HIGH PERFORMANCE FRACTIONAL MOTION ESTIMATION AND MODE DECISION FOR H.264/AVC*

Chao-Yang Kao, Huang-Chih Kuo, Youn-Long Lin
Department of Computer Science
National Tsing Hua University, Taiwan
{dr948303, mr934389, ylin}@cs.nthu.edu.tw

ABSTRACT

We propose a high performance architecture for fractional motion estimation and Lagrange mode decision in H.264/AVC. Instead of time-consuming fractional-pixel interpolation and secondary search, our fractional motion estimator employees a mathematical model to estimate SADs at quarter-pixel position. Both computation time and memory access requirements are greatly reduced without significant quality degradation. We propose a novel cost function for mode decision that leads to much better performance than traditional low complexity method. Synthesized into a TSMC 0.13 μm CMOS technology, our design takes 56k gates at 100MHz and is sufficient to process QUXGA (3200x2400) video sequences at 30 frames per second (fps). Compared with a state-of-the-art design operating under the same frequency, ours is 30% smaller and has 18 times more throughput at the expense of only 0.05db in PSNR difference.

1. INTRODUCTION

H.264 advanced video coding [1] is the latest video coding standard of ITU-T Video Coding Experts Group (VCEG) and ISO/IEC Moving Picture Experts Group (MPEG). Its new features include variable block-size motion estimation (ME) with multiple reference frames, integer 4x4 discrete cosine transform, in-loop deblocking filter and context-adaptive binary arithmetic coding (CABAC). H.264/AVC can save up to 50% bit-rate compared to MPEG-4 simple profile at the same video quality level. However, large amount of computation is required. Profiling report shows that motion estimation (5 reference frame, ±16 search range) consumes more than 90% of the encoding time. There are many fast algorithms and efficient architectures proposed for integer motion estimation (IME) but few for fractional motion estimation (FME), which accounts for about 40% of motion estimation time. Therefore, an efficient hardware accelerator for fractional motion estimation is necessary in real-time applications.

After motion estimation, mode decision determines the encoding cost of each mode, as shown below, and chooses the mode with the minimal cost.

$$\text{P frame: MODE} \in \begin{Bmatrix} 16\text{x}16, 16\text{x}8, 8\text{x}16, 8\text{x}8, \\ 8\text{x}4, 4\text{x}8, 4\text{x}4, \text{SKIP} \end{Bmatrix}$$

$$\text{B frame: MODE} \in \begin{Bmatrix} 16\text{x}16, 16\text{x}8, 8\text{x}16, 8\text{x}8, \\ 8\text{x}4, 4\text{x}8, 4\text{x}4, \text{Direct} \end{Bmatrix}$$

The rest of this paper is organized as follows. In Section 2, we briefly survey related work. In Section 3, we present our approach for fractional motion estimation and mode decision. In Section 4, we propose our VLSI architecture. Experimental results are presented in Section 5. Finally, we draw conclusions and point to possible directions for future research in Section 6.

2. RELATED WORK

In conventional motion estimation, integer ME (IME) is first performed and then half-pixel precision search is applied to eight neighboring half-pixel positions around the best full-pixel position. The process of quarter-pixel precision search is then performed similarly. This method requires time consuming half-pixel and quarter-pixel interpolation in advance. Large amount of memory access and computation become obstacle to real time application. Although there are several fast fraction-pixel precision searching methods to decrease complexity caused by interpolation, they usually lead to large estimation error and result in poor video quality.

There are two popular mode decision algorithms. Rate-Distortion Optimized (RDO) mode decision [2] considers the distortion and bit-rate by carrying out the entire encoding loop. It is computationally expensive but results in better quality and compression rate. Low complexity mode decision on the other hand only considers sum of absolute difference (SAD) and estimated bit-rate for encoding

**Supported in part by the National Science Council of Taiwan under grants no. NSC94-2215-E-007-029, -007, -009 and -017, the Ministry of Economic Affairs of Taiwan under grant no. 94-EC-17-A-01-S1-038, and Taiwan Semiconductor Manufacturing Company under grant no. NTHU-0416.*

motion information. By avoiding going through the whole encoding loop, this approach speeds up the decision process at the expense of poor quality and compression rate compared with the RDO approach.

