



A fast CU partition algorithm based on sum of region-directional dispersion for virtual reality 360° video

Mengmeng Zhang¹ · Yuhao Wang¹ · Zhi Liu¹ · Zhao Wang²

Published online: 2 September 2020
© Springer-Verlag GmbH Germany, part of Springer Nature 2020

Abstract

Virtual reality 360° video has an ultra-high resolution (usually 4–8 K), which takes more coding time than traditional video. Considering the degree of horizontal stretching in a different area of the ERP projected video, a fast coding unit (CU) partition algorithm based on the sum of region-directional dispersion is proposed. The relationship between the current block and its adjacent ones is measured based on the degree of horizontal stretching, and the current frame is divided into three regions. A new metric named the Sum of Region-directional Dispersion is defined to measure the complexity of the current block in different regions and directions. In the proposed algorithm, the optimal fast partition threshold is determined for each region to achieve the goal of early termination partition. Compared with the original reference software HM16.20, the proposed algorithm can reduce the time in virtual reality video coding by 38.5%, while the BD-rate only increases by 0.40%.

Keywords 360° video · The sum of region-directional dispersion · Fast CU partition

1 Introduction

As consumer-level virtual reality (VR) devices are officially introduced to the market, users' demand for full three-dimensional and immersive experience of virtual reality technology continues to grow. At present, the content of virtual reality is mainly based on video game experience. It can be divided into two types, one is pure virtual scene video which is created by game engines such as Unity or Unreal; the other is 360° video obtained by shooting natural scene through camera arrays [1]. Among them, the content of virtual reality 360° video has grown rapidly. Google and Facebook have not only launched the platform of 360° video but also released their own 360 panoramic cameras; it is foreseeable that 360° video will become a novel content carrier and occupy an important position in the field of virtual reality.

The applications of 360° video mainly include acquisition, splicing, encoding, transmission and playback. The acquisition of 360° video requires multiple cameras to complete simultaneously. After the synchronous acquisition, the video captured by multiple cameras should be spliced. The images taken at different angles are not on the same projection plane, and the seamless stitching of overlapping images will destroy the visual consistency of scene [2, 3]. Therefore, it is necessary to perform projection transformation on the images before stitching. Projection transformation refers to the mapping process of expanding a three-dimensional scene onto a two-dimensional plane. As a carrier of panoramic content, the projected image should not only contain the entire content of the shooting but also avoid the loss of playback quality caused by excessive distortion. Several projection methods are used in the process of JVET coding standard, including ERP Projection, CMP Projection and SSP Projection. Currently, the most widely used projection method in 360° video is ERP. ERP is a simple projection method that maps longitude lines to vertical lines with constant spacing and latitude lines to horizontal lines with constant spacing, as shown in Fig. 1. The projection has good compatibility with natural images, and even the projected 2D video is very intuitive and convenient for users to observe. However, the same number of sampling points are used for each latitude line of spherical video, which leads to

✉ Mengmeng Zhang
zmm@ncut.edu.cn

✉ Zhi Liu
lzliu@ncut.edu.cn

¹ North China University of Technology, Beijing, China

² Institute of Information Science, Beijing Jiaotong University, Beijing, China

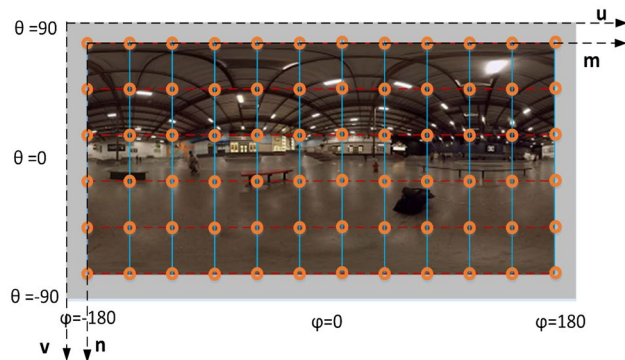


Fig. 1 ERP format projection

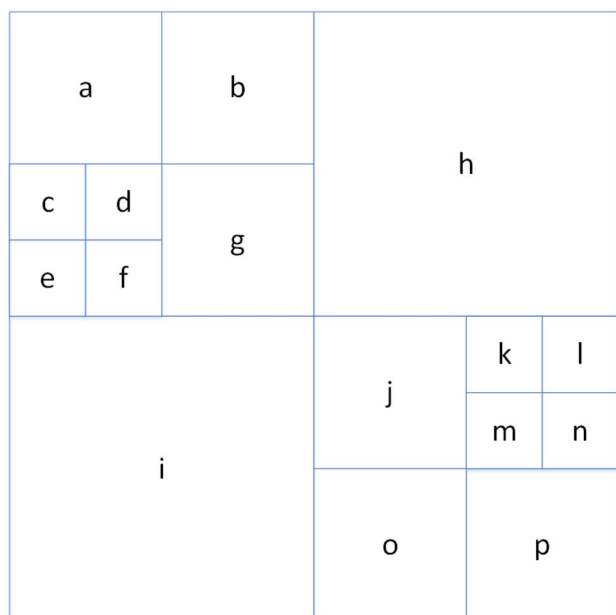


Fig. 2 The partition of CTU

more redundant sampling points for latitude line closer to the pole [4]. Therefore, the ERP projection method increases the space occupied by the video which leads to higher distortion.

In the traditional HEVC coding framework, processing a frame of video image requires first dividing it into multiple CTUs (CTU is usually 64×64 in size), and then encoding each CTU in turn. Each CTU is divided recursively in turn, through calculating encoding cost of the current depth to determine whether to continue the division. CTU can be divided into units as small as 8×8 , as shown in Figs. 2 and 3. The encoder determines whether to continue the division by comparing the rate distortion cost (RD-cost) value of the depth. If the total RD-cost of four sub-CUs in the next depth is greater than the current CU, the encoder does not continue the division. Otherwise, it continues to divide until the end

of the division. The traditional method has a high coding complexity, so many people are committed to the research of fast partitioning algorithm.

In the fast partition of CU in traditional HEVC coding, Li et al. [5] proposed a fast CU partition algorithm based on depth spatial correlation and rate distortion. The algorithm uses the strategy of early partition and early termination to divide CU and estimates the candidate depth range of CU using the depth correlation of spatial neighboring CUs. The algorithm can reduce the encoding time by 43.37% and increase BD-rate by 0.61%. Chen et al. [6] proposed a fast intra algorithm based on depth range prediction and decreasing prediction mode. The algorithm analyzes the pattern information of the previous frame to obtain a new feature and establishes a new model with the depth range of coding tree unit (CTU) to skip needless segmentation. The algorithm can reduce the time by 43.2% and increase BD-rate by 0.47%. Liao et al. [7] proposed a two-part fast intra algorithm, which first uses CTU depth spatial correlation and Bayesian decision rules to accelerate CU partitioning, and then skips unnecessary mode according to the current CU depth information and direction information. The algorithm reduces encoding time by 52.9%. Although these studies adopted different methods to accelerate CU partitioning, they can not combine well with the characteristics of 360° video in ERP format. Therefore, this paper studies the relationship between the current block and its adjacent ones based on the degree of horizontal stretching in three regions, thus the fast CU partition algorithm based on the sum of region-directional dispersion for 360° video is proposed.

