PAPER Special Section on Solid-State Circuit Design — Architecture, Circuit, Device and Design Methodology

An 8 × 8/4 × 4 Adaptive Hadamard Transform Based FME VLSI Architecture for 4 K × 2 K H.264/AVC Encoder

Yibo FAN[†a)], *Member*, Jialiang LIU[††], Dexue ZHANG[††], Xiaoyang ZENG[†], *and* Xinhua CHEN[††], *Nonmembers*

SUMMARY Fidelity Range Extension (FREXt) (i.e. High Profile) was added to the H.264/AVC recommendation in the second version. One of the features included in FREXt is the Adaptive Block-size Transform (ABT). In order to conform to the FREXt, a Fractional Motion Estimation (FME) architecture is proposed to support the 8×8/4×4 adaptive Hadamard Transform (8 × 8/4 × 4 AHT). The 8 × 8/4 × 4 AHT circuit contributes to higher throughput and encoding performance. In order to increase the utilization of SATD (Sum of Absolute Transformed Difference) Generator (SG) in unit time, the proposed architecture employs two 8-pel interpolators (IP) to time-share one SG. These two IPs can work in turn to provide the available data continuously to the SG, which increases the data throughput and significantly reduces the cycles that are needed to process one Macroblock. Furthermore, this architecture also exploits the linear feature of Hadamard Transform to generate the quarter-pel SATD. This method could help to shorten the long datapath in the second-step of two-iteration FME algorithm. Finally, experimental results show that this architecture could be used in the applications requiring different performances by adjusting the supported modes and operation frequency. It can support the real-time encoding of the seven-mode 4 K × 2 K@24 fps or six-mode 4 K × 2 K@30 fps video sequences.

key words: *fractional motion estimation, adaptive block-size hadamard transform, H.264/MPEG4 AVC, 4 K × 2 K, quad full high definition*

1. Introduction

H.264/AVC is the latest video-coding standard jointly developed by ITU-T Video Coding Experts Group (VCEG) and ISO/IEC Moving Picture Experts Group (MPEG). After the first version, H264/AVC didn't stop its development. Afterwards Fidelity Range Extension (i.e. High Profile), Scalable Video Coding (SVC) and Multi-view Video Coding (MVC) extensions were added to H.264/AVC standard in succession. Nowadays, H.264/AVC standard can be widely applied to all kinds of multimedia devices or occasions because of its more predominant performance.

Variable block size motion estimation (VBSME) is one of the most important techniques employed by H.264/AVC standard. And this technique fully exploits the temporal redundancy to provide much better compression performance. VBSME technique splits one Macroblock (MB) into one 16 × 16, two 16 × 8, two 8 × 16, or four 8 × 8 partitions, and that every 8 × 8 partition can be further split into one 8 × 8, two 8 × 4, two 4 × 8 or four 4 × 4 sub-partitions. There are total 41 sub-blocks, as shown in Fig. 1. VBSME needs to find out the best quarter-pel precision motion vector (MV) for every sub-block.

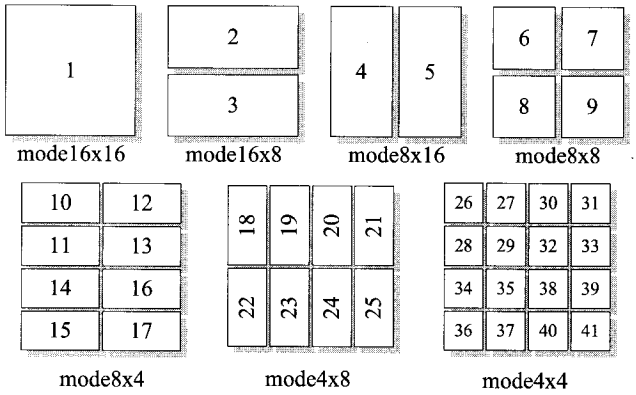


Fig. 1 41 sub-blocks and 7 modes in one MB.

However, H.264/AVC standard doesn't specify how to implement motion estimation (ME). In general, most of the implementation works separate Motion Estimation into two steps: integer motion estimation (IME) and fractional motion estimation (FME). Firstly, IME performs motion search within the entire search window to find a best matching block pointed to by integer-pel precision motion vector (IMV) for every 41 sub-blocks. In the second step, FME performs motion search around the center pointed to by IMV and further refines 41 IMVs into quarter-pel precision [4]. In the whole ME process, FME plays an important role in improving the video quality [2], [4] and reducing the bit rate [3]. However, FME will occupy almost over 45% of the computational complexity of encoding process [2], [4]. As a result, in order to meet the requirement of high performance, many hardware implementations have been proposed, which will be introduced in Sect. 2.2.

In the other hand, Adaptive Block-size Transform (ABT) was added to the later FREXt (i.e. High Profile). ABT technique requires that the transform adaptively switches between 8 × 8 and 4 × 4 transforms. According to our experiments on high definition (HD) application, this technique can increase about 0.017 dB PSNR and reduce about 0.8% bit rate on average. In addition, 8 × 8 and 4 × 4 adaptive Hadamard Transform (8 × 8/4 × 4 AHT) is a subset of ABT technique, and Hadamard Transform (HT) is the main part of Sum of Absolute Transformed Difference (SATD) which

Manuscript received August 10, 2011.
Manuscript revised November 4, 2011.

[†]The authors are with the State Key Lab. of ASIC & System, Fudan University, Shanghai, 200240, China.
[††]The authors are with the College of Information Science and Engineering, Shandong University of Science and Technology, Qingdao, 266510, China.
a) E-mail: fanyibo@fudan.edu.cn
DOI: 10.1587/transele.E95.C.447

is widely used for estimating cost of rate distortion. In this paper, a FME architecture that applies the $8 \times 8/4 \times 4$ AHT scheme to the half-pel SATD calculation is proposed. And this architecture can be applied to those applications supporting FRExt feature.

