Fast Inter-frame Prediction Based Array Processor for Depth Maps in 3D-HEVC

Yun Zhu*, Lin Jiang†1, Hui Song*, Xiaoyan Xie*, Anqi Wang*, Xubang Shen††
* Xi'an University of Posts and Telecommunications, Xi'an, China
E-mail: zhuyun@xupt.edu.cn, 1255532038@qq.com, xxy@xupt.edu.cn, 982725333@qq.com
†Xi'an University of Science and Technology, Xi'an, China
E-mail: jianglin@xust.edu.cn
††Xi'an Microelectronic Technology Research Institute, Xi'an, China
E-mail: shenxubang@163.net

***Abstract*— For the characteristics of the depth map with a large smooth area, it is not necessary in the 3D-High-Efficiency Video Coding (3D-HEVC) to use TZ search for inter-frame prediction. According to the parallelism of depth map inter-frame algorithms, analyzed the parallelism of full search, three-step search and diamond search in the array structure, this paper proposes a parallel implementation method of 3D-HEVC depth map inter-frame prediction based on array processor. Considering only the depth map encoding, the experimental results show that the synthesized frequency under the BEE4 FPGA chip is as high as 100.261MHZ. Compared with the software 3D-HEVC Test Model (HTM) version 16.1, without affecting the video quality, the speedup ratio of the full search mode can reach 35, the speedup ratio of the three-step search mode can reach 160, and the speedup ratio of the diamond search can reach 233.**

Keywords—3D-HEVC, Depth map, Inter-frame prediction, Fast search algorithm.

I. INTRODUCTION

The Joint Collaborative Team on 3D Video Coding Extension Development (JCT-3V) developed the 3D-High Efficiency Video Coding (3D-HEVC)[1-2]. The multi-view video plus depth (MVD) data format has been standardized as the 3D video representation, which contains multiple views of texture within the associated depth map for each view[3]. The depth maps coding techniques can be classified into two categories depending on the relationship with the corresponding texture video, independent coding and texture-assisted depth coding[4]. The encoding techniques used in depth maps are basically the same as those of texture frames. The motion parameter inheritance technique is used to exploit the motion similarity between depth and texture videos. However, compared to texture video, depth video exhibits distinct characteristics, which describes the geometry of the 3D scene[5]. It is usually composed of large portions of flat regions separated by sharp edges[6-7]. In order to avoid the generation of new depth values and ringing artifacts at depth maps edges, the motion compensation doesn't include an interpolation. The motion vectors are coded with sample instead of quarter-sample accuracy[8].

Since video pictures and depth maps represent different properties of the same video scene, the motion characteristics should be similar. The best block matching of each block of the current frame is found in the reference frame using inter-frame prediction[9], as shown in Figure 1. A new inter coding mode for depth maps is added in which the partitioning of a block of sub-blocks as well as the associated motion parameters are inferred from the co-located block in the associated video picture[10]. HTM that is 3D-HEVC reference software[11] uses test zone search (TZS)[12] for depth maps motion estimation. TZS is very irregular in accessing stored values during the calculation of sum of absolute differences (SAD) values and in the search process using a variety of algorithms, the storage of constant reading and writing operation, resulting in great energy consumption. In TZS, the search region is defined by the vector predictive value, either diamond search, scanning or other search approach has been employed. However, the coding complexity is still burden on TZS. Therefore, the parallel implementation of optimized search algorithms has attracted more attention.

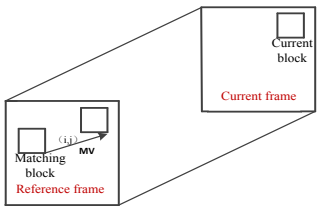


Fig.1 Motion estimation block matching process

Parallelizing 3D-HEVC coding on a parallel computing platform is an effective method to solve the high coding complexity. At present there are two kinds of parallel methods, Global Parallel Method (GPM)[13-14] and Local Parallel Method (LPM)[15-16].GPM is widely used in the H.264 / AVC motion estimation algorithm because it does not consider the data dependency between prediction units (PUs), so the degree of parallelism (DP) is very high but the coding

[1] Corresponding author.

loss is also large. LPM guarantees the rate-distortion performance, but the parallelism is low. It cannot make full use of the many processing units of the parallel computing platform, which seriously affects the efficiency of parallel processing.

In order to reduce the coding complexity, a new method of optimization algorithm, hardware and parallel mapping strategy is designed. This paper first optimizes the inter-frame predict algorithm for 3D-HEVC depth maps is optimized, then proposes a video processor consisting of 1024 thin core processing units (PEs) interconnected by adjacent components to form 32×32 array processing[17], which is used by the communication mechanism of global on-chip network and local shared memory, finally the fast search algorithm to parallel mapping is implemented on this video array processor. The design flow is shown in Figure 2.