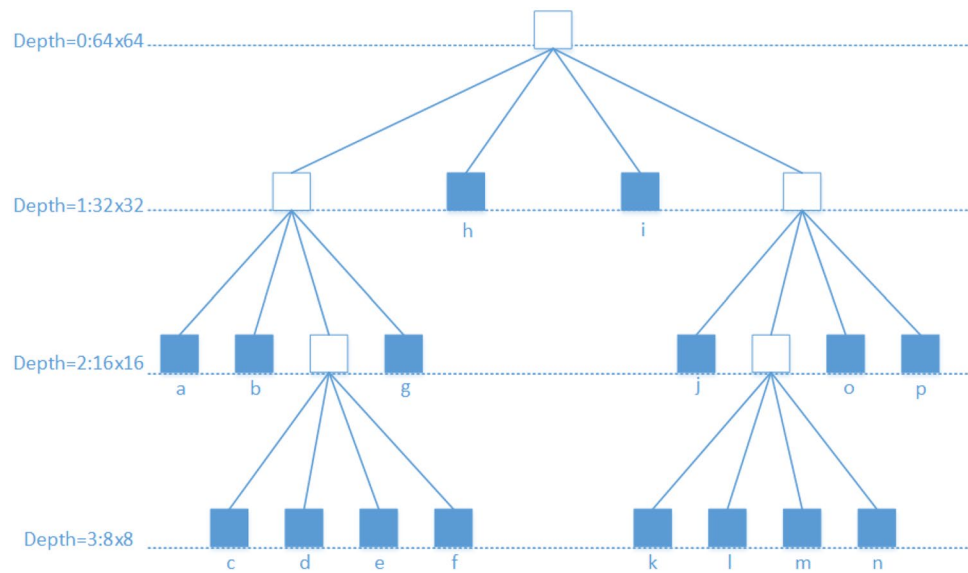
The rest of the paper is organized as follows. The second part introduces related work. The third part explains the specific steps of the proposed algorithm in detail. The experimental results are shown in the fourth part, and the conclusion is given in the fifth part.

2 Related work

The optimization research on 360° video coding of virtual reality is mainly carried out from the aspects of projection format, region filtering optimization, fast coding, transmission optimization and subjective-objective evaluation.

In terms of the projection format of 360° video in virtual reality, Duan et al. [8] proposed an adjustment cube projection (ACP) format and introduced transformation parameters to improve sampling uniformity, aiming at the non-uniformity of CMP projection sphere sample density distribution. Yu et al. [9] combined convolutional neural network (CNN) with classical interpolation method to convert the cube into ERP projection format, and derived the optimal threshold of the boundary according to its geometric features, so as to enhance the conversion performance. He et al. [10] proposed

Fig. 3 Quad-tree partition



a novel hybrid cube projection (HCP) to improve the efficiency of 360° video coding. It conducted adaptive sampling in the horizontal and vertical directions within each cube surface. Compared with the ERP projection format, its BD-rate decreased by 8.0%. Sreedhar et al. [11] studied a variety of virtual reality video projection schemes including pyramid projection, and proposed a multi-resolution encoding method based on ERP and CMP.

In terms of the regional filtering optimization of 360° video in virtual reality, Sauer et al. [12] proposed a method to reduce artifacts while encoding 360° video. The geometric correction of deblocking filtering makes the correct geometric pixels used for edge filtering. This method has little effect on coding performance, and visual quality is significantly enhanced.

In the aspect of fast coding of 360° video in virtual reality, Storch et al. [13] divided the 360° video into the equator and the polar region according to the severe stretching of the polar region under ERP projection format. It used the horizontal direction mode to predict the polar region and reduced encoding time by 16.5% while BD-rate is basically unchanged. Zhang [14] et al. proposed a fast CU partition algorithm based on 360° video weighted mean square error (WMSE). The algorithm used WMSE as the basis to calculate the similarity between the current CU and the sub-CU and to carry out early termination partition of CU. The algorithm reduced encoding time by 31%, and the BD-rate only increased by 0.3%. Guan et al. [15] proposed a fast intra algorithm for CU partitioning and mode selection under the HEVC coding framework, which reduced encoding time by 54% and increased BD-rate by only 1.4%. Li et al. [16] considered the optimal rate–distortion relationship in the spherical domain and analyzed its influence on the rate–distortion optimization process, and derived the optimal rate–distortion

relationship in the spherical domain. Finally, they proposed the optimal solution based on HEVC/H.265. Excellent solution. Wang et al. [17] proposed a fast intra algorithm based on the characteristics of ERP format video. It saved about 24.5% encoding time in full intra-frame mode.

In the optimization of 360° video transmission, Yang et al. [18] proposed a 360° video adaptive transmission method. They established a 360° video region priority division scheme which based on users' visual features and head motion characteristics. Different priority regions send different quality content as block formats, and then make a 360° video adaptive transmission decision which based on the bandwidth estimation. Fuihashi et al. [19] proposed a transmission scheme that skipped quantization and entropy coding. The scheme directly transmitted the combined linear transform signal based on 3D discrete cosine transform and wavelet cosine transform, which improved the transmission quality of 360° video.

In terms of subjective–objective evaluation of 360° video, Hanhart et al. [20] conducted different experiments to illustrate the influence of different projection formats and codecs on subjective–objective quality based on the 360° quality framework established by Joint Video Exploration Group (JVET) of ITU-T VCEG and ISO/IEC MPEG.

3 Fast CU partition algorithm based on the sum of region-directional dispersion

In 360° video coding, spherical video is projected into 2D rectangular format. Due to the simple mapping relationship of ERP format, most test sequences are stored in ERP format and compressed by video encoding. For this type of

projection, the degree of horizontal stretching becomes more and more severe from the equator region to the polar region. According to this feature, this paper firstly counts the depth relationship between current CU and adjacent CUs, then divides the different regions according to the weight of WS-PSNR, and finally calculates the sum of region-directional dispersion in each region and obtain the threshold for early termination partition of CU.

3.1 Region division of 360° video

In view of the different horizontal stretching degree of different regions in the 360° video under ERP projection format, we consider whether there is a method suitable for CU fast partition in this format.

First, we count the depth of each 8×8 CU and its adjacent upper, upper-left and left CUs. The significance of the statistical depth is that if the number of CU depth the same as adjacent CU in a certain direction is more, it shows that the reference significance of CU partition in this direction is relatively bigger. Here, taking the first frame of the test sequence Gaslamp as an example. The sequence is 8 K video, and one frame has 524,288 8×8 CUs. The samples are sufficient to reflect the depth relationship between current CU and adjacent CUs of different directions in different regions. Here we intend to divide the image into the polar region, the mid-latitude region and the equator region. The number of CU depth the same as adjacent CUs in each region is shown in Table 1.

It can be seen from the table that the depth relationship between current CU and adjacent CUs in different regions has different characteristics. Here we introduce the depth matching ratio (the coincidence rate of the current CU and adjacent CUs at each depth) to better represent this feature. If the depth of current CU and adjacent CU is the same, it indicates that the adjacent CU has a greater significance for CU partition. Meanwhile, to further explore the relationship between current CU and adjacent CUs in different regions, the depth matching ratios in three regions are more intuitively represented in the form of the histogram shown in Fig. 4.