The rest of this paper is organized as follows. In Sect. 2, we firstly point out the challenges of FME and then introduce some other's efforts. Section 3 describes the related algorithms in our design. In Sect. 4, the detail of proposed FME architecture is described. The hardware implementation results and comparisons are shown in Sect. 5. Finally, the conclusion is drawn in Sect. 6.

2. Challenges and Previous Works

2.1 The Challenges of FME

As one part of Motion Estimation, FME process isn't explicitly specified in the H.264/AVC standard. As JM [15] has provided a two-iteration algorithm, most of designs adopt this algorithm for its easiness to implement in hardware, such as Refs. [2]–[4]. The search process of this algorithm is separated into two steps, as shown in Fig. 2. In the first step, half-pel refinement is performed to search the best half-pel candidate around the best integer-pel candidate derived from IME. In the second step, the quarter-pel refinement for the best quarter-pel candidate around the best half-pel candidate is performed. The disadvantage of this algorithm is that the second step doesn't work until the accomplishment of the first step. As a result, this two-step method seriously limits the searching speed.

Another challenge of FME comes from the multiple

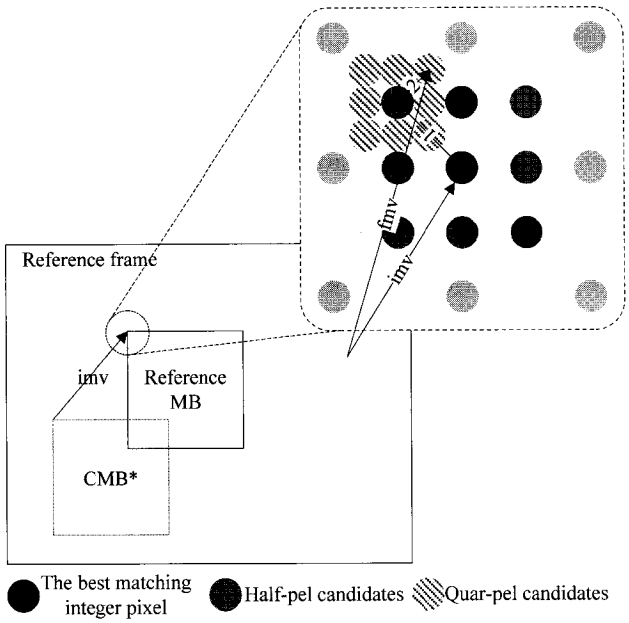


Fig. 2 Two-step search of FME in JM software.

modes caused by VBSME. As mentioned in Sect. 1, VBSME splits one MB into 41 sub-blocks and these sub-blocks are organized as 7 modes, as shown in Fig. 1. What FME has to do is to find out the best quarter-pel matching blocks for these 41 sub-blocks and to get the best partition of corresponding MB according to the minimum cost. For one thing, we must design an architecture that supports all cases of 7 modes. For another thing, if this architecture supports all 7 modes, it will need very high performance to meet the requirement of the real-time HD video encoding.

2.2 Previous Works

Nowadays, many VLSI architectures for FME have been proposed. Some of them support two-step full modes search, and most of them employ low complexity algorithm for high performance applications at the cost of visual quality drop.

Chen et al. [2] proposed an architecture based on the two-step full search algorithm. This architecture unifies different size blocks into 4×4 size blocks, which obtains 100% hardware utilization. Meanwhile, Chen et al. initiatively employed 9 separate SATD Calculation Units (SCUs) to comprise the SATD Generator (SG). Besides, Chen's architecture also supports full modes, however, which also limits its performance. Finally, they adopted one Advanced Mode Pre-Decision Algorithm (AMPA) to reduce 7 modes to 3 modes only for SDTV specification [2]. Wu's architecture [3] also supports all modes. It employs three interpolator (IP) engines to share two SGs comprised of 9 separate SCUs, which obtains the high performance by increasing the data throughput. However, it still takes more than 600 cycles @154 MHz only to support the 1080p specification. Furthermore, to schedule the 3 IPs to share two SGs will be a complicated task.

The full-mode and two-step search algorithm limits the performance of FME. So many architectures based on the low complexity algorithm are proposed. Wang et al. [14] exploited the unimodal error surface feature of FME and only explored the neighbor positions around the minimum one to skip other unlikely ones. Though this method reduces the hardware cost by reducing search points, it doesn't resolve the problem caused by the two-iteration search. Lin et al. [4], Kuo et al. [5] and Ta et al. [6] proposed their single-iteration architectures based on the prediction-based directional FME algorithm. This algorithm only searches six candidates, so it is hard to find the best matching candidates, especially for the high motion video sequences. Huang et al. [8] proposed a high parallel architecture meeting SHV application. They firstly removed the modes below 8×8 and then exploited the edge detection technique to further reduce modes. Yang et al. [10] proposed a Adaptive Mode Select (AMS) scheme to reduce the encoding modes on the purpose of encoding $4\,\mathrm{K} \times 2\,\mathrm{K}$@30 fps in real-time. The AMS scheme can select the encoding modes adaptively according to the encoding modes of neighborhood encoded MBs. Tsung et al. [9] adopted a completely different so-

Table 1 The number of modes supported by some others' works and the methods to reduce the modes.

Ref.	Methods	Modes[1]	Max. Spec.
[2]	MPD[2]	3	SDTV@30fps
[5]	MPD reduce candidates	2	1080p@60fps
[7]	remove some modes	5-7	1080p@30fps
[8]	remove some modes & edge detect	2	4Kx4K@60fps
[10]	AMS[3]	1-7	4Kx2K@30fps
[3][4]	full search	7	1080p@20fps
our work	single iteration	6	4Kx2K@30fps
		7	4Kx2K@24fps

1: The number of the modes supported by other works;

2: Mode Pre-Decision;

3: Adaptive Mode Select;

lution. They abandoned the search method defined by JM reference software, and then utilize the linearity of HT to obtain the HT coefficients of sub-pels. This method reduces the high bandwidth caused by 6-tap FIR, but at the same time, their architecture results in the dramatic increase of the hardware cost.