Fig.2 Optimize fast inter-frame prediction across algorithm, hardware and parallel mapping strategy

The remainder of the paper is organized as follows: Section II introduces the recent related work of 3D-HEVC depth maps inter prediction algorithm. Section III introduces the parallel implementation of the 3D-HEVC depth maps inter-frame algorithm in the video array structure, including detailed introduction the overall structure and communication mechanism of video array processor. In Section IV presents the implementation results and analyzes in detail. Finally, Section V summarizes the full text.

II. RELATED WORKS

A. Depth Map Features

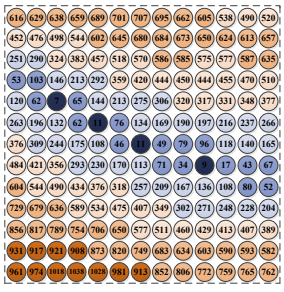


Fig.3 Texture's SAD heat map

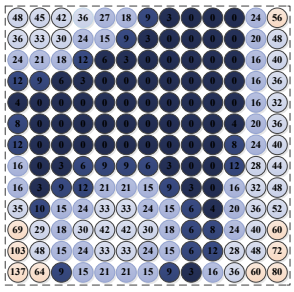


Fig.4 Depth map's SAD heat map

In 3D-HEVC, depth maps have sharp edges and large range smoothing regions[18]. The 3D-HEVC depth maps test sequences of *depth_balloons*, *depth_Newspaper* and *Poznan_Street* are analyzed, the distribution of their SADs is calculated as shown in Figure 3 and Figure 4. It can be found that the SAD distribution of texture mapping is more complicated, and there are local optimal values. It is difficult for fast algorithms to find the best matching block. However, the SAD value distribution of the corresponding depth maps is relatively regular, and there is a large area of smooth area. So the fast algorithm can quickly find the optimal SAD value. Saldanha M[19] pointed out that the motion estimation of depth maps accounts for 19.74% of the total encoding time. This is due to the complexity of the search algorithm, without considering the characteristics of the depth maps. He put forward diamond search which can be used to reduce coding complexity in the premise of ensuring coding efficiency. In this paper, three fast algorithms, including full search (FS), three-step search (TSS), and diamond search (DS), are analyzed and compared. The comparison results obtained using the video array processor is presented in Section V.

B. Algorithm Parallelism Analysis

The parallelism of LPM is less than GPM, but the encoding quality is much higher. The LPM introduces a motion estimation region (MER), and each coding tree unit (CTU) can be divided into a plurality of non-overlapping MERs.

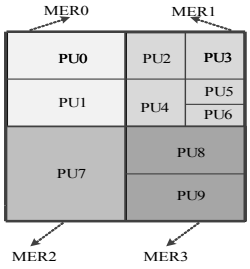


Fig.5 MER sketch map

Within the same MER, all PUs is independent of one another and block matching can be performed in parallel. As shown in Figure 5, a CTU can be divided into four MERs, which are executed sequentially from MER0 to MER3. PU8 and PU9 in MER3 are independent of each other. When processing MER3, PU8 and PU9 can be processed in parallel. The *DP* in local parallelism can be denoted in Equation (1).

$$DP = \frac{Size(MER)}{\min(Size(PU))} \tag{1}$$

The size of MER is usually 16×16 or 8×8. The minimum size of PU is 4×4. The maximum *DP* of local parallelism is 16, so the parallelism of the video array processor cannot be fully utilized. Considering data dependencies without affecting code quality, how to make full use of multi-core array structure is the key to research. Traditional LPM does not affect video quality, but its parallelism is too low. Therefore, CTU level parallelism and MER level parallelism are added.

1) CTU Level Parallel

In 3D-HEVC, CTU is the largest encoding unit. Adjacent CTUs have data dependencies, while non-adjacent CTUs are independent of each other and can be executed in parallel. Figure 6 shows the sequence in which CTUs are executed in parallel. CTUs with the same tag can be executed in parallel.

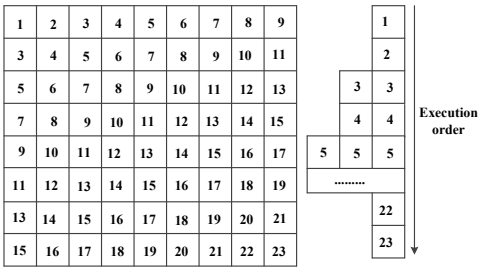


Fig.6 CTU level parallel

The degree of parallelism at the CTU level is related to the resolution of the image. The resolution is higher; the degree of parallelism is higher. The calculation process of *DP* is shown in Equation (2). *W* and *H* represent the width and height of the image divided according to the CTU, respectively. Therefore, the DP_{CTU} of the 1024×768 sequence is 8, and the DP_{CTU} of the 1920×1088 sequence is up to 15.

$$DP_{CTU} = \min(ceil(\frac{W}{2}), H) \tag{2}$$

2) MER Level Parallel

MER level parallel is similar to CTU level. Non-adjacent MERs can be executed in parallel without data dependence. As shown in Figure 7, a CTU is evenly divided into sixteen MERs. The same labeled MER without data dependency can be processed in parallel. So DP_{MER} is 2.

