

國立交通大學

電機與控制工程學系

博士論文

以內容特徵為基礎之
運動向量估測演算法及架構研究

On Study of Content-Based ME
Algorithms and Architectures

研究生：鄭顯文

指導教授：董蘭榮 博士

中華民國九十四年七月

以內容特徵為基礎之
運動向量估測演算法及架構研究
On Study of Content-Based ME
Algorithms and Architectures

研究生：鄭顯文

Student : Hsien-Wen Cheng

指導教授：董蘭榮

Advisor : Lan-Rong Dung

國立交通大學

電機與控制工程學系



Submitted to Department of Electrical and Control Engineering

College of Electrical Engineering and Computer Science

National Chiao Tung University

in partial Fulfillment of the Requirements

for the Degree of

Doctor of Philosophy

in

Electrical and Control Engineering

July 2005

Hsinchu, Taiwan, Republic of China

中華民國九十四年七月

指導教授推薦函

一、事由：推薦電機與控制工程系博士班研究生鄭顯文提出論文以參加國立交通大學博士論文口試。

二、說明：本校電機與控制工程系博士班研究生鄭顯文已完成博士班規定之學科及論文研究訓練。

有關學科部分，鄭君已修畢應修學分（請查學籍資料），通過資格考試；有關論文方面，鄭君已完成“以內容特徵為基礎之運動向量估測演算法及架構研究”初稿。其論文已為下列：

- [1] **Hsien-Wen Cheng** and Lan-Rong Dung, "A Vario-Power ME Architecture Using the Content-Based Subsample Algorithm," Consumer Electronics, IEEE Transactions on, Volume 50, Issue 1, pp.349 - 354, Feb 2004. [2 點]
- [2] **Hsien-Wen Cheng** and Lan-Rong Dung, "EFBLA: A Two-phase matching algorithm for FAST motion estimation," IEE Electronic Letter, Volume 40, Issue 11, pp.660 - 661, May 2004. [1 點]
- [3] **Hsien-Wen Cheng** and Lan-Rong Dung, "A Power-Aware Motion Estimation Architecture Using Content-based Subsampling," Journal of Information Science and Engineering (accepted). [1 點]
- [4] **Hsien-Wen Cheng** and Lan-Rong Dung, "A Content-Based Methodology for Power-Aware Motion Estimation Architecture," Circuits and Systems II: Analog and Digital Signal Processing, IEEE Transactions on. (accepted). [2 點]

等期刊接受。

三、總言之，鄭君已具備國立交通大學電機與控制工程系博士班研究生應有之教育及訓練水準，因此推薦鄭君參加國立交通大學電機與控制工程系博士論文口試。

此致

國立交通大學電機與控制工程系

電機與控制工程系助理教授

董 蘭 榮



中華民國 94 年 6 月

國立交通大學

論文口試委員會審定書

本校電機與控制工程學系博士班鄭顯文君

所提論文：以內容特徵為基礎之運動向量估測演算法
及架構研究

合於博士資格水準，業經本委員會評審認可。

口試委員：林 承 隆 譚 良 春
董 芝 榮 王 聖 智
繆 紹 綱 蔣 迪 豪
林 廷 燈

指導教授：董 芝 榮

系主任：楊 谷 澤

中華民國 94 年 7 月 8 日

Department of Electrical and Control Engineering
National Chiao Tung University
Hsinchu, Taiwan R.O.C.

Date: July 08, 2005

We have carefully read the dissertation entitled On Study of Content-Based ME Algorithms and Architectures submitted by Hsien-Wen Cheng in partial fulfillment of the requirements of the degree of **DOCTOR OF PHILOSOPHY** and recommend its acceptance.

Y Lin

Ling-Bee Ch

Jung Hsu

Sheng-Jyh Wang

Shengyang Miao

PR

Chi-Jay Lin

Tihao Ching

Thesis Advisor:

Jung Hsu

Chairman:

Jung Hsu

Contents

Abstract(Chinese)	viii
Abstract(English)	x
Acknowledgement	xiii
1 Overview	1
1.1 Background	1
1.1.1 Video Coding System	1
1.1.2 Motion Estimation	3
1.2 Objectives	5
1.3 Organization of this Dissertation	6
2 Related Works	7
2.1 Full Search Block Matching Algorithm	7
2.2 Fast Search Algorithm	9
2.2.1 Reduce the Searching Steps	10
2.2.2 Simplifying the Matching Criterion	13
2.2.3 Two-Phase Algorithm	18

3	Edge-driven Two-Phase Motion Estimation	20
3.1	Introduction	20
3.2	Algorithm	22
3.2.1	Edge-matching phase	22
3.2.2	Block-matching phase	29
3.3	Architecture	29
3.3.1	The First Phase	29
3.3.2	The Second Phase	37
3.4	Performance Analysis	39
3.5	Brief Summary	40
4	Power-Aware Algorithm and Architecture	44
4.1	Motivation	45
4.2	Battery Properties	49
4.3	Generic Subsample Algorithm	50
4.4	Content-Based Subsample Algorithm	53
4.4.1	Gradient Filter	53
4.4.2	Edge Determination	56
4.4.3	Adaptive Control Mechanism	57
4.4.4	Matching Step	63
4.5	Results	64
4.6	Power Aware Architecture	69
4.6.1	Architecture	71
4.6.2	Implementation Results	76
4.7	Performance Analysis	78
4.7.1	Power Model	78

<i>CONTENTS</i>	iii
4.7.2 Results	80
4.8 Summary	81
5 Conclusion	84

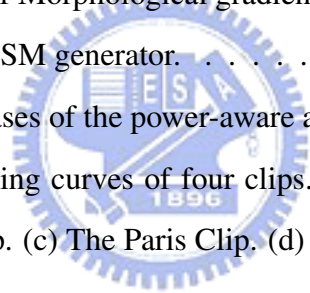


List of Figures

1-1	Main Processing flow in JPEG, MPEG, H.261, and H.263 encoding.	2
2-1	Block matching motion estimation process.	8
2-2	The Three Step Search.	11
2-3	The New Three Step Search.	12
2-4	The Four Step Search.	14
2-5	The Diamond Search.	15
3-1	Flow chart of the EFBLA algorithm	23
3-2	The reusability of quantized data in EFBLA	27
3-3	Two scan directions employed in EFBLA	28
3-4	Block diagram of the edge-driven two-phase motion estimation.	30
3-5	Architecture of Edge Generator Unit.	32
3-6	Architecture of Shift Register Array and Low-Resolution Quantization.	33
3-7	Architecture of Processing Element Array.	34
3-8	Architecture of Processing Element.	35
3-9	Architecture of UEPC Adder Tree and SMVs Selector. We assume that N is 16.	36

3-10	Architecture of the second phase.	38
3-11	Execution of the Accumulator Cells Array in the condition of $N = 8$ and $p = 8$	39
3-12	MAD curves of FS, LRQ and EFBLA for four clips. (a) The Akiyo Clip. (b) The Children Clip. (c) The Stefan Clip. (d) The Weather Clip.	43
4-1	The system block diagram of a portable, battery-powered multi-media device.	47
4-2	This diagram represents the non-linear discharging properties of battery. (a) Rate capacity effect. (b) Recovery effect.	50
4-3	The subsample mask of the generic subsample rate 8-to-6.	52
4-4	The flow chart of content-based subsample algorithm	54
4-5	The content-based subsample algorithm	55
4-6	The components of a content-based subsample mask (CSM)	58
4-7	The block diagram of the edge-determination unit with adaptive control mechanism.	60
4-8	Response time of four clips. (a) The Dancer Clip. (b) The News Clip. (c) The Paris Clip. (d) The Weather Clip.	62
4-9	Quality degradation curves of four clips. (a) The Dancer Clip. (b) The Hall Monitor Clip. (c) The News Clip. (d) The Paris Clip.	67
4-10	Average quality degradation curve of 21 test clips.	68
4-11	The 26th frame of dancer clip. (a) GSA with 8-to-3 subsample rate (b) Residual of motion compensation by GSA (c) CSA with 8-to-3 subsample rate (d) Residual of motion compensation by CSA	68

4-12	The 18th frame of table-tennis clip. (a) GSA with 8-to-3 subsample rate (b) Residual of motion compensation by GSA (c) CSA with 8-to-3 subsample rate (d) Residual of motion compensation by CSA	69
4-13	The block diagram of the power-aware ME architecture driven by the content-based subsample algorithm.	70
4-14	The architecture of PEs array and RMB buffer.	72
4-15	The structure of a PE.	73
4-16	The architecture of high-pass filter.	74
4-17	The architecture of Sobel filter.	75
4-18	The architecture of Morphological gradient filter.	75
4-19	The structure of CSM generator.	76
4-20	The execution phases of the power-aware architecture.	77
4-21	The power switching curves of four clips. (a) The Dancer Clip. (b) The News Clip. (c) The Paris Clip. (d) The Waterfall Clip. . .	82



List of Tables

3.I. Quality degradation analysis for different video clips.	41
3.II. Computational load analysis for different video clips.	42
4.I. Analyze the effect of edge threshold parameter $m1$ on subsample pixels.	59
4.II. Average stationary error for 21 video clips with $K_p = 0.2$	61
4.III. Analyze the effect of controlled edge threshold parameter on sub- sample pixels.	63
4.IV. Quality performance of PSNR by GSA for different video clips.	65
4.V. Quality performance of PSNR by ACSA with high pass filter for different video clips.	66
4.VI. Implementation of the power-aware architecture.	78
4.VII. Power analysis of the power-aware architecture	80

Abstract(Chinese)

運動向量估測在視訊壓縮編碼上已經成爲一關鍵性的重要技術，其最主要的目的是消除相鄰畫面間在時間軸上的重複資訊，以達到高壓縮比的目的，這樣的技術已經被廣泛的應用在各種視訊壓縮標準中，但是在過去相關的研究中，不論是全域搜尋演算法或者是快速搜尋演算法，皆未曾考慮視訊內容特徵的應用。因此本論文以視訊內容特徵爲基礎來探討其在運動向量估測中的應用。本論文中共提出兩種的演算法及其架構，一爲兩階段式快速搜尋法則/架構，二爲電源知覺運動向量估測演算法/架構，兩者都是利用每一影像區塊內容之不同而做不同之處理，以達到降低運算量及電源知覺的目標。後者更利用一簡單之控制機制以收斂最終的功率消耗，達到最佳的電源知覺效能。

兩階段式快速搜尋演算法中，利用每一區塊中高頻部份做低解析度量化處理，並利用此一低解度量化後高頻資訊的特性決定搜尋掃描方向，以有效的重複利用低解析度量化資料，然後再運用這些低解度量化高頻資訊做爲比對的標準，快速篩選出最有可能的位移向量供第二階段精確比對用。第二階段則是利用一般全區域式區塊搜尋演算法常用的 SAD (sum of absolute) 比對標準，在第一階段所產出的可能位移向量中，求出最佳的運動位移向量。這樣的作法可以大大降低整體的運算量，並且在所提出的實現架構中，對於資料的存取，亦能保持相當程度的重複運用以及規則存取特性，最重要的是在運動向量補償品質上，僅犧牲些許的失真。

電源知覺能力之運動向量估測演算法/架構則是依據不同視訊內容特徵，將依區塊內的資料區分為高頻資訊及低頻資訊，並且僅對低頻資訊作取樣處理，再透過一控制機制達到保持整體取樣率於一定範圍，由於取樣率在運動向量估測架構下等比於功率消耗，因此在此一演算法下實現的架構，在以電池為主電力來源的攜帶式視訊裝置上，能依照電池特性做調整使之具備較佳之電源功耗（電池使用時間），同時維持較好的運動向量補償品質。

在最後的模擬結果，本論文所提出的以視訊內容特徵為基礎下，不論是兩階段式快速搜尋演算法/架構或電源知覺能力之運動向量估測演算法/架構，在可攜式多媒體的應用領域上，都能有較佳的電源及品質效能。



Abstract(English)

The major objective of this thesis is to apply content-based approaches for motion estimation algorithms and architectures. Motion Estimation (ME) has been proven to be effective to exploit the temporal redundancy of video sequences and, therefore, becomes a key component of many multimedia standards, such as MPEG-X and H.26X standards. In such multimedia systems, the motion estimation dominates huge computation load and tends to consume much power. This issue has become a significant problem. In order to solve this problem, to develop fast searching algorithms and power-aware architectures becomes a most important issue for such video systems, especially for a portable video device which is powered by battery. Although a great deal of effort has been made on this field, considering the content property of video source on the motion estimation application is still seems to be lacking. In this thesis, we adopted a content-base methodology to meet the requirement of fast searching ME and power-aware ME for such portable video devices. This thesis proposes a edge-driven two-phase ME algorithm based on the content of video sources to reduce computation load in the matching procedure and a content-based power-aware algorithm which adaptively subsamples the background pixels only to perform grace trade-offs between quality degradation and power consumption. By employing the content-based methodology, these proposed algorithms, either for fast searching algorithm or power-aware al-

gorithm, can achieve better results than the non-content-based ones.

In the proposed two-phase motion estimation, to match the low resolution quantized edge pixels of a macro-block is used in the first phase. According to the edge pixels span, the algorithm makes decision of suitable search scan-direction to reuse the quantized data more efficiently. Then it generates the survived motion vectors for the second phase which employs the SAD as the error criteria to perform accurate matching. This content-driven algorithm can reduce the significant computational load comparing with the full-search algorithm and still be more efficient than the existed two-phase algorithm. The content-based power-aware algorithm performs power-aware function by disable/enable processing elements according to the subsample mask based on the content of the video sources. The power-aware approach extracts the edge pixels of a macro-block and subsamples the non-edge pixels only to maintain the quality performance in acceptable level. Since the power consumption is proportional to the subsample rate, this content-based algorithm adopts a close-loop control mechanism to avoid the diverse problem of subsample rate in various video sources and hence keep the subsample rate in stationary state. Founded on the proposed content-based algorithm, the power-aware architecture can dynamically operate at different power consumption modes with little quality degradation only according to the remaining capacity of battery pack to achieve better battery discharging property.

Motivating from the applications of content methodology, this thesis proposes a fast algorithm and a power-aware algorithm to implement the corresponding architectures for conquering the drawbacks without employing content-based technique for the portable multimedia devices. As the simulation results showed, the proposed content-based ME algorithms and architectures can achieve better power

and quality performance for the portable multimedia applications than those without adopting the content-based methodology.



Acknowledgement

在大學畢業後 8 年，沒想到還有這個機會重拾課本充實自我，並且順利拿到學位，對我而言，這是上天對我最大的恩賜。能有這樣的機會完成學位，首先必須感謝父母對我的支持，這段期間父母並未對我有任何經濟上的要求，讓我能兼顧學業及家庭，減輕相當大的經濟負擔，這樣的恩情，非三言兩語能道盡內心的感恩，特別在自己開始養育一對兒女後，更加能深刻感受到父母對我的恩情。

其次必須謝謝董老師這段期間來的指導及教誨，讓我能盡情地發揮想像力及創造力，董老師總能在我碰到瓶頸時給我適切的提醒及指點，使我能順利突破每個關卡，完成這份成果。再來也要感謝口試委員王聖智教授、林永隆教授、林進燈教授、陳良基教授、蔣迪豪教授、繆紹綱教授及羅佩禎教授，謝謝您們的指導及寶貴意見，使得這篇的論文能更加完備。

而系統晶片實驗室的學長、同學、以及學弟妹們，這段期間以來，必須感謝你們在課業上的互相砥礪及切磋，每每總能在與你們的討論之間，激盪出我研究的火花。包括博士班學長傅生、旭榕、宗錫，碩士班學長廷勳、彥霖、仁俊、仁傑、峻銘、國鼎，碩士班同學逸凡、偉勝、宥霖，以及學弟妹們介皇、騰轟、學之、一誠、盟淳、智凱、柏涵、健釗、毅慧、玉書、昭維、樹德、松樹、芳彥、明峰、育聖、岳璋及智偉，非常感謝你們！

也要感謝之前服務公司圓剛科技的總經理郭重松先生、張永哲副總、趙

永松副總以及陳坤洲先生 . . . 等這些老長官們，在我退伍後至重拾學業的這段工作期間，您們給我的栽培及訓練，使我從工程師到部門主管這一路走來能盡情發揮，沒你們協助我打下的基礎，我不能有今日這樣的成就，能順利取得學位，您們是功不可沒！謝謝您們！

最後，必須感謝我太太香誼對我無怨無悔的支持，這一路走來，能有你牽手陪伴，是我人生最大的幸福，謹以此論文獻給我心愛的妳及我們倆可愛的寶貝晴萱及軒劫。

篇幅有限而感恩的心無限，實在無法一一道盡內心的感謝，只能套用國中課本陳之藩「謝天」文章中的話：“因為需要感謝的人太多了，就感謝天吧！”

顯文 於台北土城家中 七月, 2005



Chapter 1

Overview

1.1 Background

1.1.1 Video Coding System

Video coding system has been developed to reduce the transmission rate or stored bits for over twenty years and proven to achieve the objective. Many standards are defined to implement the video coding system such as ISO/IEC MPEG-1, MPEG-2, MPEG-4 and the CCITT H.261 / ITU-T H.263, etc [1–6]. The demand of these video standards is to remove the redundancies of the video sources and compress the video sources to meet the constraints of limited transmission rate and stored bits. In order to achieve this demand, transform coding and predictive coding have become important strategies to identifying the large amount of spatial dependency and temporal redundancy in the video sources.

Figure 1-1 illustrates a typical block diagram of an JPEG, MPEG, H.261, and H.263 video coding system. The video encoder system contains several major components, including discrete cosine transform (DCT), inverse DCT (IDCT),

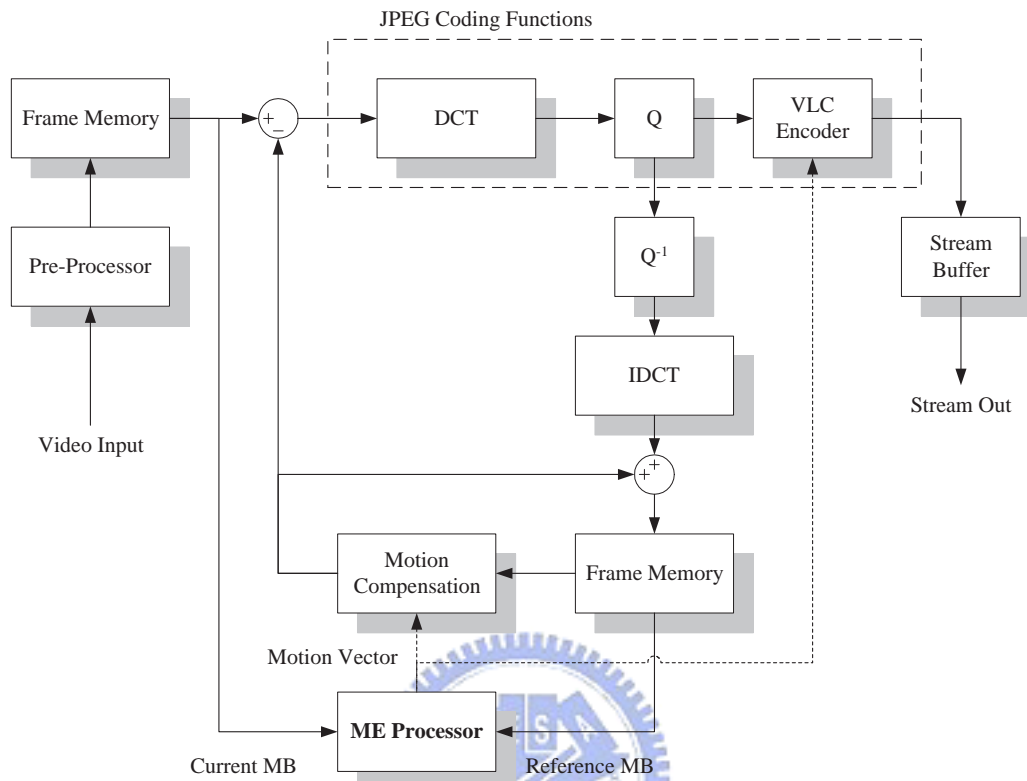


Figure 1-1: Main Processing flow in JPEG, MPEG, H.261, and H.263 encoding.

motion estimation (ME), motion compensation (MC), quantization (Q), inverse quantization (Q^{-1}), and variable-length coding (VLC) encoder.

In those components of such video systems, motion estimation is a key processor employing predictive coding technique to eliminate the temporal redundancy and it is the most computationally expensive part. An encoder creates a predictive frame of the current frame based on the reference frame (either previous or future frame) and forms a residual between the predictive frame and the reference frame. The bits to encode the residual can be fewer than to encode the original frame if the prediction is successful. Although the motion estimation based on the predictive technique can achieve high compression rate, it consumes much computa-

tional complexity in the matching procedure if it performs exhaustively searching strategy in the searching area. According to the complexity analysis, the motion estimation part is up to over 50% computational load of a MPEG or a H.261 coding system [4, 7]. Thus, many motion estimation algorithms, to optimize either the full search or fast search block matching, had become an important research field and been developed to meet various requirements for video applications.

1.1.2 Motion Estimation

As mentioned above, Motion Estimation (ME) has been proven to effectively eliminate the temporal redundancy of video sequences and therefore becomes a central part of the ISO/IEC MPEG-1, MPEG-2, MPEG-4 and the CCITT H.261 / ITU-T H.263 video compression standards. The motion estimation achieves very high compression rate by identifying the temporal redundancy and eliminating them since there is large amount of correlation between successive frames in a video sequence. Of various approaches for motion estimation, the block-matching algorithm is widely performed in those video coding systems since its regular property. The block matching approach first divides a frame into non-overlapped blocks regularly of the same size and find the motion vector by finding the most like block in the searching area. According to the motion vectors of the macro-blocks in a frame, the coding system encodes the residual part between the original frame and motion-compensated frame to raise the compression ratio.

In those block-matching algorithms, the full-search block-matching (FSBM) algorithm is the most popular approach because of its considerably good quality and regular data path. Many works addressed on implementing architectures of the full-search algorithms. Yang et al. presented 1D-array architectures [8] and

many researchers addressed on 2D-array architectures [9–11]. Lai et al. proposed architecture with data reuse scheme which accesses the reference pixels more efficiently but has restriction in the searching area [12]. Some works focused on the discussion of low-power design of FSBM architectures [13, 14]. Paper of Tuan et al. provides a data reuse analysis of FSBM architectures and proposed one access architecture to achieve optimal memory bandwidth [15]. A fast full search algorithm with adaptive scan direction is presented to speed up the conventional full-search algorithm [16]. Those researches performed significant achievements in the implementation of full-search block-matching algorithms.

Although the FSBM algorithm has the benefits of considerably good quality, it dominates huge computation load and tends to consume significant power because of its exhaustive search scheme. In order to solve this problem, to develop fast searching algorithms becomes a most important issue for these video systems, especially for a portable video device which is powered by battery. Many fast search algorithms were proposed for alleviating the heavy computational load of FSBM by reducing the search steps, such as the three-step search (TSS) [17], the new three-step search (NTSS) [18], the one-dimensional full search (1DFS) [19], the four-step search (4SS) [20], and the diamond search (DS) [21–23], etc. Some researchers developed fast algorithms by simplifying the matching criterion [24–27]. Those fast algorithms conquered the drawbacks of full-search algorithm and accomplished great achievements in the application of video coding system.