Table 1 lists the number of supporting modes in some previous works. After analyzing their architectures, it is found that mode reduction is one effective method to improve processing performance. However, this method will cause dramatic degradation in visual quality. In this paper, a FME architecture supports full modes is proposed. Of course, our architecture also can adjust the number of the supporting modes according to the performance requirement.

3. Related Algorithms

3.1 Reusable Relationship Between 8 × 8 and 4 × 4 HT

Firstly, we give out the 8 × 8 and 4 × 4 HT matrices, Eq. (1) and Eq. (2). Here we ignore the scaling factor of transform matrices.

$$\mathbf{H}_{8\times8} = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 & 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 \\ 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 \\ 1 & -1 & 1 & -1 & -1 & 1 & -1 & 1 \\ 1 & 1 & -1 & -1 & -1 & -1 & 1 & 1 \\ 1 & -1 & -1 & 1 & -1 & 1 & 1 & -1 \end{bmatrix} \quad (1)$$

$$\mathbf{H}_{4\times4} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \end{bmatrix} \quad (2)$$

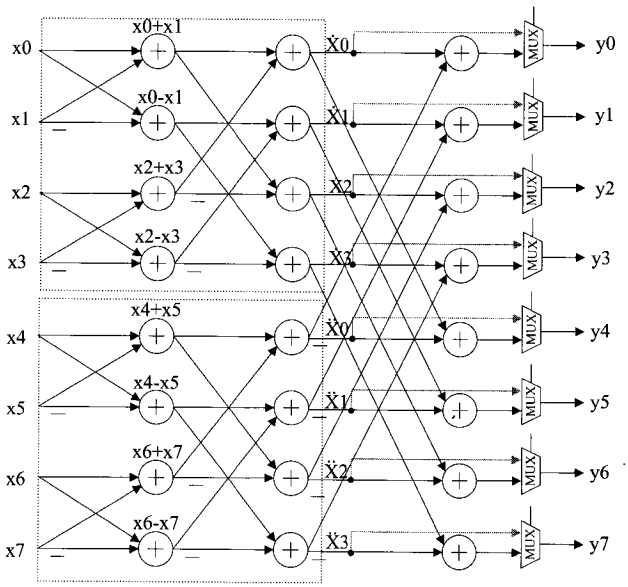


Fig. 3 1D 8 × 8/4 × 4 HT circuit.

By analyzing the relationship between $H_{4\times4}$ matrix and $H_{8\times8}$ matrix, we can easily get the expression of $H_{8\times8}$ matrix as Eq. (3).

$$\mathbf{H}_{8\times8} = \begin{bmatrix} \mathbf{H}_{4\times4} & \mathbf{H}_{4\times4} \\ \mathbf{H}_{4\times4} & -\mathbf{H}_{4\times4} \end{bmatrix} \quad (3)$$

Assume we have a 8 × 1 vector, $\vec{x}$, which can be expressed as two 4 × 1 vectors, $\vec{x}_0$ and $\vec{x}_1$, as shown in Eq. (4).

$$\vec{x} = \begin{bmatrix} \vec{x}_0 \\ \vec{x}_1 \end{bmatrix} \quad (4)$$

Now, we operate one dimension (1D) HT on vector $\vec{x}$, which is expressed as Eq. (5).

$$\vec{X} = \mathbf{H}_{8\times8} \cdot \vec{x} = \begin{bmatrix} \mathbf{H}_{4\times4} \cdot \vec{x}_0 + \mathbf{H}_{4\times4} \cdot \vec{x}_1 \\ \mathbf{H}_{4\times4} \cdot \vec{x}_0 - \mathbf{H}_{4\times4} \cdot \vec{x}_1 \end{bmatrix} \quad (5)$$

From Eq. (5), It can be seen that one 1D 8 × 8 HT unit contains two 1D 4 × 4 HT units. A reusable circuit that supports one 8 × 8 HT and two 4 × 4 HT can be designed. Figure 3 shows this 1D 8 × 8/4 × 4 HT circuit. The dotted line part indicates the 1D 4 × 4 HT. In our design, this 1D 8 × 8/4 × 4 adaptive HT circuit is used to implement the 2D 8 × 8/4 × 4 HT in SATD calculation.

3.2 The Linearity of HT

The linearity of HT is defined as Eq. (6):

$$HT(a\vec{x} + b\vec{y}) = aHT(\vec{x}) + bHT(\vec{y}) \quad (6)$$

Here, HT() presents HT operation. "a" and "b" are constants. "$\vec{x}$" and "$\vec{y}$" are transformed vectors.

Let "**Q**" and "**C**"denote the quarter-pel candidate block (QCB) matrix and the current encoding block (CB) matrix respectively. "**H**′" and "**H**″" represent the half-pel

(or integer-pel) candidate block matrix used for generating quarter-pel candidate block matrix, "**Q**", by bilinear filter that is defined in standard [1], as shown in Eq. (7).

$$\mathbf{Q} = floor((\mathbf{H}' + \mathbf{H}'' + \mathbf{1})/2) \tag{7}$$

Now, the HT of the differences between the QCB and the CB can be expressed as Eq. (8).

$$\begin{aligned} HT(\mathbf{Q} - \mathbf{C}) &= HT\left(floor\left(\frac{\mathbf{H}' + \mathbf{H}'' + \mathbf{1}}{2}\right) - \mathbf{C}\right) \\ &= HT\left(floor\left(\frac{\mathbf{H}' - \mathbf{C}}{2} + \frac{\mathbf{H}'' - \mathbf{C}}{2} + \frac{1}{2}\right)\right) \end{aligned} \tag{8}$$