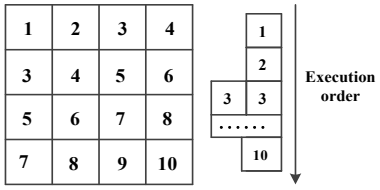


Fig.7 MER level parallel

3) PU Level Parallel

The MER can be divided into four 8×8, eight 8×4 and sixteen 4×4 PUs. Since there is no data correlation between all PUs inside the MER, they can be processed in parallel. As shown in Figure 8, the *DP* of the PU level can reach 28 (DP_{PU} =4+8+16=28).

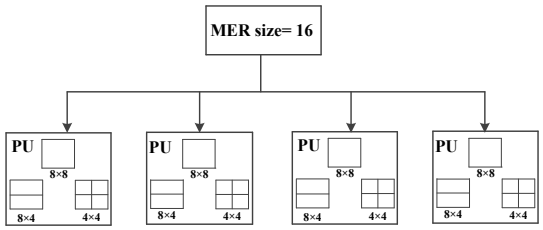


Fig.8 Local parallel (PU level) scheme

In summary, the parallelism of the 3D-HEVC depth maps is equal to the product of DP_{PU}, DP_{CTU}, and DP_{MER}, that is, the number of PUs that can be processed simultaneously, as in Equation (3). The maximum parallelism of the 1024×768 depth maps sequence is 448, the maximum parallelism of the 1920×1088 depth maps sequence can reach 840. This parallelism is not as good as GPM, but it is enough to take full advantage of the video array structure and the coding loss is small.

$$DP_{depthmap} = DP_{PU} \times DP_{CTU} \times DP_{MER} \tag{3}$$

In [20] and [21], a new parallel method for motion estimation based on LPM is proposed. Compared to the serial implementation of 1920×1080 and 2560×1600 video sequences, experiments based on 64-core systems show that the acceleration ratios in parallel mode are 30 and 40 times, respectively. And the quality of the code has also improved. A new fast and effective depth video compression method based on 3D image morphing is proposed[22], which can guarantee the quality of reconstructed depth maps, reduce inter prediction error and ensure real-time performance. However, this improvement is only in algorithm optimization and does not consider hardware implementation.

In this paper, a 32×32 video array processor is proposed, which composed of 1024 thin core processing units (PEs) interconnected by adjacent. It utilized a global on-chip network and a local shared memory communication mechanism. The fast search algorithms implement parallel mapping on the video array processor.

III. HARDWARE IMPLEMENTATIONS

A. *Structure of Video Array Processor*

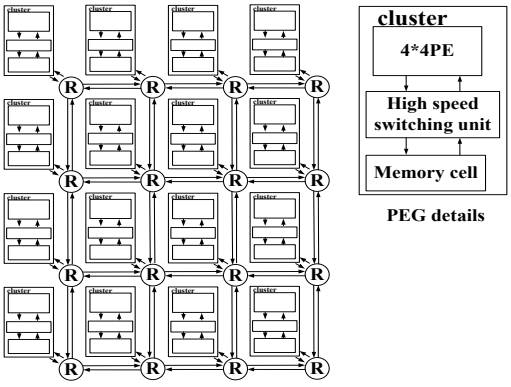


Fig.9 Partial architecture of video array processor units

The video array processor consists of 1024 light core PEs, in which each 4×4 PEs is one processor cluster (PEG). The communication mechanism based on local shared memory on global network. Intra cluster data interaction adopted adjacent interconnection and shared storage. Inter cluster telecommunication is used by network on chip. Its partial

structure is shown in Figure 9. The PE processing unit is implemented by extracting, decoding, executing, and writing back to the four-stage pipeline. It consists of a 32-bit instruction register, an 8×16-bit register file, a 256×16-bit data/instruction memory, an arithmetic logic unit, and other parts.

