JOURNAL OF INFORMATION SCIENCE AND ENGINEERING 22, 799-818 (2006)

A Power-Aware Motion Estimation Architecture Using Content-based Subsampling*

HSIEN-WEN CHENG AND LAN-RONG DUNG
Department of Electrical and Control Engineering
National Chiao Tung University
Hsinchu, 300 Taiwan
E-mail: lennon@cn.nctu.edu.tw

This paper presents a novel power-aware motion estimation architecture for battery-powered multimedia devices. As the battery status changes, the proposed architecture adaptively performs graceful tradeoffs between power consumption and compression quality. The tradeoffs are considered to be graceful in that the proposed architecture is scalable with changing conditions and the compression quality is slightly degraded as the available energy is depleted. The key to such tradeoffs lies in a content-based subsample algorithm, first proposed in this paper. As the available energy decreases, the algorithm raises the subsample rate for maximizing the battery lifetime. Differently from the existing subsample algorithms, the content-based algorithm first extracts edge pixels from a macro-block and then subsamples the remaining low-frequency part. By doing so, we can alleviate the aliasing problem and, thus, limit the quality degradation as the subsample rate increases. Given a power consumption mode, the proposed architecture first performs edge extraction to generate a turn-off mask and then uses the turn-off mask to reduce the switch activities of processing elements (PEs) in a semi-systolic array. The reduction of switch activities results in significant power consumption savings. To achieve a high degree of scalability and qualified power-awareness, we use an adaptive control mechanism to set the threshold value for edge determination and make the reduction of switch activities rather stationary. As shown by experimental results, the architecture can dynamically operate in different power consumption modes with little quality degradation according to the remaining capacity of the battery pack while the power overhead of edge extraction is kept under 0.8%

Keywords: motion estimation, image processing, VLSI architecture, video compression, power-aware system

1. INTRODUCTION

Motion estimation (ME) has been notably recognized as the most critical part of many video compression applications, such as MPEG standards and H.26x [1], since it tends to dominate the computational and hence power requirements. With increasing demand for battery-powered multimedia devices, an ME architecture that can be flexible in both power consumption and compression quality is highly required. This requirement is driven by the user-centric perspective [2]. Basically, users have two views on using portable devices. Sometimes, users want extremely high video quality at the cost of reduced battery lifetime. At other times, users want acceptable quality with extended battery lifetime. This paper, therefore, presents a novel power-aware ME architecture that

Received February 9, 2004; revised July 6, 2004; accepted July 27, 2004.
Communicated by Pau-Choo Chung.
* This work was supported in part by the National Science Council of Taiwan, R.O.C., under grant No. NSC 92-2220-E-009-033.

uses a content-based subsample algorithm, which can adaptively perform tradeoffs between power consumption and compression quality as the battery status changes. The proposed architecture is driven by a content-based subsample algorithm that allows the architecture to work in different power consumption modes with acceptable quality degradation. Since the control mechanism and data sequences in different power consumption modes are the same in the architecture, the power-aware algorithm can switch power consumption modes very smoothly on the fly. The block diagram shown in Fig. 1 illustrates a typical application of the proposed power-aware ME architecture. The host processor monitors the remaining capacity of the battery pack and switches power consumption modes. According to the power mode, the power-aware architecture sets the subsample rate and calculates the motion vector (MV) for motion compensation. Note that most portable multimedia devices, in practice, have a battery monitor unit and power management subroutines. The host processor and battery monitor unit should not be considered as the overhead of using the power-aware architecture.

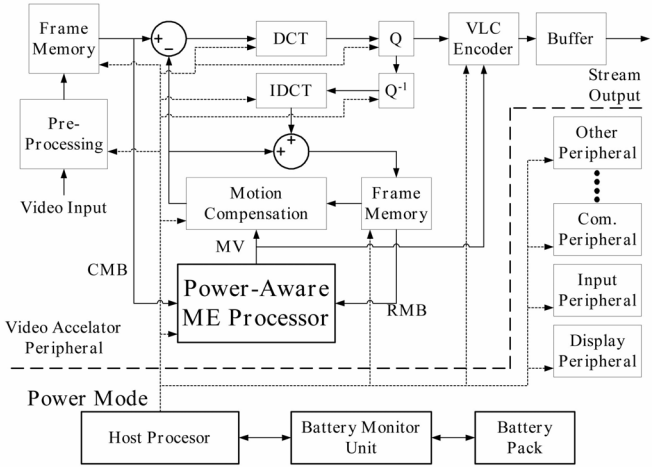


Fig. 1. The system block diagram of a portable, battery-powered multimedia device.

Many published papers have presented efficient algorithms for VLSI implementation of motion estimation, based on either high performance or low power design. However, most of them cannot dynamically adapt the compression quality to different power consumption modes. Among these proposed algorithms, the Full-Search Block-Matching (FSBM) algorithm with the Sum of Absolute Difference (SAD) criterion is the most popular approach to motion estimation because of its good quality. It is particularly attractive when extremely high quality is required. Many types of architectures have been proposed for the implementation of FSBM algorithms [3-6]. However, they require a huge number of *comparison/difference* operations and result in a large computation load and high power consumption. To reduce the computational complexity of FSBM, researchers have proposed various fast algorithms. They either reduce the number of search steps [7-12] or simplify the calculation of the error criterion [13-16]. By combining step-reduction and criterion-simplification, some proposed two-phase algorithms balance

the performance between complexity and quality [17-19]. They first use FSBM with a simplified matching criterion to generate candidate vectors and then select the best motion vector from among these candidates using the SAD criterion. These fast-search algorithms successfully improved the block matching speed while limiting the quality degradation, thus achieving low power implementation. However, a low power implementation is not necessarily a power-aware system in that a power-aware system should adaptively modify its behavior according to the change of the power/energy status and achieve a balance between quality and battery life [20]. The requirement of ME algorithms to be suitable for power-aware designs is high degree of scalability in performance tradeoffs. Unfortunately, the fast algorithms mentioned above do not meet this requirement.

The authors in [21, 22] presented subsample algorithms that significantly reduce the computation cost with low quality degradation. The reduction of the computation cost implies a savings in power consumption. Since the power consumption can be reduced by simply increasing the subsample rate, the subsample algorithms have a high degree of scalability and are very suitable for power-aware ME architectures. However, applying subsample algorithms for power-aware architectures may suffer from aliasing problem in the high frequency band. The aliasing problem degrades the compression quality rapidly as the subsample rate increases. To alleviate this problem, we extend traditional subsample algorithms to obtain a content-based algorithm, called the content-based subsample algorithm (CSA). In this algorithm, we first use edge extraction techniques to separate the high-frequency band from a macro-block and then subsample the low-frequency band only. By combining the edge pixels and subsample pixels, the algorithm generates a turn-on mask for the architecture to limit the switch activities of processing elements (PEs) in a semi-systolic array. By doing so, we can achieve significant power consumption savings and limit the quality degradation as the subsample rate increases. Because the number of high-frequency pixels varies with different video clips, we use an adaptive control mechanism to set a threshold value for edge determination and make the number of masked pixels stationary for a given power mode.