It can be seen from the statistical results that in the polar region, the depth matching ratio between current CU and left CU is the highest, and upper CU is the next; in the mid-latitude region, upper-left CU has the highest

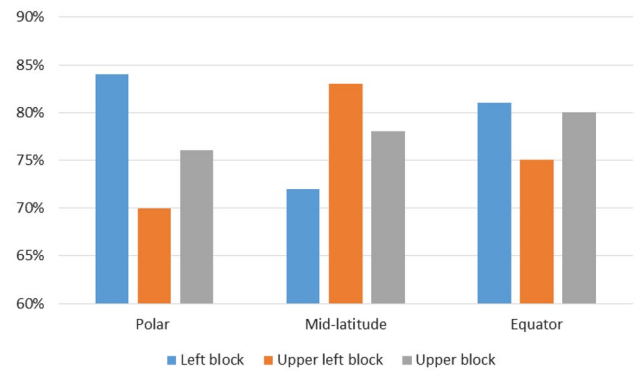


Fig. 4 Depth matching ratio for different regions

depth matching ratio with current CU, and upper CU is the next; in the equator region, left CU and upper CU have basically the same depth matching ratio with current CU, and are higher than upper-left CU. To divide three regions more accurately, this paper divides the equator region, the mid-latitude region and the polar region by calculating the weight w in the WS-PSNR. The formula of the weight w is as follows:

$$w(i,j) = \cos \left[\left(j - \frac{N}{2} + \frac{1}{2} \right) \times \frac{\pi}{N} \right], \quad (1)$$

where N is the height of the CTU, and j is the height of the pixel location (ranging from 0 to the image height).

The region determination algorithm is as follows: Take LCU as the unit, calculate the sum of ownership weights in each LCU in a column, and record it as w_0 . Take w_0 as the weight of each LCU.

The experiment shows that w_0 takes 0.3 and 0.8 as the boundary values of three regions. When $0 < w_0 < 0.3$, it is judged as the polar region. When $0.3 < w_0 < 0.8$, it is judged as the mid-latitude region. When $0.8 < w_0 < 1$, it is judged as the equator region. Considering the maximum size of CU in the process of encoding is 64×64 pixels, where 64 pixels is taken as one line. The 4 K sequence has a total of 26 lines. Here we divide the image into three areas: lines 1–3 and 24–26 of the image are tentatively defined as the pole area, lines 4–8 and 19–23 is designated as the mid-latitude region and lines 9–18 are designated as the equator area. The dividing range of the three regions is shown in Fig. 5.

The depth matching ratio between current CU and its adjacent ones is counted by the size of 8×8 , and the relationship between each CU and adjacent CUs in different areas is found. And the division of CU starts from 64×64 CU and then decreases successively to 8×8 CU. Therefore, the index of depth matching ratio can be used to quickly divide CU and improve coding efficiency.

Table 1 The number of CU depth the same as adjacent CUs

Region	Left CU	Upper-left CU	Upper CU
Polar	440,402	377,487	424,673
Mid-latitude	367,001	435,159	393,216
Equator	398,459	408,944	419,430

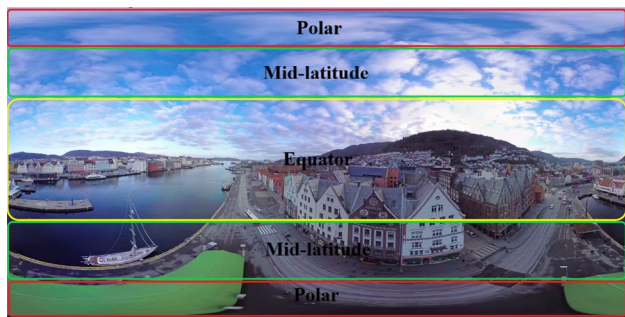


Fig. 5 The division of region

3.2 Definition of the sum of region-directional dispersion

After deriving the depth matching ratio between current 8×8 CU and adjacent ones in different regions, it can be used to divide CU from the largest size of 64×64 coding to the smallest size of 8×8 . The method that can better reflect image feature is to extract the texture information. It can reflect the homogeneity of the image and the arrangement property that changes slowly or periodically in the image. Currently, the methods widely used for texture feature extraction are LBP (local binary pattern), Gray level co-occurrence matrix, Gray-gradient co-occurrence matrix and Gabor wavelet texture. These methods have the advantages of rotation invariance and gray invariance, they can not meet the needs of different sizes and frequency textures, in other words they can't adapt well to different sizes of CU. The traditional calculation texture complexity uses variance as the basis for judgment. The formula is as follows:

$$\text{Var}(x) = \frac{\sum_{x=1}^n [i(x) - \bar{m}]^2}{n - 1}, \tag{2}$$

where

$$\bar{m} = \frac{\sum_{x=1}^n i(x)}{n}. \tag{3}$$

However, due to the high complexity of variance calculation, this paper uses the sum of region-directional dispersion (SRD) as an index to calculate the texture change of the image. It can characterize the degree to which a random variable value (such as a set of data) deviates or spreads about a central value (usually taken as a mathematical expectation). Calculating SRD can characterize the texture change trend of the image in a different direction.

Considering that the depth relationship between current CU and its adjacent CUs in a different region of 360° video is different, we calculate the sum of region-directional dispersion from four direction: Horizontal, Vertical, Up Horizontal

and Down Horizontal. The angle between UH direction, DH direction and the horizontal direction is close to 30 degree. The four calculation directions are shown in Fig. 6. The SRD calculation formula of the image is as follows:

$$\text{SRD} = \frac{\sum_{y=0}^{M-1} \sum_{x=0}^{M-1} |p(x, y) - \bar{m}|}{M^2}, \tag{4}$$

where x and y represent the x th row and the y th column of the image, respectively, and $p(x, y)$ represents the pixel luminance value of the current CU at the position (x, y) . The sum of region-directional dispersion in four directions is represented by SRD_{Hor} , SRD_{Ver} , SRD_{UH} and SRD_{DH} , respectively. The up horizontal (UH) direction stands for Mode 6 in the angle prediction mode. The down horizontal (DH) direction stands for Mode 14 in the angle prediction mode. There is no difference in the order of calculating SRD in four direction.

The following describes the calculation method of SRD in four direction. The calculation of SRD in the vertical direction is performed using 4-pixel strips shown in Fig. 6(a). The value in 4×4 block is calculated as follows:

$$\bar{m}_{\text{Ver}}(n) = \frac{1}{N} \sum_{i=0}^{N-1} P(i, n), n = 0, 1, 2, 3, \tag{5}$$

$$\text{SRD}_{\text{Ver}}(n) = \frac{1}{N} \sum_{i=0}^{N-1} |P(i, n) - \bar{m}_{\text{Ver}}(n)|, \tag{6}$$

$$\text{SRD}_{\text{Ver}} = \frac{1}{4} \sum_{n=0}^3 \text{SRD}_{\text{Ver}}(n). \tag{7}$$

The calculation of SRD in the horizontal direction is performed using four-pixel strips shown in Fig. 6b. The value in 4×4 block is calculated as follows:

$$\bar{m}_{\text{Hor}}(n) = \frac{1}{N} \sum_{i=0}^{N-1} P(n, i), n = 0, 1, 2, 3, \tag{8}$$