B. *Communication Mechanism*

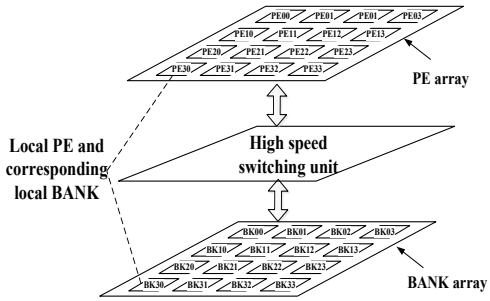


Fig.10 High speed switching unit

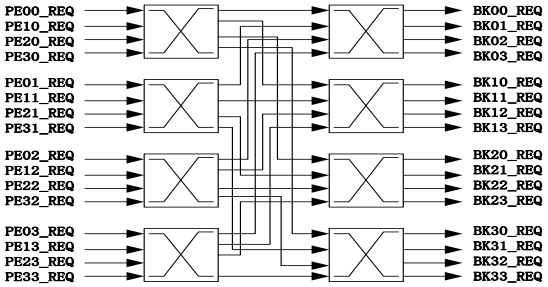


Fig.11 Two stage switching fabric

First, determine the location of the PE access storage unit. In this paper, intra cluster the data access is implemented by a high-speed switching unit, the structure is shown in Figure 10. Each local PE corresponds to one local storage bank. In view of the high frequency of data interaction in the cluster, local PE is set as the highest priority access to the local storage bank. The local PE accesses other banks in the PEG through a two-stage switch fabric and supports 16-channel read/write parallel access, as shown in Figure 11. The four PE requests in the same column is performed the first level arbitration. Four requests from different rows are sent to the second level arbitrator at the same time. The polling strategy is used to arbitrate when there are multiple requests for the same row's storage bank. After the first level arbitration, the four requests of the same row are subjected to second level arbitration. If the four requests are aimed different banks, they are directly exported to the corresponding storage bank. If multiple requests are in the same bank, polling strategy is applied to arbitrate.

Inter-cluster data communication is achieved through an on-chip network of low latency virtual channel routers. The inter cluster communication mechanism consists of three parts, namely the main part of PEG, network adapters and on-chip network routers. In order to reduce power consumption and design complexity, only the PE in the lower right corner of the 4×4 PE array is allowed to transmit data to the on-chip network. The remaining fifteen PEs can write to this PE storage through shared storage access or continuous interconnect access and then initiate inter cluster communication. The main function of the network adapter is to connect PEG and router. The output data and destination address in the packets whose data format that can be received by the routing network, are sent to the routing network. The router could send remote data packets from the source node to the target node. It uses the XY deterministic routing algorithm without deadlock and the packet switching network.

IV. PARALLEL IMPLEMENTATION

A. *Parallel implementation based on FS*

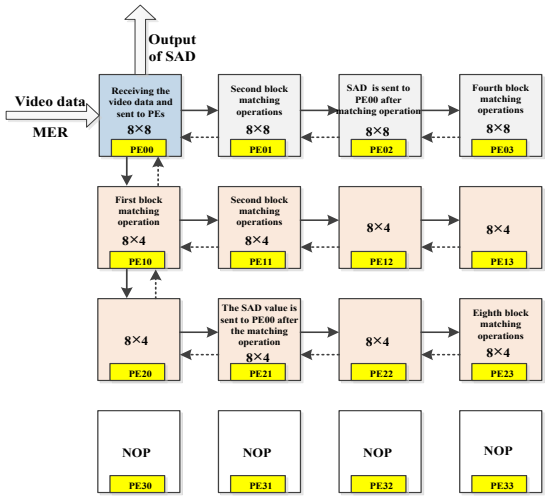


Fig.12 Parallel operation of PEG01

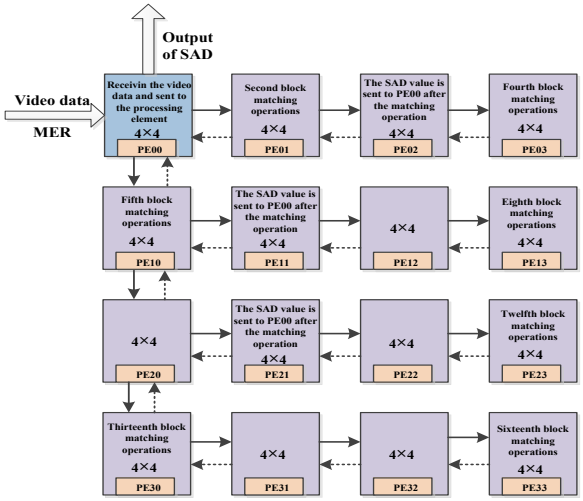


Fig.13 Parallel operation of PEG10

The FS depth maps motion estimation algorithm in a video array processor mapping scheme is shown in Figure 12 and Figure 13. The parallel processing of MER is done with two clusters PEG01 and PEG10, where PEG01 processes four 8×8

and eight 8×4 parallel processing in parallel, and PEG10 processes 16 4x4 parallel processing in parallel. The mapping process on PEG01 is as follows.

Step 1: Loading raw data. PEG01 has a 16-bit external memory connected to PE00 for storing raw video data. The CTU is divided into sixteen 16×16 MERs and stored in the order in which they are executed. The first MER is stored in addresses 0-255.

Step 2: Loading reference data. PE00, PE01, PE02, and PE03 perform four 8×8 block matching operations, respectively, and build 16×16 search windows centered on four 8×8 blocks in the reference frame, and then load four search windows. For each of the four PE data storage units, the storage address is 256-511. PE10, PE11, PE12, PE13, PE20, PE21, PE22, and PE23 perform eight 8×4 block matching operations, respectively, and build a 16×16 search window centered on these eight 8×4 blocks. The search windows are loaded into the respective data storage units of the 8 PEs, and the storage addresses are 256-511.