The CSA can be used in most existing ME architectures by turning off PEs according to the subsample rate. In this paper, we present a semi-systolic architecture with gated PEs. The proposed architecture shows that the CSA algorithm can dynamically alter the subsample rate as the power consumption mode changes. As shown by experimental results, the proposed architecture can work in different power consumption modes with acceptable and smooth quality degradation while keeping the power overhead of edge extraction under 0.8%.

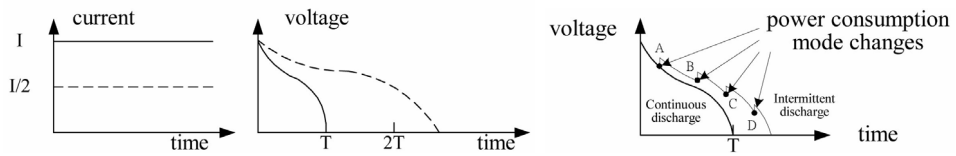
The rest of the paper is organized as follows. In section 2, we introduce the background of the power-aware paradigm. Section 3 presents subsample algorithms in detail. Section 4 describes the proposed power-aware architecture and gives experimental results. Finally, in section 5, we draw conclusions of this work.

2. BACKGROUND

2.1 Battery Properties

One may simply consider a battery as a capacitor in which the charge capacity is

linearly proportional to the output voltage. However, in practice, the behavior of a battery is less than ideal due to the variation in voltage and capacity. Two other important properties of batteries are the rate capacity effect and recovery effect [23]. The first effect means that the capacity of a battery is dependent on the discharging rate, and the second one means that a battery with an intermittent load may have a larger capacity than one with a continuous load. Fig. 2 (a) illustrates the rate capacity effect by plotting the cell voltage of two different discharging loads as time advances. As shown by the curves, when the load is halved the battery life can be more than two times longer. Fig. 2 (b) shows the recovery effect, in where the reduction of the load causes a raise of the voltage. Therefore, one can extend the battery lifetime by gradually stepping down the power dissipation. The Intel® SpeedStep™ technology, for instance, which is widely used in mobile CPUs, adopts the same strategy to extend the battery lifetime [24]. This technology changes the power consumption mode by scaling down the supplied voltage and operating frequency, hence degrading the performance in order to increase the battery lifetime.



(a) The rate capacity effect [25]. (b) The recovery effect.

Fig. 2. Non-ideal battery properties.

From these two properties of batteries, we can learn two things. First, we can reduce the load to achieve a longer battery lifetime because halving the current can more than double the battery lifetime. Second, optimal performance can be achieved when the battery is fully charged because the battery capacity can be recovered later by reducing the load. These properties provide strong motivation for developing power-aware designs and reason out the requirement of power-aware architecture – high degree of scalability in energy-quality tradeoffs.

2.2 Power Model

One can consider the major power consumption of a CMOS gate i as in Eq. (1), where C_i is the output capacitance, f_i is the operation frequency, $r_i(0 \leftrightarrow 1)$ is the switch activity of gate i, α and κ are constants:

$$P_{gate_i} = \alpha \cdot C_i \cdot f_i \cdot V_{DD}^2 = \kappa \cdot C_i \cdot r_i(0 \leftrightarrow 1). \tag{1}$$

For an execution unit EU_j in a VLSI system, the power consumption can be computed using Eq. (2), where $N_{gate,j}$ is the gate count of EU_j:

$$P_{EU_j} = \sum_{i=1}^{N_{gate,j}} \kappa \cdot C_i^j \cdot r_i^j (0 \leftrightarrow 1). \tag{2}$$

After considering the activity of execution units, the total power consumption can be expressed as in Eq. (3) and approximated as in Eq. (5) by assuming that the switch activities are uniform within an execution unit; that is, $r_i^k(0 \leftrightarrow 1) = r^k(0 \leftrightarrow 1)$, $\forall r_i^k(0 \leftrightarrow 1)$. Since the average output capacitances of each execution unit (C_{avg}^k) are nearly the same as the average output capacitances of the total system (C_{avg}), the total power consumption can be approximated to Eq. (8). Therefore, we can obtain an approximate power estimation model as shown in Eq. (9), where ε_{gp} is defined as the gate power coefficient. In this paper, we use the gate power coefficient as the unit for estimating power dissipation:

$$P_{total} = \sum_{\forall \text{inactive} EU_j} P_{EU_j} + \sum_{\forall \text{active} EU_k} P_{EU_k} \tag{3}$$

$$= \sum_{\forall \text{inactive} EU_j} \kappa \sum_{i=1}^{N_{gate,j}} C_i^j \cdot 0 + \sum_{\forall \text{active} EU_k} \kappa \sum_{i=1}^{N_{gate,k}} C_i^k \cdot r_i^k(0 \leftrightarrow 1) \tag{4}$$

$$\cong \kappa \sum_{\forall \text{active} EU_k} r^k(0 \leftrightarrow 1) \sum_{i=1}^{N_{gate,k}} C_i^k \tag{5}$$

$$= \kappa \sum_{\forall \text{active} EU_k} r^k(0 \leftrightarrow 1) \times \frac{\sum_{i=1}^{N_{gate,k}} C_i^k}{N_{gate,k}} \times N_{gate,k} \tag{6}$$

$$= \kappa \sum_{\forall \text{active} EU_k} r^k(0 \leftrightarrow 1) \times C_{avg}^k \times N_{gate,k} \tag{7}$$

$$\cong (\kappa \cdot C_{avg}) \sum_{\forall \text{active} EU_k} r^k(0 \leftrightarrow 1) \times N_{gate,k} \tag{8}$$

$$= \varepsilon_{gp} \sum_{\forall \text{active} EU_k} r^k(0 \leftrightarrow 1) \times N_{gate,k}. \tag{9}$$

3. SUBSAMPLE ALGORITHMS

3.1 Generic Subsample Algorithm

Many published papers have presented efficient algorithms for VLSI implementation of motion estimation [1, 3, 5, 6, 15, 19]. The FSBM algorithm with the SAD criterion is the most popular approach to motion estimation because of its good quality and regular data path. The algorithm uses Eqs. (10) and (11) to compare each current macro-block (CMB) with all the reference macro-blocks (RMB) in the search area to determine the best match and the motion vector is found in Eq. (11):

$$SAD(u, v) = \sum_{i=0}^{N-1} \sum_{j=0}^{N-1} |S(i+u, j+v) - R(i, j)|, \ -p \le u, v \le p-1. \tag{10}$$

The motion vector is found using Eq. (11):

$$\overrightarrow{MV} = (u,\ v)\,|_{\min_{-p\le u,v\le p-1} SAD(u,v)}, \tag{11}$$

where the macro-block size is N-by-N and $R(i, j)$ is the luminance value at (i, j) of the current macro-block (CMB). $S(i + u, j + v)$ is the luminance value at (i, j) of the reference macro-block (RMB), which offsets (u, v) from the CMB in the search area $2p$-by-$2p$.