This thesis will illustrate parts of those motion estimation algorithms in Chapter 2 in detail.

1.2 Objectives

Although lots of effort has been stressed on the research field of motion estimation, employing the content methodology on the motion estimation application is still seemed to be lacking. The major objective of this thesis is to focus on the employment of content property upon motion estimation. Upon this argument, we concentrate on two parts, one is for the fast searching algorithm and another is for the power-aware application. As mention above, the fast algorithm can reduce the computational complexity of the video system so this topic is still worth to develop and stress, especially employing the content methodology. Another issue for power-aware application has become very important since the demand of portable video devices powered by battery raises these recent years. The power-aware mechanism performs switching the power consumption modes with grace quality degradation according to the non-idea battery properties to extend the battery life in such portable devices. This thesis proposed a content-based algorithm to implement power-architecture to meet this requirement for portable applications.

In the proposed content-based fast algorithm, to match the low resolution quantized edge pixels of a macro-block is used in the first phase. According to the edge pixels span, the algorithm makes decision of suitable search scan-direction to reuse the quantized data more efficiently. Different video content causes different scan-direction in which the reusability of quantized data is better. Then the first phase removes the most impossible candidates and generates the survived motion vectors for performing accurate match which employs the SAD as the error criteria in the second phase. This content-driven fast algorithm can reduce the significant computational load comparing with the full-search algorithm and be

more efficient than the existed two-phase algorithm.

In the second part, a content-based algorithm performing power-aware function by disable/enable processing elements according to the content-based subsample mask is presented. In order to avoid the aliasing drawbacks of general subsample technique, the proposed content-based approach extracts the edge pixels of a macro-block and subsamples the non-edge pixels only to maintain the quality performance in acceptable level. Since the power consumption is proportional to the subsample rate, the algorithm adopts a close-loop control mechanism to keep the subsample rate in stationary state. Founded on the content-based algorithm, the power-aware architecture can dynamically operate at different power consumption modes with little quality degradation according to the remaining capacity of battery pack to achieve better battery discharging property.

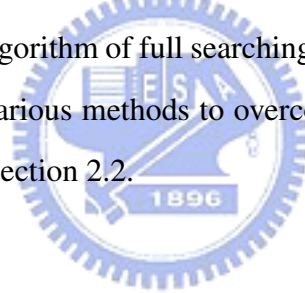
1.3 Organization of this Dissertation

The rest of this dissertation is organized as follows. In Chapter 2, we will present the related works of block-matching motion estimation. Chapter 3 illustrates the algorithm and architecture of edge-driven two-phase motion estimation. Then in Chapter 4, a content-based power-aware algorithm and architecture are addressed. Finally, conclusions and future works are shown in Chapter 5.

Chapter 2

Related Works

In this chapter, the related works of motion estimation are introduced. Section 2.1 illustrates the popular algorithm of full searching block matching (FSBM) and fast search algorithms by various methods to overcome the drawbacks of FSBM algorithm are presented in section 2.2.



2.1 Full Search Block Matching Algorithm

The FSBM algorithm with SAD criterion is the most popular approach for motion estimation because of its considerably good quality and regular data path. Figure 2-1 illustrates the representation of block matching motion estimation. In this figure, the block matching process first divides a current frame into non-overlapped blocks of the same size N -by- N called current macro-block (CMB). Then a current macro-block is exhaustively matched with all the candidate macro-blocks, called reference macro-blocks (RMBs), in the searching area of the reference frame which is either previous frame or next frame. Finally, the block matching algorithm identifies the macro-block which has the minimum distortion to the

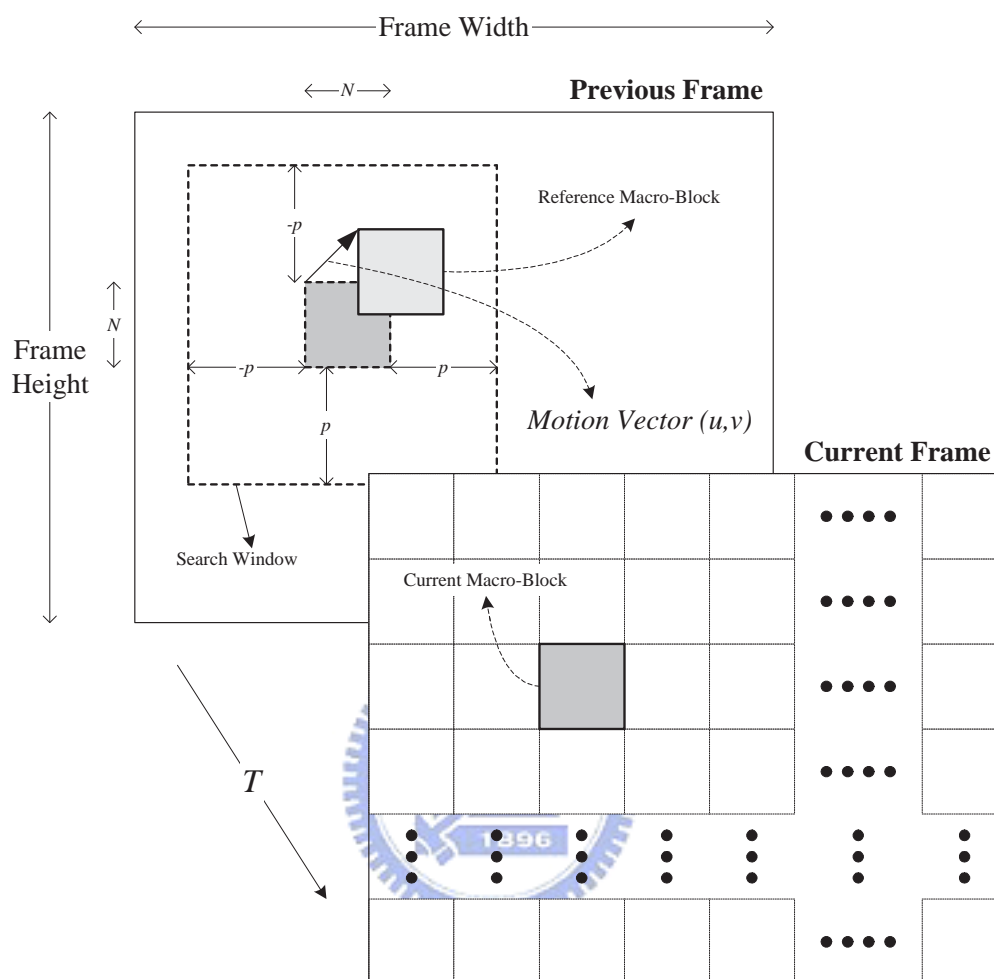


Figure 2-1: Block matching motion estimation process.

current macro-block from all the reference macro-blocks in the searching area. The desired motion vector is the offset from the reference macro-block with the minimum distortion to the current macro-block.

The full search block matching algorithm uses (2-1) and (4-14) to compare each current macro-block with all the reference macro-blocks in searching area to

determine the best match.

$$SAD(u, v) = \sum_{i=0}^{N-1} \sum_{j=0}^{N-1} |R(i + u, j + v) - S(i, j)|, \quad (2-1)$$

for $-p \leq u, v < p$ and the motion vector is figured out by (4-14)

$$\overrightarrow{MV} = (u, v) \Big|_{\min_{-p \leq u, v \leq p-1} SAD(u, v)} \quad (2-2)$$

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

2.2 Fast Search Algorithm

Although the FSBM algorithm has the benefits of considerably good quality and regular data path, its huge number of *comparison/difference* operations results in high computational complexity and power consumption. In order to meet the real-time applications, the fast search algorithms have been widely developed and studied by many researchers. These fast algorithm either reduce search steps or simplify calculations of error criterion. These fast algorithms can be divided into three main categories but not limited to them.

1. By reducing the search steps.
2. By simplifying the matching criteria.
3. Two-phase algorithm.

In the following subsections, we will present these prior works.

2.2.1 Reduce the Searching Steps

Three Step Search (TSS)

The Three Step Search (TSS) [17] uses the rectangular search patterns with logarithmically decreasing search size to test the checking points. Figure 2-2 illustrates the search patterns of the TSS with the search area from -7 to 7 . Each check point with black color means the local minimum distortion in each search step. In this illustration, the motion vector is $(-4, 3)$. The total checking points of the TSS is $25(= 9 + 8 + 8)$. With comparing to 255 search steps of the FSBM, the TSS performs considerably less computational complexity with little motion compensated quality loss.

New Three Step Search (NTSS)

The New Three Step Search (NTSS) algorithm based on the center-biased distribution of motion vector was proposed for improving the performance of the TSS since the TSS used a uniformly check points in its first step[18]. Figure 2-3(a) presents the procedure and (b) shows the check points in the first search step of NTSS. The NTSS checks the extra eight points of the search window center and uses a halfway-stop technique to speed up the matching process if the motion vector is stationary or quasi-stationary. The total number of check points of NTSS is from 17 in best case to 33 in worst case respectively.

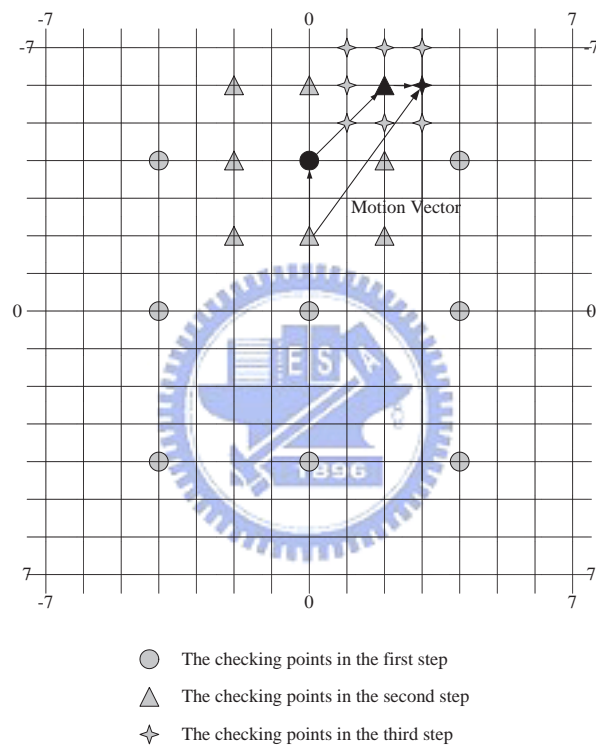


Figure 2-2: The Three Step Search.

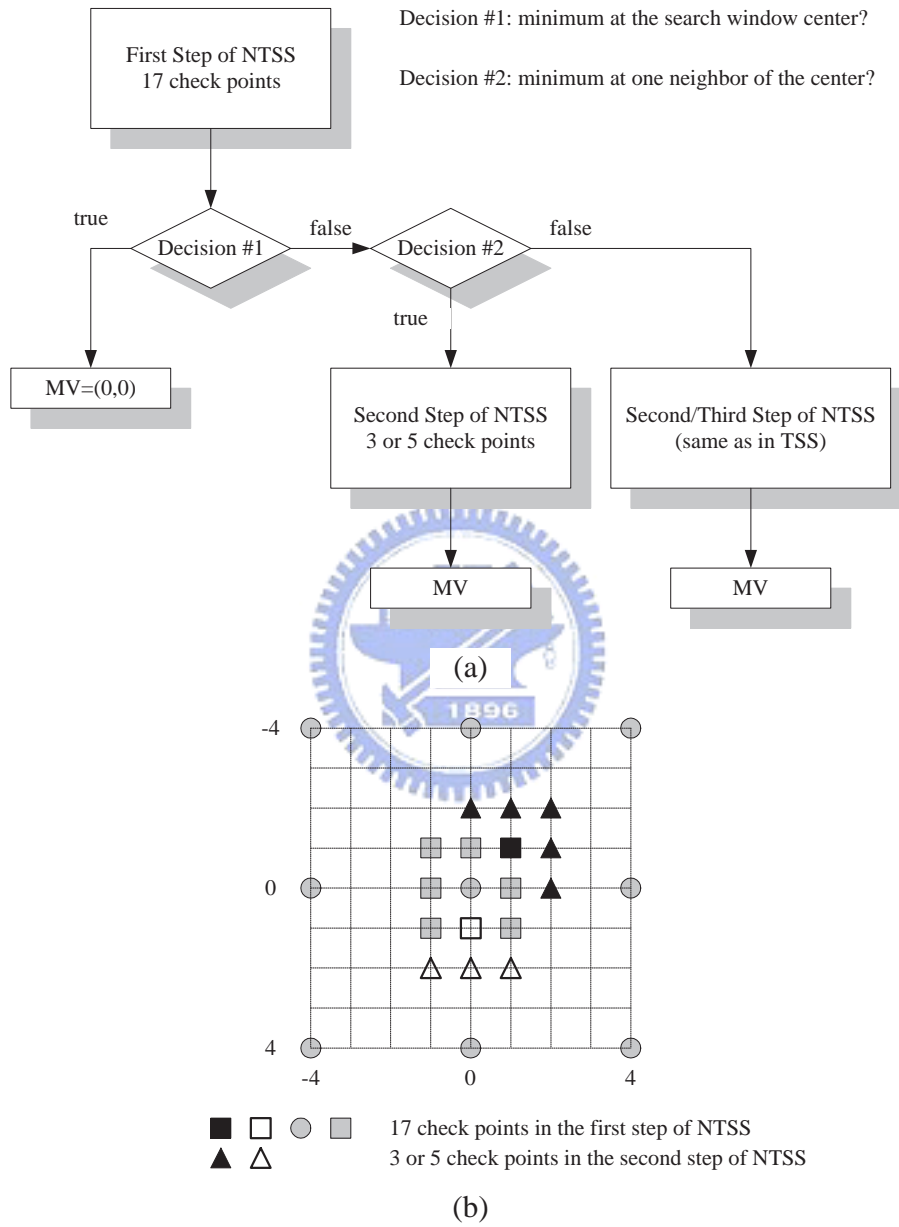



Figure 2-3: The New Three Step Search.

Four Search Step (4SS)

Similar to the NTSS, the Four Search Step (4SS) uses the attribute of center-biased distribution of motion vector and the approach of halfway-stop to save the check points [20]. Figure 2-4(a) illustrates the procedure of the 4SS and (b) shows two different search paths as examples. A black mark in each step is the point which has the minimum distortion error and as the check window center of the next step. From the first step to third step, the size of search window is 5-by-5 and the final step uses 3-by-3. The check point of the 4SS is varied from 17 to 27. It reduces the worse case check points from 33 to 27 and remains similar motion-compensated error as compared to NTSS.

Diamond Search



Diamond Search (DS) employs a diamond-shaped search pattern which is rotated from the square-shaped search pattern in 4SS by 45° [21–23]. It results in fewer check points with similar motion-compensated distortion as compared to NTSS and 4SS. The DS uses two search patterns shown in Fig. 2-5(a), one is large diamond search pattern (LDSP) and another is small diamond search pattern (SDSP). Figure 2-5(b) illustrates an example which leads to the motion vector $(4, -2)$ in five search steps, which are four times of LDSP and one time of SDSP. As the experimental results, the DS significantly improves the performance in terms of the required number of check points.

2.2.2 Simplifying the Matching Criterion

The matching criterion is employed to identify the error distortion between the current macro-block and reference macro-block. Equation (2-3) shows the crite-

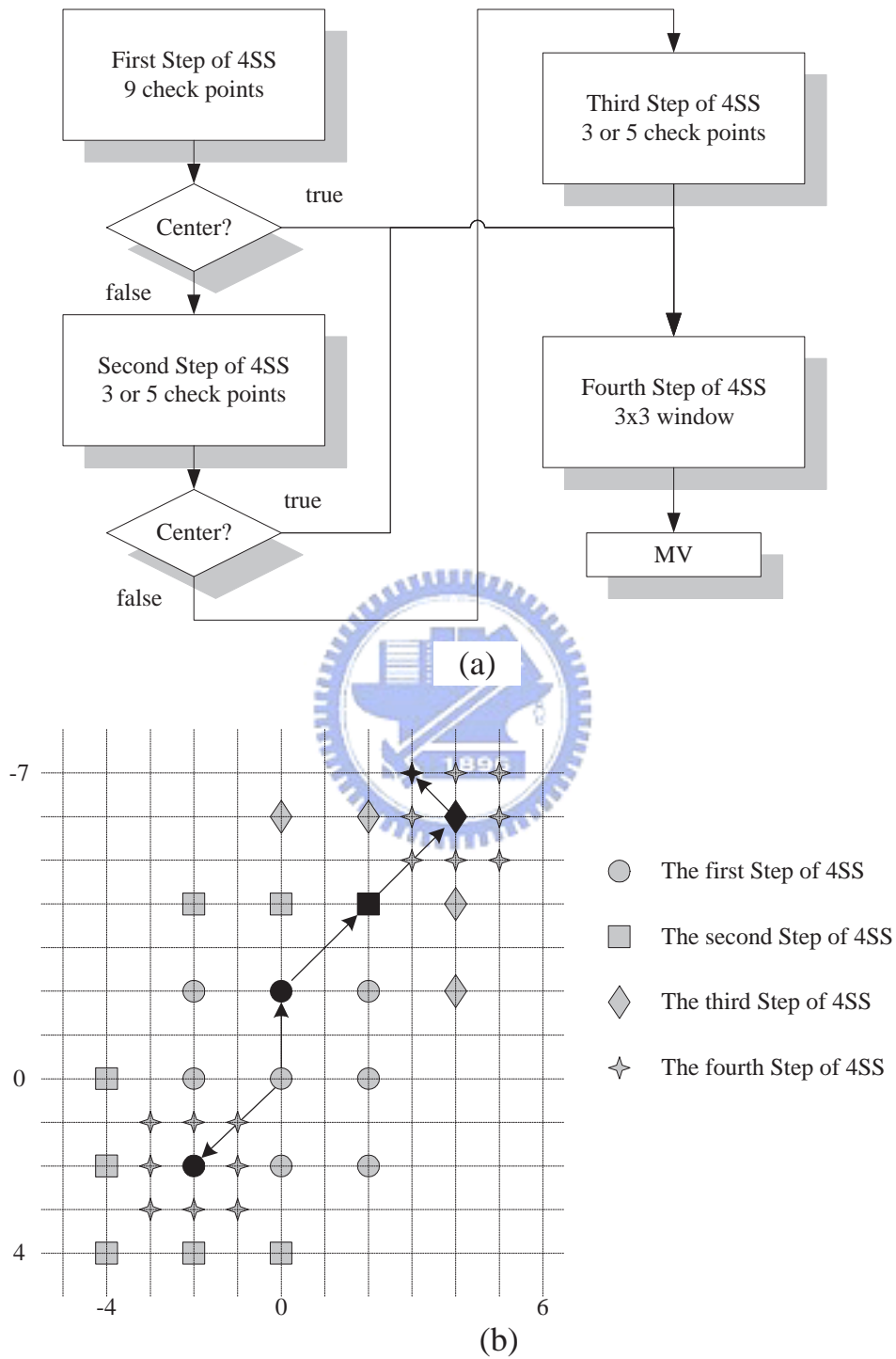


Figure 2-4: The Four Step Search.

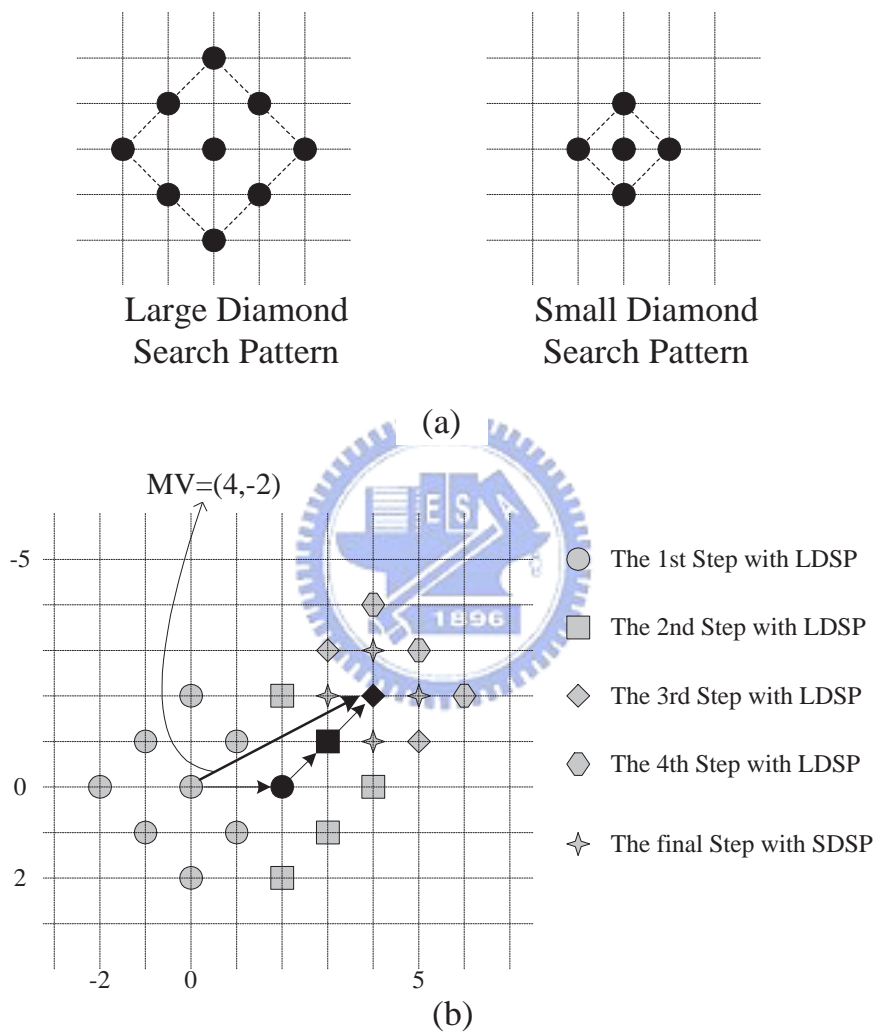


Figure 2-5: The Diamond Search.

tion of mean square error (MSE) which can achieve significant motion-compensated quality.

$$MSE(u, v) = \frac{1}{N \cdot N} \sum_{i=0}^{N-1} \sum_{j=0}^{N-1} (R(i+u, j+v) - S(i, j))^2 \quad (2-3)$$

where all the variables are defined the same as (2-1) and (4-14). However, the square operation consumes a lot of computational load by this error criterion. In order to reduce the computational complexity, the mean absolute difference (MAD) or mean absolute error (MAE) is presented which is defined as

$$MAD(u, v) = \frac{1}{N \cdot N} \sum_{i=0}^{N-1} \sum_{j=0}^{N-1} |R(i+u, j+v) - S(i, j)| \quad (2-4)$$

In practical applications, the sum of absolute difference (SAD), defined in (2-1), is usually employed instead of MAD to ignore the mean operation.

In this subsection, some techniques to conquer the drawbacks of the MSE and MAD by simplifying the error criterion in matching process are presented.