Step 3: The original MER is divided into four 8×8 PU blocks and sent to PE00, PE01, PE02 and PE03. The MER is divided into eight 8×4 PU blocks and sent to PE10, PE11, PE12, PE13, PE20, PE21, PE22, and PE23.

Step 4: PE00 sends an indication signal to each of the 12 PEs through shared memory (address 254). After PE00-PE23 receive the PE00 indication signal, each PE can start block matching. Otherwise, it can only wait.

Step 5: After the handshake succeeds, each PE starts to calculate the SAD value and writes the smaller SAD value to the address 255 of the corresponding PE data storage.

Step 6: Continue to acquire different reference blocks from the respective search windows according to the full search mode, and calculate and compare the SAD value with the original PU block until all reference blocks within the search window are acquired.

Step 7: After each PE finds the minimum SAD value and finds the best matching block, PE00 writes the minimum SAD value obtained by each PE into the PE00 data storage unit through the shared storage mechanism, and then uniformly outputs all the minimum SAD values through the shared register.

PEG10 completes the motion estimation process of sixteen 4×4 PU blocks. The parallel mapping idea is basically the same as PEG01, as shown in Figure 13.

The MER block motion estimation parallel processing is completed based on the FS mode. Then, the motion estimation of the remaining fifteen MER blocks is performed in accordance with the order in which the MER is executed. If the two MERs can operate in parallel, four clusters are processed in parallel until the CTU block matching process is completed. After processing the CTU, the next CTU block matching operation follows the order in which the CTUs are executed. If multiple CTUs can be processed in parallel, multiple clusters are used for parallel processing until motion estimation for the entire frame is completed.

B . Parallel implementation based on fast search

The parallel implementation of the depth maps motion estimation algorithm based on TSS, and DS in the video array processor is consistent with the mapping method of the full search algorithm. Through PEG01, four 8×8 blocks and eight 8×4 block matching operations are performed, and through PEG10, sixteen 4×4 block matching operations are completed. The results of analyzing and comparing the three algorithms are given in Section V.

V. EXPERIMENTAL RESULTS

A. Parallel Performance Analysis

Table 1 shows the performance comparison of parallel FS-based depth maps motion estimation algorithms in video array architecture with other implementations. It can be seen from Table 1 that the FS algorithm implemented in this paper occupies a large amount of resources and consumes a lot of power, but the operating frequency is closer to ASIC. In addition, the complete depth maps motion estimation process is implemented in this paper through reconfigurable arrays, while the dedicated hardware in [18] is limited to the calculation of SAD values, and the entire block matching process is not completed. The DP achieved by reconfigurable arrays is much higher than that of ASIC, and the block matching process can be accomplished using multiple search modes, unlike dedicated hardware that is only limited to fixed mode.