Much research has addressed subsample techniques for motion estimation in order to reduce the computation load of FSBM [21, 22]. Liu and Zaccarin, pioneers in developing subsample algorithms, applied 4-to-1 subsampling to FSBM and significantly reduced the computational load. As shown by simulation results, the 4-to-1 subsample algorithm reduces the computational load significantly while keeping the quality similar to that with exhaustive search [21]. Here, we will present a generic subsample algorithm in which the subsample rate ranges from 4-to-1 to 1-to-1. The generic subsample algorithm uses Eq. (12) as a matching criterion, called the subsample sum of absolute difference (SSAD), where $SM_{8:m}$ is the subsample mask for the subsample rate 8-to-m as shown in Eq. (13):

$$SSAD_{8:m}(u,\ v) = \sum_{i=0}^{N-1}\sum_{j=0}^{N-1} |\ SM_{8:m}(i,\ j)\cdot[S(i+u,\ j+v) - R(i,\ j)]\,|, \quad \text{for } -p \le u,\ v \le p-1, \tag{12}$$

$$SM_{8:m}(i, j) = BM_{8:m}(i \bmod 4,\ j \bmod 4). \tag{13}$$

The subsample mask $SM_{8:m}$ is generated from a basic mask as shown in Eq. (14):

$$BM_{8:m} = \begin{bmatrix} u(m-2) & u(m-5) & u(m-2) & u(m-6) \\ u(m-3) & u(m-7) & u(m-4) & u(m-8) \\ u(m-2) & u(m-5) & u(m-2) & u(m-6) \\ u(m-3) & u(m-7) & u(m-4) & u(m-8) \end{bmatrix}, \tag{14}$$

where $u(n)$, is a step function; that is,

$$u(n) = \begin{cases} 1, & \text{for } n \ge 0 \\ 0, & \text{for } n < 0 \end{cases}.$$

For example, consider the subsample rate 8-to-6. The subsample mask $SM_{8:6}$ can be expressed in Eq. (15) and is illustrated in Fig. 3:

$$SM_{8:6} = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 \end{bmatrix}. \tag{15}$$

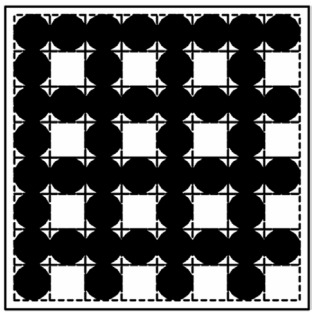


Fig. 3. The subsample mask of the subsample rate 8-to-6.

Given a subsample mask, the computational cost of the SSAD calculation can be lower than that of the SAD calculation. Since a reduction of computational cost implies reduced power consumption, the generic subsample algorithm allows the system power to scale with the changing subsample rate. The higher the subsample rate, the greater the number of inactive execution units (EUs). Accordingly, the power consumption of the system is proportional to the inverse of the subsample rate. Due to its flexibility in achieving an energy-quality tradeoff, the generic subsample algorithm is suitable for implementing power-aware architectures. However, the algorithm suffers from the aliasing problem in the high frequency band. The aliasing problem will degrade the MV quality and result in considerable quality degradation when the high-frequency band is messed up.

3.2 Content-Based Subsample Algorithm

As mentioned above, the generic subsample algorithm suffers from the aliasing problem due to the high subsample rate, leading to considerable quality degradation because the high frequency band is messed up. To alleviate this problem, we propose using the content-based subsample algorithm (CSA), which only subsamples the low-frequency band. The CSA procedure is shown in Fig. 4. We first use edge extraction to separate high-frequency pixels (or edge pixels) from a macro-block and then subsample the remaining pixels (or low-frequency pixels). The determination of edge pixels starts with gradient filtering. Three popular gradient filters [26] were also used here to execute the content-based algorithm; they are the high-pass gradient filter, the Sobel gradient filter, and the morphological gradient filter. Eqs. (16) to (18) show the calculations of the three gradient filters:

High-Pass Gradient Filter:

$$G_{hpf}(i, j) = |MF(HPF_{mask}, R)(i, j)|, \tag{16}$$

where $HPF_{mask} = \begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$.

```
// frame: t
Input current and reference frames, W × H;
for (y = 0; y < W/N; y++) {
   for (x = 0; x < H/N; x++) {
      Perform gradient filtering;
      Calculate the edge threshold:
         threshold = m1^t(x, y) · max{G(i, j)} + (1 – m1^t(x, y)) · min{G(i, j)}
      Determine edge pixels and edge mask;
      Generate content-based subsample mask (GSM);
      edge_cnt = total edges of CSM;
      // update threshold parameter for the next frame
      m1^(t+1)(x, y) = m1^t(x, y) + Kp · (csm_cnt – trg_cnt);
      if (m1^(t+1)(x, y) < 0) {m1^(t+1)(x, y) = 0};
      if (m1^(t+1)(x, y) > 1) {m1^(t+1)(x, y) = 1};
      // find MV
      SSADmin(x, y) = ∞;
         for (u = – p; u < p; u++) {
            for (v = – p; v < p; v++) {
               SSAD(u, v) = Σ_{i=0}^{N–1} Σ_{j=0}^{N–1} | CSM(i, j) · (S(i + u, j + v) – R(i, j)) |;
               if SSADmin(x, y) > SSAD(u, v)
                 {SSADmin(x, y) = SSAD(u, v); MV(x, y) = (u, v);}
            } // for loop index v
         } // for loop index u
   } // for loop index x
} // for loop index y
```

Fig. 4. The content-based subsample algorithm.

Sobel Gradient Filter:

$$G_{sobel}(i, j) = |MF(SX_{mask}, R)(i, j)| + |MF(SY_{mask}, R)(i, j)|, \tag{17}$$

where $SX_{mask} = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix}$ and $SY_{mask} = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}$.

Morphological Gradient Filter:

$$G_{morphological} = (R \oplus B) - (R \ominus B), \tag{18}$$

where $B = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$, and the operations "⊕" and "⊖" denote morphological dilation and erosion.

In Eqs. (16) and (18), the $MF(\cdot)$ function is the mask filter operation as shown in Eq. (19):

$$MF(M,R)(i,\ j) = \sum_{p=-1}^{1}\sum_{q=-1}^{1} M(p+1,\ q+1)\cdot R(i+p,\ j+q), \tag{19}$$

where M is a 3-by-3 mask and $R(i, j)$ is the luminance value at (i, j).

