	21	69	4
-2	5	1	-1	6	3	4	4	132	136	57	-5	19	0	56	38

(a) (b)

Fig. 2. (a) TU partition of residual block in uniform distribution (b) TU partition of residual block in rich texture

Based on the above analysis, we can see that the optimal RQT depth has a close relationship with dispersion. The deeper the optimal RQT depth is, the greater dispersion is. Thus the optimal RQT depth can be used to measure dispersion directly. The maximum depth of TU can be set in the configuration file, default as three. In this paper, the maximum partition depth of TU adopts the default value.

3.3 The Relationship Between CU Partition and Dispersion and Mean

As described in Sect. 3.2, dispersion is measured using optimal RQT depth denoted as $opRdepth$ which has three cases: 0, 1 and 2. The optimal RQT depth doesn't require additional computation which is the intermediate result. In order to simplify the calculation, mean is replaced by average value of ssd (short for $avgdis$) produced by HEVC, which is the sum of the squared error of original pixels and reconstructed pixels.

$$D_{split} = f(opRdepth, avgdis, CUsize) \quad (2)$$

We can get $opRdepth$ and $avgdis$ of current CU after inter encoding in $2N \times 2N$ PU mode is fulfilled. Then the splitting decision D_{split} can be represented as a function of $opRdepth$, $avgdis$ and CU size.

In order to determine the mapping relationship f , we make a mass of statistics on $opRdepth$, $avgdis$ and its corresponding CU splitting decision. Each video has 50 frames to participate in statistics. For example, detailed statistical results of BasketballDrill can be seen in Fig. 3, where the horizontal axis stands for the number of CU and the vertical axis is $avgdis$. The green dots mean the current CU need to be divided further while the red dots mean not. We can see that there is a boundary between red dots and green dots. According to statistical results of numerous videos, we propose a fast CU decision algorithm denoted as $CUDecision$ which can be seen in Table 1. TH_bottom_{ij} and TH_up_{ij} is the threshold for CU splitting decision, where i represents the depth of current CU and j is $opRdepth$.

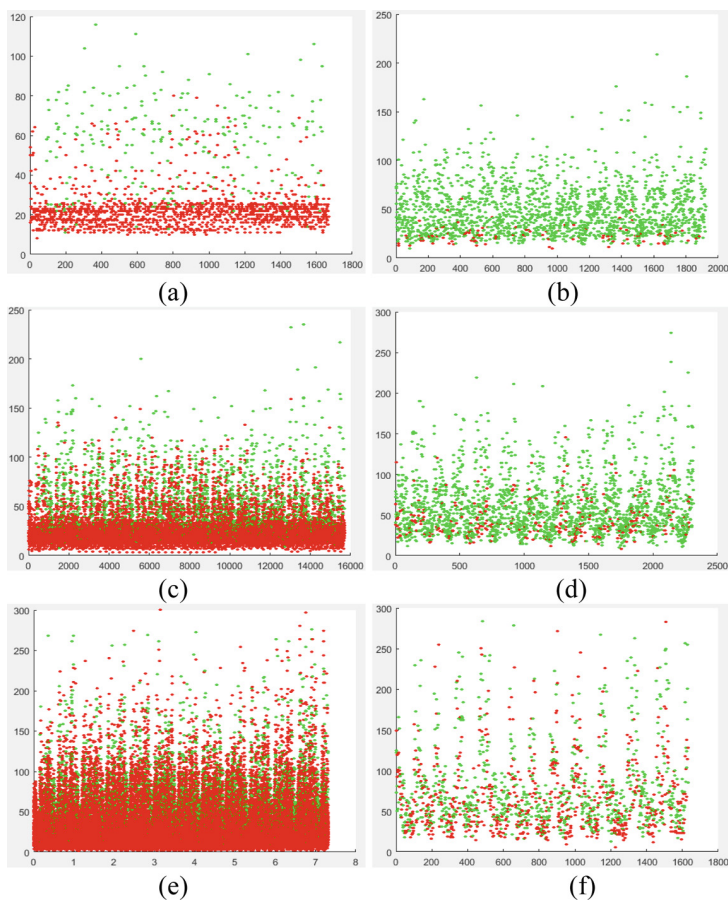


Fig. 3. BasketballDrill (a) $CUsize = 64$ $opRdepth = 0$ (b) $CUsize = 64$ $opRdepth = 2$ (c) $CUsize = 32$ $opRdepth = 0$ (d) $CUsize = 32$ $opRdepth = 2$ (e) $CUsize = 16$ $opRdepth = 0$ (f) $CUsize = 16$ $opRdepth = 2$ (Color figure online)