Table 1 Performance parameters of depth maps motion estimation

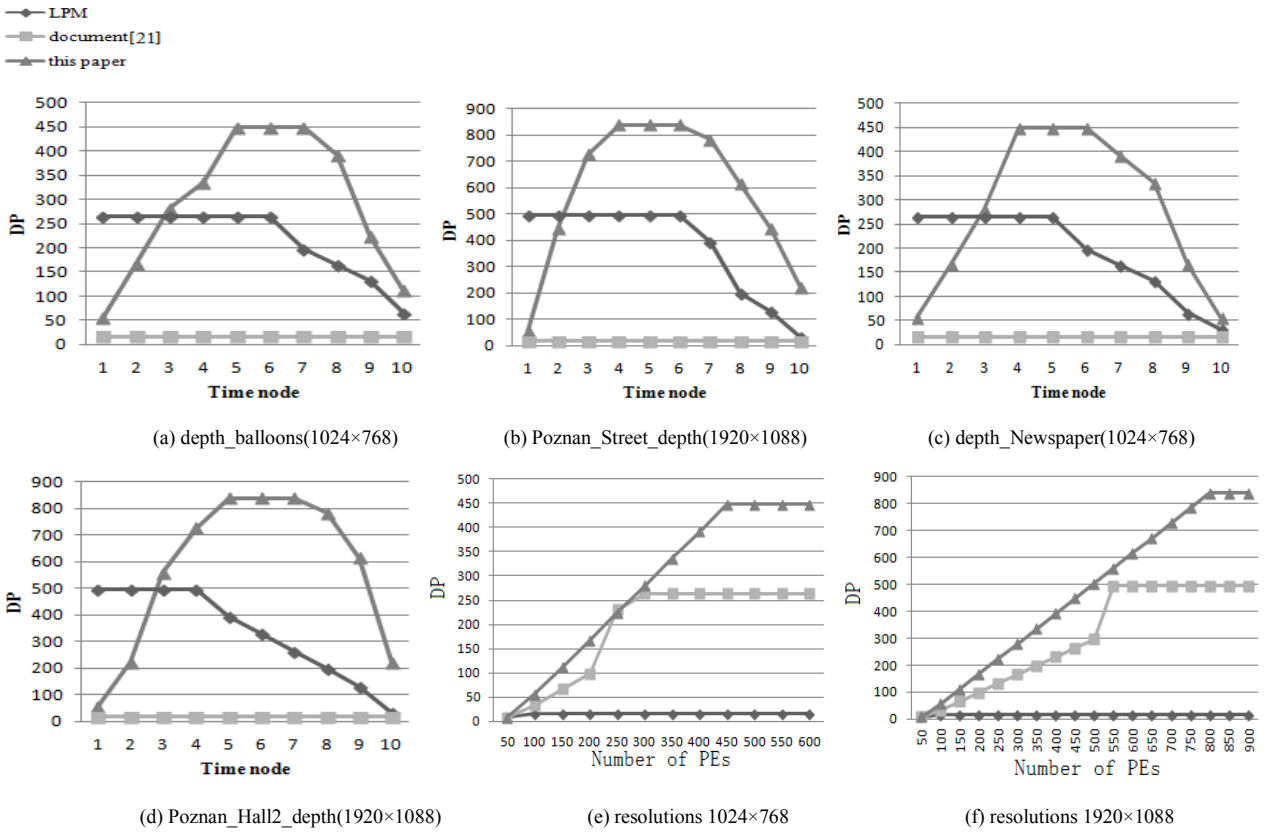
	Technology	Frequency/ MHZ	Gate count/K	Power/mW
[18] (TZS)	45nm	100	62	2.56
[18] (SDSP)	45nm	100	34	1.38
[23]	0.18μm	180	160	14.7
This paper (FS)	FPGA（Bee4）	100.261	280	667

B. Parallel Discussion

After functional simulation and FPGA testing, the DP of the FS-based motion estimation algorithm is analyzed and compared. As shown in Figure 14, the 3D-HEVC depth maps in the four test sequences (*depth_balloons*, *Poznan_Street_depth*, *depth_Newspaper*, and *Poznan_Hall2_depth*) of the two different resolutions (1024×768, 1920×1088) are statistically analyzed based on the results of the FS motion estimation, and compared with the traditional LPM and the literature [21]. The abscissa in (a)-(d) represents the time node at which the entire frame of the image is encoded. The execution time of one frame of image is divided into 10 time segments. The ordinate represents DP of the algorithm. (e)-(f) gives the change in parallelism as the number of processor cores increases.

As can be seen from (a)-(d), the parallelism of the parallel method designed in this paper gradually increases after the start of image coding, and then slowly decreases after reaching the maximum value. Literature [21] adds CTU level parallelism to LPM, and the DP is greatly improved. The parallel method designed in this paper will increase MER

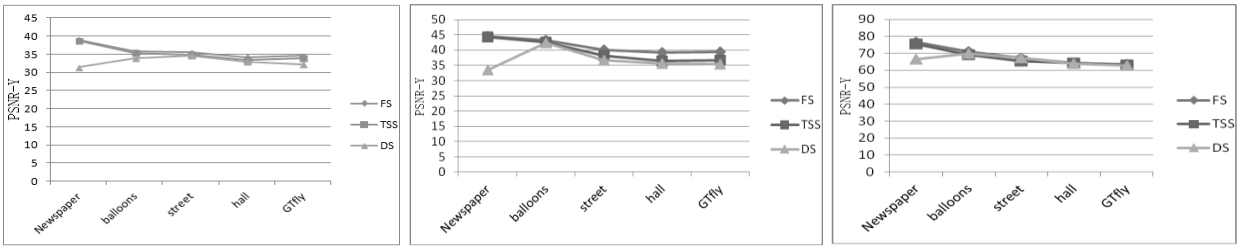
level parallelism on the basis of [21], so the degree of parallelism is higher than it. In addition, the CTU level of the DP is related to the image resolution. The higher the resolution, the higher the degree of parallelism at the CTU level. Therefore, for the method designed in this paper and [21], the higher the resolution, the larger the DP value. The DP value of LPM does not change.



(a) depth_balloons(1024×768) (b) Poznan_Street_depth(1920×1088) (c) depth_Newspaper(1024×768)

(d) Poznan_Hall2_depth(1920×1088) (e) resolutions 1024×768 (f) resolutions 1920×1088

Fig.14 test sequences based FS are compared with the LPM and [21]



(a) Coding quality of 8×8 blocks (b) Coding quality of 8×4 blocks (c) Coding quality of 4×4 blocks

Fig. 15 Comparison of PSNR-Y after three PU sizes parallel mapping

It can be seen from (e) and (f) in Figure 14 that different parallel modes have maximum DP for different resolution depth map test sequences. When the number of PEs is less than the maximum DP, the parallelism of various parallel modes will increase as the number of PEs increases. Conversely, the DP will not increase. For a test sequence with a resolution of 1024×768, the DP in this paper does not increase when the number of processor cores exceeds 450. For resolution of 1920×1088, the DP in this paper does not increase when the number of processor cores exceeds 840.