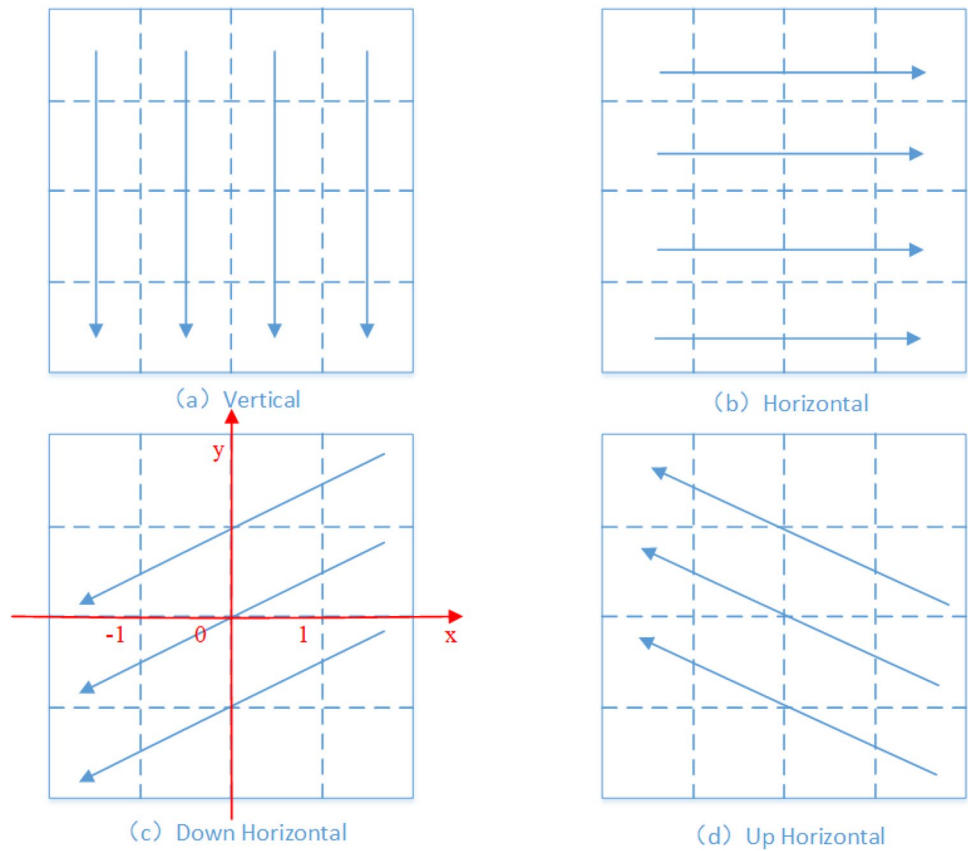
$$\text{SRD}_{\text{Hor}}(n) = \frac{1}{N} \sum_{i=0}^{N-1} |P(n, i) - \bar{m}_{\text{Hor}}(n)|, \tag{9}$$

$$\text{SRD}_{\text{Hor}} = \frac{1}{4} \sum_{n=0}^3 \text{SRD}_{\text{Hor}}(n). \tag{10}$$

The calculation of SRD in the down horizontal direction is performed using 3 pixel strips shown in Fig. 6c. The mid-point of each pixel strip is taken as the origin to establish coordinate axis. The value in 4×4 block is calculated as follows:

$$\bar{m}_{\text{DH}}(n) = \frac{1}{N} \left[\sum_{x=-\frac{N}{2}}^{-1} P(x, -1) + \sum_{x=1}^{\frac{N}{2}} P(x, 1) \right], n = 0, 1, 2, \tag{11}$$

Fig. 6 Computational direction of SRD



$$SRD_{DH}(n) = \frac{1}{N} \left[\left| \sum_{x=-\frac{N}{2}}^{-1} P(x, -1) - \bar{m}_{DH}(n) \right| + \left| \sum_1^{\frac{N}{2}} P(x, 1) - \bar{m}_{DH}(n) \right| \right], \quad (12)$$

$$SRD_{DH} = \frac{1}{3} \sum_{n=0}^2 SRD_{DH}(n). \quad (13)$$

The calculation of SRD in the up horizontal direction is performed using 3 pixel strips shown in Fig. 6d. Similarly, the midpoint of each pixel strip is taken as the origin to establish coordinate axis. The value in 4×4 block is calculated as follows:

$$\bar{m}_{UH}(n) = \frac{1}{N} \left[\sum_{x=-\frac{N}{2}}^{-1} P(x, 1) + \sum_1^{\frac{N}{2}} P(x, -1) \right], n = 0, 1, 2, \quad (14)$$

$$SRD_{UH}(n) = \frac{1}{N} \left[\left| \sum_{x=-\frac{N}{2}}^{-1} P(x, 1) - \bar{m}_{UH}(n) \right| + \left| \sum_1^{\frac{N}{2}} P(x, -1) - \bar{m}_{UH}(n) \right| \right], \quad (15)$$

$$SRD_{UH} = \frac{1}{3} \sum_{n=0}^2 SRD_{UH}(n). \quad (16)$$

where N represents the size of the block, its value is 4 in this place; $P(x, y)$ represents the pixel luminance value of the x th row and the y th column in the block; SD_{Ver} , SD_{Hor} , SD_{UH} , SD_{DH} represent the sum of dispersion in the vertical direction, the horizontal direction, the up horizontal direction and the down horizontal direction in the current block. It should be noted that it reflects the texture trend of the image in different directions. The smaller its value is, the smoother the texture of the image in this direction will be. The larger its value is, the greater the texture change in this direction will be.

3.3 Threshold determination

For 360° video in ERP format, the entire image is divided into three parts: the polar region, the mid-latitude region, and the equator region. Three methods with calculating SD are proposed for different regions.

For the polar region (the red region in Fig. 5), the depth matching ratio between the current CU and the left CU is the highest, the upper CU is second, so the region intends to use the larger weighting value of SRD_{Hor} and the smaller weighting value of SRD_{Ver} . The image complexity of the polar region is SRD_p (Pole of SRD):

$$SRD_p = \alpha_1 \times SRD_{Hor} + \beta_1 \times SRD_{Ver} \tag{17}$$

For the selection of the weighting values α_1 and β_1 in horizontal and vertical directions of the formula, a compromise value of 0.9 for α_1 and 0.1 for β_1 is obtained after a large number of experiments. Compared with other values, these two values have no significant difference in coding time. However, the distortion of the image is relatively low, so they are selected as the weighting values for the horizontal and vertical directions.

In this paper, the threshold is selected by the statistical method. For 16 test sequences, one 4 K sequence and one 6 K sequence are selected. One frame is extracted for statistics and the threshold is analyzed. Although only two videos were selected for statistics, the selected data set used to calculate SRD in each direction actually contains enough samples (64×64 , 32×32 , 16×16 and 8×8 CU samples are 6408, 25,632, 102,528, 410,112, respectively). Through statistics, it is found that the value range of threshold obtained by different sizes of video are similar, indicating that the selected threshold is universal. Here we only counted the number of CUs that should be further partitioned and the number of CUs that are early terminated when QP value is 32. The SRD_p statistical distribution of the polar region is shown in Fig. 7. This rule also applies to other QP values.

After statistical analysis, it is concluded that as the depth of CU increases, the number of CUs that should be further partitioned decreases, and the number of CUs that are early terminated increases. In other words, we can find two thresholds, one large and one small. When SRD is smaller than the small threshold S_i (Smooth), it means that the texture

is relatively simple, then CU doesn't need further division; when SRD is greater than the large threshold R_i (Rough), the texture is relatively complex, then CU terminates the partition in advance. When the value is between the two values, it is impossible to determine whether the texture is simple or complex, and the complicated process of original division and cropping is needed. For different QP and different depth of CU, the threshold selection in the polar region are shown in Table 2.

For the mid-latitude region (the green area in Fig. 5), the depth matching ratio between the current CU and upper-left CU is the highest, and the upper CU is the second. Therefore, the calculation of SRD_{UH} and SRD_{DH} are introduced. The calculation formula of SRD_m (Medium of SRD) in the mid-latitude region is as follows:

$$SRD_m = \alpha_2 \times SRD_{DH} + \beta_2 \times SRD_{UH} + \gamma_2 \times SRD_{Ver} \tag{18}$$

For the weighting values α_2 , β_2 and γ_2 of SRD in each direction in the formula, the compromise values of 0.4, 0.4 and 0.2 are selected after a large number of experiments, which reduce much coding time and cause lower image distortion.