Table 1. Detailed process of CUDecision

1. Perform inter coding in $2N \times 2N$ PU mode on the current CU, and then $opRdepth$ and $avgdis$ is obtained.
2. If $opRdepth$ is zero, jump to 3. If not, jump to 5.
3. If $avgdis < TH_{bottom_{i0}}$, splitting of CU should be terminated.
4. If $avgdis > TH_{up_{i0}}$, skip calculation of current CU and calculate CUs in next depth. If not, jump to 7.
5. If $opRdepth$ is two, jump to 6. If not, jump to 7.
6. If $avgdis > TH_{up_{i2}}$, skip calculation of current CU and calculate CUs in next depth.
7. Process CU partition with standard procedures.

3.4 Determination of Thresholds

Due to the similarity of image content, CU partition between adjacent images has a strong correlation. Thus TH_{up} and TH_{bottom} in current frame can be predicted from previous frames.

A data structure denoted as drawer is defined for describing CU partition. It contains two variables, red and green, which are arrays with a size of 60. red_i is the i -th element of red, representing the number of not divided CUs with $avgdis$ in the range of $5i$ to $5(i + 1)$. The meaning of green is similar to that of red.

Given a drawer, TH_{up} and TH_{bottom} can be obtained by threshold calculating methods $CalTHup$ and $CalTHbottom$ respectively. The detailed process of $CalTHbottom$ is listed in Table 2 and $CalTHup$ has a flow similar to $CalTHbottom$.

Table 2. Detailed process of $CalTHbottom$

1. $i=0$.
2. If $i \geq 60$, jump to 9.
3. $red_num = \sum_{j=i}^{i+3} red_j, green_num = \sum_{j=i}^{i+3} green_j$.
4. $total_num = red_num + green_num$.
5. If $red_num/total_num \geq 0.9$, $TH_bottom = (i+4)*5$ and $i=i+4$ and jump to 2.
6. $red_sub = red_i + red_{i+1}, green_sub = green_i + green_{i+1}$.
7. $total_sub = red_sub + green_sub$.
8. If $red_sub/total_sub \geq 0.9$, $TH_bottom = (i+2)*5$. If not, $TH_bottom = (i+1)*5$.
9. Return TH_bottom .

The image that serves to predict TH_{up} and TH_{bottom} is called $THimage$. $PredictSet$ is a set of $THimages$ defined as follows where P_i is the POC (Picture Order

Count) of corresponding THimage. The size of PredictSet equals to GOP (Group Of Pictures) size which is set to four in our experiments.

$$\text{PredictSet} = \{P_i | i = 0, 1, 2, 3\} \tag{3}$$

Updating PredictSet. RD1 in Eq. (4) is RD cost of current CU which is not divided and RD2 in Eq. (5) is the corresponding RD cost of divided CU. With Eq. (4) and Eqs. (5) and (6) is deduced. What's more, λ is positively correlated with the QP (Quantization Parameter), as shown in Eq. (7) where α and W_k are weighting factors related to different configurations. Accordingly we can conclude that QPs will make a difference on distribution of CU partition. The HEVC encoder sets a series of QP values for each frame in a GOP, and each GOP has the same QP configuration. Therefore, THimages in PredictSet should contain all QP values in one GOP. In our experiments, QP values are set to be 35, 34, 35 and 33 in one GOP. A PredictSet updating algorithm named UpdatePed is proposed to initialize and update PredictSet, as shown in Table 3.