The Pixel Difference Criterion (PDC)

In this technique, the matching criterion employs the Pixel Difference Counts [28]. Each pixel in a macro-block is clarified into either a matching or a mismatching pixel by

$$T_{u,v}(i, j) = \begin{cases} 1, & \text{if } |R(i+u, j+v) - S(i, j)| \leq t \\ 0, & \text{otherwise} \end{cases} \quad (2-5)$$

for $0 \leq i, j < N$ and t is the predefined threshold. Then the PDC is defined as

$$PDC(u, v) = \sum_{i=0}^{N-1} \sum_{j=0}^{N-1} T_{u,v}(i, j) \quad (2-6)$$

Since PDC counts the number of matching pixels between current macro-block and reference macro-block, the motion vector is defined as the maximum PDC shown in following equation.

$$\overrightarrow{MV} = (u, v) \mid_{\max_{-p \leq u, v \leq p-1} PDC(u, v)} \quad (2-7)$$

Integral Projection-Matching(IPM)

Integral Projection Matching (IPM) was employed to extract the features of a macro-block as the matching criterion instead of the matching criterion mentioned above [26, 29, 30]. The major principle of projection matching is to create a cost function by summing up the luminance value of the row and the column. Equation (2-8) and (2-9) show the integral projection of the current macro-block.

$$H_k(m) = \sum_{i=0}^{N-1} S(i, m) \quad (2-8)$$

$$V_k(n) = \sum_{j=0}^{N-1} S(n, j) \quad (2-9)$$

for $0 \leq m, n < N$. As the same manner, equation (2-10) and (2-11) illustrate the feature projection of the reference macro-block with searching area parameter p .

$$H_{k-1}(m, u, v) = \sum_{i=0}^{N-1} R(i+u, m+v) \quad (2-10)$$

$$V_{k-1}(n, u, v) = \sum_{j=0}^{N-1} R(n+u, j+v) \quad (2-11)$$

for $0 \leq m, n < N$ and $-p \leq u, v < p$. The $R(\cdot)$ and $S(\cdot)$ are the luminance value, which have been defined above, of reference and current macro-block. After the cost functions of integral projection are calculated, IPM perform the matching step by (2-12) to (2-15).

$$D_H(u, v) = \sum_{m=0}^{N-1} |H_k(m) - H_{k-1}(m, u, v)| \quad (2-12)$$

$$D_V(u, v) = \sum_{n=0}^{N-1} |V_k(n) - V_{k-1}(n, u, v)| \quad (2-13)$$

$$MV_y = v|_{\min_{-p \leq u, v < p} D_H(u, v)} \quad (2-14)$$

$$MV_x = u|_{\min_{-p \leq u, v < p} D_V(u, v)} \quad (2-15)$$

2.2.3 Two-Phase Algorithm

Low Resolution Quantization Method

A two-phase fast algorithm by low-resolution quantized scheme was presented by Lee et al [31]. In the first phase, each pixel value of the current macro-block and the reference macro-blocks was quantized as two-bit low resolution by

$$\hat{f}_k(i, j) = Q_2(f_k(i, j) - Avg_k), \quad (2-16)$$

where Avg_k is the total pixel average of the current macro-block, which is defined in (3-6).

$$Avg_k = \frac{\sum_{i=0}^{N-1} \sum_{j=0}^{N-1} f_k(i, j)}{N^2} \quad (2-17)$$

Then the first phase matched the low-resolution quantized value by

$$DPC(u, v) = \sum_{i=0}^{N-1} \sum_{j=0}^{N-1} \delta[\hat{f}_k(i, j), \hat{f}_{k-1}(u + i, v + j)] \quad (2-18)$$

where

$$\hat{f}_{k-1}(u + i, v + j) = Q_2(f_{k-1}(u + i, v + j) - Avg_k), \quad (2-19)$$

and

$$\delta[\hat{f}_k, \hat{f}_{k-1}] = \begin{cases} 0, & \text{for } \hat{f}_k = \hat{f}_{k-1} \\ 1, & \text{otherwise} \end{cases} \quad (2-20)$$

After the low resolution matching scheme, the first phase generates the pre-defined number of survived motion vectors with minimum DPC in each row for the further accurate matching of the second phase. In the second phase, the algorithm figures out the motion vector from the survived motion vectors by matching with SAD criterion.

Chapter 3

Edge-driven Two-Phase Motion Estimation

3.1 Introduction



This chapter presents an edge-driven two-phase algorithm and architecture, called Edge-matching First Block-matching Last Algorithm (EFBLA), for fast motion estimation[32, 33]. In the proposed two-phase motion estimation, the major matching criterion in the first phase is low resolution quantized edge pixels of a macro-block. According to the edge pixels span, the algorithm makes decision of suitable search scan-direction to reuse the quantized data more efficiently. Then it generates the survived motion vectors for the second phase which employs the SAD as the error criteria to perform accurate matching. This content-driven algorithm can reduce significant computational load comparing with the full-search algorithm and still be more efficient than the existing two-phase algorithm.

Many papers have proposed different ways to reduce the computation com-

plexity of the full search algorithm. Most of them target on the elimination of impossible motion vectors, such as SEA[34] and LRQ[31, 35, 36], and only perform complete matching for the possible candidates. They have done great jobs on the reduction of block-matching evaluations and further save the computation power and cost. Applying this philosophy, the thesis proposes a two-phase algorithm employing content methodology to remove the impossible candidates. The edge-driven two-phase algorithm contains two major procedures, one is the edge matching and the other is the block matching. Our goal is to decrease the number of block-matching evaluations without degrading the video quality much such that the computation load can be significantly reduced. Hence, how to effectively remove the impossible motion vectors becomes the key to solve the cost-consuming problem of the full search algorithm.

The edge-matching procedure does not require complex computation; it only needs shift, quantization, comparison and threshold operations. The edge-matching procedure first performs high-pass filter on a macro-block of the current frame, called a current macro-block, and then determines edge-pixels that have larger value than threshold. According the distribution of edge-pixels, the procedure determines the scan direction for high degree of data reusability. Then, the EFBLA starts matching the current macro-block with those reference macro-blocks in the searching area of the reference frame. The matching order is based on the scan direction. The matching criterion is unmatched edge-pixel count (UEPC). An unmatched edge-pixel is the pixel of the current macro-block whose low-resolution quantized value is different from that of the corresponding edge-pixel of the reference macro-block. Obviously, the smaller the UEPC value the more similar the target block to the reference block. Thus, the EFBLA only picks the mo-

tion vectors with lower UEPC as the survived motion vectors (SMVs). Following the edge-matching phase, the proposed algorithm then performs accurately block matching with the SAD criteria on those SMVs. As results of simulating MPEG video clips, the EFBLA requires fewer addition operations than the full search algorithm.

This chapter is organized as follows. In this Section, we introduced the background and motivation of the two-phase algorithm. Section 3.2 presents the EFBLA in details and Section 3.3 proposes hardware architecture based on the EFBLA. Section 3.4 shows the experimental results and a brief summary of this work is given in Section 3.5 finally.

3.2 Algorithm

Figure 3-1 illustrates the flow chart of the Edge-matching First Block-matching Last Algorithm (EFBLA). Assume that the macro-block size is N -by- N and the searching window is $2p$ -by- $2p$. The orientation of the current macro-block is (x, y) .

3.2.1 Edge-matching phase

The edge-matching phase of EFBLA contains five steps which will be described as below:

Step 1. Perform high-pass filter on the current macro-block.

In the first phase, the proposed algorithm first performs the edge extraction using the general high-pass spatial filter mask, as shown in (3-1)[37]. In (3-1), the $S(i, j)$ represents the intensity of the pixel at (i, j) in current macro-block. Note

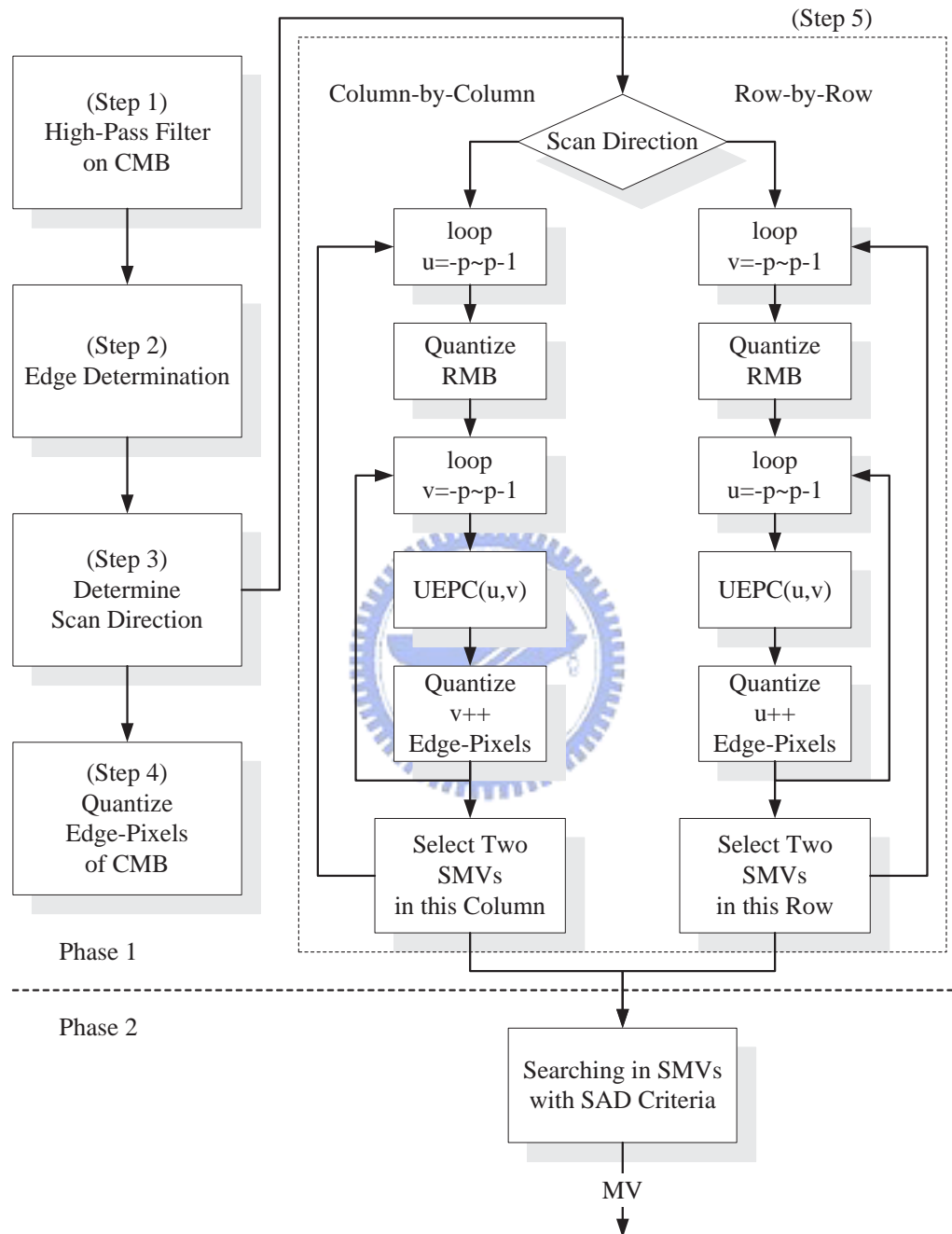


Figure 3-1: Flow chart of the EFBLA algorithm

that $G(i, j)$ expresses the gradient of the pixel at (i, j) and the larger the value of $G(i, j)$ the more possible is the pixel on the edge.

$$G(i, j) = \left| \sum_{\Delta i=-1}^1 \sum_{\Delta j=-1}^1 c \cdot S(i + \Delta i, j + \Delta j) \right|, \quad (3-1)$$

$$\text{where } \begin{cases} c = 8 & , \text{when } (\Delta i, \Delta j) = (0, 0) \\ c = -1 & , \text{otherwise} \end{cases}$$

for $0 \leq i, j < N$.

Step 2. Edge Determination

In the EFBLA, we use the local edge-determination method in current macro-block. It calculates edge threshold defined as (3-2) to determine the edge pixels. Basically, the algorithm considers those pixels with $G(i, j)$ greater or equal than E_{th} are the edge pixels, as shown in (3-3). If the pixel at (i, j) is the edge pixel, $\alpha(i, j)$ is set to 1; otherwise, $\alpha(i, j)$ is set to 0.

$$E_{th} = \frac{\max(G(i, j)) + \min(G(i, j))}{2} \quad (3-2)$$

$$\alpha(i, j) = \begin{cases} 1 & , \text{if } G(i, j) \geq E_{th} \\ 0 & , \text{otherwise} \end{cases} \quad (3-3)$$

In order to increase the accuracy of the edge-matching, the EFBLA also regards the pixels around pixels with $G(i, j)$ greater than E_{th} as the edge pixels as well. Thus, the EFBLA employs the edge extension as shown in (3-4) to mark the edge pixels.

$$\alpha(i, j) = \begin{cases} 1 & , \text{if } G(i \pm 1, j \pm 1) \geq E_{th} \\ 0 & , \text{otherwise} \end{cases} \quad (3-4)$$

Step 3. Determine the scan direction.

The data reusability is highly dependent on the scan direction because it employs the criteria of unmatched edge-pixel count (UEPC), which will be illustrated in step 5. Before UEPC, EFBLA has to quantize the edge pixels in the macro-block first. There are highly duplicated data in the successive searching steps. Fig.3-2 shows the impact of the scan direction to the data reusability as an example. If the edge pixels are widely distributed along with the y-coordinate, searching along with x-coordinate can reuse the quantized data efficiently. In Fig.3-2(a), a macro-block which size is 8-by-8 and black circles means an edge pixels, that is, the $\alpha(i, j)$ is equal to 1 in the position marked as black circle. In Fig.3-2(b) and (c), we assumed that the searching position shifts from A to B. The gray and black marks represent the edge pixels when the referent macro-block is at the position A. The black and white marks represent the edge pixels when the target block is at the position B. Therefore, the quantized data at the black marks can be reused in matching step that uses the criteria of UEPC. Obviously, it just needs to calculate the quantized edge pixels in the white marks only, then removes the unmatched edge pixel count in gray marks and plus these in white marks. So the scan direction in Fig.3-2(b) has higher degree of data reusability than that in Fig.3-2(c).

The EFBLA has two scan directions: column-by-column and row-by-row, as illustrated in Fig.3-3. To decide the scan direction, this step first determine the span width of edge pixels with x-coordinate, named the x-span, and the span width of edge pixels with y-coordinate, named the y-span. If the x-span is smaller than y-span, the step selects the column-by-column scan as the direction; otherwise, the scan direction will be row-by-row. As the example shown in Fig.3-2, the value of x-span is equal to four and the y-span is eight, and therefore the efficient

scan direction is column-by-column.

Step 4. Quantize the edge pixels of the macro-block.

This step quantizes the pixel values at the edge pixels for the low-resolution computation. The philosophy of two-phase motion estimation is to eliminate impossible motion vectors at the lowest computation cost. Hence, the EFBLA utilizes low-resolution computation to perform the edge matching.

Equation (3-5) represents the quantization of the reference blocks where $\hat{S}(i, j)$ is the value of two most significant bits (MSBs) of $(S(i, j) - Avg_k)$. The reason that the step quantizes $(S(i, j) - Avg_k)$ instead of $S(i, j)$ is because the former has higher variance than later. The higher variance leads to higher degree of accuracy for edge matching.

$$\hat{S}(i, j) = Q_2(S(i, j) - Avg_k), \forall \alpha(i, j) = 1, \quad (3-5)$$

where Avg_k is the total pixel average of the current macro-block, which is defined as

$$Avg_k = \frac{\sum_{i=0}^{N-1} \sum_{j=0}^{N-1} S(i, j)}{N^2} \quad (3-6)$$

Step 5. Perform edge matching and generate SMVs.

Upon the completion of the step 3 and 4, the first phase starts to perform edge matching. First, the EFBLA matches the motion vectors along with the scan direction obtained by the step 3. The edge matching employs the criteria of unmatched edge-pixel count (UEPC), as shown in (3-7). In (3-7), $\hat{R}(u+i, v+j)$ is

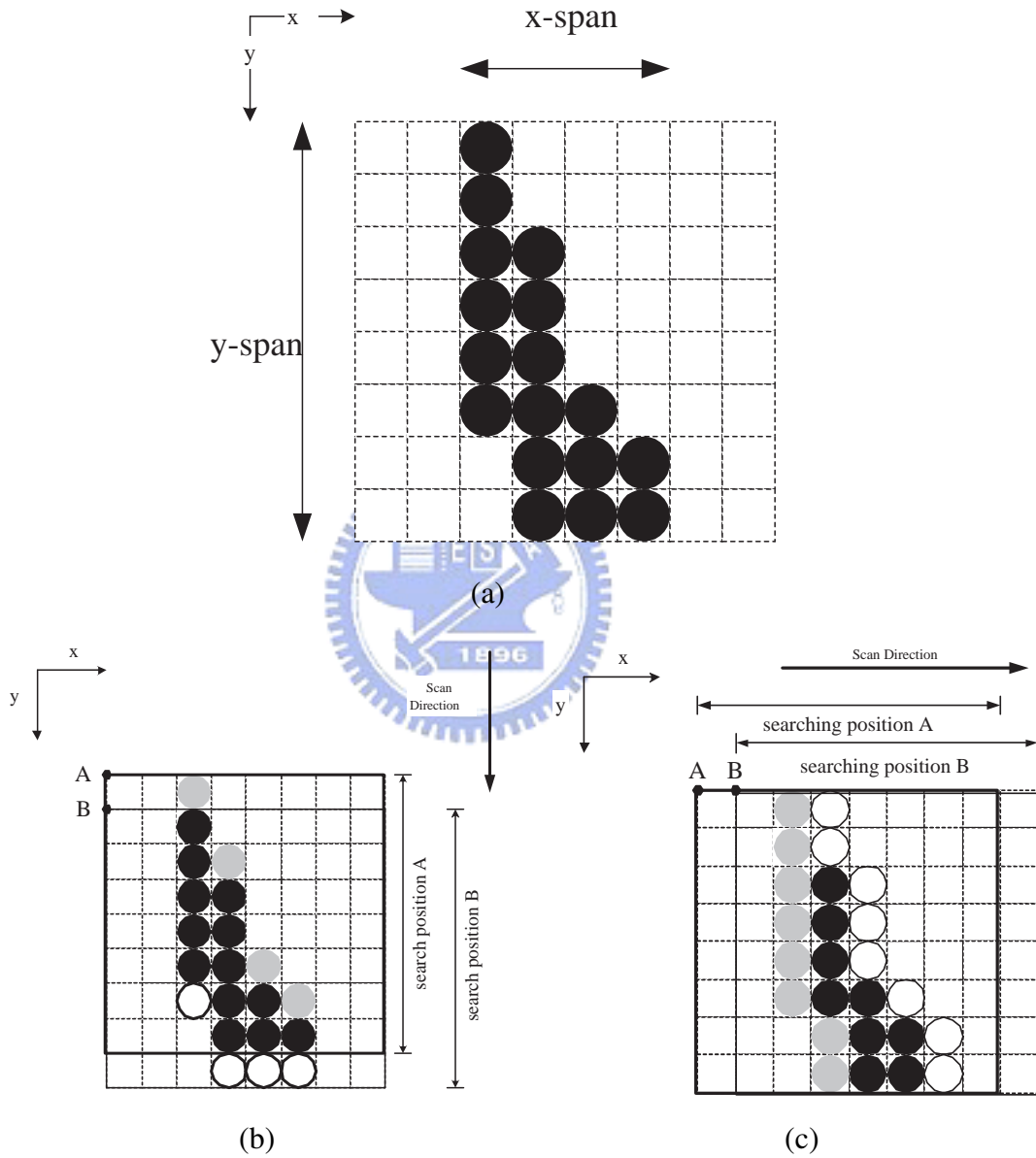


Figure 3-2: The reusability of quantized data in EFBLA

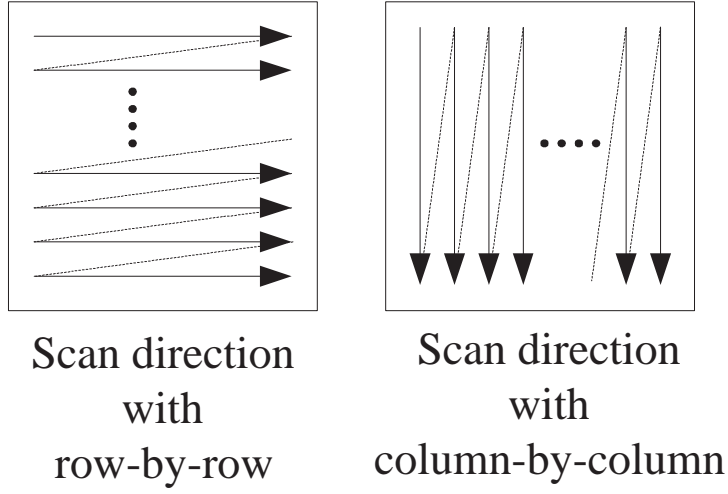


Figure 3-3: Two scan directions employed in EFBLA

the quantization result of the reference macro-block with the motion vector (u, v) .

$$UEPC(u, v) = \sum_{i=0}^{N-1} \sum_{j=0}^{N-1} \alpha(i, j) \cdot \delta[\hat{S}(i, j), \hat{R}(u + i, v + j)], \quad (3-7)$$

where

$$\hat{R}(u + i, v + j) = Q_2(R(u + i, v + j) - Avg_k), \forall \alpha(i, j) = 1, \quad (3-8)$$

and the *delta* function is defined as

$$\delta[\hat{S}, \hat{R}] = \begin{cases} 0 & , \text{for } \hat{S} = \hat{R} \\ 1 & , \text{otherwise} \end{cases} \quad (3-9)$$

Next, this step generates a pair of SMVs for each scan line, either row or column. The motion vectors with the high UEPCs on a scan line are most likely impossible ones. Thus, the EFBLA only picks the motion vectors with the lowest two UEPC

reference macro-blocks as the survived motion vectors (SMVs).

3.2.2 Block-matching phase

Following the edge-matching phase, the second phase of EFBLA performs block-matching with SAD criteria on SMVs. Note that the block-matching requires much less evaluations than the traditional full search block-matching because the first phase has eliminated a large amount of impossible motion vectors.

3.3 Architecture

According to the Edge-matching First Block-matching Last Algorithm depicted in the previous section, this thesis proposes a two-phase VLSI architecture and its block diagram is showed in Fig. 3-4. In order to achieve the goal of parallel processing and avoid multiple data access from the off-chip frame memory, the proposed architecture is base on a two-dimensional systolic array in both phases and saves the data of current/reference macro-block in the CMB/RMB buffer. In the following subsections, the architecture and behavior of each block will be illustrated.

3.3.1 The First Phase

The architecture of the first phase contains a Current Macro-Block Buffer, an Edge Generator Unit, a UEPC PEs Array, a Reference Macro-Block Buffer, a Quantization Unit, an Adder Tree and a Survived Motion Vectors Selector. After the edge matching processing, the first phase generates two survived motion vectors in each searching row/column for the second phase to perform more accurate matching.