Similarly, we count the number of CUs that should be further partitioned and the number of CUs that are early terminated partitioning when QP value is 32. The SRD_m statistical distribution of the mid-latitude region is shown in Fig. 8. For different QP and different depth of CU, the threshold selection in the mid-latitude region are shown in Table 3.

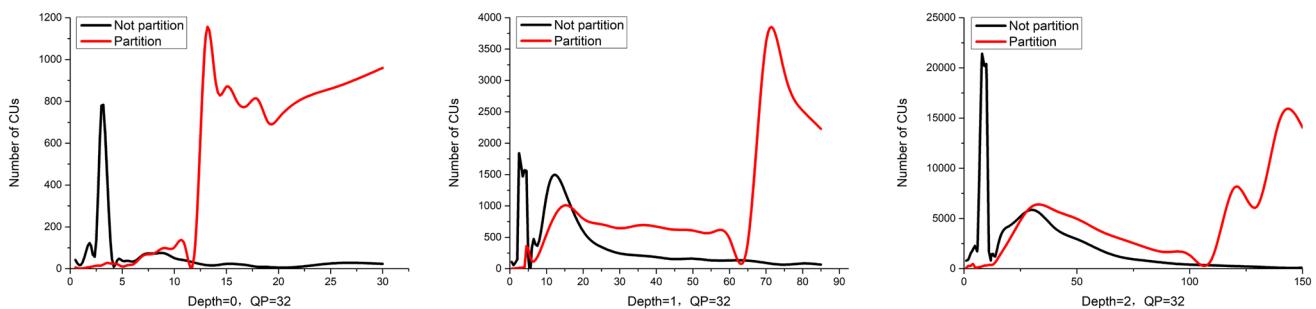


Fig. 7 SRD in polar region

Table 2 Threshold selection in polar region

QP	Depth = 0		Depth = 1		Depth = 2	
	S_1	R_1	S_1	R_1	S_1	R_1
22	1.6	6	2.5	30	5	35
27	3	10	4	35	6	55
32	3	13	4	70	10	140
37	5	17	5	110	18	170

For the equator region (the yellow region in Fig. 5), the depth matching ratio between current CU and left CU, upper CU is basically the same, and the upper-left CU is the lowest. Therefore, SRD_{Hor} and SRD_{Ver} are selected for calculation. The SRD_e (Equator of SRD) calculation formula for the equatorial region is as follows:

$$SRD_e = \alpha_3 \times SRD_{Hor} + \beta_3 \times SRD_{Ver}. \tag{19}$$

For the weighting values α_3 and β_3 in horizontal and vertical directions in the above formula, the theoretical weights should be the same. But after a large number of experiments, it is found that α_3 is 0.6, β_3 is 0.4, compared with α_3 is 0.5, β_3 is 0.5, reduces more coding time and causes smaller image distortion. Therefore, these two values are selected as the weighting values in two directions.

We count the number of CUs that should be further partitioned and the number of CUs that are early terminated

when the QP value is 32. The SRD_e statistical distribution of the equator region is shown in Fig. 9. For different QP and different depth of CU, the threshold selection in the equator region are shown in Table 4.

In summary, we obtain S_i and R_i in different regions. By judging the size between the value of each CU and two values, we can directly further partition or early terminate CU to achieve the goal of reducing coding time.

3.4 The whole diagram of algorithm

The algorithm proposed in this paper needs to calculate SRD in different directions according to the region where CTU is located. First, according to the weighting value w used in WS-PSNR, CTU is located in three different regions which are used in horizontal, vertical, upper horizontal and lower horizontal directions, respectively. SRD_i calculated in different regions is compared with the two thresholds S_i and R_i

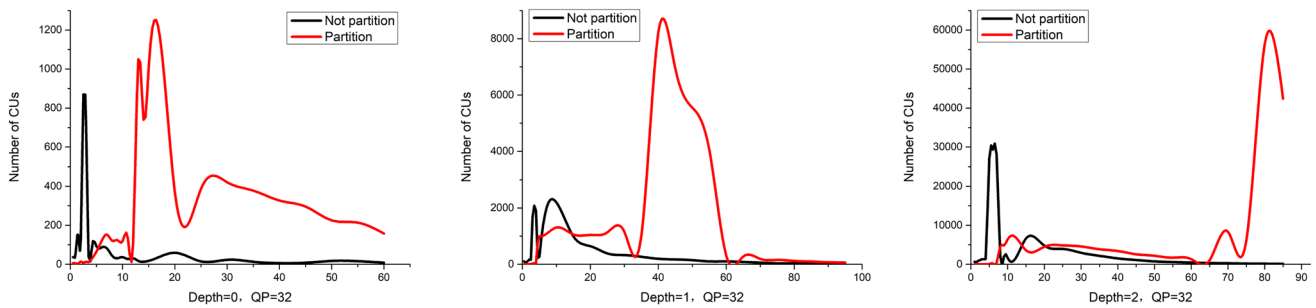


Fig. 8 SRD in mid-latitude region

Table 3 Threshold selection in MID-Latitude region

QP	Depth=0		Depth=1		Depth=2	
	S_2	R_2	S_2	R_2	S_2	R_2
22	1.5	5	1.5	25	3.5	30
27	2.5	10	2.5	30	5	45
32	3	12	4	40	7	85
37	4	18	7	90	15	100

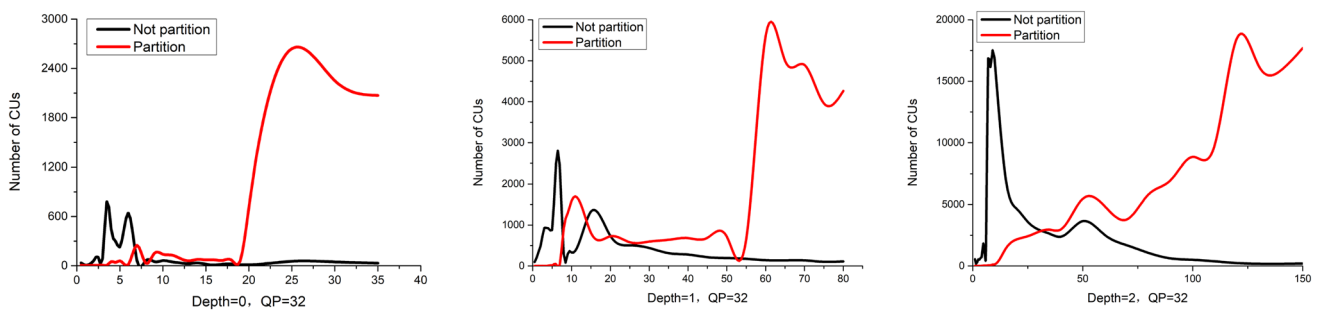


Fig. 9 SD in equator region

Table 4 Threshold selection in equator region

QP	Depth=0		Depth=1		Depth=2	
	S_3	R_3	S_3	R_3	S_3	R_3
22	3	8	2.5	45	5	40
27	4	12	4	50	6	65
32	6	20	7	60	10	120
37	8	30	10	100	20	170

of this region. If SRD_i is smaller than S_i , the image features of CU are relatively simple, and CU is early terminated. If SRD_i is greater than S_i , the comparison with R_i is continued. If SRD_i is larger than R_i , the image features of CU are complicated, and the next step is directly divided. If SRD_i is smaller than R_i , it is impossible to judge whether CU needs to be divided or not, and the traditional coding method is used to determine. The schematic diagram of the algorithm is shown in Fig. 10.