C. Coding quality analysis

In order to analyze the coding quality of the fast algorithm, the Y-component of peak signal to noise ratio (PSNR-Y) of the depth map motion estimation algorithm based on full search, three-step search and diamond search is compared. Figure 15 shows the PSNR-Y values of the motion estimation process when the algorithm is divided into 8×8, 8×4, and 4×4 PU blocks in parallel mapping.

For all five test sequences, the PSNR-Y value of the motion estimation process based on FS mode is the largest and the coding quality is the best. As the size of the PU block becomes smaller, the larger the PSNR-Y value, the higher the matching accuracy, and the searched matching block are almost the same as the original block. This also explains the large area smooth area of the depth map and the small variation range. For the 3D-HEVC depth map, the motion estimation process based on the fast search mode such as TSS is small, and even the matching accuracy of the FS can be achieved. Therefore, for the depth map, a fast algorithm can be used to estimate the motion.

D. Analysis of coding speed

After analyzing the encoding quality of the fast algorithm, the encoding speed of the fast algorithm is also analyzed and compared. Table 2 calculates the encoding time of the HTM16.1 and the depth maps motion estimation algorithm designed in five test sequences. Compared with the software HTM16.1 implementation, the motion estimation algorithm based on video array processor has greatly improved the coding speed. Among them, the speedup ratio of the FS mode can reach 35, and the speedup ratio of the TSS mode is 160, and the speedup ratio of DS can reach 233.

Table 2. Parallel implementation results evaluation on HTM 16.1 random access mode

Sequence	FS		TSS		DS	
	△ PSNR-Y(dB)	Speed-up	△ PSNR-Y(dB)	Speed-up	△PSNR-Y(dB)	Speed-up
depth_Newspaper	10.80	27	10.11	122.04	1.12	176.02
depth_balloons	5.62	22	4.72	99.44	4.36	143.42
depth_Poznan_Street	2.28	41	0.75	186.18	0.72	267.22
depth_Poznan_Hall2	-4.36	44	-5.59	199.80	-6.06	286.78
depth_GhostTownFly	-4.26	45	-5.62	204.34	-6.68	293.29
Average	2.016	35.8	0.874	162.36	-1.308	233.346

VI. CONCLUSIONS

For the motion estimation of 3D-HEVC depth maps, the fast algorithm can not only approach the precision of FS, but also increase the coding speed greatly. That is to say, fast algorithm can improve encoding speed and reduce coding time under the condition of guaranteeing the encoding quality of depth maps. This paper has a significant improvement in maximum parallelism based on the new parallel approach proposed by LPM. From the experimental results and analysis, it can be verified that the depth map has a large area of smooth area, so a fast search can be used for block matching of the depth maps. This parallel scheme greatly saves the time required for 3D depth maps coding, and improves computational efficiency and resource utilization. The next step in this work will continue to explore the characteristics of the depth map fast search algorithm, which can take full advantage of the characteristics of the reconfigurable array structure in the mapping process.

ACKNOWLEDGMENT

This work was supported by the National Natural Science Foundation of China (No. 61834005, 61772417, 61802304, 61602377, 61874087, 61634004), Shaanxi Provincial Co-ordination Innovation Project of Science and Technology (No. 2016KTZDGY02-04-02), Shaanxi Provincial Key R&D Plan (No. 2017GY-060), Shaanxi International Science and Technology Cooperation Program (No. 2018KW-006).

REFERENCES

[1] Sullivan, G. J., Boyce, J. M., Chen, Y., Ohm, J. R., Segall, C. A., & Vetro, A. “Standardized extensions of high efficiency video coding (HEVC)”. *IEEE Journal of selected topics in Signal Processing*, (2013).*7*(6), 1001-1016.

[2] Tech, G., Chen, Y., Müller, K., Ohm, J. R., Vetro, A., & Wang, Y. K. “Overview of the multiview and 3D extensions of high efficiency video coding”. *IEEE Transactions on Circuits and Systems for Video Technology*, (2016).*26*(1), 35-49.

[3] Müller, K., Schwarz, H., Marpe, D., Bartnik, C., Bosse, S., Brust, H., & Tech, G. “3D high-efficiency video coding for multi-view video and depth data”. *IEEE Trans. Image Processing*, (2013).*22*(9), 3366-3378.

[4] Lei, J., Li, S., Zhu, C., Sun, M. T., & Hou, C. “Depth Coding Based on Depth-Texture Motion and Structure Similarities”. *IEEE Trans. Circuits Syst. Video Techn.*, (2015). *25*(2), 275-286.