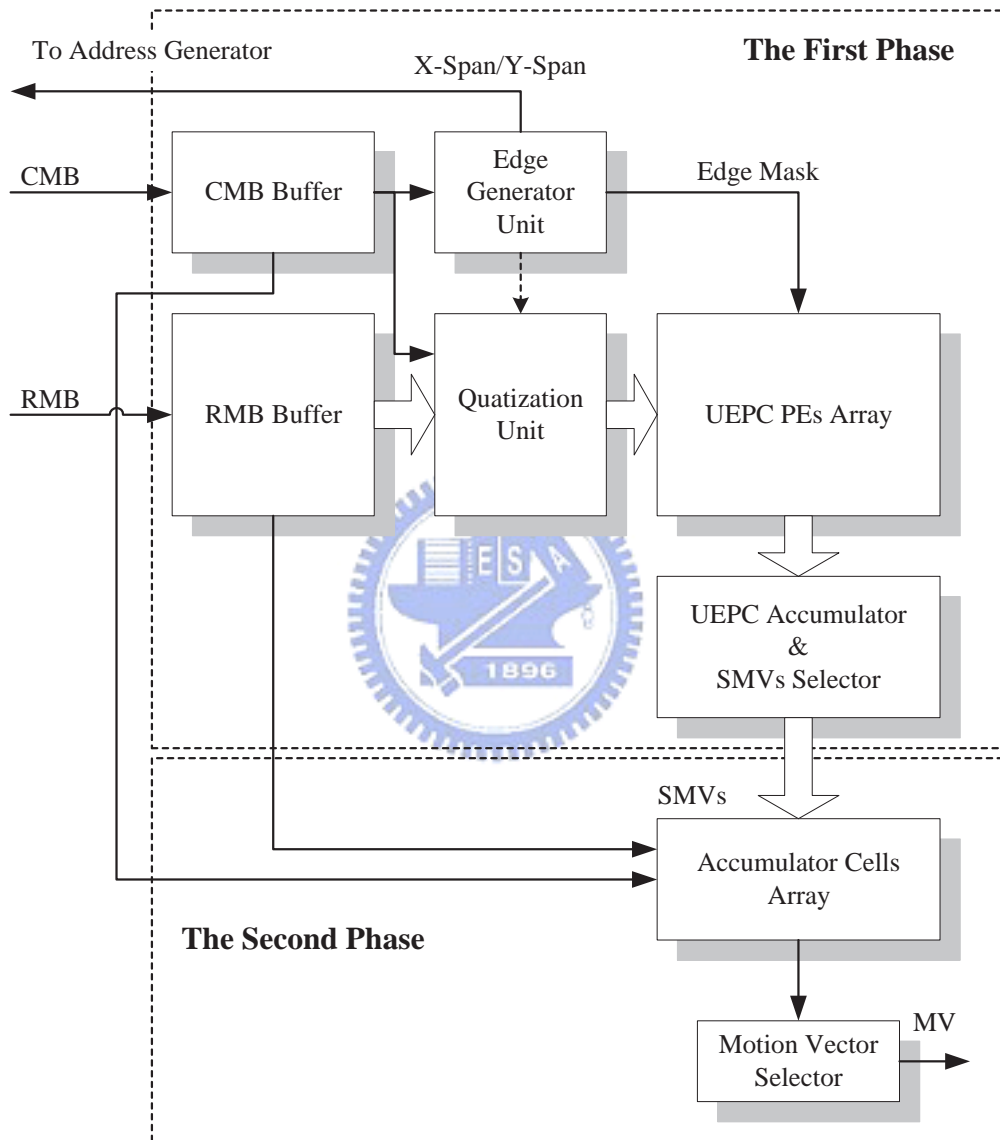


Figure 3-4: Block diagram of the edge-driven two-phase motion estimation.

Edge Generator Unit

In Fig. 3-5, we presented the architecture of Edge Generator Unit which is used to produce the edge mask and makes decision of search scan-direction described from **Step 1** to **Step 3** in section 3.2. This unit contains two main blocks, the high-pass filter block and edge determination block. The former block calculates the gradient of each pixel in current macro-block as shown in equation (3-1). The later one is used to determine the edge mask and X-Y span depicted in the **Step 3** of EFBLA.

According to (3-1), the high-pass filter calculates the gradient of a target pixel with eight neighbor pixels around it. The data paths, CMB_1 , CMB_2 , and CMB_3 , are the input interface of previous line, current line and next line from the CMB buffer. The left and right pixels can be reserved by simply delay elements. In order to avoid the boundary error when the target pixel is in the border, the proposed architecture uses multiplexers to switch the null data out of the current macro-block to existent pixels instead. The black-dot in each multiplexer indicates the switching path when the filter unit is processing a border pixel. To calculate the gradient value of a target pixel needs total six equivalent adder operations, which are five adder operations and one absolute operation. We treated the computational load of an absolute operation as an adder operation. The computational load of $\times 8$ is ignored since it can be implemented with simple shift operation.

The edge determination unit, whose structure is illustrated in the right part of Fig. 3-5, implements two main functions. The first one is to figure out the maximum and minimum of the gradient value of the current macro-block and then determines the threshold value according to the equations from (3-2) to (3-4). The second one is to decide the searching scan-direction depicted in **Step 3** of EFBLA.

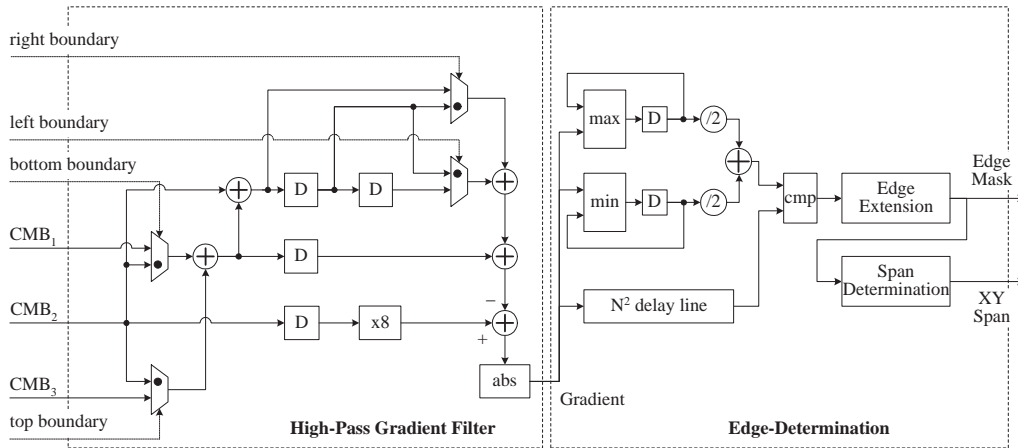


Figure 3-5: Architecture of Edge Generator Unit.

The determination of scan-direction contains simple logic OR gate and look-up table (LUT) to figure out the XY-span. The edge determination unit generates the edge mask and scan direction for the UEPC matching in the first phase.

RMB Buffer and Quantization Unit

Figure 3-6 illustrates the architecture of RMB buffer and Quantization Unit. The reference macro-block (RMB) buffer has two major functions; one is to provide the parallel data for UEPC PEs arrays in the first phase. The second function is to buffer the data of reference macro-block for the second phase since by this way we can ensure that it accesses the data from the reference frame memory only once. In each clock period, the RMB buffer provides N pixels at the same time to the Quantization Unit and the Quantization Unit transfers them to low-resolution data for the matching procedure of UEPC.

In order to save the hardware resource, the quantization procedure for the current macro-block shares the same quantization cell with the reference macro-block. At the initial time, there are $(N + 2P - 1) \times (N - 1)$ cycles to store the

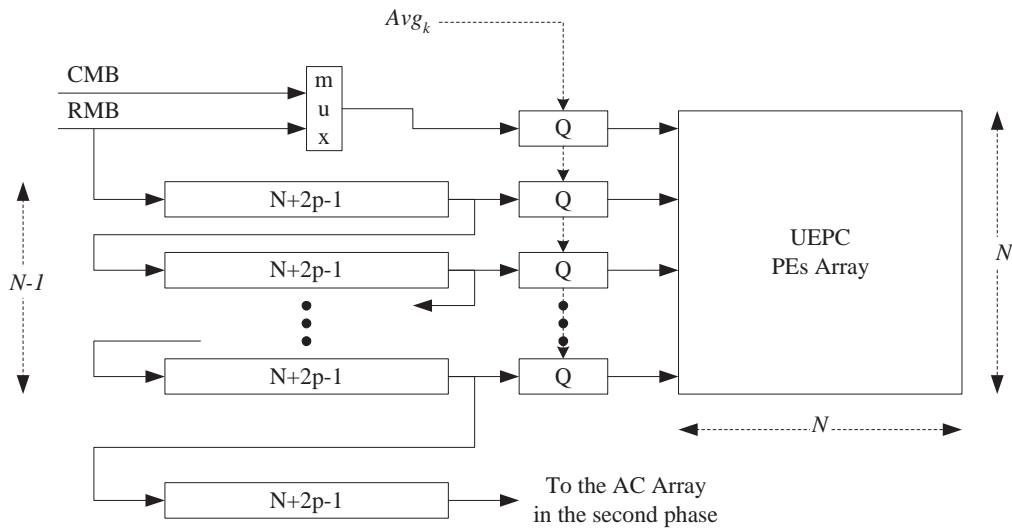
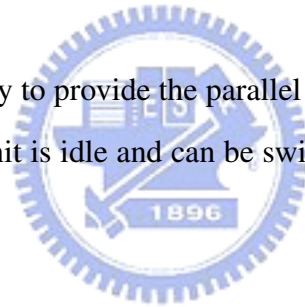


Figure 3-6: Architecture of Shift Register Array and Low-Resolution Quantization.

reference macro-block ready to provide the parallel data for the PEs array. In this period, the Quantization Unit is idle and can be switched to quantize the current-macro-block.



Processing Elements Array

The architecture of Processing Elements Array is illustrated in Fig. 3-7. The array is composed of N -by- N processing elements to calculate the criteria of unmatched edge-pixel count shown in (3-7). The data path of CMB in the tail of a row is linked to the head of the next row and thus it needs N^2 cycles to shift all the quantized data of current macro-block into the UEPC PEs array. By this linked data path, to quantize the current macro-block only needs to active one Quantization Unit.

Since the first phase uses the criteria of unmatched edge-pixel count, the PEs array activates the processing element while corresponding pixel is an edge, that

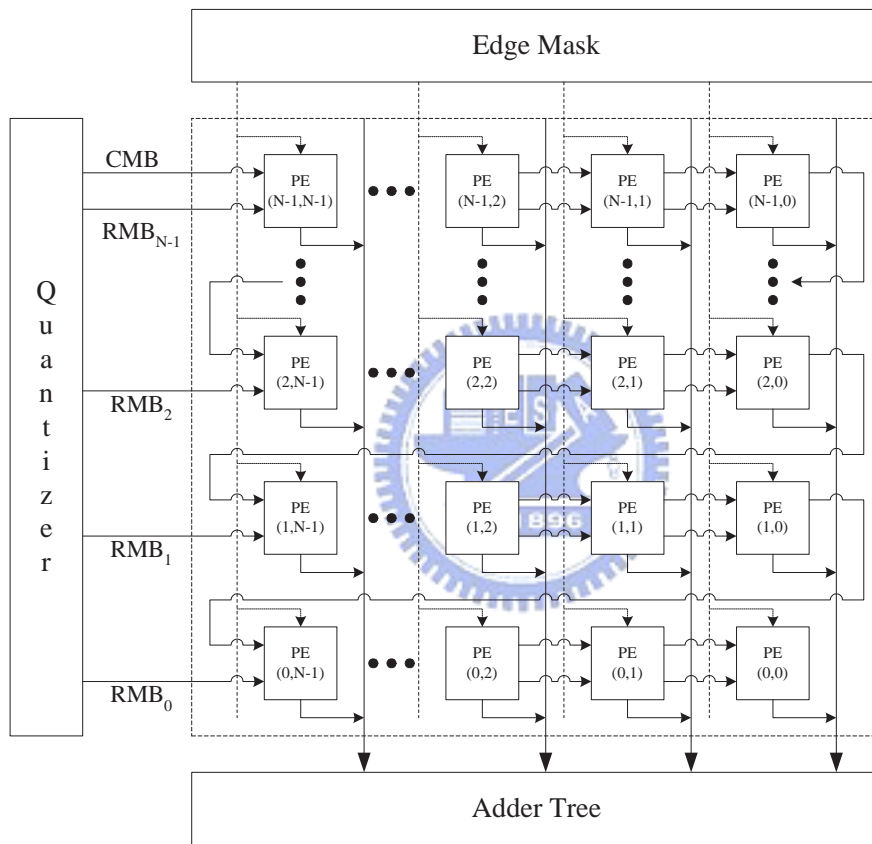


Figure 3-7: Architecture of Processing Element Array.

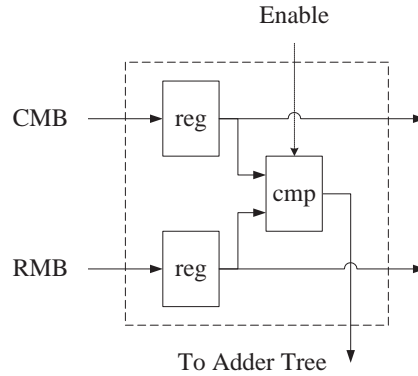


Figure 3-8: Architecture of Processing Element.

is, the edge mask $\alpha(u, v)$ is equal to 1 shown in (3-4). The turn-on/off signal is from the Edge Mask generated from the Edge Generator Unit. The processing element, which architecture is shown in Fig. 3-8, performs the unmatched edge-pixel comparison and produces a signal 1 if the quantized data of current macro-block is not identical to that of reference macro-block. The architecture of the processing element to calculate the unmatched edge-pixel count is shown in Fig. 3-8. Each processing element contains two two-bit shift register to store the low-resolution information of current macro-block and reference macro-blocks. The compared circuit in a PE can be implemented with two exclusive-OR gates and one OR gate. After the matching process, each processing element generates one bit signal to the adder tree and SMVs selector for further evaluating the correlation between the current and reference macro-block.

UEPC Accumulator and SMVs Selector

The UEPC accumulator is used to accumulate the unmatched edge-pixel signal from each processing element. There is a look-up table (LUT) in each column to transfer the unmatched signals to a binary number which counts that how many

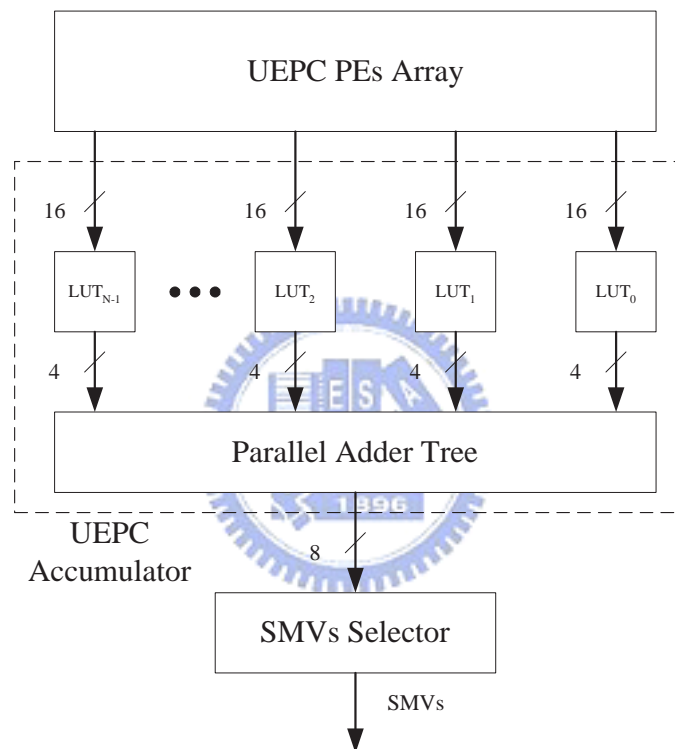


Figure 3-9: Architecture of UEPC Adder Tree and SMVs Selector. We assume that N is 16.

unmatched pixels in this column. Then the binary number can be summed up by a parallel adder tree to measure the total unmatched edge-pixels in the macro-block. The SMVs selector uses these unmatched edge-pixel counts to pick up two survived motion vectors in each column for further detail matching in the second phase. So the first phase figures out 2 -by- $2p$ survived motion vectors which are the most possible motion vectors to the second phase. The architecture of the UEPC accumulator and SMVs selector are shown in Fig. 3-9.

3.3.2 The Second Phase

In the second phase, it consists of an Accumulator Cell Array and a Motion Vector Selector. The former unit is used to accumulate the SAD in position of the survived motion vectors and it is composed of 2 -by- N accumulator cells. The later one, Motion Vector Selector, is used to compare the SAD calculated from the Accumulator Cell Array and to pick a best motion vector up with the minimum distortion in the SMVs.

The Accumulator Cells Array

The second phase performs further matching with SAD criteria between those SMVs generated from the first phase. Figure 3-10 shows the block diagram of this phase. It consists of an accumulator cells array with dimension N -by- 2 and a controller. Each accumulator cell is used to calculate the SAD value of a row in a macro block. Architecture of the accumulator cell is shown in the dash circle on the right hand. The enable signal from the controller is used to active the accumulator cell when the data in the RMB bus is in the range of the searching position of corresponding survived motion vector generated form the first phase.

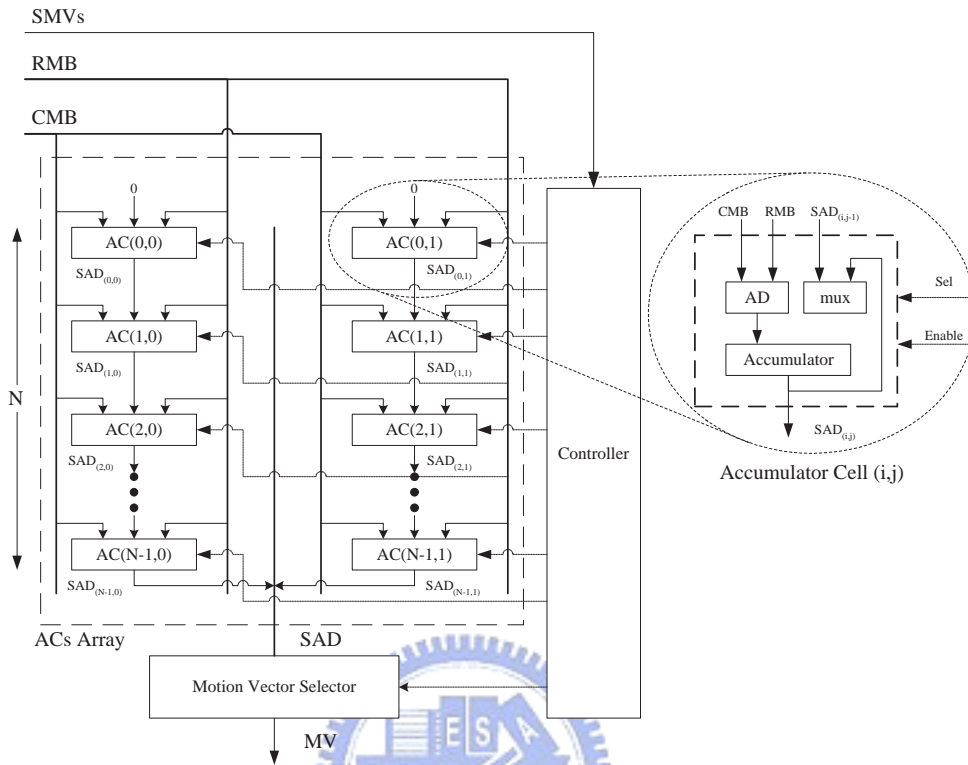


Figure 3-10: Architecture of the second phase.

When the index counter is in the range of the SMVs, the controller generates the enable signal to the corresponding accumulator cell to calculate the SAD value at this searching position. The control signal named as Sel is used to switch the multiplexer to receive the partial SAD value from the previous accumulator cell.

Figure 3-11 shows the execution of the accumulator cells array in the second phase. In this diagram, it assumes that the macro block size N is 8 and the searching window is from -8 to 7 . The searching scan-direction is row direction. In this illustration, the SMVs are $(-8, -2)$ and $(-8, 4)$ in the first row, $(-7, -7)$ and $(-7, 7)$ in the second row and so on. In order to make the graph concise, the diagram does not show every box of all SMVs as illustration.

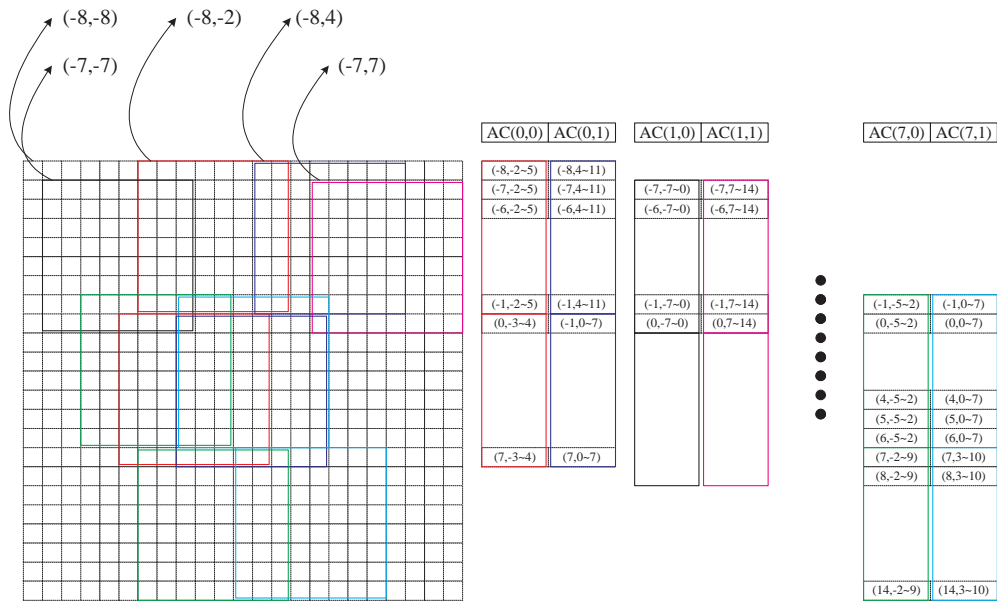


Figure 3-11: Execution of the Accumulator Cells Array in the condition of $N = 8$ and $p = 8$.

The Motion Vector Selector

In the final step, the Motion Vector Selector receives the matching results from the accumulator cells array and identify the SADs step by step to figure out the the motion vector with minimum distortion from those survived motion vectors.

3.4 Performance Analysis

The proposed algorithm significantly reduces the number of motion vectors that requires costly evaluations. To compare with the other motion estimation algorithms, this paper uses two metrics: computation cost and the mean absolute difference (MAD). Since the major operation of motion estimation algorithms is addition, we approximately consider the total number of equivalent addition, denoted as ϵ_{adder} , required for each macro-block as the computation cost. In this

chapter, the 21 MPEG video clips of CIF format as test bench [38]. Each frame has 352 by 288 pixels and each pixel value is 8-bit gray resolution. The macro-block size N is 16-by-16, and the search window range is from $(-16, -16)$ to $(15, 15)$. The full search algorithm (FS) and a two-phase algorithm, the low resolution quantization algorithm (LRQ), are as comparisons with the proposed algorithm EFBLA.

Table 3.I. and 3.II. show the quality performance and computational load of these test clips. The results shown in the two tables are the average of 100 frames for each test clip. Obviously, the EFBLA significantly saves 17.47% of the computation cost while the MAD degradation is only 0.065 per pixel averagely with comparing to LRQ. Fig. 3-12 (a) to (d) demonstrate the MAD curves of four typical clips and shows that the quality of the EFBLA is very close to the quality of the others. The Akiyo and Weather clips are slow motion and the Children clip is belong to middle motion type. The test sequence Stefan is a type of fast motion. These results show that the EFBLA is capable of lower computational load and having a good quality as well.