4 Experimental results

The effectiveness of the proposed algorithm is determined on the basis of the latest HM16.20 plus 360Lib4.0. JVET recommended 16 standard test sequences. The test sequences include 4 K, 6 K, 8 K, and 8 K 10-bit sequences. The sequences used in this paper are provided by GoPro [21], InterDigital [22], Nokia [23] and Letin VR [24]. To facilitate the encoding process, we have to convert the test sequences into a low-resolution ERP. We use the test computer which model is Mechrevo X6Ti with Intel® Core™ i7-7700HQ CPU @ 2.80 GHz, 8 GB RAM.

We use the encoding parameters of All Intra frame and BD-rate. For each sequence, we select the first 100 frames to judge the encoding performance compared with original HM16.20. At the same time, we select 22, 27, 32, and 37 four quantization parameters to test the sequences. There is a unified standard to descript the change of image quality called WS-PSNR. ΔT represents the difference between the encoding time of our proposed algorithm and the original algorithm. BD-rate can reflect the quality of the algorithm compared with the original algorithm.

$$\Delta T(\%) = \frac{T_{HM16.20} - T_{proposed}}{T_{HM16.20}} \times 100\%, \tag{20}$$

$$\Delta WS\text{-PSNR}_Y = WSPSNR_{proposed} - WSPSNR_{HM16.20}, \tag{21}$$

$$BD\text{-rate}(\%) = \frac{BDrate_{proposed} - BDrate_{HM16.20}}{BDrate_{16.20}} \times 100\%. \tag{22}$$

The algorithm in this paper is compared with HM16.20. The experimental data is shown in Table 5.

Different test sequences reduce different percentages of time mainly because the sequences have different texture and content. The results in Table 4 show that the algorithm has an average time reduction of 38.52% compared with HM, while BD-rate only increases by 0.40%. This is because the paper first divides the image into three regions according to the location of CTU. Then different regions use different directions to calculate SRD, and obtain a threshold that can be early terminated and a threshold that can be further divided. By comparing the relationship between SRD_i of each coding unit and two thresholds, the unnecessary division processes is skipped, thereby achieving the effect of saving time without causing large distortion.

The algorithm in this paper reduces much time in Balboa, Broadway, Landing2, Gaslamp, Harbor, SkateboardInLot, SkateboardTrick, Train sequence. The main reason is that the texture of this sequence is relatively simple. More importantly, they conform well to the feature that the relationship between the current block and adjacent blocks in a different region is different. However, in PoleVault, BranCastle2 and KiteFlite sequences, the algorithm reduces relatively less time because the texture of these sequences is relatively complex and the selected thresholds cannot well skip the division process of complex CU. The relatively special result is that PoleVault and Trolley sequence have a large increase in BD-rate, while the percent of time reduction is small. The main reason is that the value of SRD calculated by the algorithm is between two thresholds, and the traditional process of CU partition can't be skipped. Figure 11 is a comparison of rate-distortion performance with HM16.20 of Broadway, Landing2, PoleVault, and Trolley. The black spline expresses the RD performance of HM16.20 encoder, and the red spline expresses the RD performance of the proposed algorithm. The splines are basically coincident, indicating that performance is basically unchanged.

Table 6 shows the performance comparison of other algorithms and proposed algorithm. Li [26] proposed the algorithm based on spatial correlation and RD-cost. The algorithm can reduce encoding time by 43.3%. Belghith [27] proposed a CU fast partition algorithm based on edge