The floor operation in Eq. (8) doesn't have the linear feature. Here, we have to make a change as Eq. (9).

$$\begin{aligned} &HT(\mathbf{Q} - \mathbf{C}) \\ &= HT\left(floor\left(\frac{\mathbf{H}' - \mathbf{C}}{2} + \frac{\mathbf{H}'' - \mathbf{C}}{2} + \frac{1}{2}\right)\right) \\ &\approx floor(HT((\mathbf{H}' - \mathbf{C}) + (\mathbf{H}'' - \mathbf{C}) + 1)/2) \\ &= floor((HT(\mathbf{H}' - \mathbf{C}) + HT(\mathbf{H}'' - \mathbf{C}) + 1)/2) \end{aligned} \tag{9}$$

Here, $HT(\mathbf{H}' - \mathbf{C})$ and $HT(\mathbf{H}'' - \mathbf{C})$ are the HT coefficients matrixes of the differences between half-pel candidate block (HCB) and CB. From Eq. (9), it is seen that the HT of the differences between QCB and CB can be derived from the HTs of differences between two neighboring HCBs and CB. These two neighboring HCBs are used for generating the QCB by bilinear filter. According to Eq. (9), $HT(\mathbf{Q} - \mathbf{C})$ can be generated following the generation of $HT(\mathbf{H}' - \mathbf{C})$ and $HT(\mathbf{H}'' - \mathbf{C})$. In this way, based on Eq. (9), our design can simplify the second iteration process of original two-iteration FME algorithm. And the bottleneck caused by two-iteration process can be solved.

The effect on the encoding efficiency after introducing the change in Eq. (9) is listed in Table 2. This change results in about 0.022 dB visual quality degradation and 0.67% bitrate raise. The calculation method is referred from Ref. [16].

Table 2 The effect on the encoding performance caused by the change in Eq. (9). (IPPP GoP, 30 fps, 50 frames, QP=16, 20, 24, 28, RD off Transform 8 × 8 Mode on)

Test Sequences (1920x1080)	PSNR change (dB)	bit-rate inc (%)
riverbed	-0.0107	+0.19
blue_sky	-0.0106	+0.29
pedestrian_area	-0.0184	+0.59
rush_hour	-0.0215	+1.00
station2	-0.0297	+0.85
sunflower	-0.0326	+1.12
tractor	-0.0291	+0.65
Average	**-0.0218**	**+0.67**

4. The Proposed FME Design

4.1 The Proposed Architecture

The proposed FME architecture is illustrated in Fig. 4. In this architecture, two 8-pel interpolators are employed. These two interpolators can continuously produce the half-pel candidates to SATD Generator (SG) in turn, which can increase the utilization of SG in unit time. In Fig. 4, SG is divided into two parts: one part is half-pel SATD calculation (HPU, Processing Unit for half pixels) and the other is quarter-pel SATD calculation (QPU, Processing Unit for quarter pixels). 9 HPUs can simultaneously process 9 candidates output from interpolators. Here, we employ 16 QPUs following the HPUs to generate the quarter-pel SATDs based on the linearity of HT. The HT circuit in HPU can support 8 × 8/4 × 4 adaptive Hadamard Transform. The bigger 8 × 8 blocks (including 8 × 8 block) are divided into 8 × n (n = 8, 16) and arranged into 8 × 8 HT, while the smaller 8 × 8 blocks run at 4 × 4 HT. Compared with those architectures which only supports the 4 × 4 HT, our design increase the data throughput and encoding efficiency. In addition, because we make use of the interval time of SATD calculation, the output of quarter-pel SATDs can be delayed to the end of half-pel search process, which could avoid the comparing operation of 9 half-pel candidates together with 16 quarter-pel candidates.

4.2 The Time-Sharing Scheme for SATD Generator

Figure 5 shows the conventional 4-pel interpolator [12]. In our architecture, two parallel 4-pel interpolators comprise one 8-pel interpolator. Our 8-pel interpolator can support

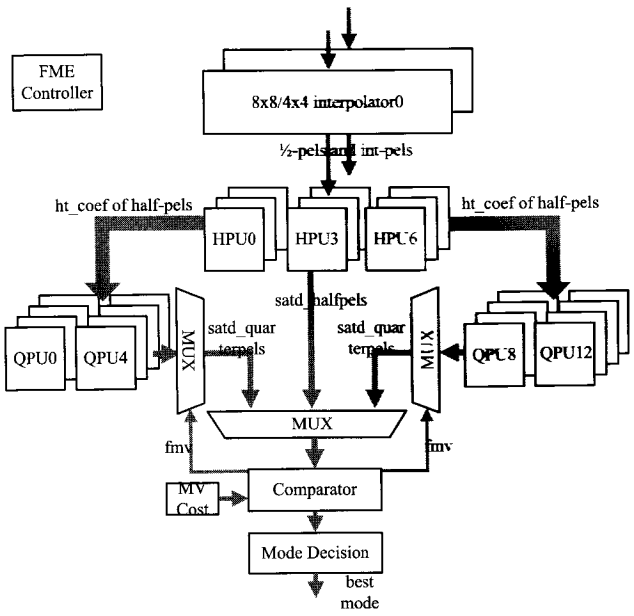


Fig. 4 The proposed FME architecture.