3.5 Brief Summary

This chapter proposes a two-phase algorithm and architecture to significantly reduce the computational load of motion estimation by removing the unlikely motion vectors in the first phase. As the result of simulating video clips, the quality degradation is very little comparing with FS, only degrading 0.435 per pixel in MAD averagely. In addition, the algorithm features adaptive choosing for the scan direction; it turns out a high degree of data reusability and low memory requirement.

Table 3.I.: Quality degradation analysis for different video clips.

Clips	FS	LRQ	EFBLA	vs. FS	vs. LRQ
akiyo	0.605	0.645	0.652	0.047	0.007
children	2.572	2.882	2.930	0.358	0.048
coastguard	5.341	6.309	6.390	1.049	0.081
container	1.564	1.578	1.591	0.027	0.013
dancer	2.696	3.963	3.974	1.278	0.011
destruct	4.022	4.439	4.475	0.454	0.036
flower	6.000	6.367	6.491	0.491	0.124
foreman	2.838	3.614	3.684	0.846	0.070
hall monitor	2.543	2.678	2.686	0.143	0.008
mobile	8.837	9.053	9.445	0.608	0.392
mother daughter	1.496	1.646	1.645	0.148	-0.001
news	1.197	1.336	1.349	0.151	0.013
paris	2.500	2.732	2.782	0.282	0.049
sean	1.647	1.713	1.725	0.078	0.012
silent	1.723	1.923	1.930	0.207	0.007
singer	0.821	0.885	0.885	0.064	-0.000
stefan	6.615	7.429	7.715	1.099	0.286
table tennis	4.388	5.262	5.298	0.910	0.036
tempete	5.685	6.181	6.336	0.651	0.155
waterfall	2.948	3.152	3.150	0.202	-0.002
weather	0.797	0.830	0.847	0.050	0.017
Average	3.183	3.553	3.618	0.435	0.065

Table 3.II.: Computational load analysis for different video clips.

Clips	FS	LRQ	EFBLA	vs. FS	vs. LRQ
akiyo	711341	69105	57138	-91.97%	-17.32%
children2	711341	69105	55823	-92.15%	-19.22%
coastguard	711341	69105	58278	-91.81%	-15.67%
container	711341	69105	57416	-91.93%	-16.92%
dancer	711341	69105	58929	-91.72%	-14.72%
destruct	711341	69105	56073	-92.12%	-18.86%
flower	711341	69105	57918	-91.86%	-16.19%
foreman	711341	69105	56121	-92.11%	-18.79%
hall monitor	711341	69105	56671	-92.03%	-17.99%
mobile	711341	69105	56742	-92.02%	-17.89%
mother daughter	711341	69105	57223	-91.96%	-17.19%
news	711341	69105	56064	-92.12%	-18.87%
paris	711341	69105	56042	-92.12%	-18.90%
sean	711341	69105	56911	-92.00%	-17.65%
silent	711341	69105	56887	-92.00%	-17.68%
singer	711341	69105	56734	-92.02%	-17.90%
stefan	711341	69105	57402	-91.93%	-16.94%
table tennis	711341	69105	57469	-91.92%	-16.84%
tempete	711341	69105	56905	-92.00%	-17.65%
waterfall	711341	69105	58352	-91.80%	-15.56%
weather	711341	69105	56631	-92.04%	-18.05%
Average	711341	69105	57035	-91.98%	-17.47%

Unit: Equivalent Adder (ε_{adder}).

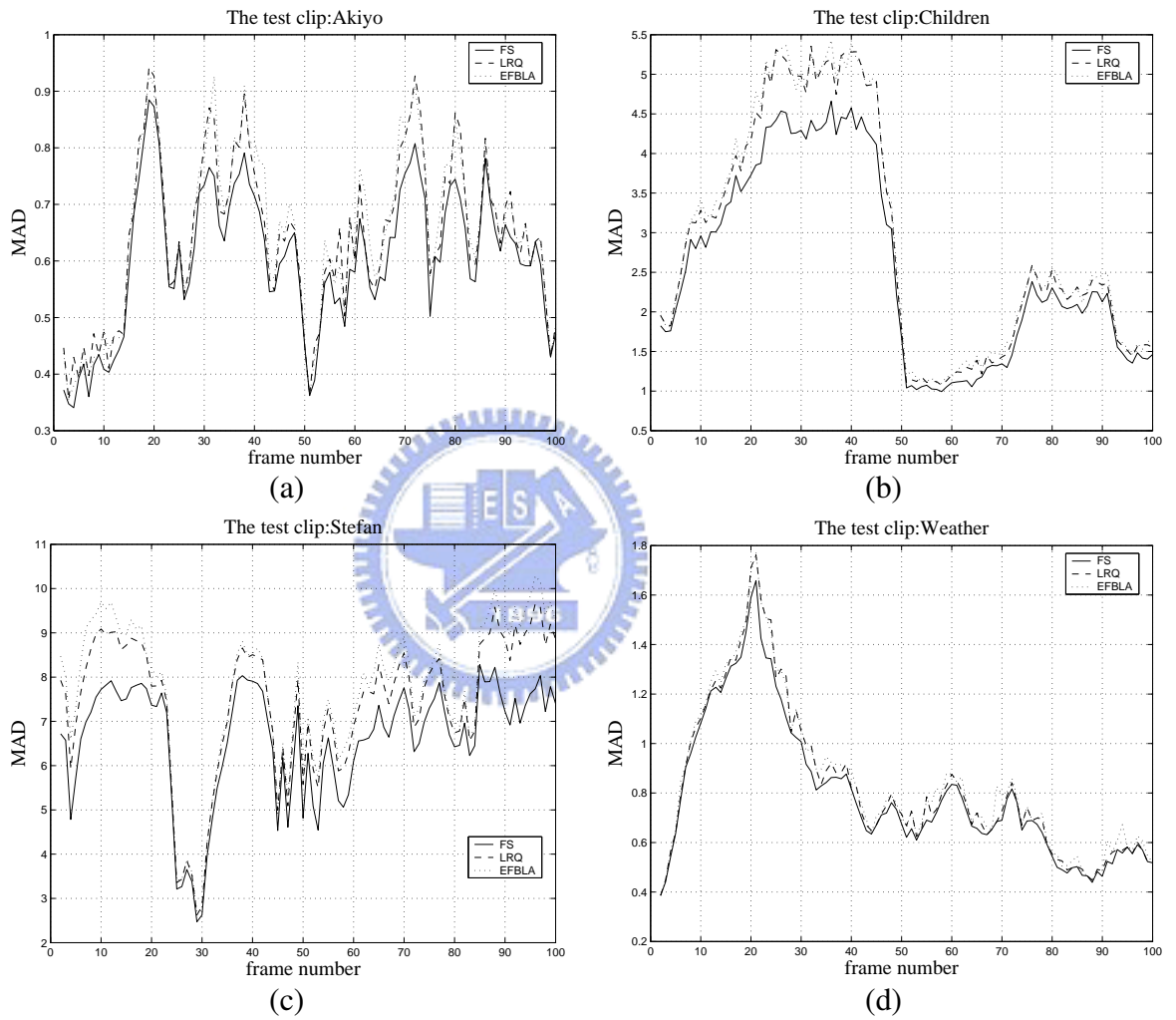


Figure 3-12: MAD curves of FS, LRQ and EFBLA for four clips. (a) The Akiyo Clip. (b) The Children Clip. (c) The Stefan Clip. (d) The Weather Clip.

Chapter 4

Power-Aware Algorithm and Architecture

This chapter presents a power-aware architecture based on subsample algorithms to perform graceful tradeoffs between power consumption and compression quality while the battery status changes [39–41]. As the available energy decreases, the algorithm raises the subsample rate for maximizing battery lifetime. As shown in experimental results, the proposed algorithm and architecture can dynamically operate at different power consumption modes with little quality degradation according to remaining capacity of battery pack.

This chapter is organized as follows. In Section 4.1 and 4.2, we will introduce the motivation and background of power-aware paradigm. Section 4.3 and 4.4 present generic and content-based subsample algorithms in detail. Section 4.6 describes the proposed power-aware architecture and section 4.7 shows the performance analysis. Finally, Section 4.8 is the conclusion of this work.

4.1 Motivation

Motion estimation (ME) has been notably recognized as the most critical part in many video compression applications, such as MPEG standards and H.26x, which tends to dominate most computational load and hence power requirements. With increasing demand of battery-powered multimedia devices, an ME architecture that can be flexible in both power consumption and compression quality is highly required. The requirement is driven by user-centric perspective [42]. Basically, users have two thoughts on using portable devices. Sometimes, users might want extremely high video quality at the cost of reduced battery lifetime. At other times, users might want acceptable quality for extending battery lifetime.

This chapter, therefore, intends to present a novel power-aware ME architecture using 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 at different power consumption modes with acceptable quality degradation. Since the control mechanism and data sequences at 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. 4-1 illustrates a typical application of the proposed power-aware ME architecture. The host processor monitors the remaining capacity of battery pack and switches the 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 the battery monitor unit and power management subroutines. Besides the power-aware motion estimation unit,

all the units marked as gray background also can be designed with power-aware capability to facilitate this portable system to be friendlier for the battery usage. In this chapter, the thesis focuses the target to the power-aware motion estimation based on the content property.

Lots of published papers have presented efficient algorithms for VLSI implementation of motion estimation, on either high performance or low power design. Yet, 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 Sum of Absolute Difference (SAD) criterion is the most popular approach for motion estimation because of its considerably good quality. It is particularly attractive to the ones who require extremely high quality. There are many types of architectures that have been proposed for the implementation of FSBM algorithms [8, 11, 12, 15]. However, they require a huge number of *comparison/difference* operations and result in high computation load and power consumption. To reduce the computational complexity of FSBM, researchers have proposed various fast algorithms. They either reduce search steps [17–19, 21, 43, 44] or simplify calculations of error criterion [13, 29, 34, 45]. By combining step-reduction and criterion-simplifying, some researchers proposed two-phase algorithms to balance the performance between complexity and quality [31, 32, 46]. They first use FSBM with a simplified matching criterion to generate candidate vectors and then select the best motion vector from these candidates with SAD criterion. These fast-search algorithms have successfully improved the block matching speed while the quality degradation is little and, thus, lead to a 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

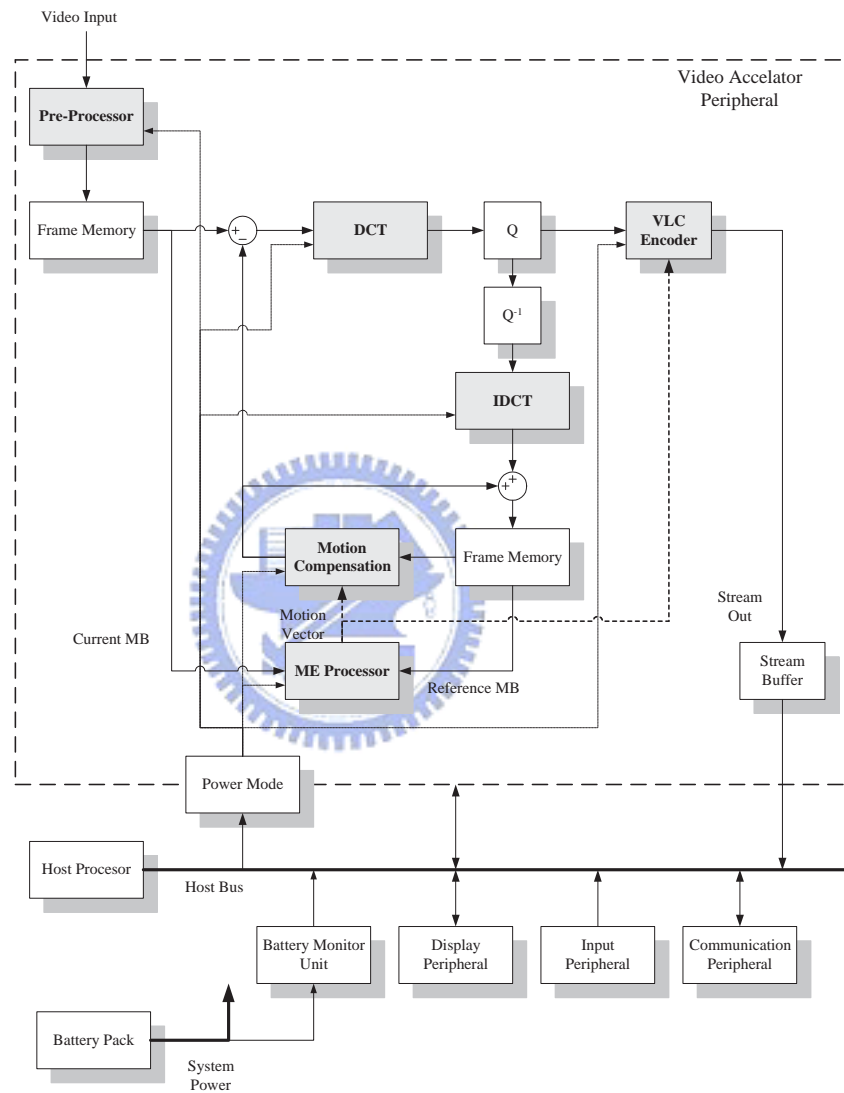


Figure 4-1: The system block diagram of a portable, battery-powered multimedia device.

behavior with the change of power/energy status and balance the performance between quality and battery life [47]. The requirement for ME algorithms to be suitable for power-aware design is high degree of scalability in performance tradeoffs. Unfortunately, the fast algorithms mentioned above do not meet the requirement.

Articles in [24, 48] present subsample algorithms to significantly reduce the computation cost with low quality degradation. The reduction of computation cost implies the saving of power consumption. Since the power consumption can be reduced by simply increasing the subsample rate, the subsample algorithms have high degree of scalability and are very suitable for power-aware ME architecture. However, applying subsample algorithms for power-aware architecture may suffer from aliasing problem in high frequency band. The aliasing problem degrades the compression quality rapidly as the subsample rate increases. To alleviate the problem, we extend traditional subsample algorithms to a content-based algorithm, called the content-based subsample algorithm (CSA). In the 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. 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 have significant power consumption save and keep the quality degradation little 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 the 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 accordingly with subsample rate. In this chapter, we will 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.

4.2 Battery Properties

One may simply consider battery as a capacitor in which the charge capacity is linearly proportional to the output voltage. However, in practice, the behavior of battery is non-ideal for its variable voltage and capacity. There are two most important properties of battery, the rate capacity effect and recovery effect [49]. The first effect states that the capacity of battery is dependent on the discharging rate and the second one means that a battery with an intermittent load may have larger capacity than that with a continuous load. Figure 4-2(a) illustrates the rate capacity effect by plotting the cell voltage for two different discharging loads with time advancing. As shown in the curves, when the load is halved the battery life can be more than two times longer. Figure 4-2(b) shows the recovery effect in which the reduction of load causes the 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 the mobile CPU has the same strategy to extend the battery lifetime [50]. The technology changes the power consumption mode by scaling down the supplied voltage and operating frequency, and hence degrades the performance to gain battery lifetime.

From these two properties of battery, we can learn two things. First, one can reduce the load for longer battery lifetime because halving the current can more than double the battery lifetime. Second, it is worthy to have optimal performance

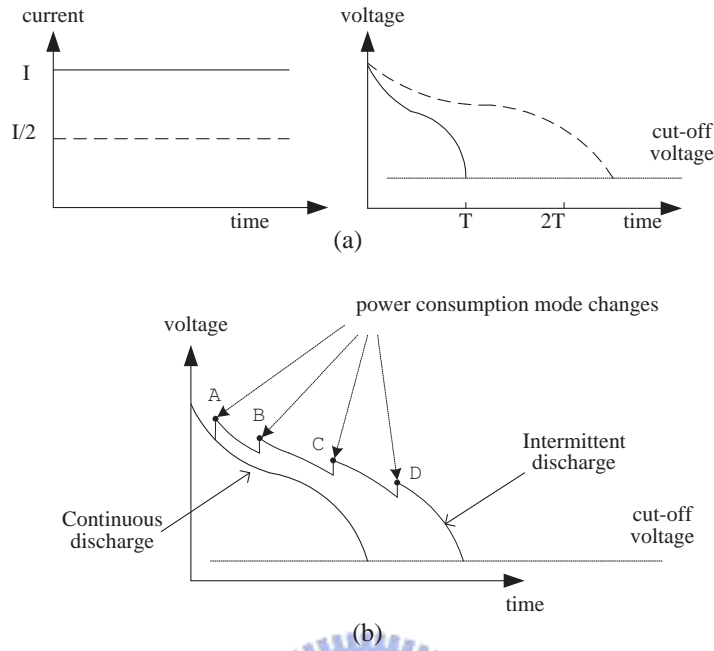


Figure 4-2: This diagram represents the non-linear discharging properties of battery. (a) Rate capacity effect. (b) Recovery effect.

when the battery is fully charged because the battery capacity can be recovered later by reducing the load. These form a good motivation for power-aware design and reason out the requirement of power-aware architecture — high degree of scalability in energy-quality tradeoffs.

4.3 Generic Subsample Algorithm

Much research addressed subsample techniques for motion estimation to reduce the computation load of FSBM [24, 48, 51]. Liu and Zaccarin, as pioneers of subsample algorithms, applied 4-to-1 subsampling to FSBM and significantly reduced the computation load. As the simulation results, the 4-to-1 subsample algorithm reduces the computation load significantly while the quality is similar to

that of exhaustive search.

Here, we presented a generic subsample algorithm in which the subsample rate ranges from 4-to-1 to 1-to-1. The generic subsample algorithm uses (4-1) as a matching criterion, called as subsample sum of absolute difference (SSAD), where $SM_{8:m}$ is the subsample mask for the subsample rate 8-to- m as shown in (4-2). The macro-block size is N -by- N and $S(i, j)$ is the luminance value at (i, j) of the current macro-block (CMB). The $R(i + u, j + v)$ is the luminance value at (i, j) of the reference macro-block (RMB) which offsets (u, v) from the CMB in the searching area $2p$ -by- $2p$.

$$SSAD_{8:m}(u, v) = \sum_{i=0}^{N-1} \sum_{j=0}^{N-1} |SM_{8:m}(i, j) \cdot [R(i + u, j + v) - S(i, j)]|, \quad (4-1)$$

for $-p \leq u, v \leq p - 1$ and the subsample mask $SM_{8:m}$ is defined as

$$SM_{8:m}(i, j) = BM_{8:m}(i \bmod 4, j \bmod 4) \quad (4-2)$$

and the $SM_{8:m}$ is generated from basic mask as shown in (4-3).

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

where $u(n)$ is a step function; that is, $u(n) = \begin{cases} 1, & \text{for } n \geq 0 \\ 0, & \text{for } n < 0. \end{cases}$

For example, consider the subsample rate 8-to-6. The subsample mask $SM_{8:6}$ can

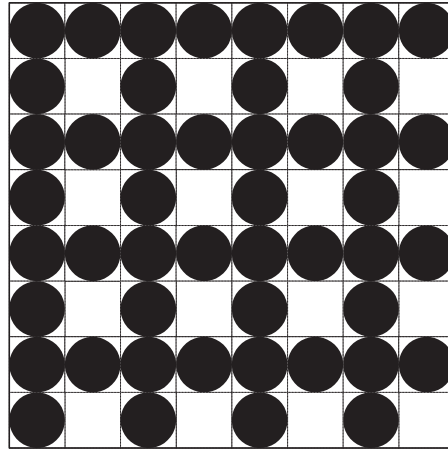


Figure 4-3: The subsample mask of the generic subsample rate 8-to-6.

be expressed as (4-4) and illustrated in Fig. 4-3. The dot circle means the pixel will be considered in the subsample matching step.

$$SM_{8:6} = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 \end{bmatrix} \quad (4-4)$$

Given a subsample mask, the computation cost of SSAD calculation can be lower than that of SAD calculation. Since the reduction of computation cost implies the saving of power consumption, the generic subsample algorithm allows the system power to scale with changing subsample rate. The higher the subsample rate, the more the number of inactive processing elements will be. Accordingly, the power consumption of the system is proportional to the inverse of the subsample rate. Due to its flexibility in energy-quality tradeoffs, the generic sub-

sample algorithm is suitable for implementing power-aware architectures. However, the algorithm suffers from aliasing problem for 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.

4.4 Content-Based Subsample Algorithm

As mentioned in the previous section, the generic subsample algorithm has aliasing problem for high subsample rate and leads to considerable quality degradation because the high frequency band is messed up. To alleviate the problem, this section presents the content-based subsample algorithm (CSA) which can overcome the drawback obviously. The procedure of the content-based subsample algorithm is described in Fig. 4-4. The CSA first performs gradient filter to extraction high-frequency pixels (or edge pixels) from a current macro-block and determines the edge pixels by these gradient value. Then the proposed algorithm subsamples the low-frequency pixels only and remains the high-frequency pixels to generate the content-based subsample mask. Finally, the CSA uses the content-based subsample mask to calculate the subsample SAD and employs the subsample SAD as matching criteria to figure out the motion vector. Detailed procedure of the content-based subsample algorithm is given in Fig. 4-5. We will describe the procedure in the following subsections.

4.4.1 Gradient Filter

This thesis uses three popular gradient filters to exercise the content-based algorithm; they are the high-pass gradient filter, the Sobel gradient filter, and the

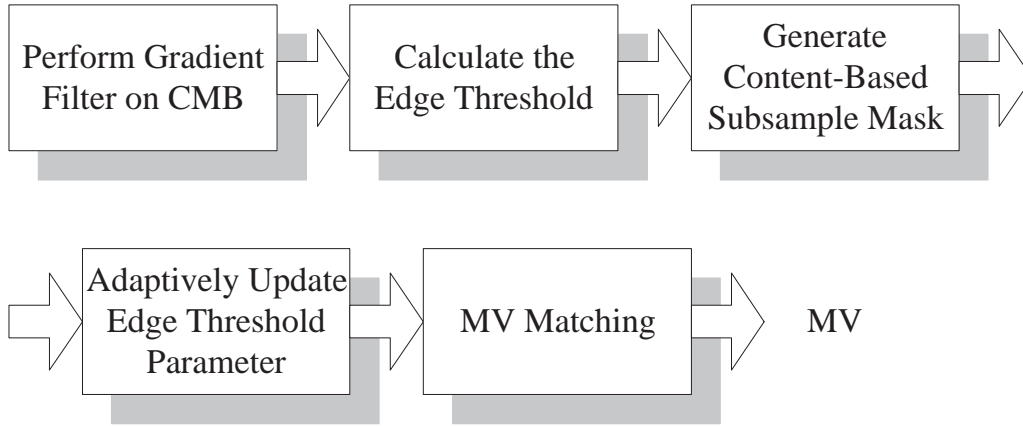


Figure 4-4: The flow chart of content-based subsample algorithm

morphological gradient filter [37]. Equations (4-5)-(4-7) list the calculations of these three gradient filters.