After obtaining the gradients, G, instead of using a constant threshold, we use a floating threshold to determine the edge pixels of the CMB. The floating threshold makes edge extraction more robust when video content varies. Eq. (21) shows the calculation of the floating threshold:

$$threshold = m_1^t(x, y) \cdot \max\{G(i, j)\} + (1 - m_1^t(x, y)) \cdot \min\{G(i, j)\}, \text{ for } 0 \le m_1^t \le 1, \tag{20}$$

where $m_1^t(x, y)$ is the threshold parameter of macro-block (x, y) in the *t-th* frame.

Following the threshold setting step, the algorithm uses the threshold value to pick the edge pixels and produce the edge mask as shown in Eq. (21):

$$EdgeMask(i,\ j) = \begin{cases} 1, & \text{for } G(i,\ j) \ge threshold \\ 0, & \text{otherwise} \end{cases}. \tag{21}$$

Finally, the contend-based subsample mask (CSM) is generated by merging the edge mask and the subsample mask, as shown in Eq. (22). In Eq. (22), the operator $\vee$ means logic a OR operation. According to the calculation of the CSM, the subsample rate in the CSA (CSR), denoted as R_s, is N^2-to-*csm_cnt*, where *csm_cnt* is the number of 1's in CSM and N^2 is the macro-block size. Fig. 5 shows an example of a CSM where the subsample rate is 64-to-27:

$$CSM(i, j) = SM_{8:m}(i, j) \vee EdgeMask(i, j),\ 0 \le i, j \le N - 1. \tag{22}$$

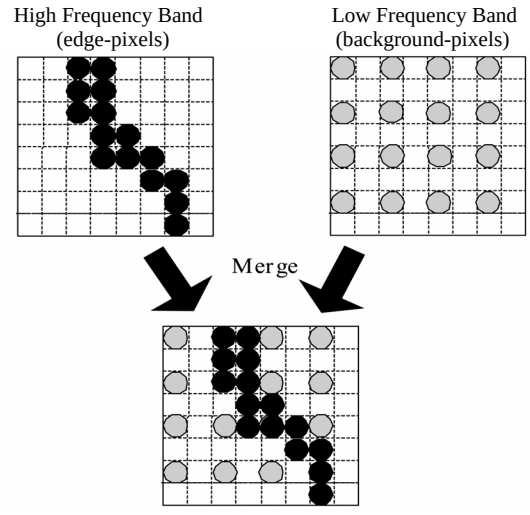


Fig. 5. The components of a content-based subsample mask (CSM).

Once the CSM is generated, the algorithm can then determine the motion vection (MV) with the content subsample sum of the absolute difference (CSSAD) criterion. The CSSAD criterion is similar to SSAD mentioned in section 3.1 and shown in Eq. (23):

$$CSSAD_{8:m}(u,\ v) = \sum_{i=0}^{N-1}\sum_{j=0}^{N-1} |\ CSM_{8:m}(i,\ j)\cdot[S(i+u,\ j+v) - R(i,\ j)]\ |, \quad \text{for } -p \le u,\ v \le p-1. \tag{23}$$

The results of simulation show that the CSA can significantly reduce the computation complexity with little quality degradation. However, there will exist a non-stationary problem with CSA when a power-aware architecture is implemented if the designer uses constant threshold parameters m_1^t and statically sets the floating threshold for a given power mode. Since different video clips with the same threshold parameters will have different subsample rates, setting the threshold value without considering the content variation of the video clip will make the subsample rate non-stationary; that is, power consumption will not converge within a narrow range for a given power mode. The divergence of power consumption can result in a poor power-awareness. To solve this non-stationary problem, we use an adaptive control mechanism to adaptively adjust the threshold parameters so that the subsample rate can be stationary. The adaptive control mechanism used here is a run-time process that adjusts the threshold parameters fittingly according to the difference between the current subsample rate and the desired subsample rate (or target subsample rate).

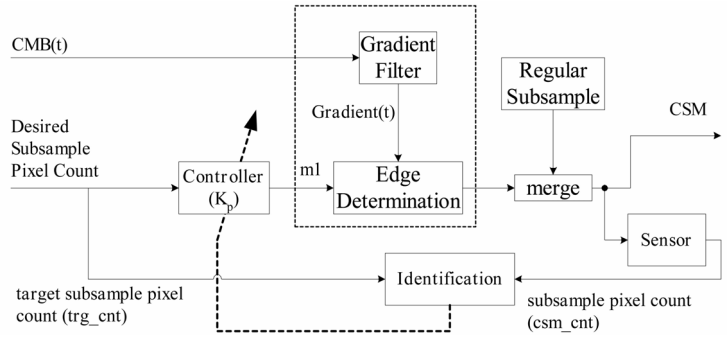


Fig. 6. A block diagram of the edge-extraction unit with an adaptive control mechanism.

Fig. 6 shows a block diagram of the adaptive control mechanism. Given the battery status, the host processor sets the power mode and the target subsample rate as well. The target subsample rate is N^2-to-*trg_cnt*, where *trg_cnt* is the target number of 1's in the CSM. Then, the controller recursively updates the threshold parameter, $m_1^{t+1}(x, y)$, based on the current $m_1^t(x, y)$ and the difference of *csm_cnt* and *trg_cnt*, as shown in Eq. (24):

$$\begin{aligned} &m_1^{t+1}(x, y) = m_1^t(x, y) + K_p \cdot (csm\_cnt - trg\_cnt); \\ &\text{if } (m_1^{t+1}(x, y) < 0)\ \{m_1^{t+1}(x, y) = 0\}; \\ &\text{if } (m_1^{t+1}(x, y) > 1)\ \{m_1^{t+1}(x, y) = 1\}; \end{aligned} \tag{24}$$

where $m_1^{t+1}(x, y)$ is the threshold parameter of macro-block (x, y) in the $(t + 1)$-*th* frame and K_p is the control parameter. The control parameter K_p will affect the settling time and steady-state error of the subsample rate.

3.3 Simulation Results

Figs. 7 and 8 show the simulation results for four 352-by-288 MPEG clips with the parameters $N = 16$ and $p = 32$. The control parameter K_p was set as 0.3. The target subsample rates were set to (4:1), (8:3), (2:1), (8:5), (4:3), (8:7), and (1:1); that is, the target subsample pixel counts were 64, 96, 128, 160, 192, 224, and 256, respectively. Note that the target subsample pixel counts were proportional to the power consumption. Thus, the figures can also be read as charts of power versus PSNR. The dashed lines indicate the results obtained using the generic subsample algorithm, and the solid lines indicate the results obtained using the content-based subsample algorithm with the three gradient filters. As shown by the results, the quality degradation due to the content-based algorithm was less than that due to the generic subsample algorithm, and the type of gradient filter did not significantly affect the performance of the proposed algorithm. In addition, the adaptive control mechanism kept the subsample rate quite stationary. Tables 1 to 3 show the CSR errors with four 40-frame clips. From the results shown in tables, the

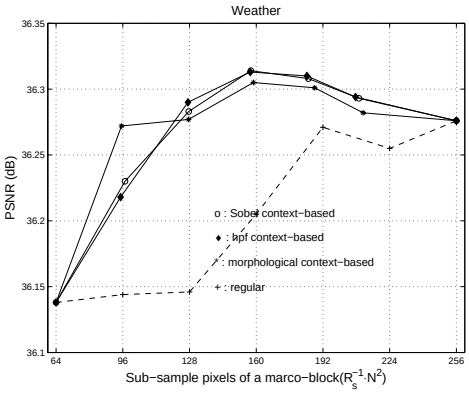


Fig. 7. The quality degradation of the weather clip.