[5] Zhu, C., Li, S., Zheng, J., Gao, Y., & Yu, L. “Texture-aware depth prediction in 3D video coding”. *IEEE Transactions on Broadcasting*, (2016). *62*(2), 482-486.

[6] Mora, E. G., Jung, J., Cagnazzo, M., & Pesquet-Popescu, B. “Initialization, limitation, and predictive coding of the depth and texture quadtree in 3D-HEVC”. *IEEE Transactions on Circuits and Systems for Video Technology*, (2014).*24*(9), 1554-1565.

[7] Conceição, R., Avila, G., Corrêa, G., Porto, M., Zatt, B., & Agostini, L. “Complexity reduction for 3D-HEVC depth map coding based on early skip and early DIS scheme”. (2016, September). In *Image Processing (ICIP), 2016 IEEE International Conference on* (pp. 1116-1120). IEEE.

[8] Schwarz, H., Bartnik, C., Bosse, S., Brust, H., Hinz, T., Lakshman, H., ... & Tech, G. “3D video coding using advanced prediction, depth modeling, and encoder control methods”. (2012, May). In *Picture Coding Symposium (PCS), 2012*(pp. 1-4). IEEE.

[9] Yu, F., Hui, M., Han, W., Wang, P., Dong, L. Q., & Zhao, Y. J. “The application of improved block-matching method and block search method for the image motion estimation”. *Optics Communications*, (2010).*283*(23), 4619-4625.

[10] Müller, K., Schwarz, H., Marpe, D., Bartnik, C., Bosse, S., Brust, H., & Tech, G. "3D high-efficiency video coding for multi-view video and depth data". *IEEE Trans. Image Processing*, (2013).*22*(9), 3366-3378.

[11] Chen, Y., Tech, G., Wegner, K., & Yea, S. "Test model 9 of 3D-HEVC and MV-HEVC". (2014).*Joint Collaborative Team on 3D Video Coding Extension Development of ITU-T SG*, *16*.

[12] Tang, X. L., Dai, S. K., & Cai, C. H. "An analysis of TZSearch algorithm in JMVC". (2010).*International Conference on Green Circuits and Systems* (pp.516-520). IEEE.

[13] E. Marth and G. Marcus, "Parallelization of the x264 encoder usingopenCL,"in Proc.ACM SIGGRAPH, Jul. 2010, pp.1–72.

[14] Xiao Z, Baas B M. "A 1080p H.264/AVC Baseline Residual Encoder for a Fine-Grained Many-Core System". *IEEE Transactions on Circuits & Systems for Video Technology,* 2011, 21(7):890-902.

[15] Minhua, Z. "Configurable and CU-group level parallel merge/skip". (2012). *JCTVC-H0082, JCT-VC of ITU-T SG16WP3and ISO/IEC, CA, USA*.

[16] Yu, Q., Zhao, L., & Ma, S.. "Parallel AMVP candidate list construction for HEVC". (2012, November) *Visual Communications and Image Processing (VCIP), 2012 IEEE*(pp. 1-6). IEEE.

[17] Yun, Z. , Jiang, L. , Wang, S. , Huang, X. , Song, H. , & Li, X. . "Design of reconfigurable array processor for multimedia application". *Multimedia Tools and Applications,* 2018, 77(3): 3639-3657.

[18] Saldanha, M., Sanchez, G., Zatt, B., Porto, M., & Agostini, L. "Energy-aware scheme for the 3d-hevc depth maps prediction". *Journal of Real-Time Image Processing,* (2017).13(1), 1-15.

[19] Saldanha, M., Sanchez, G., Zatt, B., Porto, M., & Agostini, L. "Complexity reduction for the 3D-HEVC depth maps coding". (2015, May). *Circuits and Systems (ISCAS), 2015 IEEE International Symposium on* (pp. 621-624). IEEE.

[20] Yan, C., Zhang, Y., Dai, F., & Li, L. "Highly parallel framework for HEVC motion estimation on many-core platform". (2013, March). *Data Compression Conference (DCC), 2013* (pp. 63-72). IEEE.

[21] Yan, C., Zhang, Y., Xu, J., Dai, F., Zhang, J., Dai, Q., & Wu, F. "Efficient parallel framework for HEVC motion estimation on many-core processors". *IEEE Transactions on Circuits and Systems for Video Technology*, (2014). 24(12), 2077-2089.

[22] Wang, X., Şekercioğlu, Y. A., Drummond, T., Natalizio, E., Fantoni, I., & Frémont, V. "Fast Depth Video Compression for Mobile RGB-D Sensors". *IEEE Transactions on Circuits and Systems for Video Technology*, (2016).26(4), 673-686

[23] Li, Peng , and H. Tang . "A low-power VLSI implementation for fast full-search variable block size motion estimation." *International Journal of Electronics ,* 100.7-9(2013):1240-1255.