$$\text{RD1} = D1 + \lambda R1 \tag{4}$$

$$\text{RD2} = D2 + \lambda R2 \tag{5}$$

$$\lambda > = \frac{D2 - D1}{R1 - R2} \tag{6}$$

$$\lambda = \alpha \cdot W_k \cdot 2^{\frac{(QP-12)}{3}} \tag{7}$$

Table 3. Detailed process of UpdatePed

- | |
|---|
| <ol style="list-style-type: none"> 1. The first frame with POC=0 is encoded in intra coding. 2. CurPOC=1. 3. The frame is encoded in inter coding using initial TH_up and TH_bottom. 4. CurPOC= curPOC+1. 5. If curPOC<=4, jump to 3. 6. PredictSet = {1, 2, 3, 4} and TH_poc=5. 7. If CurPOC< TH_poc, the frame is encoded using updated TH_up and TH_bottom and jump to 12. 8. If (CurPOC = TH_poc), the frame is encoded using initial TH_up and TH_bottom. 9. The minimum value in PredictSet is replaced by TH_poc. 10. Update TH_up and TH_bottom using PredictSet. 11. TH_poc= TH_poc+5. 12. CurPOC= CurPOC+1. 13. Jump to 7. |
|---|

Threshold Calculation. Each THimage $P_i (i = 0, 1, 2, 3)$ in PredictSet has its corresponding drawer set D_i defined in Eq. (8) where d_{ijk} is the drawer of CUs in depth j with opRdepth being k . Dsum is the sum of D_i , defined in Eq. (9). $TH_{up_{j0}}$ and $TH_{up_{j2}}$ can be obtained by inputting $dsum_{j0}$ and $dsum_{j2}$ into CalTHup respectively.

$$D_i = \{d_{ijk} | j = 0, 1, 2; k = 0, 2\} \tag{8}$$

$$Dsum = \{(dsum_{jk} = \sum_{i=0}^3 d_{ijk}) | j = 0, 1, 2; k = 0, 2\} \tag{9}$$

$TH_{up_{j0}}$ and $TH_{up_{j2}}$ are predicted without considering the influence of QPs. Differing from prediction of TH_up, the calculation of TH_bottom takes QPs into consideration and each QP value has its corresponding prediction threshold. $TH_{GOPbottom_{ijk}}$ and $TH_{QPbottom_{ijk}}$ are defined for i -th QP value where j is the depth of CUs and k is opRdepth. In fact, we just need to calculate $TH_{GOPbottom_{ij0}}$ and $TH_{OPbottom_{ij0}}$.

Inputting $dsum_{j0}$ into CalTHbottom can get $TH_{bottom_{j0}}$ and Inputting d_{ij0} can output $TH_{QPbottom_{ij0}}$. $TH_{GOPbottom_{ijk}}$ is the optimal threshold selected from them. In most cases, $TH_{bottom_{j0}}$ is an appropriate prediction threshold. However if $TH_{QPbottom_{ij0}}$ is greater more than $TH_{bottom_{j0}}$, maybe $TH_{QPbottom_{ij0}}$ is more accurate (see Fig. 4.), which is caused by QP difference. According to derivation process of CU partition in last section, we can see the larger the QP value is, the greater the threshold is. Similarly the larger the QP value is, the more not divided TUs are. Therefore the QP value is measured by the number of CUs with opRdepth being zero denoted as $CUNum_i$. Thus the optimal threshold can be chosen through $CUNum_i$.

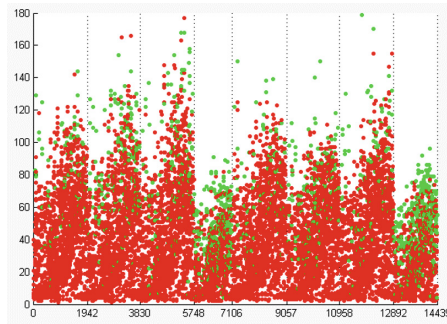


Fig. 4. CU partition of BQTerrace in depth one

Based on the above analysis, the method of threshold calculation denoted as CalTHpred describes the flow for calculating $TH_{GOPbottom_{ijk}}$, shown in Table 4.