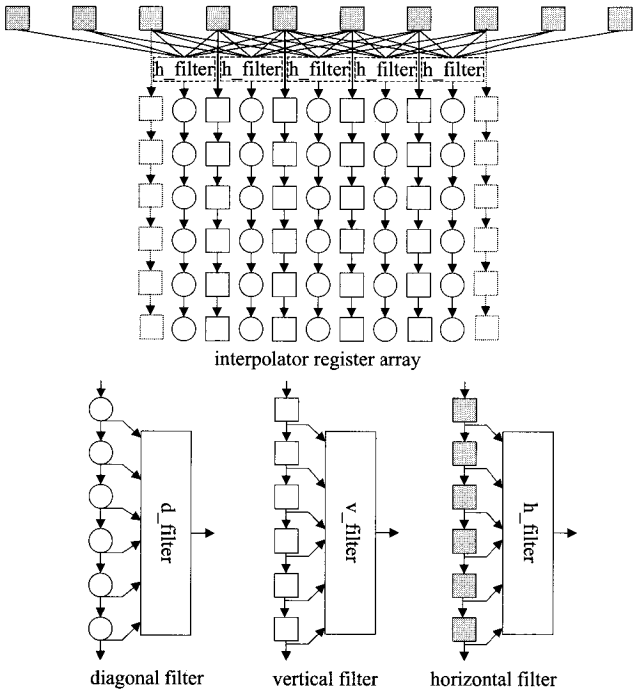


Fig. 5 The conventional 4-pel interpolator.

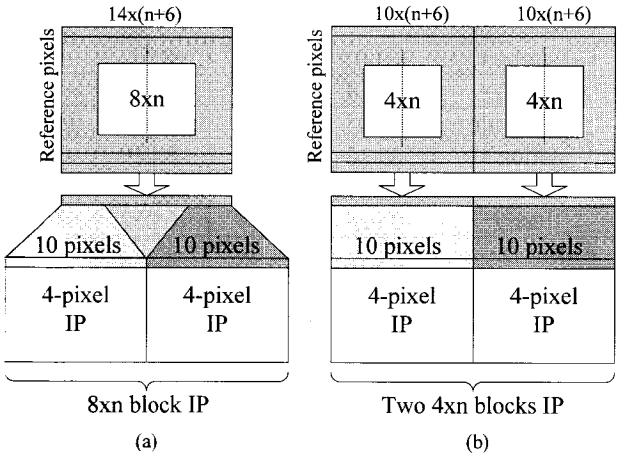


Fig. 6 The 8-pel interpolator comprised of two 4-pel interpolator and the distribution of the input pixel data.

not only the 8 pixels input interpolation but also two 4 pixels input interpolation. The bigger 8×8 blocks can be split into $8 \times n$ ($n = 8, 16$) blocks and perform 8-pel interpolation and 8×8 HT. The rest of small sub-blocks will be unified into $4 \times n$ ($n = 4, 8$) blocks, and two neighboring $4 \times n$ blocks perform 4-pel interpolation and 4×4 HT simultaneously. The distribution of the reference pixels for $4 \times n$ blocks or $8 \times n$ blocks is shown in Fig. 6.

The conventional architectures based on single interpolator contain a problem. The available data produced by the interpolator are not continuous, while SATD Generator has the capacity of accepting the data continuously. Only one interpolator will not meet the requirement of the SG's

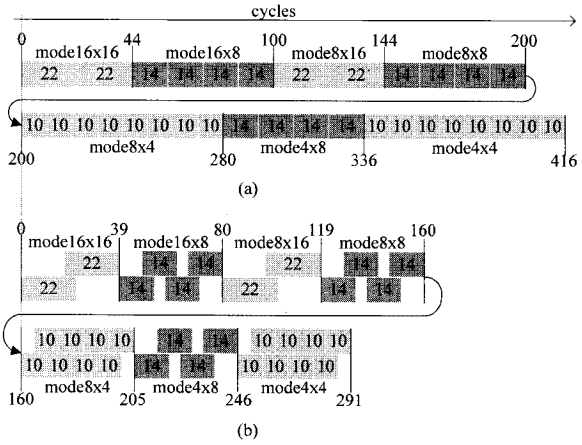
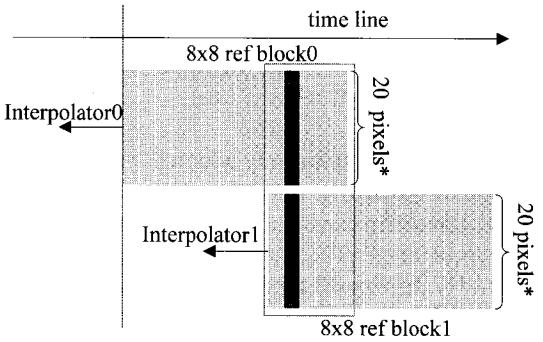


Fig. 7 The schedule of loading reference pixels for all the partitions in every mode (a) the case of the single interpolator (b) our two-interpolator scheme.



*: 20 pixels are expanded from 14 pixels for 8xn block

Fig. 8 The dataflow of 4×8 block0 switching to 4×8 block1.

throughput. To solve this problem, two interpolators are introduced in our architecture, which can continuously send the available data in turn to SATD Generator. These two interpolators keep SATD Generator working continuously. Figure 7 shows the schedule of loading reference pixels for all partitions in every mode. From Fig. 7(a), it is seen that if only one interpolator is applied to the architecture, it will take 416 cycles in loading reference data. However, by using two-interpolator, the number of cycles can be reduced to 291. This is because a pre-loading strategy is adopted.

In general, the minimum output delay of IP architecture [12] is 8 cycles on the condition that the outputs of the diagonal and vertical filters are registered, as shown in Fig. 5. The delay of IP varies with the pipeline level of the diagonal and vertical filters. In our interpolator circuit, the delay is 8 cycles. According to this delay period, one interpolator needs to load reference pixels ahead of 5 cycles before the end of the other. One example is given in Fig. 8, which is the moment at which one 8×8 block switches the next 8×8 block.