3. PROPOSED ALGORITHM

3.1. Fractional Motion Estimation

In Reference [3], the authors proposed a mathematical model to estimate SADs at half-pixel precision according to neighboring integer-pixel precision SADs. This method avoids half-pixel interpolation and reduces computation time. We apply this method and extend it to quarter-pixel precision.

In Figure 1, squares denote integer pixels (f_1, f_2, …, f_9) and triangles denote half pixels(h_1, h_2, …, h_9). Let (0, 0) be the position of integer motion vector (MV) determined by IME.

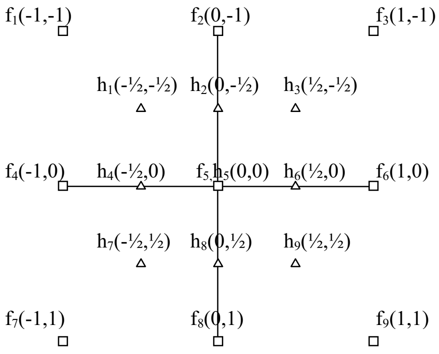

Fig.1. Integer and Half-pixel

The mathematical model used to approximate the surface defined by the nine integer pixels is as following:

$$f(x,y) = c_1x^2y^2 + c_2x^2y + c_3x^2 + c_4xy^2 + c_5xy + c_6x + c_7y^2 + c_8y + c_9 \quad (1)$$

Writing down the 9 SADs, we can get Eq. (2) below:

$$\begin{bmatrix} f_1 \\ f_2 \\ f_3 \\ f_4 \\ f_5 \\ f_6 \\ f_7 \\ f_8 \\ f_9 \end{bmatrix} = \begin{bmatrix} 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & -1 & 1 \\ 1 & -1 & 1 & 1 & -1 & 1 & 1 & -1 & 1 \\ 0 & 0 & 1 & 1 & 0 & -1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 & -1 & -1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} c_1 \\ c_2 \\ c_3 \\ c_4 \\ c_5 \\ c_6 \\ c_7 \\ c_8 \\ c_9 \end{bmatrix} \quad (2)$$

Its 9 coefficients can be determined by 9 integer-pixel precision SADs around (0, 0). We can obtain 9 coefficients by the inverse matrix of Eq. (2), as shown in Eq. (3).

$$\begin{bmatrix} 1/4 & -1/2 & 1/4 & -1/2 & 1 & -1/2 & 1/4 & -1/2 & 1/4 \\ -1/4 & 1/2 & -1/4 & 0 & 0 & 0 & 1/4 & -1/2 & 1/4 \\ 0 & 0 & 0 & 1/2 & -1 & 1/2 & 0 & 0 & 0 \\ -1/4 & 0 & 1/4 & 1/2 & 0 & -1/2 & -1/4 & 0 & 1/4 \\ 1/4 & 0 & -1/4 & 0 & 0 & 0 & -1/4 & 0 & 1/4 \\ 0 & 0 & 0 & -1/2 & 0 & 1/2 & 0 & 0 & 0 \\ 0 & 1/2 & 0 & 0 & -1 & 0 & 0 & 1/2 & 0 \\ 0 & -1/2 & 0 & 0 & 0 & 0 & 0 & 1/2 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} f_1 \\ f_2 \\ f_3 \\ f_4 \\ f_5 \\ f_6 \\ f_7 \\ f_8 \\ f_9 \end{bmatrix} = \begin{bmatrix} c_1 \\ c_2 \\ c_3 \\ c_4 \\ c_5 \\ c_6 \\ c_7 \\ c_8 \\ c_9 \end{bmatrix} \quad (3)$$

We substitute these 9 coefficients into the original mathematical model (Eq.(2)). In the next step, SADs at the neighboring half-pixel positions (h_1, h_2, …, h_9) can be obtained by replacing x and y in Eq.(2) with its coordinates. The position which causes minimum SAD is the half-pixel precision MV. Besides, to reduce computation time, each polynomial of Eq. (2) can be calculated in advance as below.