Fig. 10 Fast CU partition algorithm

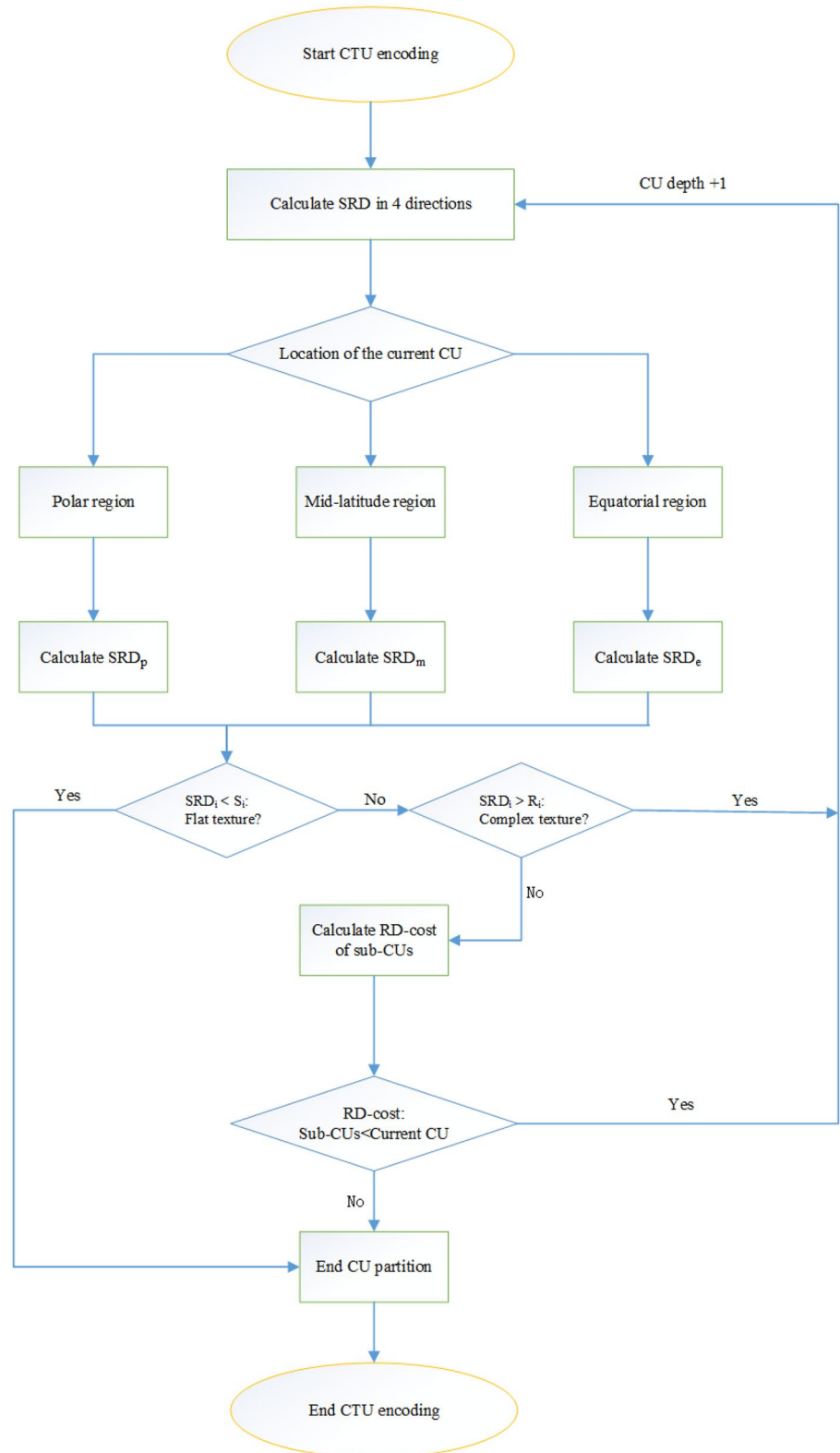


Table 5 Experimental results

Class	Test sequence	BD-rate(%)	ΔT (%)	Δ WS-PSNR _Y
4 K	AerialCity	0.50	32.69	-0.008
	DrivingInCity	0.30	38.92	-0.001
	PoleVault	0.60	29.22	-0.020
	DrivingInCountry	0.10	35.64	-0.010
6 K	Balboa	0.60	46.25	-0.004
	BranCastle2	0.70	26.92	-0.015
	Broadway	0.80	43.00	-0.004
	Landing2	0.60	43.70	-0.005
8 K	ChairliftRide	0.10	32.73	-0.001
	Gaslamp	0.30	39.40	-0.006
	Harbor	0.20	41.93	-0.006
	KiteFlite	0.40	28.90	-0.008
	SkateboardInLot	0.10	43.42	-0.001
	SkateboardTrick	0.30	48.19	-0.015
	Train	0.30	46.69	-0.013
	Trolley	1.10	32.69	-0.032
Average		0.40	38.52	-0.009

detection, which defines the parameter of texture. The algorithm can reduce time by 31.3%. Compared with these reference algorithms, the proposed algorithm reduces more encoding time and the loss of image quality is negligible.

Figure 12 shows the CU partitioning comparison of DrivingInCountry, ChairliftRide, Harbor and SkateboardInlot. HM represents the division of CU under HM16.20 encoder. Proposed represents the division of CU under the proposed algorithm. The red block indicates CUs that are early terminated, and the green block indicates CUs that need to be further divided. Obviously, the proposed algorithm can reduce unnecessary partition and deeply divide CUs that should be accurately partitioned.

We can see the accuracy of CU partitioning in the proposed algorithm for each test sequence from Fig. 13. The last term is the average CU partitioning accuracy of 16 sequences. It can be seen that as the depth gradually deepens, the accuracy of the CU partition gradually becomes higher, and the accuracy can basically reach 80% or more. In general, the algorithm skips a lot of unnecessary partitioning,

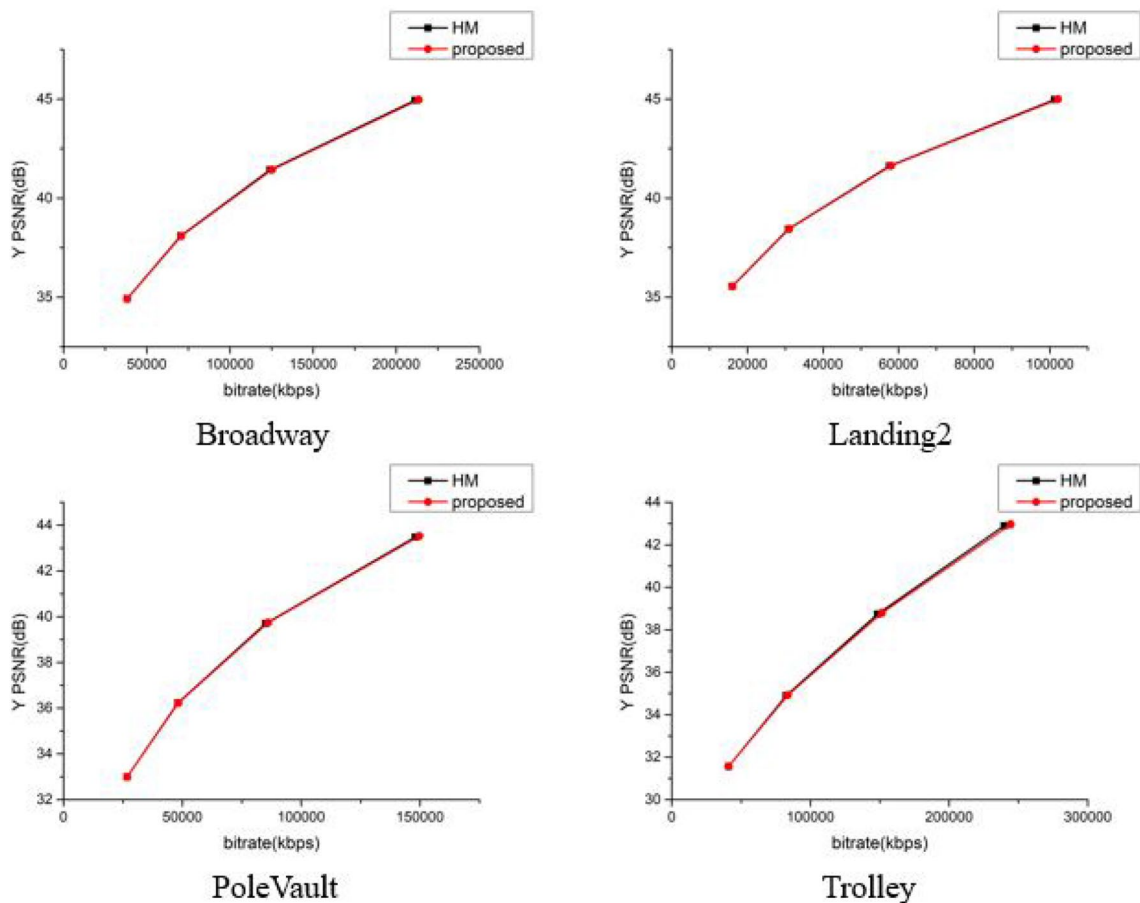


Fig. 11 Comparison diagram of RD performance

Table 6 Comparison of reference algorithms

	Li [26]	Belghith [27]	Proposed
ΔT (%)	43.3	31.3	38.5
BD-rate (%)	0.6	0.76	0.4
$\Delta WS\text{-}PSNR_Y$	-0.027	-0.09	-0.009

greatly reduces the encoding time, while still maintaining a high CU partitioning accuracy and only a small loss for BD-rate.

5 Conclusion

In this paper, a fast intra algorithm based on the sum of region-directional dispersion for 360° video is proposed, which makes full use of the characteristics of 360° video under ERP projection format to optimize coding performance and reduce the computational complexity of HEVC intra encoder. The algorithm divides the video into different regions and processes them separately, which reduces unnecessary CU partitioning and reduces computational complexity while keeping BD-rate almost unchanged. Compared with the original HM16.20 coding, the average time can be reduced by 38.5%, while BD-rate is only increased by 0.40%. In conclusion, the algorithm in this