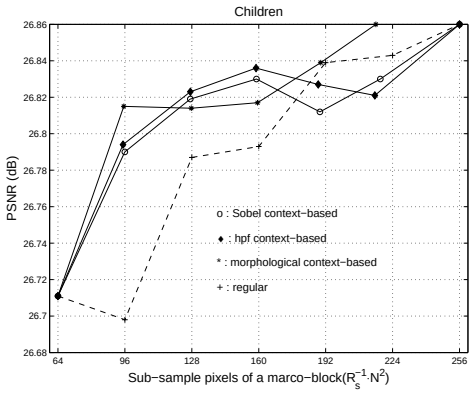


Fig. 8. The quality degradation of the children clip.

Table 1. The CSR error of the content-based subsample algorithm (where the control parameter K_p = 0.3).

Video Clip	Weather		News		Table-Tennis		Children	
Target CSR	Average CSR	CSR Error	Average CSR	CSR Error	Average CSR	CSR Error	Average CSR	CSR Error
96	95.416	0.61%	95.366	0.66%	95.398	0.63%	95.490	0.53%
128	127.521	0.37%	127.720	0.22%	127.467	0.42%	127.754	0.19%
160	158.678	0.83%	159.795	0.13%	159.533	0.29%	159.430	0.36%
192	188.037	2.06%	191.313	0.36%	191.429	0.30%	189.632	1.23%
224	211.274	5.68%	221.224	1.24%	222.893	0.49%	216.001	3.57%

The edge-extraction unit uses the high-pass gradient filter.

Table 2. The CSR error of the content-based subsample algorithm (where the control parameter K_p = 0.3).

Video Clip	Weather		News		Table-Tennis		Children	
Target CSR	Average CSR	CSR Error	Average CSR	CSR Error	Average CSR	CSR Error	Average CSR	CSR Error
96	94.985	1.06%	94.728	1.33%	95.361	0.67%	95.114	0.92%
128	127.264	0.58%	127.688	0.24%	127.445	0.43%	127.304	0.54%
160	157.104	1.81%	159.766	0.15%	159.415	0.37%	159.702	0.381%
192	184.295	4.01%	191.183	0.43%	191.249	0.39%	188.451	1.85%
224	207.612	7.32%	220.949	1.36%	222.685	0.59%	215.536	3.78%

The edge-extraction unit uses the Sobel gradient filter.

Table 3. The CSR error of the content-based subsample algorithm (where the control parameter K_p = 0.3).

Video Clip	Weather		News		Table-Tennis		Children	
Target CSR	Average CSR	CSR Error	Average CSR	CSR Error	Average CSR	CSR Error	Average CSR	CSR Error
96	97.072	1.12%	95.547	0.47%	96.040	0.04%	95.959	0.04%
128	127.626	0.29%	127.718	0.22%	127.120	0.69%	127.267	0.57%
160	157.401	1.62%	159.659	0.21%	158.986	0.63%	158.885	0.70%
192	185.013	3.64%	191.477	0.27%	190.765	0.64%	189.279	1.42%
224	209.300	6.56%	222.434	0.70%	222.405	0.71%	218.118	2.63%

The edge-extraction unit uses the morphological gradient filter.

average CSR error was as low as 1.12%, and the CSR error variance was as low as 0.00024. Because the subsample rate could be kept nearly stationary with given target subsample rate and power mode, we conclude that the power-awareness of the proposed algorithm is very good, and that the CSA can be applied in a power-aware architecture. The results also show that the selection of K_p was proper for controlling the threshold parameters. The following section will further address on the selection of the control parameter.

3.4 Selection of the Control Parameter

As mentioned in sections 3.3 and 3.4, we use an adaptive control mechanism to obtain a stationary subsample rate while keeping the quality acceptable. However, if the control parameter is not properly selected, the settling time will be too long to achieve real-time switching, and the CSR error will be so large as to make the setting of the power consumption mode inaccurate and the power-awareness worse. The control parameter, K_p, in Fig. 6 is the major factor affecting the settling time and the CSR error. After four 30-frame video clips were simulated with 1:1 of the initial subsample rate and 8:5 of the target subsample rate, the effects of the K_p selections were as shown in Figs. 9 and 10. Obviously, the higher the value of K_p, the shorter the settling time and the worse the stability of the CSR. As shown by the results, the suitable range of K_p was from 0.1 to 0.5.

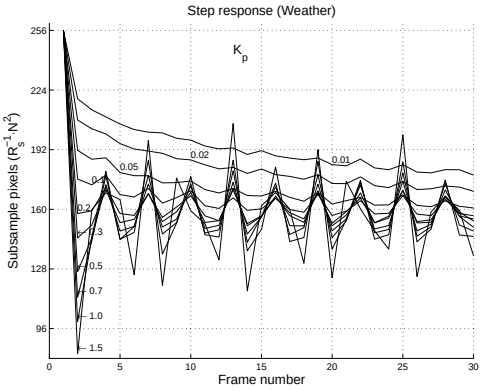


Fig. 9. The step response for varying K_p in the case of the weather clip.

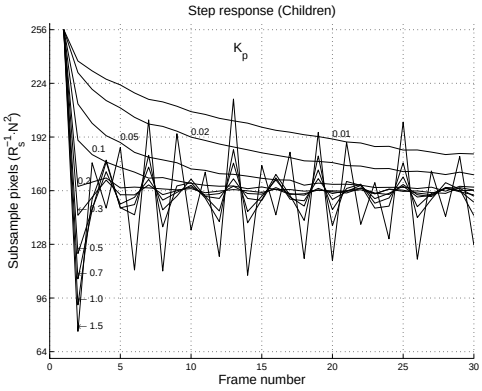


Fig. 10. The step response for varying K_p in the case of the children clip.