$$\begin{bmatrix} h_1 \\ h_2 \\ h_3 \\ h_4 \\ h_5 \\ h_6 \\ h_7 \\ h_8 \\ h_9 \end{bmatrix} = \begin{bmatrix} 1/16 & -1/8 & 1/4 & -1/8 & 1/4 & -1/2 & 1/4 & -1/2 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1/4 & -1/2 & 1 \\ 1/16 & -1/8 & 1/4 & 1/8 & -1/4 & 1/2 & 1/4 & -1/2 & 1 \\ 0 & 0 & 1/4 & 0 & 0 & -1/2 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1/4 & 0 & 0 & 1/2 & 0 & 0 & 1 \\ 1/16 & 1/8 & 1/4 & -1/8 & -1/4 & -1/2 & 1/4 & 1/2 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1/4 & 1/2 & 1 \\ 1/16 & 1/8 & 1/4 & 1/8 & 1/4 & 1/2 & 1/4 & 1/2 & 1 \end{bmatrix} \begin{bmatrix} c_1 \\ c_2 \\ c_3 \\ c_4 \\ c_5 \\ c_6 \\ c_7 \\ c_8 \\ c_9 \end{bmatrix} \quad (4)$$

After finding the minimum SAD at half-pixel level, we can reset the origin (0, 0) to the position pointed to by half-pixel MV and find the minimum SAD at quarter-pixel precision similarly.

Our mathematical-model-based approach greatly reduces computation time requirement. Table 1 compares the computation costs between our mathematical approach and the traditional sub-pixel interpolation one for processing one 16x16 macroblock.

Table1: Computation Costs Comparison

		# Shift	# Add
2-tap Interpolation	1/2-pixel	833	4088
	1/4-pixel	1666	8176
Mathematical Model	1/2-pixel	16	85
	1/4 pixel	32	170

For 2-tap interpolation at half-pixel level, 833 half-pixel are generated. Besides, secondary search at 8 half-pixel positions costs 8 x (256 add + 255 sub) = 4088 operations. The cost of mathematical model is reduced by resource sharing and shown later in our architecture.

Another advantage of this model is to use the same equations to refine MVs of variable size blocks. This feature further reduces hardware costs. By this mathematical model and a pipelined architecture, we can refine 41 variable block-size integer MVs to quarter-pixel precision in 45 clock cycles.

3.2. Mode Decision

Although RDO mode decision results in better performance, it is very complicated for hardware and software implementation. We set it in our road map as a long term goal. Presently, we propose to enhance the low complexity mode decision with more accurate Lagrange cost function as shown in Eq. (5):

$$Mode\_Cost = SAD + \lambda \cdot BitUsage \qquad (5)$$

SAD is the sum of absolute difference between current and reference block, λ is a weight parameter, and Bit-Usage represents the bit-rate used to encode the motion information. Instead of looking up the MVBITS table as traditional low complexity mode decision, we formulate λ as a function of quantization parameter (QP) for better rate-distortion estimation. This is similar to the RDO approach [4]. Bit-usage is a function of reference index (Ref_idx) and motion vector difference (MVD). The reference index points to the referenced frame among all reference frames. We take motion vector difference (MVD) instead of MV into account because the entropy coder (CAVLC or CABAC) encodes the MVD instead of MV for bit-rate saving. The functions of Bit-Usage and λ are shown below.

$$\lambda = \sqrt{0.85 \cdot 2^{(QP-12)/3}} \qquad (6)$$

$$BitUsage = 2 \cdot Ref\_idx + Bit\_to\_encode\_MVD \qquad (7)$$

4. ARCHITECTURE

Figure 2 depicts our proposed top-level architecture. It consists of two parts: FME and mode decision. For each variable-size block, the FME receives from IME an integer MV and nine associated SADs (one for the best integer position and eight around the best) and outputs a fractional MV and a minimum SAD at quarter-pixel precision. The mode decision engine receives SAD and fractional MV from FME and reference index from IME. It produces the chosen modes and associated MVs of macroblock and submacroblocks.