High-Pass Filter

$$G_{hpf}(i, j) = |MF(HPF_{mask}, R(i, j))|, \quad (4-5)$$

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

Sobel Filter

$$G_{sobel}(i, j) = |MF(SX_{mask}, R)(i, j)| + |MF(SY_{mask}, R)(i, j)|, \quad (4-6)$$

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

```

//frame:t
Input current and reference frames,  $W \times H$  ;
for(y = 0; y < W/N ; y ++){
  for(x = 0; x < H/N ; x ++){
    Perform gradient filtering;
    Calculate the edge threshold :
     $threshold = m_1^t(x, y) \cdot \max \{G(i, j)\} + (1 - m_1^t(x, y)) \cdot \min \{G(i, j)\}$ 
    Determine edge pixels and edge mask;
    Generate content - based subsample mask (CSM);
     $csm\_cnt = total\ edges\ of\ CSM;$ 
    //update threshold parameter for the next frame
     $m_1^{t+1}(x, y) = m_1^t(x, y) + K_p \cdot (csm\_cnt - trg\_cnt);$ 
    if ( $m_1^{t+1}(x, y) < 0$ ) { $m_1^{t+1}(x, y) = 0$ };
    if ( $m_1^{t+1}(x, y) > 1$ ) { $m_1^{t+1}(x, y) = 1$ };
    //find MV
     $SSAD_{min}(x, y) = \infty;$ 
    for(u = -p; u < p; u ++){
      for(v = -p; v < p; v ++){
         $SSAD(u, v) = \sum_{i=0}^{N-1} \sum_{j=0}^{N-1} |CSM(i, j) \cdot (S(i+u, j+v) - R(i, j))|;$ 
        if  $SSAD_{min}(x, y) > SSAD(u, v)$ 
          { $SSAD_{min}(x, y) = SSAD(u, v); MV(x, y) = (u, v);$  }
      }//for loop index v
    }//for loop index u
  }//for loop index x
}//for loop index y

```

Figure 4-5: The content-based subsample algorithm

Morphological Gradient Filter

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

$$\text{where } B = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \quad (4-7)$$

where the operations ' \oplus ' and ' \ominus ' denote the morphological dilation and erosion.

In equations (4-5) and (4-6), the $MF(\cdot)$ function is the mask filter operation as shown in (4-8).

$$MF(M, R)(i, j) = \sum_{p=-1}^1 \sum_{q=-1}^1 M(p+1, q+1) \cdot R(i+p, j+q) \quad (4-8)$$

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

4.4.2 Edge Determination

After obtaining the gradients G , instead of using a constant threshold, we use a floating threshold to determine the edge pixels of the current macro-block. The floating threshold makes the edge extraction more robust with video content varying than the constant threshold. Equation (4-9) describes the calculation of the floating threshold.

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

$$\text{for } 0 \leq m_1^t \leq 1, \quad (4-9)$$

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 (4-10).

$$EdgeMask(i, j) = \begin{cases} 1, & \text{for } G(i, j) \geq threshold \\ 0, & \text{otherwise} \end{cases} \quad (4-10)$$

Finally, the contend-based subsample mask (CSM) is generated by merging the edge mask and the subsample mask, as shown in (4-11). According to the calculation of CSM, the subsample rate in CSA (CSR), denoted as R_s , is N^2 -to- esm_cnt , where esm_cnt is the number of 1's in CSM and N^2 is the macro-block size. Figure 4-6 demonstrates an example of CSM in which the subsample rate is 64-to-27.

$$CSM(i, j) = SM_{8:m}(i, j) \vee EdgeMask(i, j), \quad (4-11)$$

$$0 \leq i, j \leq N - 1.$$

4.4.3 Adaptive Control Mechanism

In the previous subsection, the content-based subsample mask is generated by merging the regular generic subsample mask with edge mask of high frequency pixels. Since the edge pixels is determined by the edge threshold parameter, the lower the edge threshold parameter, the more the edge pixels will be took. In the prior work, we used constant edge threshold parameter to determine the edge mask [39]. However, the same edge threshold parameter turns out diverse number of edge pixels for various video clips. In Table 4.I., we analyzed the effect of edge threshold parameter on the subsample pixels by setting the edge threshold parameter from 0.05, 0.1, 0.2, 0.3, 0.4, 0.5 to 0.6. As the simulation results shown in the table, standard deviation is up to 13.846 between these 21 test clips for the same edge threshold parameter. Since the subsample rate (N^2 -to-subsample

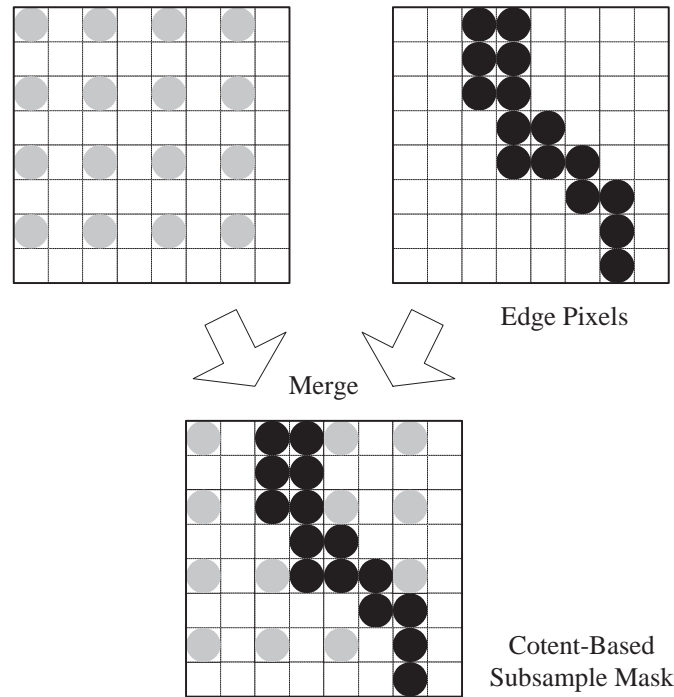


Figure 4-6: The components of a content-based subsample mask (CSM)

pixels) is related to how many corresponding processing elements will be disabled for saving power consumption, the diverse property of edge threshold parameter causes obstacle to implement accurate power-aware architecture.

In order to conquer the obstacle, the content-based subsample algorithm proposes an adaptive mechanism to self-optimize the edge threshold parameter to converge the subsample pixels. Figure 4-7 shows the block diagram of the adaptive control mechanism. As described in the previous section, the host processor receives the status of battery pack and decides the best target subsample pixels (trg_cnt) according to the battery profile. By the target subsample pixels, the adaptive mechanism recursively updates the edge threshold parameter $m_1^{t+1}(x, y)$ according to the difference between target subsample pixels and current subsam-

Table 4.I.: Analyze the effect of edge threshold parameter m_1 on subsample pixels.

m_1	0.05	0.1	0.2	0.3	0.4	0.5	0.6
akiyo	205.414	176.522	136.769	110.035	92.805	82.260	74.973
children	178.668	149.623	119.201	101.613	89.612	81.177	74.757
coastguard	226.269	200.438	157.329	125.502	103.123	87.978	77.955
container	211.320	183.773	145.082	118.056	99.330	86.446	77.758
dancer	213.506	191.270	153.491	123.626	102.685	88.545	77.913
destruct	204.833	170.554	128.369	103.568	88.558	79.085	72.849
flower	204.992	177.966	140.167	113.887	95.937	84.277	75.844
foreman	190.506	153.741	116.262	96.951	85.588	78.253	73.124
hall monitor	201.773	172.598	136.236	112.599	96.687	85.678	77.743
mobile	184.738	158.141	124.977	104.341	90.300	80.638	74.081
mother daughter	208.210	178.797	139.426	113.509	96.484	85.003	76.194
news	197.857	168.008	130.289	107.245	92.678	82.839	75.606
paris	187.107	157.579	124.165	104.327	91.431	82.612	76.093
sean	203.642	173.333	135.699	110.464	94.222	83.595	75.872
silent	203.741	173.749	132.628	106.703	90.764	80.832	74.483
singer	207.794	179.213	141.512	116.214	98.662	86.565	77.972
stefan	203.708	177.468	141.179	115.379	97.696	85.556	77.092
table tennis	215.165	184.407	138.681	109.122	91.020	80.171	73.374
tempete	197.329	165.714	128.063	104.663	89.659	79.982	73.699
waterfall	221.736	191.497	144.527	113.015	93.324	81.379	74.054
weather	169.356	149.140	125.092	108.309	97.317	87.226	80.078
Average	201.794	173.025	135.197	110.435	94.189	83.338	75.786
σ	13.749	13.846	10.522	7.052	4.646	3.059	1.967

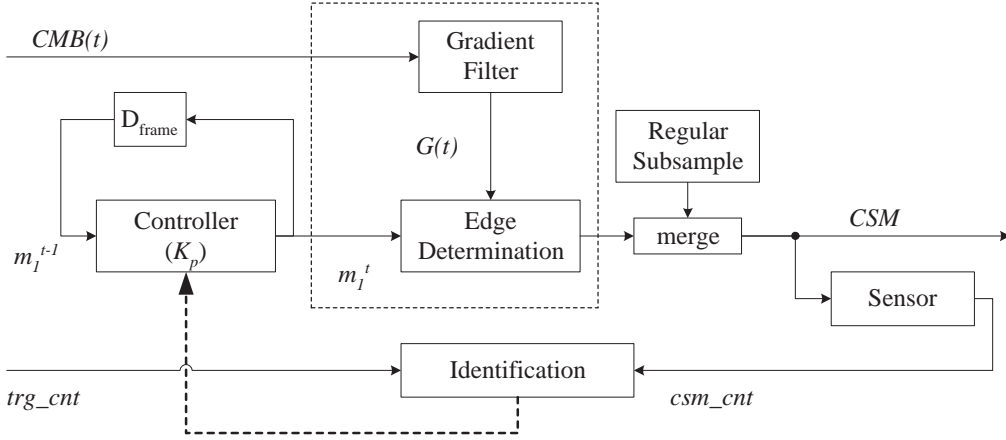


Figure 4-7: The block diagram of the edge-determination unit with adaptive control mechanism.

ple pixels (csm_cnt) frame by frame, as shown in (4-12).

$$\begin{aligned}
 m_1^{t+1}(x, y) &= m_1^t(x, y) + K_p \cdot (csm_cnt - trg_cnt); \\
 if(m_1^{t+1}(x, y) < 0) \{m_1^{t+1}(x, y) &= 0\}; \\
 if(m_1^{t+1}(x, y) > 1) \{m_1^{t+1}(x, y) &= 1\};
 \end{aligned} \tag{4-12}$$

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. Figure 4-7 illustrates the block diagram of the the edge-determination unit with the proposed adaptive control mechanism.

As shown in (4-12), the control parameter K_p that will affect the settling time and stationary state error of subsample rate. If the control parameter is not well-selected, the settling time would be too long to have real-time switching and the CSR error would be so large to make the setting of power consumption mode inaccurate and the power-awareness worse. The control parameter K_p in Fig. 4-7 is the major factor to affect the settling time and the CSR error. In order to analyzing the effect of K_p upon the subsample rate and stationary error, we simulated 21 test

Table 4.II.: Average stationary error for 21 video clips with $K_p = 0.2$.

Target	96.000	128.000	160.000	192.000	224.000
HPF	95.913	127.827	159.771	191.359	221.731
Error	0.091%	0.135%	0.143%	0.334%	1.013%
SBL	96.026	127.818	159.712	191.261	221.659
Error	0.027%	0.142%	0.180%	0.385%	1.045%
MPH	96.553	128.015	159.734	191.370	222.179
Error	0.576%	0.012%	0.166%	0.328%	0.813%

clips for 30 frames with 1 : 1 of the initial subsample rate and 8 : 5 of the target subsample rate. Figure 4-8 shows the effect of the K_p selections of four clips as illustrations.

Obviously, the higher the value of K_p , the shorter the settling time and the worse the stability of the CSR are. If the K_p is too large, the real subsample pixels will be overshoot and oscillatory around the target subsample pixels. Although the settling time is under three frames, the overshoot is up to 32 pixels worse in the weather clip and hard to converge. On the other hand, the settling time will be longer than twenty frames while the K_p is set too tiny. As shown in the response plot of various K_p , the suitable range of K_p is from 0.1 to 0.3. In this range, all the test clips can be converge to the target subsample pixels under ten frames and less overshooting. That is, for a real-live video clip with 30 frames each second, the system can be operated on the target power mode under 0.33 second. Table 4.II. presents the average subsample pixels and station error of three filters with the proposed adaptive mechanism. The simulation results clearly show that the stationary error can be kept less than 1.045%.

Finally, Table 4.III. shows the simulation results by the proposed adaptive control mechanism. In order to comparing with the fixed edge threshold parameter shown in Table 4.I., we set the target subsample pixels as the average subsam-

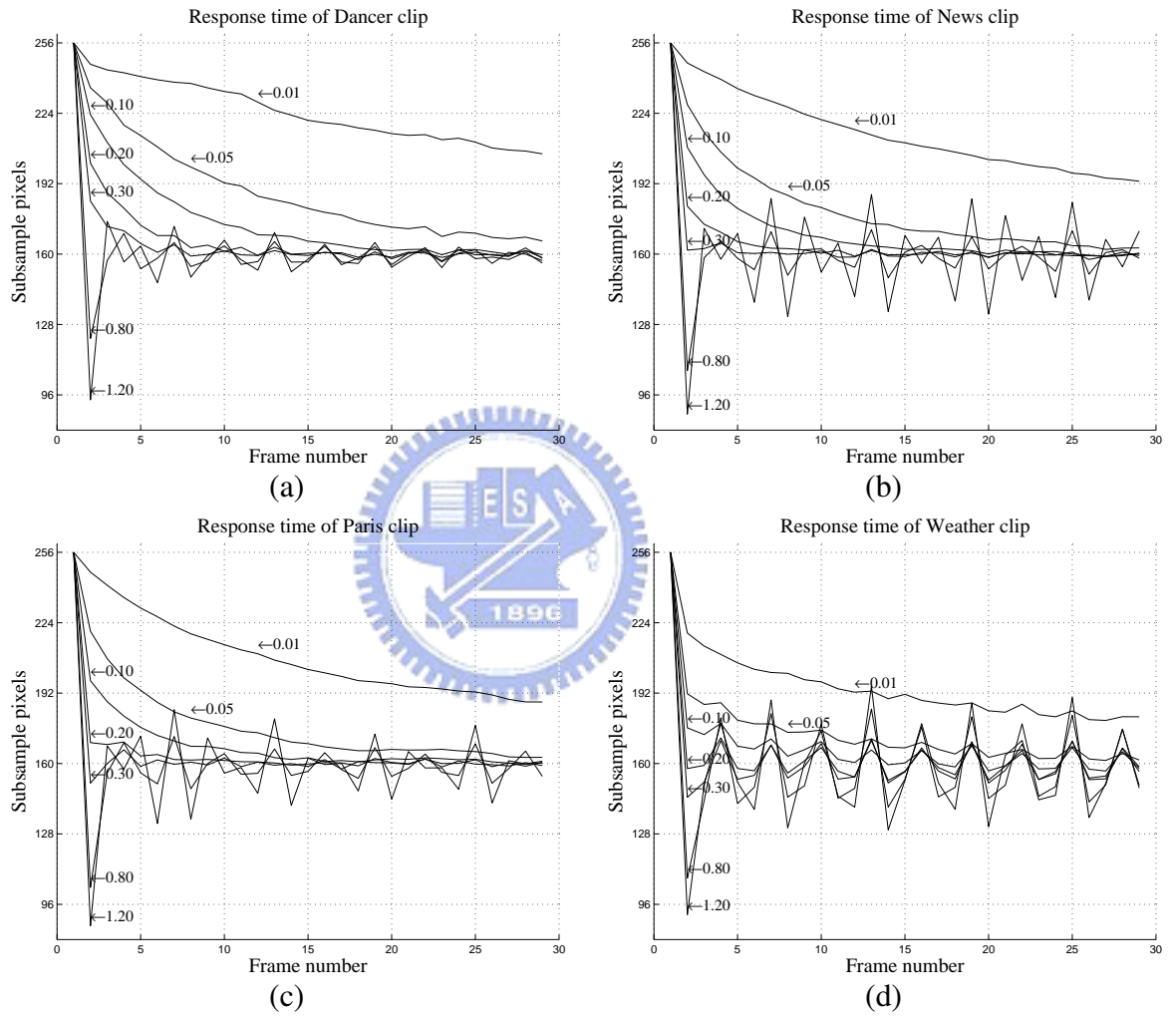


Figure 4-8: Response time of four clips. (a) The Dancer Clip. (b) The News Clip. (c) The Paris Clip. (d) The Weather Clip.

Table 4.III.: Analyze the effect of controlled edge threshold parameter on subsample pixels.

Target	201.794	173.025	135.197	110.435	94.189	83.338	75.786
akiyo	201.430	172.929	135.121	110.373	94.155	83.813	76.256
children2	200.458	172.210	135.142	111.296	95.576	85.236	78.055
coastguard	201.198	172.537	134.989	110.477	94.563	84.081	76.705
container	200.960	172.690	135.206	110.601	94.411	84.011	76.518
dancer	200.848	172.613	135.572	111.167	95.083	84.667	77.370
destruct	200.837	172.315	134.867	110.587	94.899	84.504	77.361
flower	201.033	173.026	135.515	110.816	94.898	84.843	78.578
foreman	201.463	172.843	134.975	110.415	94.773	84.639	77.552
hall monitor	201.043	172.787	135.279	110.733	94.602	84.321	76.948
mobile	200.948	172.907	135.129	110.859	95.157	84.401	76.925
mother daughter	200.324	172.884	135.071	110.562	94.665	84.139	76.720
news	201.223	172.951	135.313	110.496	94.291	83.890	76.603
paris	200.924	172.855	135.240	110.786	94.741	84.260	76.926
sean	200.409	171.941	134.242	109.320	93.614	83.268	76.264
silent	199.972	172.756	135.039	110.336	94.025	83.898	76.366
singer	201.256	172.798	135.075	110.273	94.159	83.605	76.281
stefan	201.532	173.145	135.469	110.790	94.985	84.371	77.060
table tennis	200.460	171.941	134.546	110.115	94.315	83.886	76.715
tempete	201.527	173.155	135.454	110.875	94.894	84.144	76.853
waterfall	201.794	173.080	135.254	110.526	94.368	83.790	76.459
weather	198.632	172.282	136.317	113.543	98.618	88.744	81.883
Average	200.870	172.697	135.182	110.712	94.800	84.405	77.162
σ	0.690	0.365	0.404	0.763	0.981	1.089	1.234

ple pixels of these 21 test clips with the fixed edge threshold parameter. As the results showed, the standard deviation can be reduced to 0.365 and the worse is only 1.234, which is much better than the result 13.846 with uncontrolled edge threshold parameter.

4.4.4 Matching Step

Once the CSM is generated, the algorithm can then determine the motion vector (MV) with the subsample sum of absolute difference (SSAD) criterion. The

SSAD criterion is similar to that mentioned in Section 4.3 and shown in (4-13).

$$SSAD(u, v) = \sum_{i=0}^{N-1} \sum_{j=0}^{N-1} |CSM(i, j) \cdot [R(i+u, j+v) - S(i, j)]|, \quad (4-13)$$

for $-p \leq u, v \leq p-1$. In this equation, $S(i, j)$, $R(i+u, j+v)$, N and p are defined the same as (4-1). As the SSAD of each reference macro-block is calculated, the motion vector can be selected with the minimum error distortion by

$$\overrightarrow{MV} = (u, v) \Big|_{\min_{-p \leq u, v \leq p-1} SSAD(u, v)} \quad (4-14)$$

4.5 Results

In this thesis, we simulate 21 352-by-288 MPEG clips with parameters $N = 16$ and $p = 16$ for 50 frames to analyze the quality performance of motion compensation. The control parameter K_p in the adaptive control mechanism is set as 0.2 as we have presented the suitable K_p is ranged from 0.1 to 0.3 in section 4.3. The target subsample rates are set as (4 : 1), (8 : 3), (2 : 1), (8 : 5), (4 : 3), (8 : 7), and (1 : 1); that is, the target subsample pixel counts are 64, 96, 128, 160, 192, 224 and 256, respectively.

Table 4.IV. and 4.V. are the quality results of GSA and ACSA with high-pass filter respectively. Figure 4-9 shows the quality degradation curve of four video clips, the Dancer clip, the Hall Monitor clip, the News clip and the Paris clip. In order to illustrate more robust performance, we showed the average quality degradation curve in Fig 4-10. The dashed line in the figure is the result of the generic subsample algorithm and the solid lines are the results of the content-based subsample algorithm with three filters. As shown in the results, the quality

Table 4.IV.: Quality performance of PSNR by GSA for different video clips.

Target R_s	8:2	8:3	8:4	8:5	8:6	8:7	8:8
akiyo	42.874	42.884	42.908	42.920	42.928	42.927	42.922
children	26.743	26.724	26.813	26.809	26.848	26.858	26.870
coastguard	30.567	30.594	30.623	30.624	30.634	30.640	30.652
container	38.373	38.383	38.396	38.401	38.400	38.402	38.405
dancer	32.166	32.209	32.389	32.462	32.657	32.676	32.714
destruct	28.728	28.749	28.822	28.857	28.919	28.921	28.937
flower	26.186	26.267	26.311	26.328	26.341	26.347	26.353
foreman	33.237	33.777	33.880	33.863	33.862	33.915	33.938
hall monitor	34.370	34.397	34.462	34.502	34.564	34.571	34.573
mobile	23.641	23.732	23.814	23.919	23.959	23.971	23.988
mother daughter	41.589	41.607	41.662	41.679	41.714	41.719	41.735
news	38.415	38.410	38.506	38.536	38.596	38.592	38.597
paris	30.508	30.524	30.587	30.608	30.659	30.668	30.676
sean	38.538	38.566	38.577	38.590	38.623	38.619	38.636
silent	36.452	36.759	36.780	36.793	36.800	36.796	36.811
singer	35.215	35.281	35.317	35.334	35.368	35.385	35.411
stefan	26.204	26.234	26.321	26.336	26.350	26.356	26.365
table tennis	30.937	30.958	31.013	31.026	31.053	31.054	31.066
tempeste	27.295	27.322	27.349	27.360	27.381	27.379	27.387
waterfall	35.430	35.463	35.474	35.476	35.478	35.481	35.481
weather	36.701	36.725	36.756	36.792	36.845	36.836	36.840
Average	33.056	33.122	33.179	33.201	33.237	33.244	33.255

Unit: dB

degradation of the content-based algorithm is less than that of the generic sub-sample algorithm, and the type of filter does not make much difference to the performance of the proposed algorithm. Figure 4-11 and 4-12 illustrate the improvement of CSA over GSA. Because the CSA alleviates the aliasing problem for high-frequency band, the compensated frame with CSA has better quality than that with GSA on the edge of the test sequences. Besides the adaptive control mechanism does work quite well to keep the subsample rate stationary.

Table 4.V.: Quality performance of PSNR by ACSA with high pass filter for different video clips.