4. THE POWER-AWARE ARCHITECTURE

4.1 System Architecture

According to the content-based subsample algorithm, we present a semi-systolic architecture in Fig. 11, which is based on existing architectures, such as that in [5]. The architecture contains an edge-extraction unit (EXU), an array of processing elements (PEs), a parallel adder tree (PAT), a shift register array (SRA), and a motion-vector selector (MVS). Given the power consumption mode, the EXU extracts high-frequency (or edge) pixels from the current macro-block (CMB) and generates 0-1 content-based subsample masks (CSM) for the PE array to disable or enable processing elements (PEs). The structure of the PE array, as shown in Fig. 12, is used to accumulate absolute pixel differences column by column while the parallel adder tree sums up all the results to generate the value of the CSSAD. The MVS then performs a compare-and-select operation to select the best motion vector.

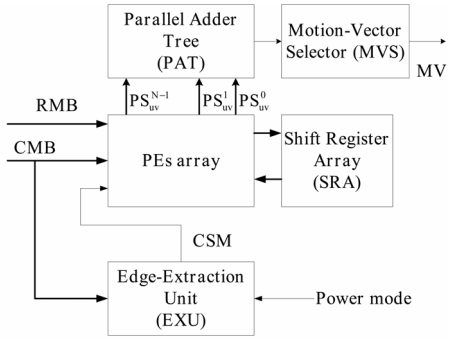


Fig. 11. The system block diagram.

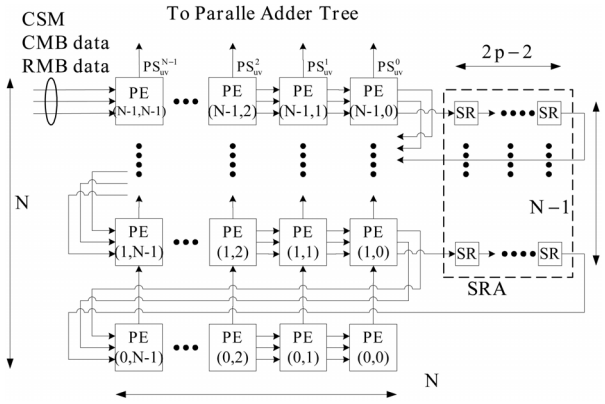


Fig. 12. The architecture of the PE array and shift register array.

Based on the semi-systolic architecture with the content-based subsample algorithm, the architecture dynamically disables some processing elements to reduce the power consumption since we assume the major power consumption is mainly determined by the switch activity of the system [15]. After edge extraction is performed first, a threshold is set as the criterion for determining whether or not to enable/disable processing elements, thus dynamically changing the switch activities of the system to reduce the power consumption. Fig. 13 shows the PE structure and indicates how the CSM disables/enables processing elements. The CSM disables the PE by using the block element (BE), implemented by means of AND gates. The BEs can nullify the input signals of data path, which consists of the absolute difference unit ($|a - b|$) and the Adder unit. When a PE is disabled during a MV searching iteration, the circuits in the PE remain still until the next iteration starts; thus, the consumption of transient power can be reduced.

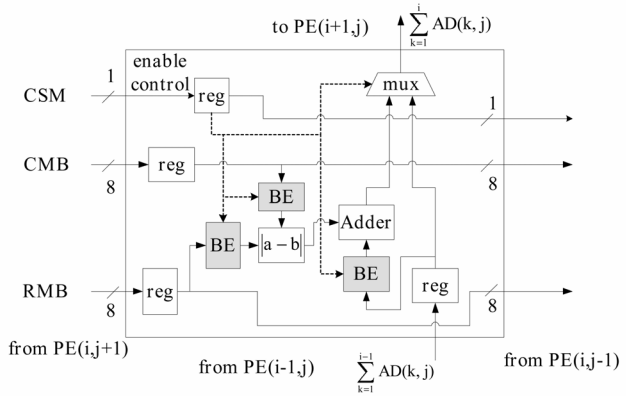


Fig. 13. The structure of a PE.

The edge-extraction unit contains two blocks: a gradient filter and a CSM generator. The gradient filter is implemented based on one of Eqs. (16) to (18). The proposed archi-

tecture only requires that a single gradient filter be embedded. However, we will show all of their implementations for the purpose of estimating the overheads and making a comparison. Figs. 14 to 16 illustrate the implementations of high-pass, Sobel, and morphological gradient filters respectively. Multiplexers are used to prevent boundary errors from occurring with border pixels of the CMB. The black dot in each multiplexer indicates the switching path used when processing a border pixel. Fig. 17 shows the structure of the CSM generator. The CSM generator first determines the threshold according to the gradient values and the power mode, and then generates the CSM by OR-merging the regular subsample pattern and the edge pattern, as shown in Eqs. (21) and (22).

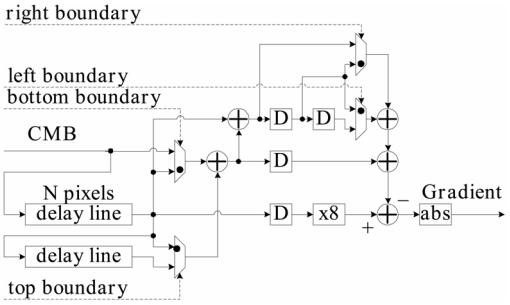


Fig. 14. The architecture of the high-pass gradient filter.

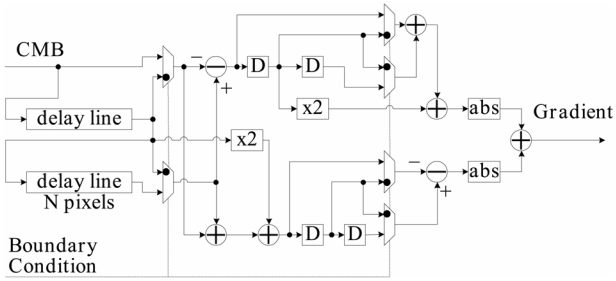


Fig. 15. The architecture of the Sobel gradient filter.

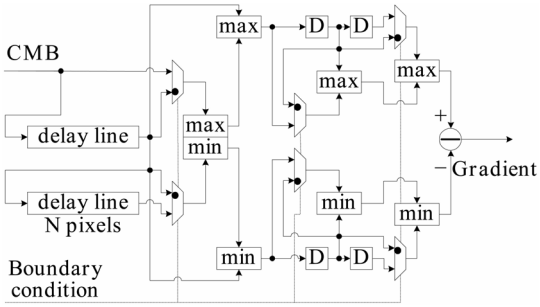


Fig. 16. The architecture of the morphological gradient filter.

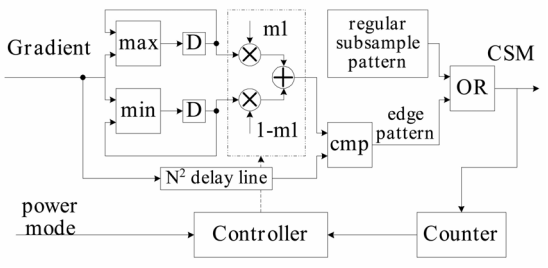


Fig. 17. The architecture of edge-determination and the CSM generator.

