An Improved HEXBS Motion Estimation Algorithm and Hardware Design for H.264/AVC

Wei Wang, Jie Hu, Jie Li, Yongmeng Dong, Guanyu Wang, Jun Yuan

College of Electronics Engineering, Chongqing University of Posts and Telecommunications, 400065, china

Abstract. Motion estimation plays an important role in the H.264 video compression, it occupies most parts of the entire computation. An improved hexagon-based search algorithm (HEXBS) and its hardware implementation is proposed. The number of the search points of the first template is reduced. The improved algorithm adopts two types search pattern: large cross search pattern and small cross search pattern. In addition, the algorithm can make full use of hardware resources, it works in a parallel way efficiently and is able to process very high definition real-time videos. The proposed architecture is simulated by Modalism 6.5 SE, synthesized to the Xilinx Virtex4 XC4VLX15 FPGA device. The simulation results show that this architecture can achieve 110MHz, and 140M pixels can be processed per second, which meets the requirements of real-time applications.

Keywords: motion estimation, Hexagon Search algorithm, H.264, FPGA.

1. Introduction

H.264/AVC (AVC) is the video compression standard [1], developed by the ITU-T Video Coding Experts Group (VCEG) and the ISO/IEC Moving Picture Experts Group (MPEG). It contains a number of new features that allow it to compress video much more effectively than older standards and to provide more flexibility for application to a wide variety of network environments [2]. Compared with other video compression standard, H.264 has high compression ratio, low bit rate, better encoding performance and more suitable for industrial application.

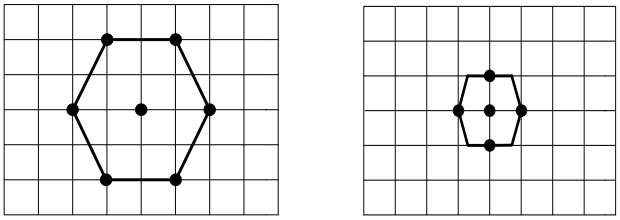
Motion Estimation (ME) is a very important part in H.264 coding technology. It explores and reduces the temporal redundancy between current frame and reference frame, which is considered as the most complex part in the H.264/AVC encoder requiring between 60% and 90% of the encoding time [3]. Full search algorithm (FS) is a key motion estimation algorithm, it has the highest search accuracy but it brings huge computational complexity. Some new fast estimation algorithms are proposed, such as three-step search (TSS) [2], four-step search (FSS) [6], diamond search algorithm (DS) [7], Hexagon-based search algorithm (HEXBS) [4] and so on. These fast algorithm improved the coding performance, reduced the search points and get a good encoding speed. However, there are some disadvantage, for example, the average search points are large, the complexity is high, and the search accuracy is low and so on.

In this paper, the traditional Hexagon Search algorithm is improved, the improved algorithm only uses two simple search patterns, large cross search pattern and small cross search pattern. At the same time, the architecture of the proposed algorithm is proposed, with full use of FPGA memory resources, the reference block and the current block of the proposed algorithm is processed and storage parallels, and this results in the search speed of motion vector is rising. The structure mainly includes the current frame and reference frame storage module, calculation module and SAD value calculation module and SAD value comparison module.

The rest of this paper is organized as follows: Section II analyzes the algorithm. The proposed architecture is presented in Section III. Section IV analyzes the simulation and synthesis results. Finally, Section V concludes the paper.

2. Improved HEXBS Motion Estimation Algorithm

HEXBS algorithm is illustrated in figure1[4], it uses two templates to search, in the motion estimation, the first search using large hexagon template matching, until the minimum block distortion point (MBD) is located at large template center, and then matching with the small template, at last the MBD point is the best matching point.



(a) Large hexagon template (b) small hexagon template

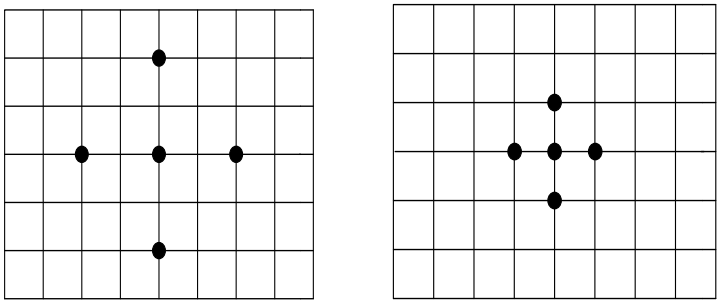
Fig.1. HEXBS Search Templates

The characteristics of the hexagon algorithm is: it analyzes the basic rule of motion vector in the video image, it uses two kinds of hexagon search plates. Compared with other fast search algorithms, it has a faster search speed, but the HEXBS do not consider adequately the relevant video characteristics of the image frames [5] and the center-biased characteristics. In addition, the first step size of HEXBS is too large, it is easy to calculate too many search points, making it possible to miss the optimum point [4]. In this paper, the improved algorithm also adopts two search patterns, as shown in figure2, one is large cross search pattern, and it consists of a central point and four points separated by a pixel around the central point. The second is small cross search pattern, it consists of center points and the surrounding four points. The processes of improved algorithm as follows:

Step1: First, the large cross search pattern is used, for processing the block as the center, in the search window compare the five search points, and get the MBD. If the MBD point is the center of the pattern, turns to step 3, otherwise go to step 2.

Step2: assuming the above MBD point as the center of a new large cross search pattern, and calculating the five points, until the best matching point in the center position. If the MBD is the center point, turns to step3. Otherwise, repeat this step continuously.

Step3: the small cross search pattern is used. the above got MBD point as the center of the small cross search pattern, then check the five search points, if the MBD is not the center point, use the MBD as new point of the small cross search pattern and search until the best matching point in the center position. Get the MBD is the final result. The flow chart of the whole system are as Fig.3.



(a) large cross search pattern (b)small cross search pattern

Fig.2. Search Pattern of Proposed Algorithm

Compared with large hexagon mode, the improved algorithm reduce two search points under the condition of the same template mobile. The template can be used to match the points both on the horizontal and vertical direction, but the large hexagon mode only used to match well on the horizontal direction.

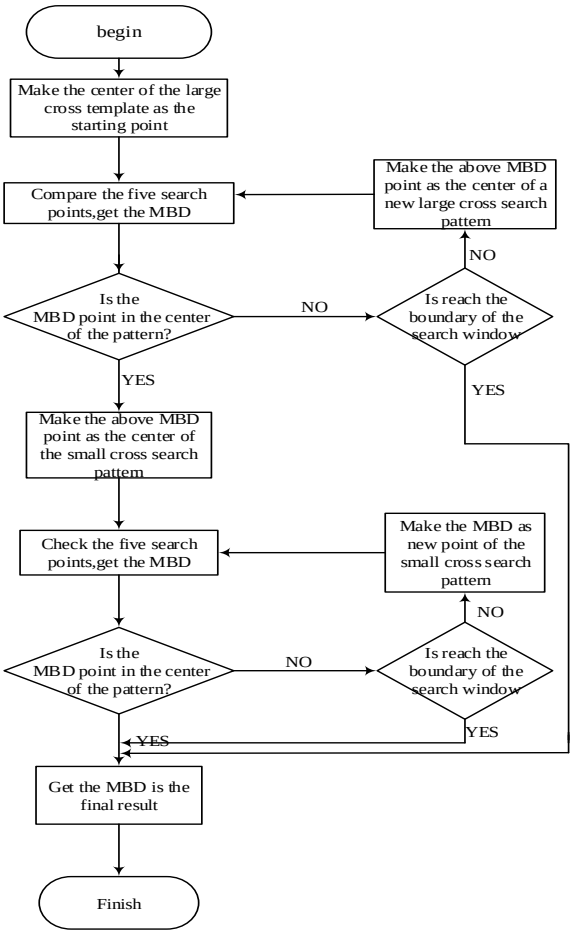


Fig.3. the flow chart of the system

3. Proposed Architecture For the Algorithm

In this section, we describe the proposed architecture of the proposed algorithm. It includes control unit, buffer unit, data switch, the reference block memory, the current block memory and comparing unit. A block diagram of the entire system is shown in Fig.4. The control module is to control the algorithm implementation of each step through a finite state machine, according to the need of the current frame pixel and reference frame pixel to be stored in the current memory and the reference memory. SAD value calculation module is designed with use of PE (processing unit module) module. Comparing unit module mainly designed with a comparator.

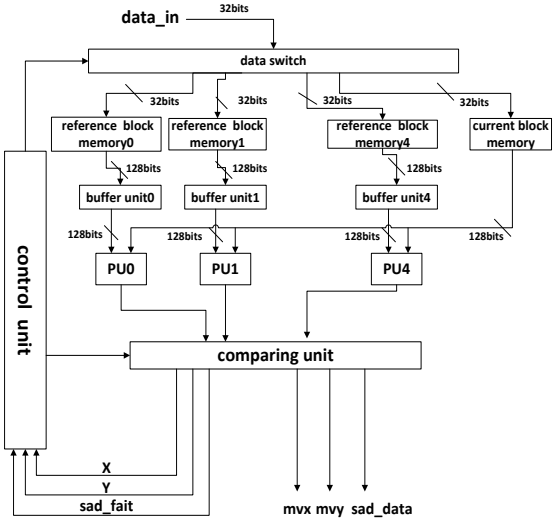


Fig.4. the hardware architecture

3.1control Unit

Controller is responsible for the control system of synchronization between the various modules and given external memory addressing the address of the signal, the control unit is designed as the finite state machine as shown in Fig.5, it includes system control state machine, large cross search state machine and small cross search state machine..