Figure 3 shows the architecture of our FME, which only costs 85 adders and 16 shifters. The architecture is a direct implementation of Eq (3) and Eq (4). In the first step, FME receives nine SADs from IME and figures out 9 related SADs at half-pixel precision. Then, the comparator finds the minimum half-pixel SAD and MV Refiner adjusts integer MV to half-pixel precision according to the comparison result. The engine then takes the half-pixel precision SADs and MV as it inputs and refines them to quarter-pixel precision

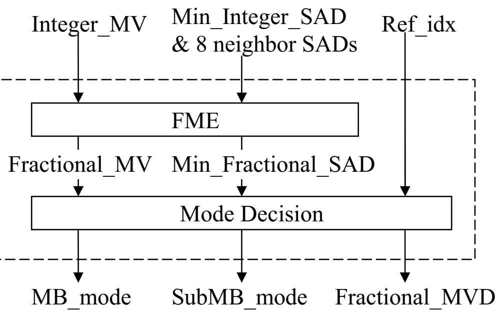

Fig. 2. Top-Level of the Proposed Architecture

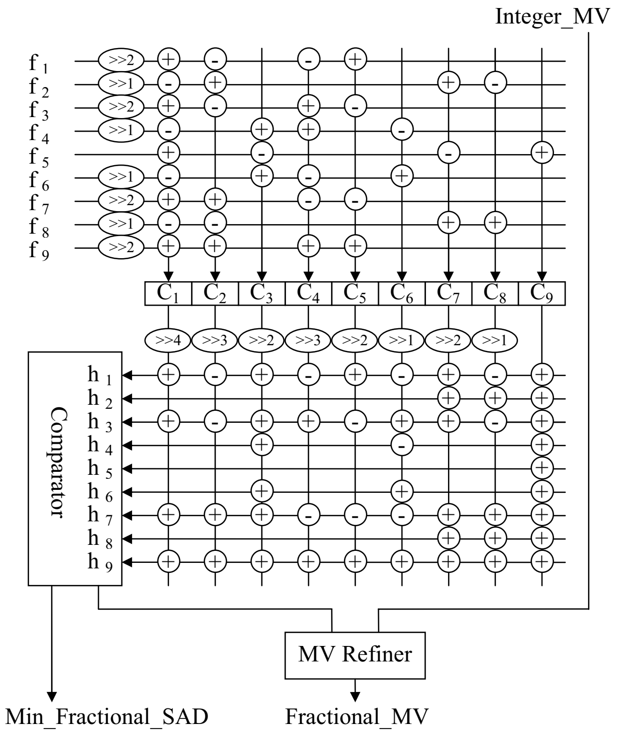

Fig. 3. FME Architecture

Figure 4 depicts the architecture of our mode decision engine. It calculates MVD = MV – MVP and looks up the table MVBITS to obtain the estimated bit-rate cost of encoding MVD. It then calculates the cost of each mode according to Eq. (5). The comparator keeps comparing the calculated cost with the best found cost so far stored in the Best Mode buffer. After all modes have been compared, it outputs the motion information (MB_mode, SubMB_mode and Fractional_MVD) for motion compensation and entropy coding.