4.2 Execution of Power-Aware ME

Power-aware motion estimation is performed in five phases: the initial CMB phase, initial RMB phase, SAD calculation phase, filtering phase, and edge-determination phase. The initial CMB phase involves loading the CMB data into a PE array, while the initial RMB phase involves filling up the PE array with RMB data to start the SAD calculation. As shown in Fig. 18, the initial CMB phase and initial RMB phase are executed in parallel with edge extraction; thus, the timing overhead of edge extraction is hidden by the initial phases. For $p > 8$, the timing overhead of edge extraction is zero.

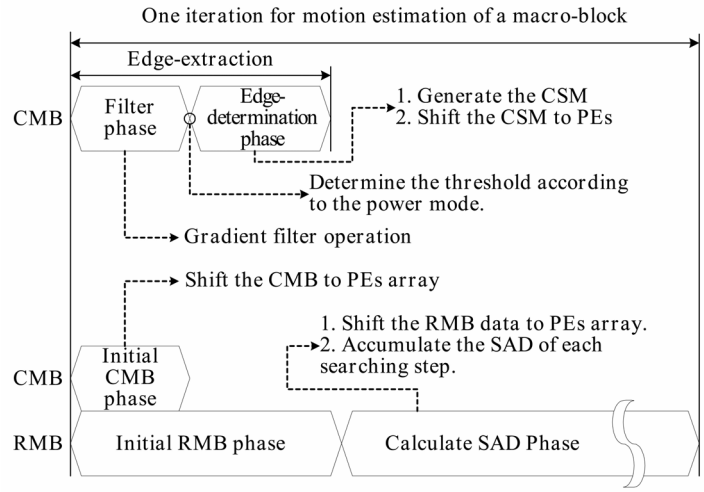


Fig. 18. The execution phases of the power-aware architecture.

4.3 Experimental Results

Table 4 shows the synthesis results obtained using the TSMC 1P4M 0.35*um* cell library, where the symbol R_s denotes the content-based subsample rate and ε_{gp} is the gate power coefficient defined in Eq. (9). Compared with the general semi-systolic architecture [5], the edge extraction unit (EXU) of the proposed architecture has the major

Table 4. Power analysis of the power-aware architecture.

EU_i	PE array		SRA	PAT + MVS	EXU
	AD + Adder	Other			
Gate Count G^i	117,760	58,708	44,640	1,800	17,121
$r^i(0\leftrightarrow1)$	$4p^2R_s^{-1}$=4096R_s^{-1}	$4p^2$=4096	$4p^2$=4096	$4p^2$=4096	N^2=256
$P^i_{consumption}$	4.8*e*8 · R_s^{-1}	2.4*e*8	1.8*e*8	7.37*e*6	4.38*e*6
$P^{all}_{consumption}(\varepsilon_{gp})$	4.8*e*8 · R_s^{-1} + 4.3*e*8				

N = 16 and p = 32. Cell library: TSMC 0.35*um* process.

overhead for the power-aware function. As mentioned above, we used one of the three gradient filters here to implement the EXU. As for the synthesis results, the gate counts of the three gradient filters were 595.33, 793.77, and 727.63, respectively. The variance of these values is very small compared to the overall gate count of the EXU. For instance, the gate count of EXU with the high-pass filter was equal to 14745. This number is much larger than the variance. This means that the selection of a gradient filter does not affect the overhead estimation very much. Therefore, we used the high pass filter to estimate the performance overhead caused by the EXU. From Table 4, one can see that the area overhead of EXU was 7.68%, while the worst-case power overhead was only 0.8% when the subsample rate was 4-to-1 for motion estimation with N = 16 and p = 32.

Fig. 19 shows the results for the video clip "table-tennis" under switching of the power consumption mode. The target subsample pixel count was reduced by 48 every 40 frames, and the control parameter K_p was set to 0.3. The results show that the adaptive control mechanism could enable the power consumption to reach the target level within 10 frames. According to the battery properties described in section 2, the curve shows that our power-aware architecture can extend the battery lifetime by gradually degrading the quality. A, B, C, and D correspond to the switching points in Fig. 2 (b), respectively.

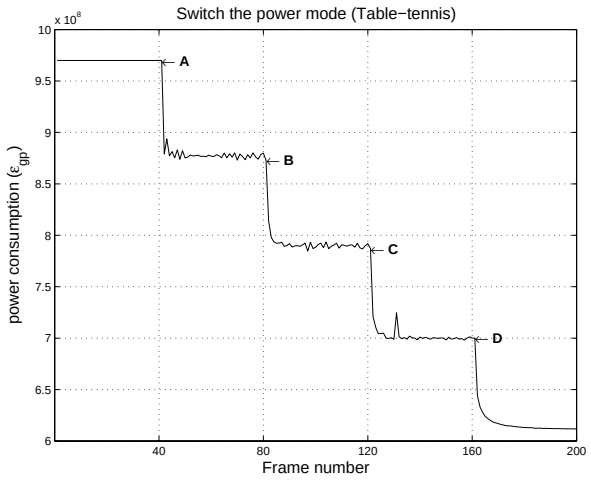


Fig. 19. An application involving switching of the power mode in the case of the video clip "table-tennis."

5. CONCLUSIONS

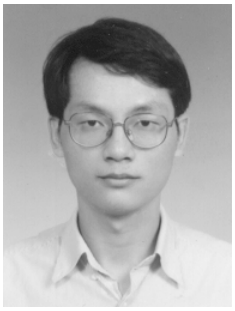
Motivated by the battery properties and the power-aware paradigm, this paper has presented an architecture-level power-aware technique based on a novel content-based subsample algorithm. When the battery capacity is full, the proposed ME architecture turns on all the PEs to provide the best compression quality. In contrast, when the battery capacity is short for full operation, instead of exhibiting an all-or-none behavior, the proposed architecture shifts to a lower power consumption mode by disabling some PEs in order to extend the battery lifetime with little quality degradation. Switching of power consumption mode can be smoothly accomplished; thus, the proposed architecture makes it possible to switch the power consumption mode with acceptable quality degradation. Although edge extraction plays a crucial role to dynamically adjusting the power consumption mode, it does not introduce much power dissipation and the timing overhead can be neglected. As shown by the simulation results, the proposed algorithm successfully improves the compression quality of the generic subsample algorithm and switches the power consumption mode by adaptively adjusting the threshold parameters.