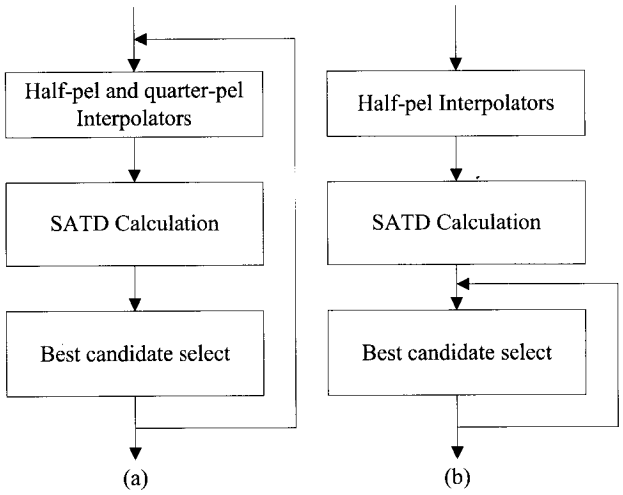


Fig. 9 (a) The conventional two-iteration process. (b) The improved two-iteration process.

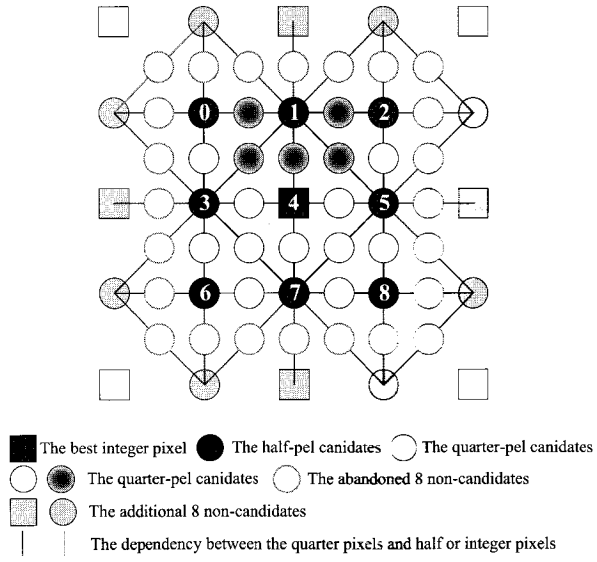


Fig. 10 The search points for FME.

4.3 SATD Generator Based on 8 × 8/4 × 4 Adaptive Hadamard Transform

Most of previous solutions only adopted 4 × 4 HT, because the minimum block size is 4 × 4. However, the variable block-size transform can better present the natural feature of the real-world image. The reusable relationship between 8 × 8 and 4×4 HT provides the feasibility of designing a sharing HT circuit. This reusable relationship has been discussed in Ref. [11]. Here, we apply it to the FME architecture in accordance with the FREXt. The 8 × 8/4 × 4 adaptive HT circuit supports one 8 × 8 HT or two parallel 4 × 4 HTs, and it together with two 8-pel interpolators increase the data throughput.

4.4 Improved Two-Iteration Process

The two-iteration search process defined in JM seriously limits the high-performance application of the FME, because the quarter-pel search process can't run until the accomplishment of the half-pel search process. Furthermore, the second iteration process needs to pass through the Interpolation, SATD Calculation and Best Candidate Select long datapath, as shown in Fig. 9(a). According to the linearity of HT (Sect. 3.2), the quarter-pel SATD calculation can be directly calculated following on half-pel SATD calculation, which helps the quarter-pel search process to skip the Interpolation and SATD Calculation datapath.

However, in the half-pel search process, only nine half-pel SATDs are generated simultaneously, as shown in Fig. 10. According to the dependency between quarter pixels and half pixels or integer pixels, only 16 quarter-pel SATDs can be generated with the help of the 9 half-pel SATDs. If we want to get the other quarter-pel SATDs, the additional 12 half-pel or int-pel SATDs are required, which will result in the dramatic increase of hardware cost. But, fortunately, experimental results show that about 77% of the best matching candidates are located at the central 25 candidates. Based on this feature, we decide to abandon the marginal 21 quarter-pel candidates for saving hardware cost.

Table 3 The visual quality drop comparison of our scheme and 25-point search scheme in Ref. [13]. (IPPP GoP. 30 fps. 50 frames, QP=16, 20, 24, 28, RDO off)

Test Sequences (1920x1080)	Ref.[13]		Our Scheme	
	Δ PSNR	Δ bit rate	Δ PSNR	Δ bit rate
riverbed	+0.005	-0.09	+0.011	-0.19
blue_sky	-0.014	+0.65	-0.023	+1.02
rush_hour	-0.008	+0.48	-0.012	+0.69
station2	-0.012	+0.61	-0.020	+1.01
sunflower	-0.017	+0.71	-0.025	+0.97
tractor	-0.024	+0.69	-0.032	+0.90
pedestrian_area	-0.006	+0.23	-0.008	+0.29
Average	**-0.011**	**+0.47**	**-0.016**	**+0.67**

Besides, the scheme of applying the linearity of HT to the quarter-pel SATD calculation can also realize the single iterative search process as Ref. [13] does. However, this single-iteration scheme will yield 9 half-pel SATDs and 16 quarter-pel SATDs simultaneously, so there will need a 25-input comparator to meet the throughput. In our design, we postpone the outputting of the quarter-pel SATDs. Just as the original two-iteration process does, our improved process also selects the required quarter-pel candidates around the best matching half-pel candidate, but the required quarter-pel candidates are only limited at the center 25 candidates. For example, if the best half-pel candidate is located at the No.1, as shown in Fig. 10, there will be 5 quarter-pel candidates around No.1 being searched. Though our scheme is two-iteration, it shortens the datapath of the second iteration to "Best Candidate Select" only, as shown

in Fig. 9(b).

Table 3 lists the visual quality drop caused by our proposed two-iteration scheme and 25-point search scheme [13], compared with the full search method in JM. The results show that all of these two schemes lead to the trivial reduction of the encoding efficiency. Comparing with the improvement of the processing performance, it is acceptable. Besides, though our scheme further reduces the search points on the basis of 25 search points, there is only about 0.006 dB PSNR drop. For the sake of reducing the hardware cost and shortening long datapath, the cost is worthy.