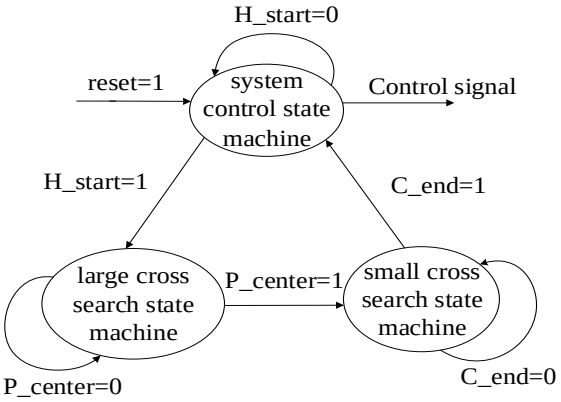


Fig.5.The state transition schematic of controller with FST

3.2 The current block memory

The current block is used to store 16x28 bits current block pixels. It includes a 128 bits register and a width of 128 bits, depth of 16 stand-up RAM. 128 bits register is used to check the current block line of pixels. 256 bits data is input from external memory, the input data of the first clock cycle is included five reference blocks and the first four pixels of the first line in the current block, the input data of the second clock cycle is five reference blocks and the fifth to the eighth pixels of the first line in the current block.

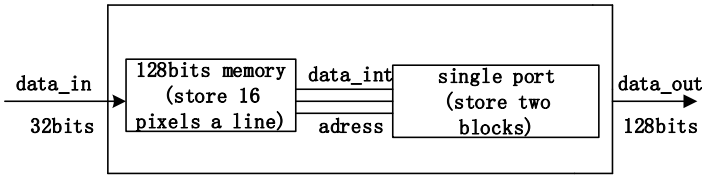


Fig.6. Current block memory

3.3 The reference block memory

In order to output the pixels of five blocks at the same time, the five reference block memories are need to store reference block. With redistribution of the input data, it can realize parallel store the five reference blocks. The structure of the memory as shown in Fig.7. It consisted of a 128bits memory and a single port. The single port interleaved two reference blocks respectively, in this way, the speed of the system was improved.

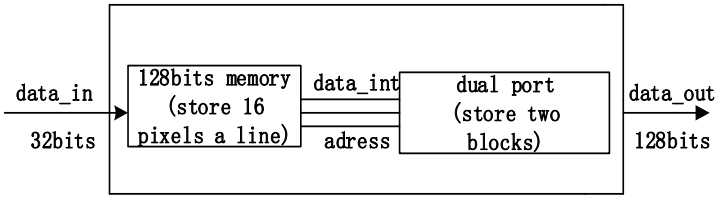


Fig.7. Reference block memory

3.4The PU

In order to calculate five SAD value of the pixel block at the same time, we need to use the five PU at the same time. Each PU consist of 16 parallel PE (Processing Element) array, a parallel adder and an accumulator. So each PU can calculation a macro block line of pixels at the same time, in order to reduce the critical path, we increased the level of pipeline stages between the adder and the accumulator. PU structure as shown in Fig.8.

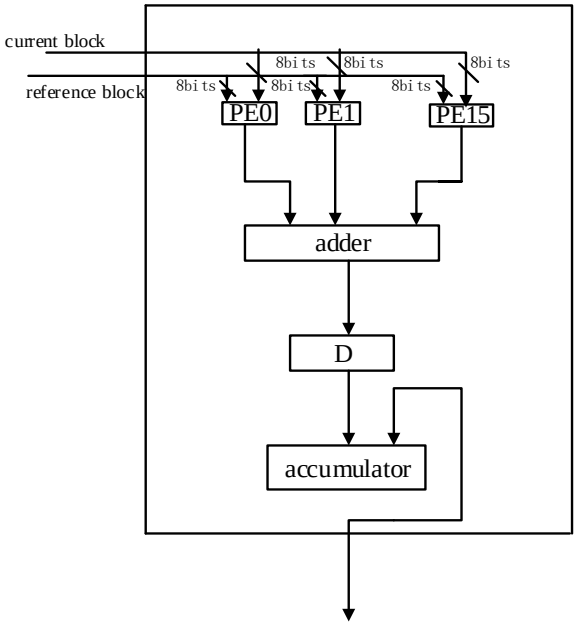


Fig.8. the PU

3.5The PE

As a small unit of PU, PE used for the poor, the absolute value between the current block and the reference block. In order to reduce the operation time and improve the speed of the system work, we use two levels of pipeline stages, PE structure diagram as shown in Fig.9.