REFERENCES

1. P. Raghavan and C. Chakrabarti, "Battery-friendly design of signal processing algorithms," in *Proceedings of the IEEE Workshop on Signal Processing Systems*, 2003, pp. 304-309.
2. M. J. Chen, L. G. Chen, and T. D. Chiueh, "One-dimensional full search motion estimation algorithm for video coding," *IEEE Transactions on Circuits and Systems for Video Technology*, Vol. 4, 1994, pp. 504-509.
3. B. Liu and A. Zaccarin, "New fast algorithms for the estimation of block motion vectors," *IEEE Transactions on Circuits and Systems for Video Technology*, Vol. 3, 1993, pp. 148-157.
4. H. W. Cheng and L. R. Dung, "EFBLA: a two-phase matching algorithm for fast motion estimation," *Advances in Multimedia Information Processing – PCM*, LNCS 2532, 2002, pp. 112-119.
5. D. Linden, *Handbook of Batteries*, 2nd ed., McGraw-Hill, Inc., New York, 1995.
6. R. C. Gonzalez and R. E. Woods, *Digital Image Processing*, Addison Wesley, New York, 1993.
7. C. H. Hsieh and T. P. Lin, "VLSI architecture for block-matching motion estimation algorithm," *IEEE Transactions on Circuits and Systems for Video Technology*, Vol. 2, 1992, pp. 169-175.
8. K. Sauer and B. Schwartz, "Efficient block motion estimation using integral projections," *IEEE Transactions on Circuits and Systems for Video Technology*, Vol. 6, 1996, pp. 513-518.
9. J. R. Jain and A. K. Jain, "Displacement measurement and its application in interframe image coding," *IEEE Transactions on Communications*, Vol. COM-29, 1981, pp. 1799-1808.
10. J. N. Kim, S. C. Byun, Y. H. Kim, and B. H. Ahn, "Fast full search motion estimation algorithm using early detection of impossible candidate vectors," *IEEE Transactions*

on Signal Processing, Vol. 50, 2002, pp. 2355-2365.
11. T. Koga, K. Iinuma, A. Hirano, Y. Iijima, and T. Ishigura, "Motion compensated interframe coding for video conferencing," in *Proceedings of the IEEE National Telecommunication Conference*, Vol. 4, 1981, pp. G5.3.1-G5.3.5.
12. Y. K. Lai and L. G. Chen, "A data-interlacing architecture with two-dimensional data-reuse for full-search block-matching algorithm," *IEEE Transactions on Circuits and Systems for Video Technology*, Vol. 8, 1998, pp. 124-127.
13. S. Lee and S. I. Chae, "Motion estimation algorithm using low-resolution quantization," *IEE Electronic Letters*, Vol. 32, 1996, pp, 647-648.
14. J. H. Luo, C. N. Wang, and T. Chiang, "A novel all-binary motion estimation (ABME) with optimized hardware architectures," *IEEE Transactions on Circuits and Systems for Video Technology*, Vol. 12, 2002, pp. 700-712.
15. *Mobile Pentium III Processor in BGA2 and Micro-PGA2 Packages Datasheet*, Intel Corporation, pp. 55.
16. P. Kuhn, *Algorithms, Complexity Analysis and VLSI Architectures for MPEG-4 Motion Estimation*, Kluwer Academic Publishers, U.S.A., 1999.
17. R. Li, B. Zeng, and M. L. Liou, "A new three-step search algorithm for block motion estimation," *IEEE Transactions on Circuits and Systems for Video Technology*, Vol. 4, 1994, pp. 438-442.
18. C. K. Cheung and L. M. Po, "Normalized partial distortion search algorithm for block motion estimation," *IEEE Transactions on Circuits and Systems for Video Technology*, Vol. 10, 2000, pp. 417-422.
19. S. Unsal and I. Koren, "System-level power-aware design techniques in real-time systems," in *Proceedings of the IEEE Special Issue on Real-Time Systems*, Vol. 91, 2003, pp. 1055-1069.
20. W. Li and E. Salari, "Succesive elimination algorithm for motion estimation," *IEEE Transactions on Image Processing*, Vol. 4, 1995, pp. 105-107.
21. J. Y. Tham, S. Ranganath, M. Ranganath, and A. A. Kassim, "A novel unrestricted center-biased diamond search algorithm for block motion estimation," *IEEE Transactions on Circuits and Systems for Video Technology*, Vol. 8, 1998, pp. 369-377.
22. J. C. Tuan, T. S. Chang, and C. W. Jen, "On the data reuse and memory bandwidth analysis for full-search block-matching VLSI architecture," *IEEE Transactions on Circuits and Systems for Video Technology*, Vol. 12, pp. 61-72.
23. V. L. Do and K. Y. Yun, "A low-power VLSI architecture for full-search block-matching motion estimation," *IEEE Transactions on Circuits and Systems for Video Technology*, Vol. 8, 1998, pp. 393-398.
24. K. M. Yang, M. T. Sun, and L. Wu, "A family of VLSI designs for the motion compensation block-matching algorithm," *IEEE Transactions on Circuits and Systems for Video Technology*, Vol. 36, 1989, pp. 1317-1325.
25. W. Zhang, R. Zhou, and T. Kondo, "Low-power motion-estimation architecture based on a novel early-jump-out technique," in *Proceedings of the IEEE International Symposium on Circuits and Systems*, Vol. 5, 2001, pp. 187-190.
26. C. Zhu, X. Lin, and L. P. Chau, "Hexagon-based search pattern for fast block motion estimation," *IEEE Transactions on Circuits and Systems for Video Technology*, Vol. 12, 2002, pp. 349-355.

27. M. Bhardwaj, R. Min, and A. P. Chandrakasan, "Quantifying and enhancing power awareness of VLSI systems," *IEEE Transactions on Very Large Scale Integration Systems*, Vol. 9, 2001, pp. 757-772.
28. C. L. Su and C. W. Jen, "Motion estimation using MSD-first processing," *IEE Proceedings of Circuits, Devices and Systems*, Vol. 150, 2003, pp. 124-133.



Hsien-Wen Cheng (鄭顯文) was born in 1968. He received the B.S. degree in Control Engineering from National Chiao Tung University, Hsinchu, Taiwan, R.O.C. in 1992. He joined the AVerMedia Technologies Inc. from 1994 to 2001. He is currently working toward the Ph.D. degree in the Electrical and Control Engineer, National Chiao Tung University. His research interests are video/image compression, motion estimation, VLSI architecture, and digital signal processing.

Lan-Rong Dung (董蘭榮) was born in 1966. He received a B.S.E.E. and the Best Student Award from Feng Chia University, Taiwan, in 1988, an M.S. in Electronics Engineering from National Chiao Tung University, Taiwan, in 1990, and Ph.D. in Electrical and Computer Engineering from Georgia Institute of Technology, in 1997. From 1997 to 1999 he was with Rockwell Science Center, Thousand Oaks, CA, as a Member of the Technical Staff. He joined the faculty of National Chiao Tung University, Taiwan in 1999 where he is currently an Associate Professor in the Department of Electrical and Control Engineering. He received the VHDL International Outstanding Dissertation Award celebrating in Washington DC in October, 1997. His current research interests include VLSI design, digital signal processing, hardware-software codesign, and System-on-Chip architecture. He is a member of Computer and Signal Processing societies of the IEEE.