5. Implementation Results and Comparison

5.1 The Implementation Results

Our proposed FME architecture has been implemented in Verilog HDL and synthesized with the SMIC 130 nm CMOS technology at the constraint of 350 MHz (The Max. frequency). The detailed hardware cost is listed in Table 4.

Simulation results show that this design has capability of realizing the real-time encoding of full-mode 4 K × 2 K@24 fps video sequences or six-mode 4 K × 2 K@30 fps. Figure 11 illustrates the performance spec under different operation frequency and supported modes, which can be used in different applications. The operation frequency is calculated at the worst case.

Table 4 The hardware cost of FME.

Modules	Gate Count (gates(NAND))
2xInterpolators	70,341
9xHPUs	167,825
16xQPUs	122,907
Best Candidate Select	25,528
Mode Decision	4,348
Controller	424
Total	**420,185**

5.2 Comparison

Table 5 lists the comparison with others' implementations. Tsung et al. [9] completely change the refining algorithm and apply the bilinear filter to the half-pel interpolation instead of the 6-tap filter. This change helps to utilize the linearity of HT to half-pel and quarter-pel SATD calculation. Though this change provides almost the same performance as our architecture, their technology is more advanced. Huang's [8] architecture can support 4 K × 4 K@30 fps, this is because they reduce the 7 modes to only 2 modes and adopt high parallelism in hardware, which caused very high hardware cost. Yang et al. [10] propose a adaptive mode select scheme, which can select the supported modes according to the feature of MB. To process one MB will take variable cycles because of the adaptive feature of AMS, so it is not good for the integration of FME. Our proposed architecture can also reach to the specifications that pervious works does. Our architecture doesn't change the interpola-

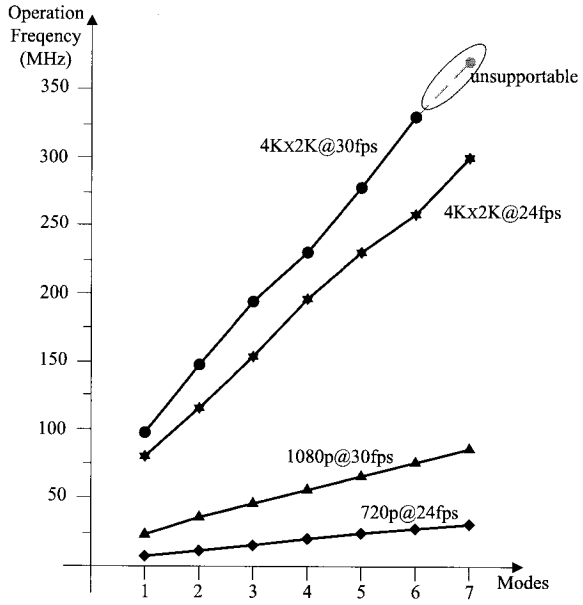


Fig. 11 The operation frequency and the supported modes in the different applications.

Table 5 The comparison with other's implementations.

	Tsung's[9]	Huang's[8]	Yang's[10]	Wu's[3]	Our Proposed
Max.Spec.	4Kx2K@24fps	4Kx4K@30fps	4Kx2K@30fps	1920x1088@30fps	(1).4Kx2K@30fps@six modes (2).4Kx2K@24fps@full modes
Supported Modes	Change the Algorithm	2 Modes	AMS*	7 modes	1-7 modes
Speed(Cycles/MB)	--	--	80-720	631	330
Operation Frequency	280MHz	145MHz	300MHz	145MHz	322MHz for Spec.(1) 300MHz for Spec.(2)
Gate Count(gates)	448K	976.5K	125.4K	321K	420.2K
PSNR drop(DP)	~0.02	0.2	0.026	0	~0.03
CMOS Tech.	TSMC 90nm	TSMC 180nm	UMC 90nm	TSMC 180nm	SMIC 130nm

* AMS: Adaptive modes select

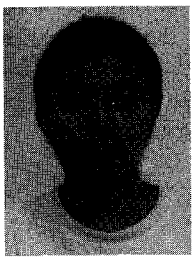
tion algorithm, and maintains the full modes. The number of the cycles for one MB is almost a constant, so it is easy to be integrated into system.

6. Conclusions

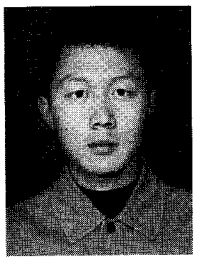
In order to conform to the FRExt feature, a FME architecture supporting $8\times8/4\times4$ adaptive Hadamard Transfrom is proposed. Compared with the conventional FME architecture only based on 4×4 Hadamard Transform, our proposed design increases the throughput and improves the processing performance. Meanwhile, in order to increase the utilization of SATD Generator, our architecture employs two interpolators that can work in turn to provide the available data to the SG continuously. Besides, the linear feature of HT is also used in the quarter-pel SATD calculation which shortens the long datapath of the quarter-pel SATD generation. As a result, these efforts reduce the cycles of processing one MB, which provides the feasibility of applying our FME architecture to the higher performance applications. Finally, under the SMIC 130 nm CMOS technology, the implementation results show that our proposed FME design accounts for 420.2 K gates and the maximum frequency can reach to 350 MHz. And it can realize the real-time encoding of full-mode $4\,K\times2\,K$@24 fps or the six-mode $4\,K\times2\,K$@ 30 fps video sequences.