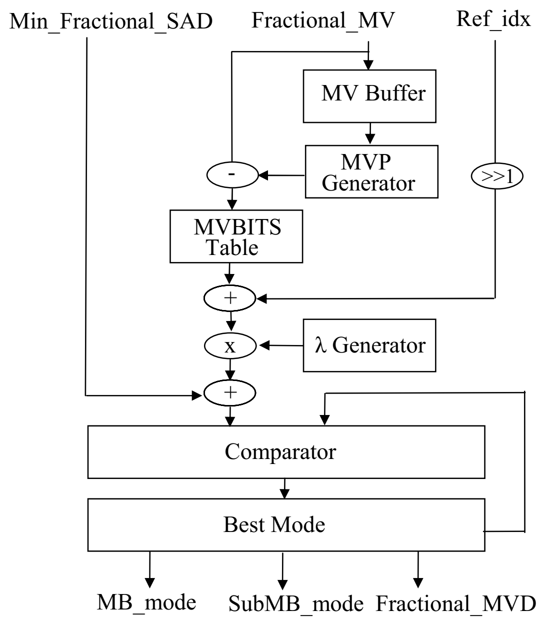

Fig.4. Proposed Mode Decision Architecture

5. EXPERIMENTAL RESULTS

We use six video sequences 'Foreman', 'Akiyo', 'News', 'Mobile', 'Tempete' and 'Carphone' to test the quality of our design. Each sequence consists of 100 frames in CIF (352x288) format. Table 2 shows the results in comparison with H.264 reference software JM9.0 [5] (GOP: IPPPP; 5 reference frames; ±16 search range). We observe that our mathematical-model-based FME causes 0.2~0.4db degradation in PSNR quality. Using the proposed mode decision, we reduce this degradation to 0.03~0.21db.

Table 2: Video Quality Comparison (PSNR)

	JM9.0	Proposed FME	Proposed FME+ MD
Foreman	38.08	37.77	37.88
Akiyo	40.71	40.44	40.68
News	39.14	38.75	39.01
Mobile	34.92	34.67	34.71
Tempete	35.64	35.41	35.45
Carphone	38.66	38.45	38.58

We have implemented the proposed FME and mode decision in synthesizable Verilog HDL. We synthesize our design with a TSMC 0.13μm cell library and under worst case operating environment (WCCOM). Table 3 shows the synthesis result and the comparison with a previous design [6]. Our design consumes 56K gates (28K each for FME and MD), which is 30% smaller than the previous work. Under the same operating frequency (100MHz), it is capable of 18 times more processing capability. The difference in PSNR drop is only 0.05db.

Table 3: Implementation Results

	Reference[6]	Proposed
Process	.18 μm	.13 μm
Gate Count	79,372	56,539
Frequency	100MHz	100MHz
Quality Drop	0.1 dB	0.15dB
Processing Capability	49k MB/s 720x480, 30fps	909k MB/s 3200x2400,30fps

6. CONCLUSION

We have presented a very high performance integrated FME and mode decision for H.264/AVC. Experimental result shows that our design is 30% smaller and having 18 times higher throughput than a state of the art previous work at the expense of only 0.05db PSNR quality difference. This design is very suitable for high-end real-time application.

In the future, we would like to extend the architecture to multi-frame system and exploit the possibility of resource sharing across frames. For applications that require more than 909k MB/s throughput, we can use multiple copies of the design with partial sharing in the MD part.

REFERENCES

[1] "Draft ITU-T Recommendation and Final Draft International Standard of Joint Video Specification (ITU-T Rec. H.264|ISO/IEC 14496-10 AVC)," JVT-G050, May. 2003.

[2] G.J. Sullivan, T. Wiegand, "Rate-Distortion Optimization for Video Compression," *IEEE Signal Processing Magazine*, vol. 15, pp. 74-90, Nov. 1998.

[3] J.W. Suh, J. Jeong, "Fast Sub-pixel Motion Estimation Techniques Having Lower Computation Complexity," *IEEE Transaction on Consumer and Electronic*, vol. 50, pp. 968-973, Aug. 2004.

[4] T. Wiegand, B. Girod, "Lagrange Multiplier Selection in Hybrid Video Coder Control," *Proceedings of International Conference on Image Processing*, vol. 3, pp. 542-545, Oct. 2001.

[5] Joint Video Team Reference Software JM 9.0, http://bs.hhi.de/suehring/tml/download/, Oct. 2004.

[6] T.C Chen, Y.W Huang, L.G Chen, "Fully Utilized and Reusable Architecture for Fractional Motion Estimation of H.264/AVC," *Proceedings of International Conference on Acoustics, Speech, and Signal Processing*, vol. 5, pp. 9-12, May. 2004.