Target R_s	8:2	8:3	8:4	8:5	8:6	8:7	8:8
akiyo	42.874	42.967	42.986	42.962	42.959	42.937	42.922
children	26.743	26.840	26.821	26.838	26.847	26.869	26.870
coastguard	30.567	30.625	30.647	30.643	30.648	30.650	30.652
container	38.373	38.567	38.568	38.569	38.569	38.569	38.405
dancer	32.166	32.630	32.717	32.735	32.724	32.750	32.714
destruct	28.728	28.815	28.865	28.921	28.944	28.948	28.937
flower	26.186	26.284	26.305	26.328	26.348	26.350	26.353
foreman	33.237	33.924	33.951	33.966	33.963	33.951	33.938
hall monitor	34.370	34.521	34.559	34.572	34.581	34.578	34.573
mobile	23.641	23.912	23.946	23.964	23.974	23.981	23.988
mother daughter	41.589	41.700	41.722	41.732	41.733	41.735	41.735
news	38.415	38.621	38.656	38.666	38.632	38.632	38.597
paris	30.508	30.649	30.670	30.694	30.690	30.691	30.676
sean	38.538	38.626	38.636	38.640	38.643	38.639	38.636
silent	36.452	36.762	36.757	36.785	36.791	36.802	36.811
singer	35.215	35.586	35.560	35.510	35.453	35.438	35.411
stefan	26.204	26.285	26.327	26.347	26.362	26.363	26.365
table tennis	30.937	31.006	31.041	31.040	31.053	31.061	31.066
tempete	27.295	27.307	27.318	27.340	27.366	27.381	27.387
waterfall	35.430	35.484	35.485	35.485	35.484	35.484	35.481
weather	36.701	36.833	36.863	36.878	36.872	36.852	36.840
Average	33.056	33.235	33.257	33.267	33.268	33.270	33.255

Unit: dB

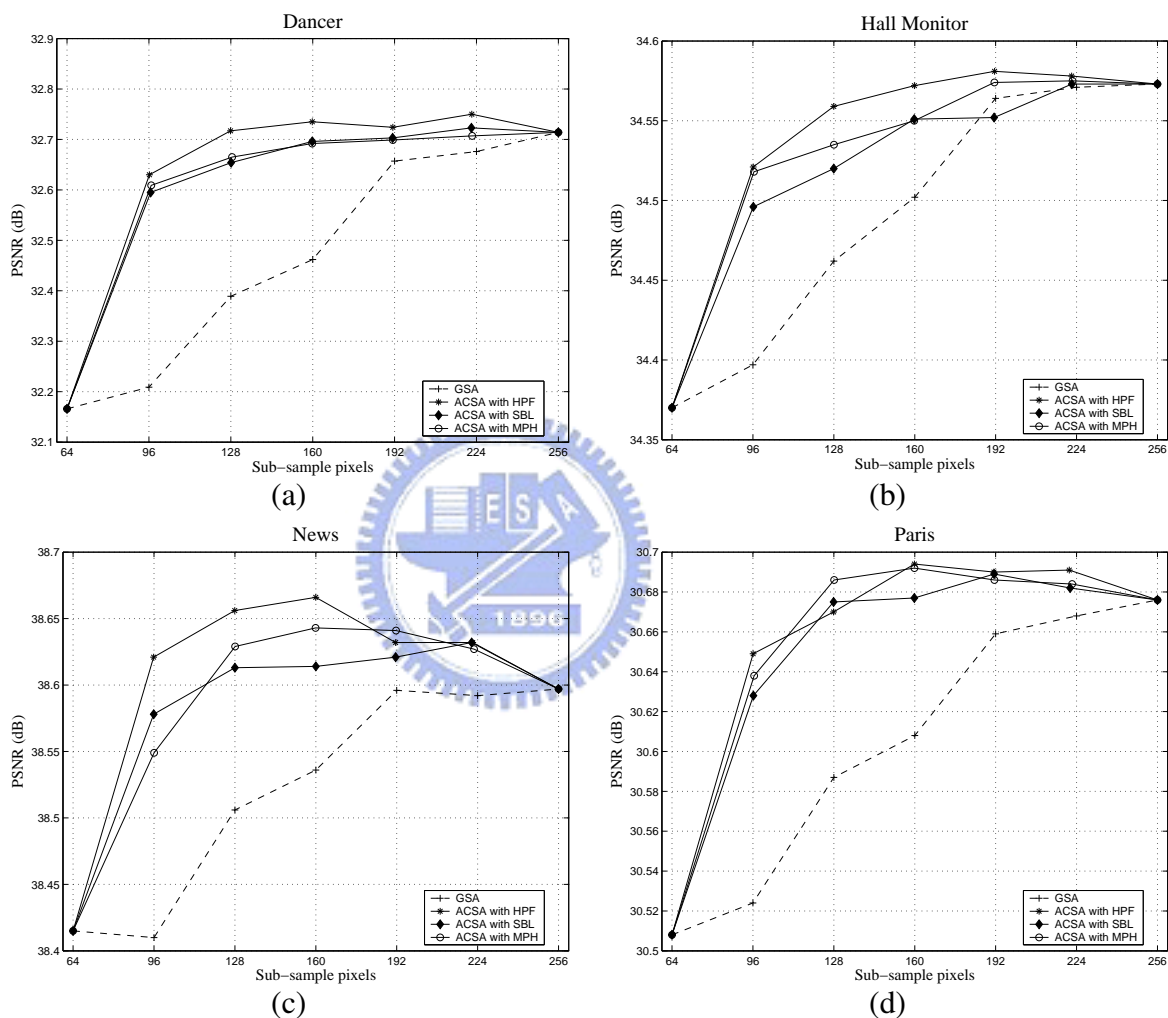


Figure 4-9: Quality degradation curves of four clips. (a) The Dancer Clip. (b) The Hall Monitor Clip. (c) The News Clip. (d) The Paris Clip.

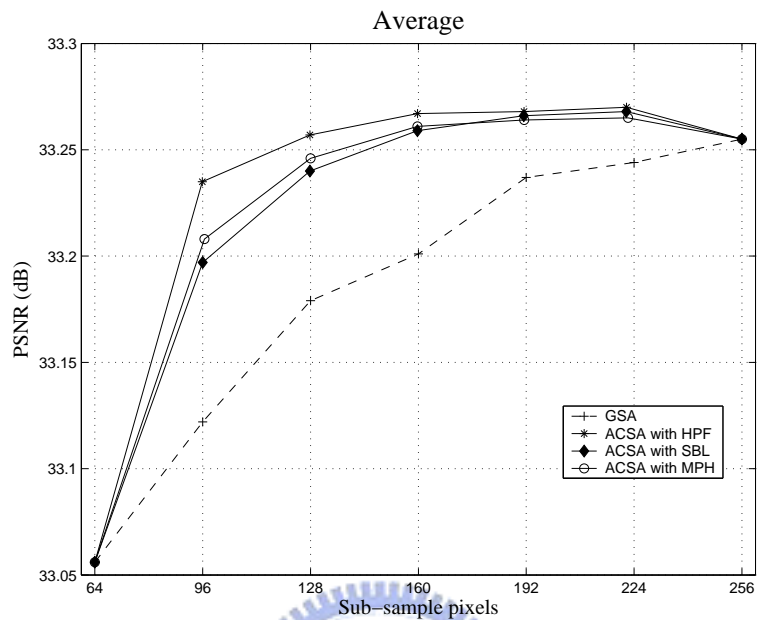


Figure 4-10: Average quality degradation curve of 21 test clips.

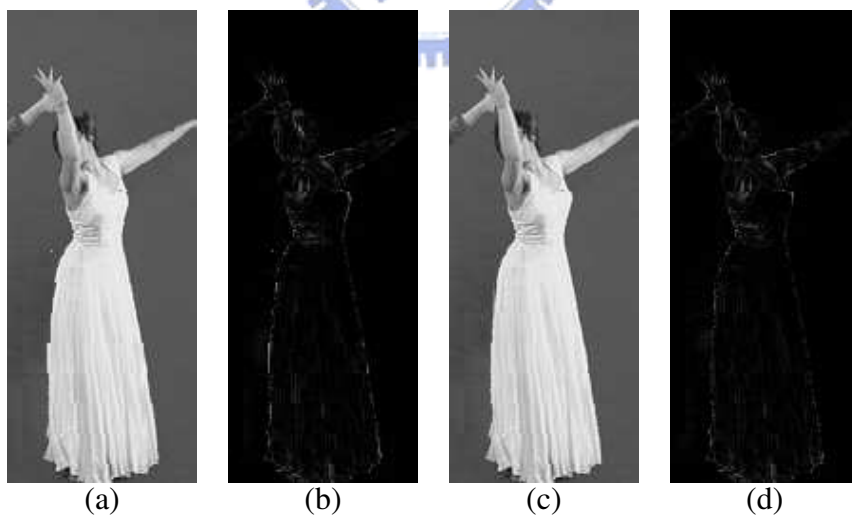


Figure 4-11: The 26th frame of dancer clip. (a) GSA with 8-to-3 subsample rate (b) Residual of motion compensation by GSA (c) CSA with 8-to-3 subsample rate (d) Residual of motion compensation by CSA

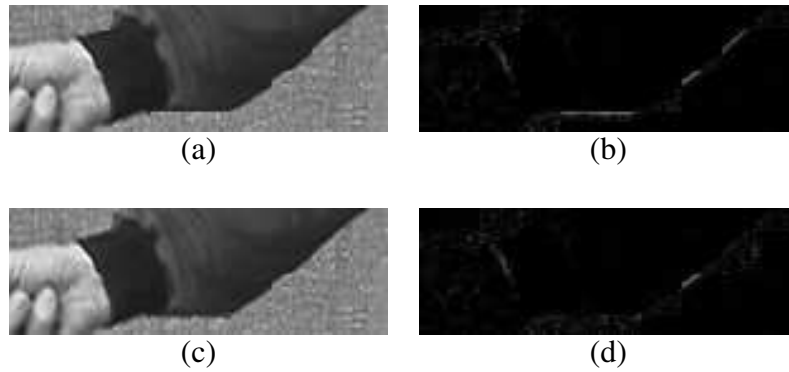


Figure 4-12: The 18th frame of table-tennis clip. (a) GSA with 8-to-3 subsample rate (b) Residual of motion compensation by GSA (c) CSA with 8-to-3 subsample rate (d) Residual of motion compensation by CSA

4.6 Power Aware Architecture

According to the content-based subsample algorithm, we presented a semi-systolic architecture as shown in Fig. 4-13, based on existed 2D array architectures, such as [11][13]. The proposed architecture contains an edge-extraction unit (EXU), an array of processing elements (PEs), a parallel adder tree (PAT), a RMB Buffer (RMBBUF), 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 PEs array is used to accumulate absolute pixel differences column by column while the parallel adder tree sum up all the results to generate the value of SSAD. The MVS, then, performs compare-and-select operation to select the best motion vector.

In the following subsections, we will illustrate these blocks in detail and the system execution schedule at the last subsection.

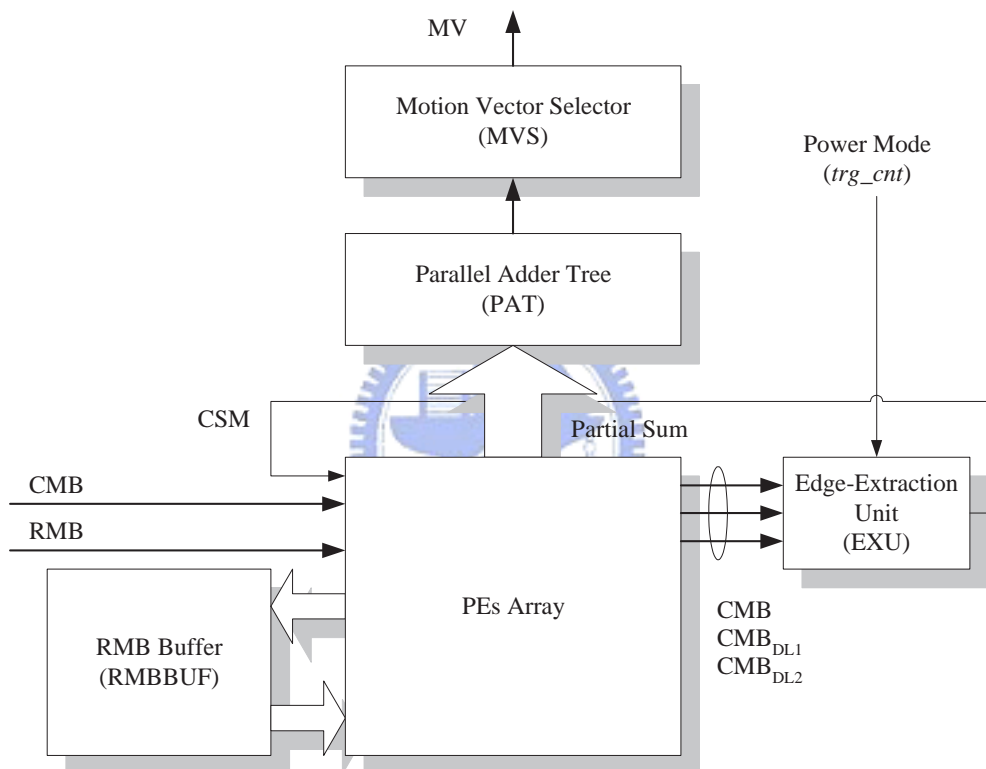


Figure 4-13: The block diagram of the power-aware ME architecture driven by the content-based subsample algorithm.

4.6.1 Architecture

Processing Element Array and RMB Buffer

The processing elements array, which block diagram is shown in the right side of Fig. 4-14, is used to accumulate absolute pixel differences column by column. It is consisted of N -by- N processing element (PE) and each processing computes the absolute difference between luminance values of a pixel in the current macro-block and a pixel in the reference macro-block if the PE is not disabling by the subsample mask. Then the PE adds the absolute difference with the partial sum from the previous PE in the below row and forwards the updated partial sum to the next PE in the above row. Finally, the PEs array accumulates the partial sums of absolute difference in each column and sends these partial sum to the parallel adder tree to sum up the final sum of absolute difference.

The reference macro-block buffer, shown in the left part of Fig. 4-14, contains $(N - 1)$ -by- $(2p - 2)$ registers to save the pixel value of the reference macro-block temporarily and it guarantees that each pixel is accessed only one time from the frame memory for a macro-block matching iteration. Although this buffer can save the frame memory bandwidth, there are still overlaps between the searching area in the reference frame of successive macro-block which causes the multi-accessing in a reference pixel. Each pixel in the reference frame is accessed for $(N + 2p - 1)^2 / N^2$ times.

In the proposed architecture, there are three specific paths in the PEs array to provide the luminance value of the current macro-block to the edge-extraction unit. Those are one bypass path, marked as CMB, and two a line-delayed paths, marked as CMB_{DL1} and CMB_{DL2}. Since each filter operation needs the previous and next lines to perform the mask operation, these paths can save the delay line

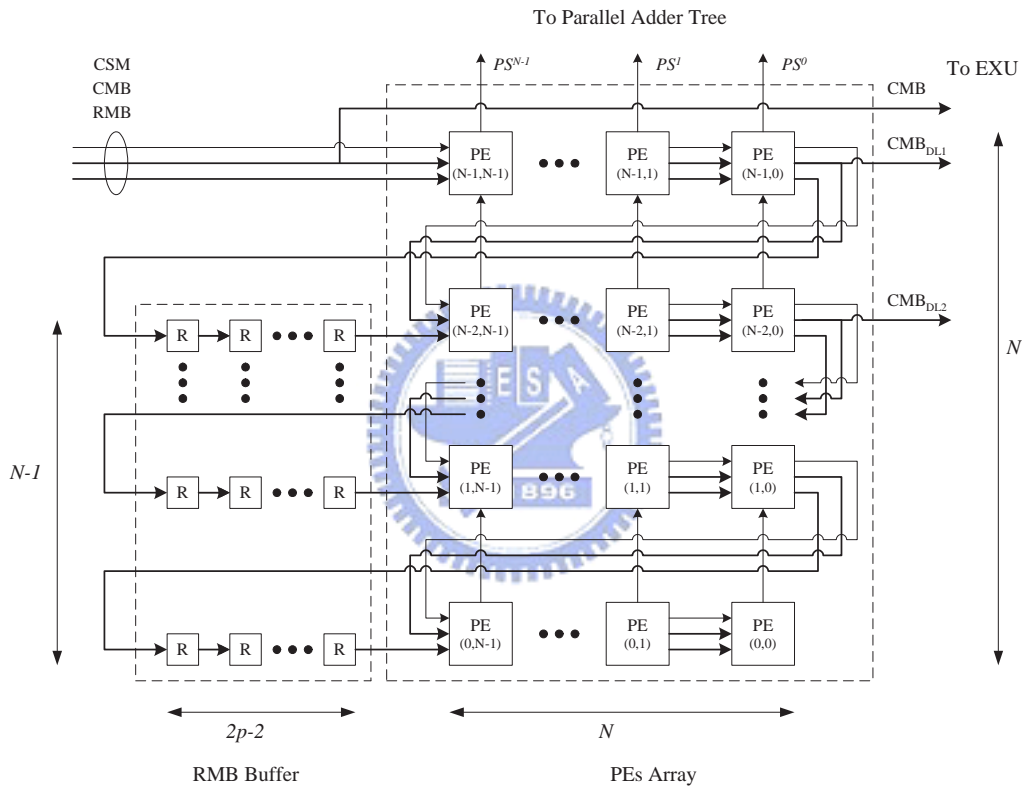


Figure 4-14: The architecture of PEs array and RMB buffer.

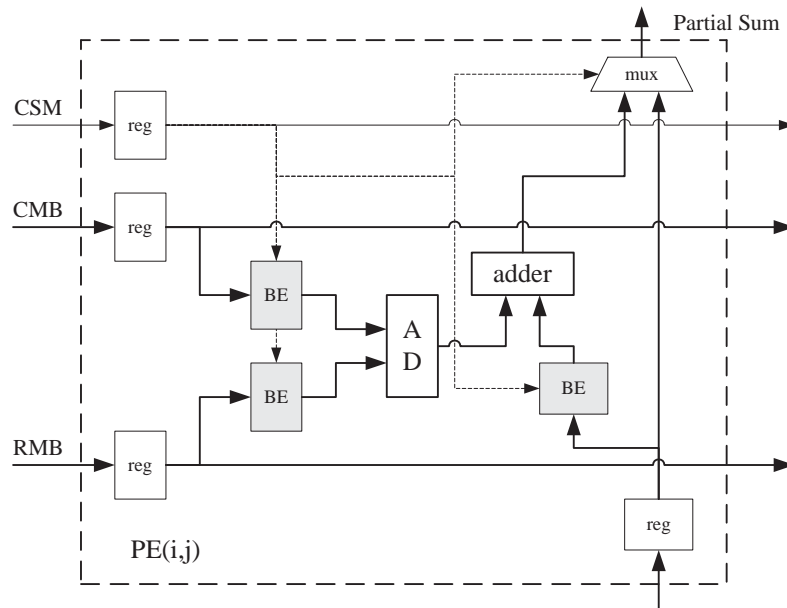


Figure 4-15: The structure of a PE.

the edge-extraction unit.

Figure 4-15 shows the PE structure and explains how the CSM disables/enables processing elements. The CSM disables the PE by using the block element (BE) marked as gray block. The BEs implemented by AND gates can nullify the input signals of data path that consists of the absolute difference unit ($|a - b|$) and the adder unit. When a PE are disabled during a MV searching iteration, the circuits in the PE remain still until the next iteration starts and, thus, the consumption of transient power can be saved.

Based on the systolic architecture with content-based subsample algorithm, the architecture dynamically disable some processing elements to reduce the power consumption in that we assume the major power consumption is determined by the switch activity of system [13]. The enable/disable mask is generated from the edge-extraction unit which will be depicted in the next subsection.

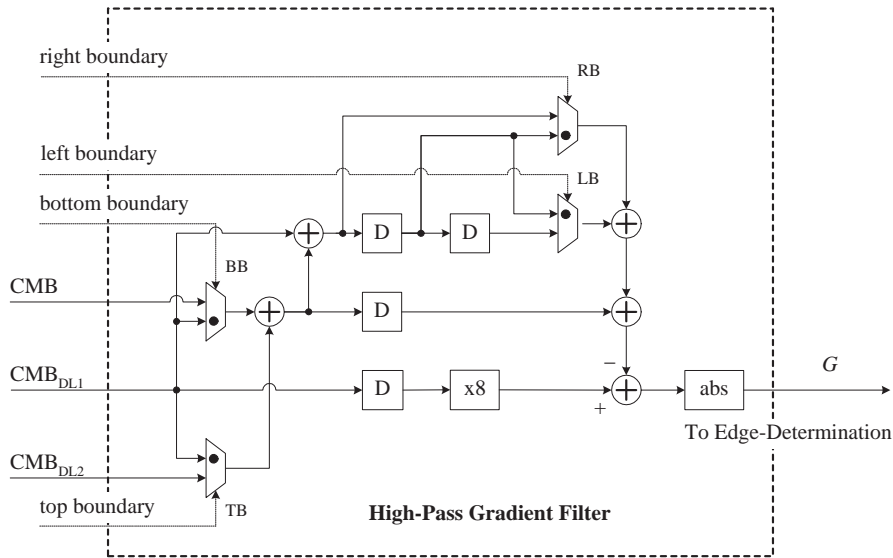


Figure 4-16: The architecture of high-pass filter.

Edge Extraction Unit

The edge-extraction unit contains two blocks, the gradient filter and the CSM generator. The implementation of gradient filter is based on one of (4-5) to (4-7). The proposed architecture only needs a single gradient filter embedded. However, we still show all of their implementations for the purposes of estimating the overheads and comparison. Figures 4-16 to 4-18, illustrate the implementations of high-pass filter, Sobel filter and morphological gradient filter respectively. The multiplexers are used to prevent from boundary errors for border pixels in performing gradient filter. The black-dot in each multiplexer indicates the switching path when processing a border pixel. Figure 4-19 shows the structure of CSM generator. The CSM generator has two steps. It 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 (4-9)-(4-11).

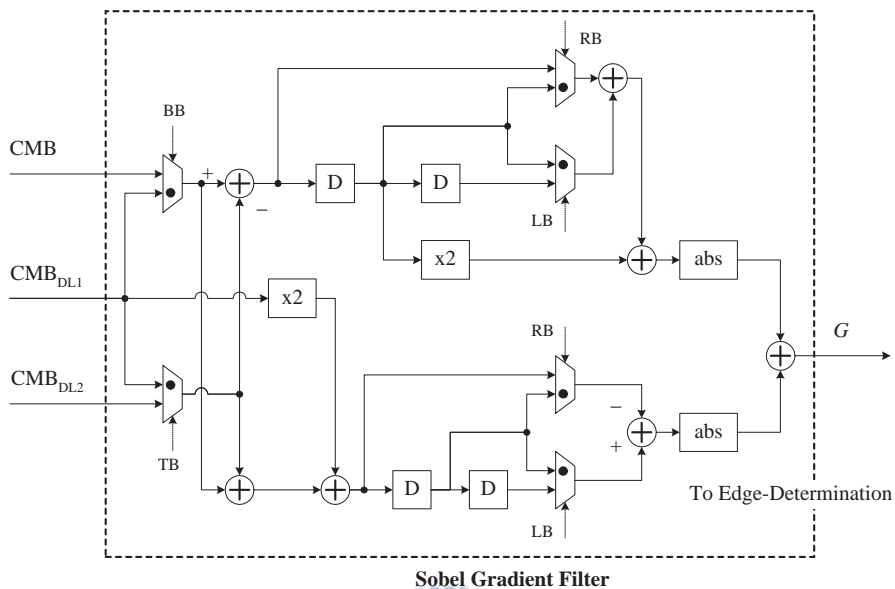


Figure 4-17: The architecture of Sobel filter.