Table 4. Detailed process of CalTHpred

<ol style="list-style-type: none"> 1. Compute TH_bottom_{j0} and $TH_QPbottom_{ij0}$. 2. Calculate $CUnum_i$. 3. $avgCUnum = (\sum_{i=0}^3 CUnum_i)/4$. 4. $TH_GOPbottom_{ij0} = TH_bottom_{j0}$. 5. If $TH_OPbottom_{ij0} - TH_bottom_{j0} \geq 10$ and $CUnum_i > avgCUnum$, $TH_GOPbottom_{ij0} = TH_QPbottom_{ij0}$.
--

4 Experiments and Results

The proposed algorithm is implemented on HM13.0 and the coding performance can be evaluated by $\Delta PSNR$ (Peak Signal to Noise Ratio), ΔBR (Bitrate), and ΔET (Encoding Time) defined in (10) to (12) under lowdelay P configuration. PSNR is calculated by Eq. (13). In our experiments, 50 frames are tested for each video.

$$\Delta PSNR = \frac{PSNR_{proposed} - PSNR_{HM}}{PSNR_{HM}} \quad (10)$$

$$\Delta BR = \frac{BR_{proposed} - BR_{HM}}{BR_{HM}} \quad (11)$$

$$\Delta ET = \frac{ET_{proposed} - ET_{HM}}{ET_{HM}} \quad (12)$$

$$PSNR = (PSNR_Y \times 4 + PSNR_U + PSNR_V)/6 \quad (13)$$

Table 5 shows the acceleration effect of the proposed algorithm. Through careful observation and analysis, we can get the following points.

1. The improved algorithm could save about 56% of encoding time on average, with 0.2034% increase of bitrate and 0.0423 reduction in PSNR.
2. If the video content changes slowly such as Flowervase, Mobisode2 and so on, the proposed method can reduce execution time by about 70% with a small change in bitrate and PSNR. In some cases, the bitrate is even reduced.
3. If the video content changes quickly such as BasketballDrill, Tennis and so on, the proposed algorithm can reduce execution time by 36%–50% and the increment of bitrate is a little higher than that of small changing videos.
4. The PSNR of SlideShow decreases by 0.1164% which is greatest in test sequences and the increment of bitrate reaches 1.4575%, a little higher than that of other videos.
5. The increment of bitrate is limited within 1.11%. This proves that the proposed algorithm can guarantee compression ratio when it realizes accelerating.

Table 5. Results of the proposed algorithm compared to HM13.0

Resolution	Video name	Δ BR (%)	Δ PSNR	Δ ET (%)
832 * 480	BasketballDrill	1.1178	-0.0329	-48.4424
	BasketballDrillText	0.9064	-0.0466	-49.4051
	FlowerVase	-1.0959	-0.0163	-70.7357
	Mobisode2	0.3659	-0.0402	-67.5394
	RaceHorsesC	0.5669	-0.0158	-39.2287
	BQMall	0.4097	-0.0331	-47.4201
	Keiba	0.6921	-0.0273	-43.8054
	PartyScene	0.2589	-0.0178	-36.2514
1024 * 768	ChinaSpeed	-0.5957	-0.0443	-50.0416
1280 * 720	FourPeople	-0.1044	-0.0323	-72.5145
	Johnny	-0.6339	-0.0479	-75.0793
	SlideShow	1.4575	-0.1164	-74.6874
	vidyo1	-0.3152	-0.0243	-74.6151
	KristenAndSara	-0.4875	-0.0456	-73.0945
1920 * 1800	Cactus	0.3813	-0.0202	-53.8501
	Kimono1	0.8674	-0.0174	-52.4435
	BQTerrace	-1.3989	-0.0231	-53.4527
	ParkScene	0.0071	-0.0206	-49.1677
	Tennis	0.9690	-0.0222	-49.8077
2560 * 1600	PeopleOnStreet	0.8509	-0.0145	-39.8343
	Traffic	0.0522	-0.0298	-57.8289
Average		0.2034	-0.0423	-56.1545

We can find that the video with small changes can achieve more reduction of computational complexity. That's because CUs in the small changing video can get good matching performance and then terminates CU partition earlier. For the video with fast content change, prediction of TH_{up} and TH_{bottom} is less accurate, leading to more bitrate increasing.

SlideShow has much more sudden changes and sometimes image content is completely different between adjacent frames. Under this circumstance, thresholds are not accurate, contributing to slightly larger image distortion and more bitrate increasing. In fact, due to big difference between adjacent frames, inter coding is not suitable for SlideShow to realize compression.

5 Conclusion

For one CTU, HEVC will traverse all possible divisions with the quadtree technique to make the coding efficiency optimized. However, the exhaustive search process greatly aggravates the computational burden. A fast CU decision algorithm based on residual analysis is proposed to predict whether current CU will be divided and then skip redundant computation process. The residual is analyzed by mean and dispersion, and

then the measure of dispersion is fulfilled by the optimal RQT depth. When the optimal RQT depth equals to 0 or 2, the decision of CU partition will be determined by comparing $avgdis$ with the corresponding threshold. What's more, the proposed algorithm could realize adaptive changes of thresholds for different videos.