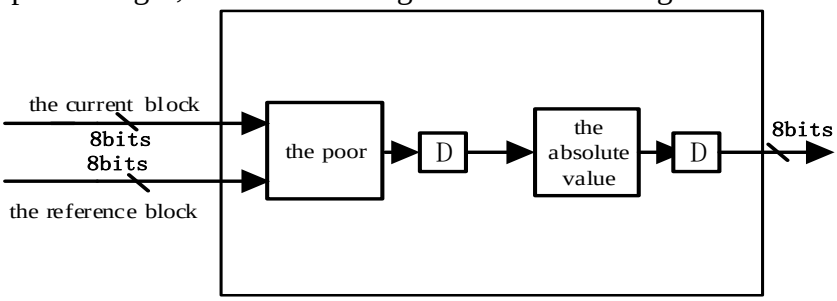


Fig.9. the PE

4. Experiment Results

The design was simulated by Modalism 6.5 SE and synthesized to a Xilinx Virtex4 XC4VLX15 FPGA device with the Simplify Pro 9.6.2 synthesis tool. The results show the clock frequency can reach 110 MHz, it can handle more than 141M pixels per second, fully satisfied HDTV (1920x1088@60fps) the requirement of real-time application.

Table 1. The average search points of proposed algorithm compared with other algorithms

	FS	TSS	NTSS	DS	HEXBS	This work
Football	1089	36	26	20	16	12
Foreman	1089	36	24	16	11	7
Highway	1089	36	24	14	11	6
Stefan	1089	36	22	15	12	8
Bridge(far)	1089	36	20	13	11	5

From the table 1 and table 2, we can see that the search points are reduced obviously, the PSNR value slightly higher than others algorithms. Compared with the HEXBS algorithm, the search points of the proposed algorithm will be lesser at average 36% and the PSNR has been slightly increased.

Table 2. PSNR values of proposed algorithm compared with other algorithms

	FS	TSS	NTSS	DS	HEXBS	This work
Football	41.11	41.10	41.09	41.06	41.05	41.01
Foreman	41.48	41.41	41.43	41.45	41.42	41.45
Highway	41.16	41.12	41.13	41.15	41.15	41.16
Stefan	39.91	39.85	39.87	38.90	39.89	39.90
Bridge(far)	40.97	40.95	40.95	40.96	40.96	40.95

5. Conclusion

An improved motion estimation algorithm and its hardware design is proposed. The pipeline parallel architecture is used, with which the speed of motion vector search and the system clock frequency is greatly improved. This architecture can handle more than 141M pixels per second, fully meet the requirements of real-time video processing.

REFERENCES

[1] 'Draft ITU-T Recommendation and Final Draft International Standard of Joint Video Specification', ITU-T Recommendation H.264 and ISO/IEC14496-10 AVC, Joint Video Team, May 2003.

[2] T. Wiegand, G.J. Sullivan, 'Overview of the H.264/AVC Video Coding Standard', IEEE Trans. Circuits and Systems for Video Technology, 13(7) p.560-576, 2003.

[3] G. Ruiz, J. A. Michell, 'An efficient VLSI architecture of fractional motion estimation in H.264 for HDTV', J. Signal Processing System, 62(3) p.443-457, 2011.

[4] Z. Wei, S. Hao-shan, Z. Xin, et al. 'A suitable H.264 high-performance block-matching motion estimation algorithm', Computer Application Research, 2007, pp.76.

[5] T. Koga, K. Iinuma, A. Hirano, Y. Iijima, and T. Ishiguro, 'Motion compensated underframe coding for video conferencing', Proc. NTC81, p. G5.3.1-G5.3.5. 1981.

[6] L.M. Po. And W. C. Ma, 'A novel four-step search algorithm for fast block motion estimation', IEEE Trans. Circuits and Systems for Video Technology, 6(3) p.313-317, 1996.

[7] S. Zhu and K.K. Ma, 'A new diamond search algorithm for fast block-matching motion estimation', IEEE Trans. Image Processing, 9(2) p.287-290, 2000.