References

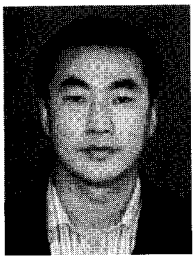
[1] ITU-T, "Advanced video coding for generic audiovisual services," http://www.itu.int/rec/T-REC-H.264, March 2005.
[2] T.C. Chen, Y.W. Huang, and L.G. Chen, "Fully utilized and reusable architecture for fractional moiton estimation of h.264/avc," Proc. IEEE Int. Conf. Acoustics, Speech, and Signal Processing, pp.V9-V12, 2004.
[3] C.Y. Wu, C.L. Wu, and Y.L. Lin, "A high-performance three-engine architecture for h.264/avc fractional motion estimation," IEEE Trans. Very Large Scale Integr. (VLSI) Syst., vol.18, no.4, pp.662–666, 2010.
[4] C.Q. Yang, S.S. Goto, and T.S. Ikenaga, "High performance VLSI architecture of fractional motion estimation in h.264 for hdtv," Proc. IEEE Int. Symp. Circuits and Systems, pp.2605–2608, 2006.
[5] Y.K. Lin, C.C. Lin, T.Y. Kuo, and T.S. Chang, "A hardware-efficient h.264/avc motion-estimation design for high-definition video," IEEE Trans. Circuits Syst., vol.55, no.6, pp.1526–1533, 2008.
[6] T.Y. Kuo, Y.K. Lin, and T.S. Chang, "SIFME A single iteration fractional-pel motion estimation algorithm and architecture for hdtv sized h.264 video coding," IEEE Int. Conf. Acoustics, Speech and Signal Processing, pp.I-1185–I1188, 2007.
[7] N.T. Ta and J.R. Ch, "High performance fractional motion estimation in h.264/avc based on one-step algorithm and 8×4 element block processing," J. Image Communication, vol.26, pp.85–92, 2011.
[8] Y.Q. Huang, Q. Liu, and T.S. Ikenaga, "Highly parallel fractional motion estimation engine for super hi-vision $4\,K\times4\,K$@60 fps," IEEE Int. Works. Multimedia Signal Processing, 2009.
[9] P.K. Tsung, W.Y. Chen, L.F. Ding, C.Y. Tsai, T.D. Chuang, and L.G. Chen, "Single-iteration full-search fractional motion estimation for quad full hd h.264/avc encoding," IEEE Int. Conf. Multimedia and Expo, pp.9–12, 2009.
[10] C.C. Yang, K.J. Tan, Y. Ch. Yang, and J.I. Guo, "Low complexity fractional motion estimation with adaptive mode selection for h.264/avc," IEEE Int. Conf. Multimedia and Expo, pp.673–678, 2010.
[11] Z.Y. Liu, D.S. Wang, and T.S. Ikenaga, "Hardware optimizations of variable block size hadamard transform for h.264/avc frext," IEEE Int. Conf. Image Processing, pp.2701–2704, 2009.
[12] S.Z. Wang, T. An. Lin, T.M. Liu, and C.Y. Lee, "A new motion compensation design for h.264/avc decoder," IEEE. Int. Symp. on Circuits and Systems, pp.4558–4561, 2005.
[13] Y.H. Chen, T.C. Chen, C.Y. Tsai, S.F. Tsai, and L.G. Chen, "Algorithm and architecture design of power-oriented h.264/avc baseline profile encoder for portable devices," IEEE Trans. Circuits Syst. Video Technol., vol.19, no.8, pp.1118–1128, Aug. 2009.
[14] Y.J. Wang, C.C. Cheng, and T.S. Chang, "A fast algorithm and its vlsi architecture for fractional motion estimation for h.264/mpeg-4 avc video coding," IEEE Trans. Circuits Syst. Video Technol., vol.17, no.5, pp.578–583, May 2007.
[15] JM 16.0 Source Code. http://iphome.hhi.de/suehring/tml/
[16] G. Bjontegaard, "Calculation of average PSNR differences between RD-Curves," ITU SG16 VCEG-M33, April 2001.



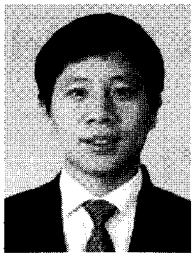
Yibo Fan received the B.E. degree in electronics and engineering from Zhejiang University, China in 2003, M.S. degree in Micro electronics from Fudan University, China in 2006, and Ph.D. degree in engineering from Waseda University, Japan in 2009. From 2009 to 2010, he worked as an Assistant Professor in Shanghai Jiaotong University, And currently, he is the Assistant Professor in Department of Microelectronics of Fudan University. His research interesting includes information security, video coding and associated VLSI architecture.



Jialiang Liu received the B.E. degree in electronic information science and technology from Shandong University of Science and Technology, China in 2009. where he is currently working toward the M.E. degree in circuits and systems. Now he is working at the State Key Lab. of ASIC and System as an exchange student in Fudan University, Shanghai, China. His research interests include video and image processing, digital signal processing, and VLSI design.



Dexue Zhang received the B.S. degree and the Ph.D. degree from University of Science and Technology of China, Hefei, China, in 2000 and 2006. He works at ShanDong University of Science and Technology, Qingdao, China and is currently a post-doctor at the State Key Lab of ASIC & System, Fudan University, P.R. China. His research interests include SoC design, network on chip design, and many-core processors design.



Xiaoyang Zeng received the B.S. degree from Xiangtan University, China in 1992, and the Ph.D. degree from Changchun Institute of Optics and Fine Mechanics, Chinese Academy of Sciences in 2001. From 2001 to 2003, he worked as a post-doctor researcher at the State Key Lab of ASIC & System, Fudan University, P.R. China. Then he joined the faculty of Department of Micro-electronics at Fudan University as an professor. His research interests in signal processing, and communication systems. Prof. Zeng is the Chair of Design-Contest of ASP-DAC 2004 and 2005, also the TPC member of several international conferences such as ASCON 2005 and A-SSCC 2006, etc.

Xinhua Chen received the B.E. degree in Electronic Automation from Shandong Mining Institute, China in 1976, where she joined the faculty of the Department of Machinery and Electronic Engineering after graduation. In 2001, as a leader, she joined to establish the major of Electronic Information Science and Technology, and hosted the post of the department head up to now. His research interests include the design of embedded system, computer architecture and VLSI design on the digital multimedia.