Experimental results show that the proposed algorithm can achieve good performance on speeding up calculation without affecting compression ratio and the quality of reconstructed videos.

References

1. Duanmu, F., Ma, Z., Wang, Y.: Fast mode and partition decision using machine learning for intra-frame coding in HEVC screen content coding extension. *IEEE J. Emerg. Sel. Topics Circuits Syst.* **6**(4), 517–531 (2016)
2. Correa, G., Assuncao, P., Agostini, L., et al.: Performance and computational complexity assessment of high-efficiency video encoders. *IEEE Trans. Circuits Syst. Video Technol.* **22** (12), 1899–1909 (2012)
3. McCann, K., Rosewarne, C., Bross, B., et al.: High efficiency video coding (HEVC) test model 16 (HM 16) improved encoder description. *JCT-VC*, Technical report JCTVC-S 1002 (2014)
4. Sullivan, G.J., Ohm, J., Han, W.J., et al.: Overview of the high efficiency video coding (HEVC) standard. *IEEE Trans. Circuits Syst. Video Technol.* **22**(12), 1649–1668 (2012)
5. Qing, A., Zhou, W., Wei, H., et al.: A fast CU partitioning algorithm in HEVC inter prediction for HD/UHD video. In: *Signal and Information Processing Association Annual Summit and Conference*, pp. 1–5. IEEE Press, Pacific (2016)
6. Wang, J., Dong, L., Xu, Y.: A fast inter prediction algorithm based on rate-distortion cost in HEVC. *Int. J. Sig. Process. Image Process. Pattern Recogn.* **8**(11), 141–158 (2015)
7. Liu, J., Jia, H., Xiang, G., et al.: An adaptive inter CU depth decision algorithm for HEVC. In: *Visual Communications and Image Processing*, pp. 1–4. IEEE Press, Singapore (2015)
8. Wang, J., Dong, L.: An efficient coding scheme for surveillance videos based on high efficiency video coding. In: *10th International Conference on Natural Computation*, pp. 899–904. IEEE Press, Xiamen (2014)
9. Wu, D., Wu, S., Lim, K.P., et al.: Block inter mode decision for fast encoding of H.264. In: *IEEE International Conference on Acoustics, Speech, and Signal Processing*, pp. iii–181. IEEE Press, Canada (2004)
10. Mallikarachi, T., Fernando, A., Arachchi, H.K.: Fast coding unit size selection for HEVC inter prediction. In: *IEEE International Conference on Consumer Electronics*, pp. 457–458. IEEE Press, Taiwan (2015)
11. Wang, C.C., Liao, Y.C., Wang, J.W., et al.: An effective TU size decision method for fast HEVC encoders. In: *International Symposium on Computer, Consumer and Control*, pp. 1195–1198. IEEE Press, Taiwan (2014)
12. Wu, X., Wang, H., Wei, Z.: Bayesian rule based fast TU depth decision algorithm for high efficiency video coding. In: *Visual Communications and Image Processing*, pp. 1–4. IEEE Press, Chengdu (2016)
13. Fang, J.T., Tsai, Y.C., Lee, J.X., et al.: Computation reduction in transform unit of high efficiency video coding based on zero-coefficients. In: *International Symposium on Computer, Consumer and Control*, pp. 797–800. IEEE Press, Xi'an (2016)

14. Correa, G., Assuncao, P.A., Agostini, L.V., et al.: Fast HEVC encoding decisions using data mining. *IEEE Trans. Circuits Syst. Video Technol.* **25**(4), 660–673 (2015)
15. Sampaio, F., Bampi, S., Grellert, M., et al.: Motion vectors merging: low complexity prediction unit decision heuristic for the inter-prediction of HEVC encoders. In: *IEEE International Conference on Multimedia and Expo*, pp. 657–662. IEEE Press, Australia (2012)
16. Gao, Y., Liu, P., Wu, Y., et al.: Quadtree degeneration for HEVC. *IEEE Trans. Multimedia* **18**(12), 2321–2330 (2016)