Fig. 12 Comparison diagram of CU partition

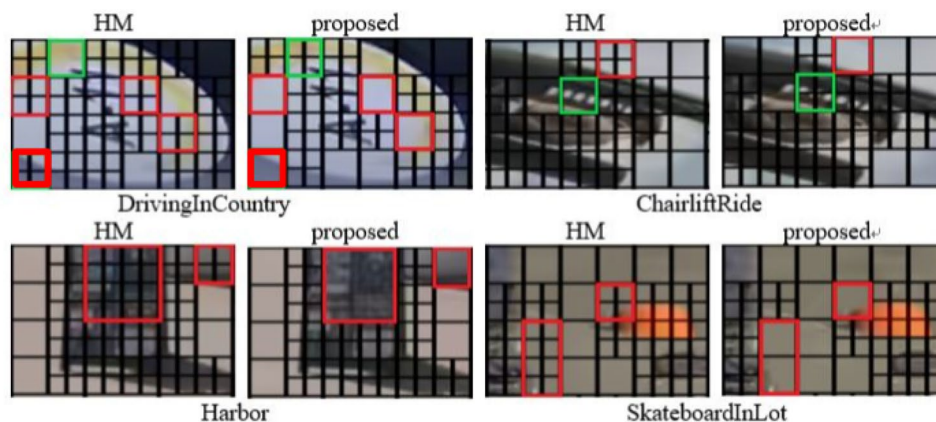
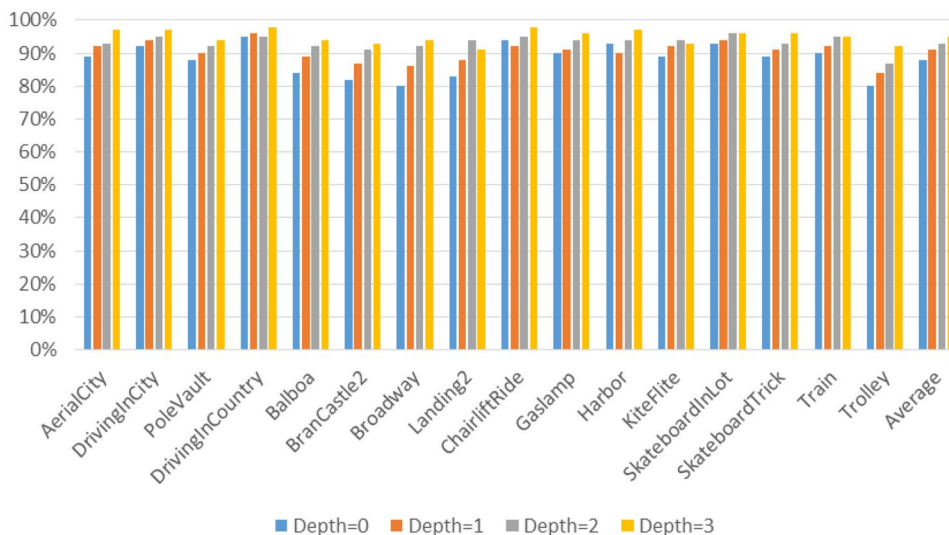


Fig. 13 Accuracy of CU partition



paper significantly improves the coding efficiency without causing large image distortion.

Acknowledgements This work is supported by the Great Wall Scholar Project of Beijing Municipal Education Commission (CIT&TCD20180304), Beijing Municipal Natural Science Foundation (No. 4202018), and the National Natural Science Foundation of China (No. 61972023).

References

- P. P. Routhier: Virtually perfect: factors affecting the quality of a VR experience and the need for a VR content quality standard, in *SMPTE 2016—Annual Technical Conference and Exhibition*, pp. 1–20, (2016)
- G. X. Jin, S. Ankur, B. Madhukar: Motion estimation and compensation for fisheye warped video, in *International Conference on Image Processing*, pp. 2751–2755, (2015)
- Ralf, S., et al.: Interactive steaming of panoramas and VR worlds. *SMPTE Motion Imaging J* **126**, 35–42 (2017)
- Y. Ramin Ghaznavi et al.: Efficient coding of 360-degree pseudo-cylindrical panoramic video for virtual reality applications, in *International Symposium on Multimedia*, pp. 525–528, (2016)
- Li, Z.M., et al.: A fast CU partition method based on CU depth spatial correlation and RD cost characteristics for HEVC intra coding. *Signal Process. Image Commun.* **75**, 141–146 (2019)
- Chen, F., et al.: Fast intra coding algorithm for HEVC based on depth range prediction and mode reduction. *Multimedia Tools Appl.* **77**, 28375–28394 (2018)
- Liao, W.H., Chen, Z.Z.: A fast CU partition and mode decision algorithm for HEVC intra coding. *Signal Process. Image Commun.* **67**, 140–148 (2018)
- F. Y. Duanmu et al.: Hybrid cubemap projection format for 360-degree video coding, in *2018 Data Compression Conference*, (2018)
- N. Yu et al.: Improving cube-to-erp conversion performance with geometry features of 360 video structure, in *2019 Data Compression Conference*, (2019)
- Y. W. He et al., “Content-Adaptive 360-Degree Video Coding Using Hybrid Cubemap Projection,” 2018 Picture Coding Symposium, 2018
- K. K. Sreedhar et al.: Standard-compliant multiview video coding and streaming for virtual reality applications, in *2016 IEEE International Symposium on Multimedia*, (2016)
- J. Sauer et al.: Geometry-Corrected Deblocking Filter for 360° Video Coding using Cube Representation, in *2018 Picture Coding Symposium*, (2018)
- I. Storch et al.: FastIntra360: a fast intra-prediction technique for 360-degrees video coding, in *2019 Data Compression Conference*, (2019)
- M. M. Zhang et al.: Fast PU early termination algorithm based on WMSE for ERP video intra prediction, in *2019 Data Compression Conference*, (2019)
- X. H. Guan et al.: Fast early termination of CU partition and mode selection algorithm for virtual reality video in HEVC, in *2019 Data Compression Conference*, pp. 576, (2019)
- T. M. Li, J. Z. Xu, Z. Z. Chen: Spherical domain rate-distortion optimization for 360-degree video coding, in *International Conference on Multimedia and Expo*, vol. 0, pp. 709–714, (2017)
- Y. B. Wang et al.: A fast intra prediction algorithm for 360-degree equirectangular panoramic video, in *Visual Communications and Image Processing*, vol. 2018, pp. 1–4, (2018)
- F. X. Yang et al.: Region priority based adaptive 360-degree video streaming using DASH, in *International Conference on Audio, Language and Image Processing*, pp. 398–405, (2018)
- F. Takuya et al.: Graceful quality improvement in wireless 360-degree video delivery, in *IEEE Global Communications Conference*, (2018)
- H. Philippe et al.: 360-degree video quality evaluation, in *2018 Picture Coding Symposium*, pp. 328–332, (2018)
- A. Abbas, B. Adsumilli: AHG8: new GoPro test sequences for virtual reality video coding, in *Joint Video Exploration Team (JVET) of ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11, 4th Meeting, document JVET-D0026, Chengdu, China*, (2016)
- E. Asbun, Y. He, Y. Ye: AHG8: InterDigital test sequences for virtual reality video coding, in *Joint Video Exploration Team (JVET) of ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11, 4th Meeting, document JVET-D0039, Chengdu, China*, (2016)
- S. Schwarz, A. Aminlou: Tampere pole vaulting sequence for virtual reality video coding, in *Joint Video Exploration Team (JVET) of ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11, 4th Meeting, document JVET-D0143 Chengdu, China*, (2016)
- W. Sun R. Guo: Test sequences for virtual reality video coding from letinvr, in *Joint Video Exploration Team (JVET) of ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11, 4th Meeting, document JVET-D0179, Chengdu, China*, (2016)
- Bai, H.H., et al.: Multiple description video coding based on human visual system characteristics. *Trans. Circuits Syst. Video Technol.* **24**, 1390–1394 (2014)
- Li, Z., Zhao, Y., Dai, Z., Rogeany, K., Cen, Y., Xiao, Z., Yang, W.: A fast CU partition method based on CU depth spatial correlation and RD cost characteristics for HEVC intra coding. *Signal Process. Image Commun.* **75**, 141–146 (2019)
- Belghith, F., Kibeya, H., Ayed, M., Masmoudi, N.: Fast coding unit partitioning method based on edge detection for HEVC intra-coding. *Signal Image Video Process* **10**, 811–818 (2016)

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.