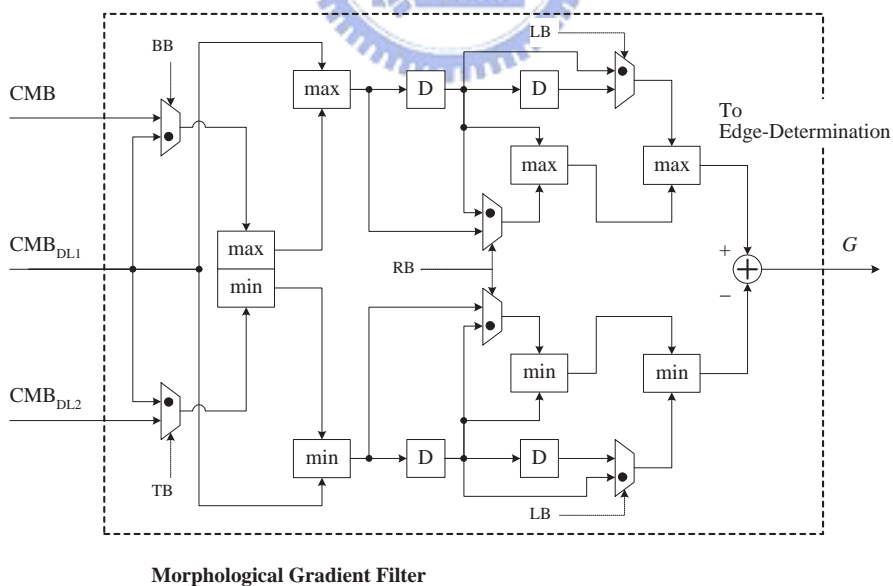


Figure 4-18: The architecture of Morphological gradient filter.

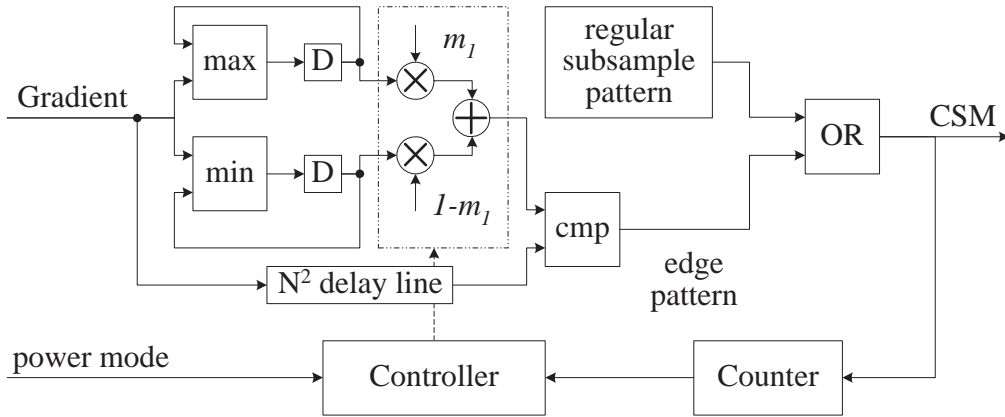


Figure 4-19: The structure of CSM generator.

Execution of Power-Aware ME

The execution of power-aware motion estimation has five phases: initial CMB phase, initial RMB phase, SAD calculation phase, filtering phase, and edge determination phase. The initial CMB phase is for loading the CMB data into PEs array while the initial RMB phase is for filling up PEs array in full with RMB data to start the SAD calculation. As shown in Fig. 4-20, the initial CMB phase and initial RMB phase are executed in parallel with the edge extraction and thus the timing overhead of edge extraction is hidden by the initial phases. For $p > 8$, the timing overhead of edge extraction is zero.

4.6.2 Implementation Results

The system architecture is implemented using Verilog HDL and synthesized with 0.35 μm cell library of TSMCTM process technology by Design Compiler[®] of Synopsys, Inc. Table 4.VI. reports the synthesized results of the power-aware architecture. In the results, we show the gate count by blocks, which are PEs Array, SRA, PAT+MVS, and EXU. In this way, we will use the synthesized gate count

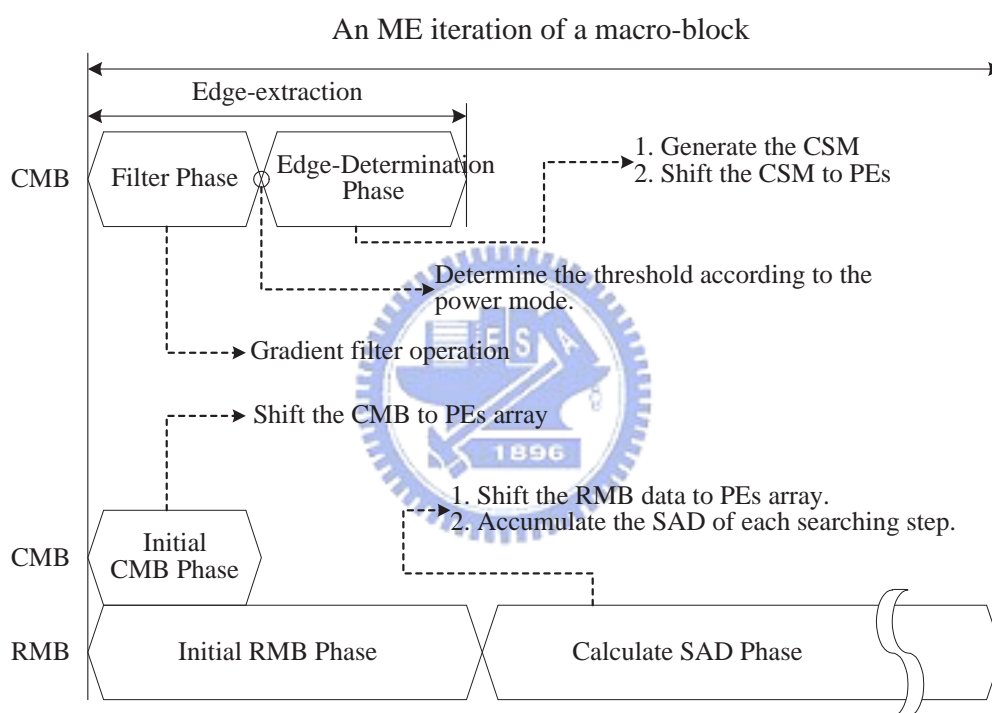


Figure 4-20: The execution phases of the power-aware architecture.

Table 4.VI.: Implementation of the power-aware architecture.

	EU_i	G^i
PEs array	AD + Adder	117,760
	Others	58,708
	SRA	44,640
	PAT+MVS	1,800
	EXU	15,393

Unit: gate count
Cell library: TSMC™ 0.35um 1P4M process

to analyze the power consumption which will be illustrated in the next section.

4.7 Performance Analysis

4.7.1 Power Model

Before depict the simulation results, this subsection presents the power model which will be used to analyze the performance in the thesis. In a VLSI system designed in CMOS technology, one can consider the major power consumption of a CMOS gate i as (4-15), where C_i is the output capacitance, f_i is the operation frequency, and $r_i(0 \leftrightarrow 1)$ is the switch activity of gate i . α and κ are constants.

$$P_{gate_i} = \alpha \cdot C_i \cdot f_i \cdot V_{DD}^2 = \kappa \cdot C_i \cdot r_i(0 \leftrightarrow 1). \quad (4-15)$$

For an execution unit EU_j in such VLSI system, the power consumption can be shown in (4-16), where G^j is the gate count of EU_j .

$$P_{EU_j} = \sum_{i=1}^{G^j} \kappa \cdot C_i^j \cdot r_i^j(0 \leftrightarrow 1). \quad (4-16)$$

After considering the activity of execution units, the total power consumption can be expressed as (4-17) and approximated as (4-19) by assuming 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 total system (C_{avg}), the total power consumption can be approximated to (4-22). Therefore, we can obtain an approximated power estimation model shown in (4-23), 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.

$$P_{total} = \sum_{\forall \text{inactive} EU_j} P_{EU_j} + \sum_{\forall \text{active} EU_k} P_{EU_k} \quad (4-17)$$

$$= \sum_{\forall \text{inactive} EU_j} \kappa \sum_{i=1}^{G^j} C_i^j \cdot 0 + \sum_{\forall \text{active} EU_k} \kappa \sum_{i=1}^{G^k} C_i^k \cdot r_i^k(0 \leftrightarrow 1) \quad (4-18)$$

$$\cong \kappa \sum_{\forall \text{active} EU_k} r^k(0 \leftrightarrow 1) \sum_{i=1}^{G^k} C_i^k \quad (4-19)$$

$$= \kappa \sum_{\forall \text{active} EU_k} r^k(0 \leftrightarrow 1) \times \frac{\sum_{i=1}^{G^k} C_i^k}{G^k} \times G^k \quad (4-20)$$

$$= \kappa \sum_{\forall \text{active} EU_k} r^k(0 \leftrightarrow 1) \times C_{avg}^k \times G^k \quad (4-21)$$

$$\cong (\kappa \cdot C_{avg}) \sum_{\forall \text{active} EU_k} r^k(0 \leftrightarrow 1) \times G^k \quad (4-22)$$

$$= \varepsilon_{gp} \sum_{\forall \text{active} EU_k} r^k(0 \leftrightarrow 1) \times G^k \quad (4-23)$$

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

EU_i	PE array		SRA	PAT+MVS	EXU
	AD + Adder	Others			
Gate Count G^i	117,760	58,708	44,640	1,800	15,393
$r^i(0 \leftrightarrow 1)$	$4p^2 R_s^{-1}$	$4p^2$	$4p^2$	$4p^2$	N^2
P^i	$1.21e8 \cdot R_s^{-1}$	$6.01e7$	$4.57e7$	$1.84e6$	$3.94e6$
$P_{total}(\varepsilon_{gp})$	$1.21e8 \cdot R_s^{-1} + 1.12e8$				

$N = 16$ and $p = 16$

Cell library: TSMC 0.35um process

4.7.2 Results

Table 4.VII. shows the synthesis result using the TSMC 1P4M 0.35um cell library, where the symbol R_s means the content-based subsample rate and the ε_{gp} is the gate power coefficient defined in (4-23). Comparing with the general semi-systolic architecture [11], the edge extraction unit (EXU) of proposed architecture is the major overhead for power-aware function. As mentioned above, this paper uses one of three gradient filters to implement the EXU. As per the synthesis results, the gate counts of the three gradient filters are 499.33, 697.77 and 631.63 respectively. The variance of these values is very little to the overall gate count of EXU. For instance, the gate count of EXU with high-pass filter is equal to 15393. The number is extremely larger than the variance. It means that the selection of gradient filter does not affect the overhead estimation much. Therefore, we selectively use high pass filter to estimate the performance overhead caused by EXU. From Table 4.VII., we can notice that the area overhead of EXU is 6.46% while the worst-case power overhead is only 2.8% when the subsample rate is 4-to-1 with $N = 16$ and $p = 16$.

Figure 4-21 shows examples of four video clips for switching the power consumption mode. The target subsample pixel count is reduced by 48 every 40

frames and the control parameter K_p is set to 0.2. The result shows that the adaptive control mechanism can make the power consumption reach the target level within 10 frames and the stationary error be under 5%. According to the battery properties described in section 4.2, the curve shows that our power-aware architecture can extend the battery lifetime by slowly and gradually degrading the quality. The marks A, B, C, and D are corresponding to the switching points in Fig. 4-2(b) respectively.

4.8 Summary

Motivated by the concept of battery properties and power-aware paradigm, this chapter presents an architecture-level power-aware technique based on a novel adaptive content-based subsample algorithm. When the battery is in the status of full capacity, the proposed ME architecture will turn 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 will shift to lower power consumption mode by disable some PEs to extend the battery lifetime with little quality degradation.

As the results of simulation, the CSA can significantly reduce computation complexity with little quality degradation. However, there exists a non-stationary problem with CSA for implementing power-aware architecture 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 video clips will make the subsample rate non-stationary; that is, the power consumption will not be converged within a narrow

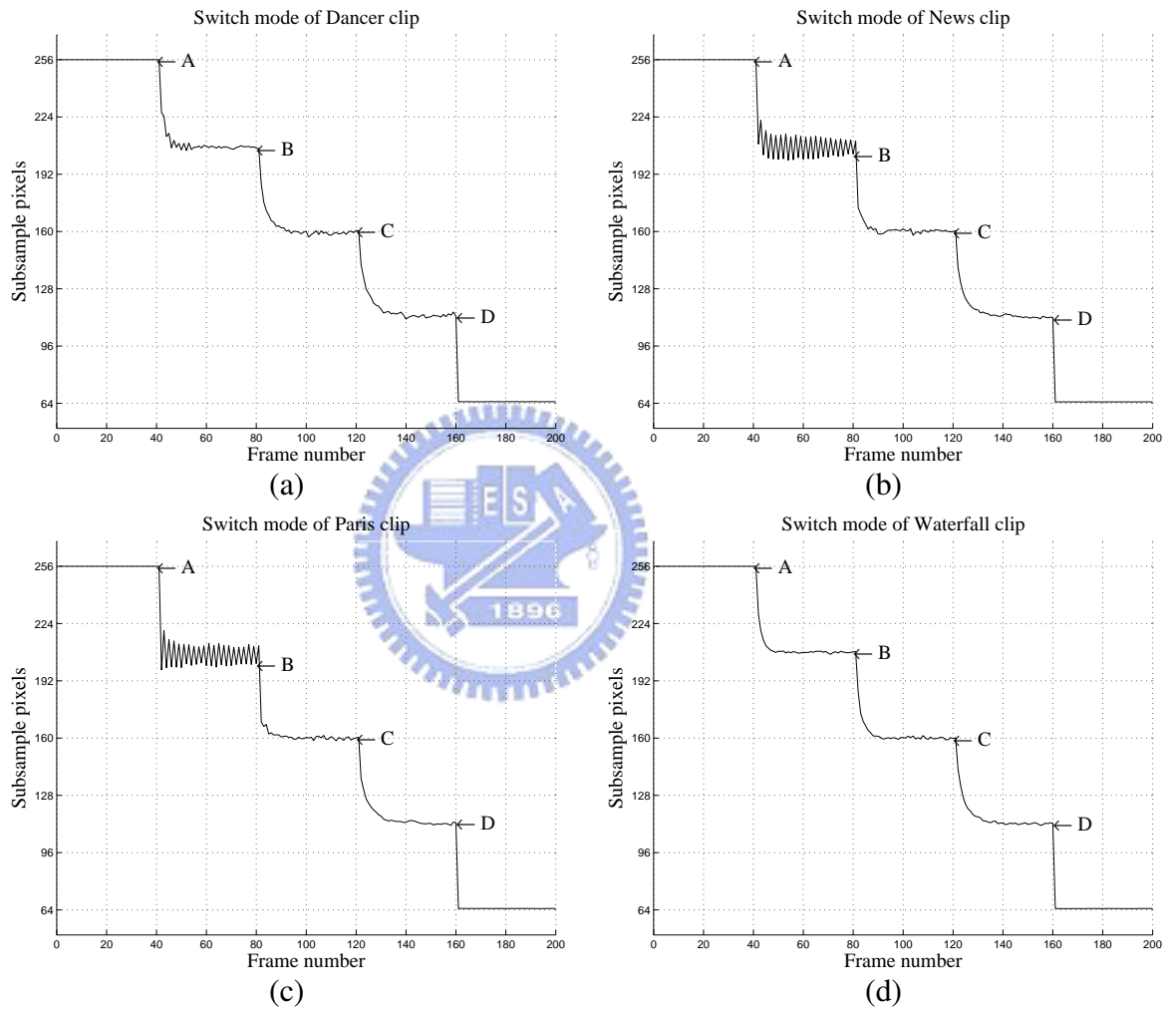


Figure 4-21: The power switching curves of four clips. (a) The Dancer Clip. (b) The News Clip. (c) The Paris Clip. (d) The Waterfall Clip.

range for a given power mode. The divergence of power consumption can result in a bad power-awareness. To solve this non-stationary problem, the paper uses an adaptive control mechanism to adaptively adjust the threshold parameters so that the subsample rate can be stationary. The adaptive control mechanism used in this work 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).

Founded on the content-based algorithm, the power-aware architecture can dynamically operate at different power consumption modes with little quality degradation according to the remaining capacity of battery pack to achieve better battery discharging property. And the control mechanism maintains the power-consumption mode in an acceptable stationary state successively.



Chapter 5

Conclusion

This thesis has presented content-based motion estimation algorithms and architectures to solve the problem of huge computation load and power-aware issue for the portable multimedia applications. The two-phase ME algorithm and the content-based power-aware ME algorithm are proposed to achieve the target. The former one uses the edge-matching approach as matching criteria of the first phase to reduce the computational complexity. The latter one applies a content-based subsample algorithm with the adaptive control mechanism to achieve the power-aware function for better quality degradation while the power mode is operated in the lower consumption level.

By employing the edge-matching criteria and the scan direction in the first phase, the edge-driven two-phase algorithm can use the quantized pixel value more efficiently and thus reduce the computation complexity. Reducing the computational complexity means reducing the power consumption. As the simulation results, the proposed two-phase algorithm can reduce the significant computation load comparing with the full-search algorithm and still can be more efficient than the exist two-phase algorithm.

The content-based power-aware algorithm performs power-aware function by disable/enable processing elements according to the subsample mask. The power-aware approach extracts the edge pixels of a macro-block and subsamples the non-edge pixels only to maintain the quality performance in acceptable level. Since the power consumption is proportional to the subsample rate, this content-based algorithm adopts a close-loop control mechanism to keep the subsample rate in stationary state. Founded on the content-based algorithm, the power-aware architecture can dynamically operate at different power consumption modes with little quality degradation according to the remaining capacity of battery pack to achieve better battery discharging property.

According to the content-based methodology, the proposed algorithms and architectures are very suitable for the portable multimedia devices which are powered by battery. The two-phase architecture is for the low-complexity application and the power-aware architecture is for extending the battery life with better quality trade-off. As the simulation results show, the proposed content-based ME algorithms and architectures can achieve better power and quality performance for the portable multimedia application.

Bibliography

- [1] *Information Technology–Coding of Moving Picture and Associated Audio for Digital Storage Media at Up to About 1.5Mbps/s–Part2: Video*, ISO/IEC 11172-2 (MPEG-1 Video), 1993.
- [2] *Information Technology–General Coding of Moving Pictures and Associated Audio Information: Video*, ISO/IEC 13818-2 (MPEG-2 Video) ITU-T H.262, 1996.
- [3] MPEG-4 Video Group, *Information Technology– Coding of Audio-Visual Objects– Part 2: Visual*, ISO/IEC 14496-2, 1999.
- [4] P. Kuhn, *Algorithms, Complexity Analysis And VLSI Architectures for MPEG-4 Motion Estimation*, Kluwer Academic Publishers, 1999.
- [5] ITU-T Recommendation H.261, *Video Codec for Audiovisual Services at Px64 Kbits*, Dec. 1990.
- [6] ITU-T Recommendation H.263, *Video Codec for Low Bitrate Communication*, May 1996.
- [7] V. Bhaskaran, *Image and Video Compression Standards–Algorithms and Architectures*, Kluwer Academic Publishers, 1997.

- [8] K.-M. Yang, M.-T. Sun, and L. Wu, "A family of VLSI designs for the motion compensation block-matching algorithm," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 36, no. 10, pp. 1317–1325, Oct. 1989.
- [9] H. Yeo and Y.-H. Hu, "A novel modular systolic array architecture for full-search block matching motion estimation," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 5, no. 5, pp. 407–416, Oct. 1995.
- [10] S.-B. Pan, S.-S. Chae, and R.-H. Park, "VLSI architectures for block matching algorithms using systolic arrays," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 6, no. 1, pp. 67–73, Feb. 1996.
- [11] C.-H. Hsieh and T.-P. Lin, "VLSI architecture for block-matching motion estimation algorithm," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 2, no. 2, pp. 169–175, Jun. 1992.
- [12] Y.-K. Lai and L.-G. Chen, "A data-interlacing architecture with two-dimensional data-reuse for full-search block-matching algorithm," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 8, no. 2, pp. 124–127, Apr. 1998.
- [13] V.-L. Do and K.-Y. Yun, "A low-power VLSI architecture for full-search block-matching motion estimation," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 8, no. 4, pp. 393–398, Aug. 1998.
- [14] J.-F. Shen, T.-C. Wang, and L.-G. Chen, "A novel low-power full-search block-matching motion-estimation design for h.263+," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 11, no. 7, pp. 890–897, Jul. 2001.

- [15] 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 Trans. Circuits Syst. Video Technol.*, vol. 12, no. 1, pp. 61–72, Jan. 2002.
- [16] J.-N. Kim and T.-S. Chio, "Adaptive matching scan algorithm based on gradient magnitude for fast full search in motion estimation," *IEEE Tran. Consumer Electronics*, vol. 45, no. 3, pp. 762–772, Aug. 1999.
- [17] T. Koga, K. Iinuma, A. Hirano, Y. Iijima, and T. Ishiguro, "Motion-compensated interframe coding for video conferencing," in *Proc. National Telecommunications Conf.*, New Orleans, LA, Nov. 1981, pp. G5.3.1–G5.3.5.
- [18] R. Li, B. Zeng, and M.-L. Liou, "A new three-step search algorithm for block motion estimation," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 4, no. 4, pp. 438–442, Aug. 1994.
- [19] M.-J. Chen, L.-G. Chen, and T.-D. Chiueh, "One-dimensional full search motion estimation algorithm for video coding," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 4, no. 5, pp. 504–509, Oct. 1994.
- [20] L.-M. Po and W.-C. Ma, "A novel four-step search algorithm for fast block motion estimation," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 6, no. 3, pp. 313–317, 1996.
- [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 Trans. Circuits Syst. Video Technol.*, vol. 8, no. 4, pp. 369–377, Aug. 1998.

- [22] A.-M. Tourapis, O.-C. Au, M.-L. Liou, G. Shen, and I. Ahmad, "Optimizing the MPEG-4 encoder-advanced diamond zonal search," in *Proc. 2000 Int. Symp. Circuits and Systems*, Geneva, Switzerland, May 2000, pp. 674–677.
- [23] S. Zhu and K.-K. Ma, "A new diamond search algorithm for fast block-matching motion estimation," *IEEE Trans. Image Processing*, vol. 9, no. 2, pp. 287–290, Feb. 2000.
- [24] B. Liu and A. Zaccarin, "New fast algorithms for the estimation of block motion vectors," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 3, no. 2, pp. 148–157, Apr. 1993.
- [25] B. Natarajan, V. Bhaskaran, and K. Konstantinides, "Low-complexity block-based motion estimation via one-bit transforms," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 7, no. 4, pp. 702–706, Aug. 1997.
- [26] S. Cucchi and D. Grechi, "A new features-based fast algorithm for motion estimation: Decimated integral projection (D.I.P.)," in *Proc. 1997 Int. Conf. Infor., Commun., and Signal Processing.*, Singapore, Sep. 1997, pp. 297 – 300.
- [27] J.-S. Kim and R.-H. Part, "Feature-based block matching algorithm using integral projections," *IEE Electron. Letter*, vol. 25, pp. 29–30, Jan. 1989.
- [28] H. Gharavi and M. Mills, "Blockmatching motion estimation algorithms—new results," *IEEE Trans. Circuits and Systems*, vol. 37, no. 5, pp. 649 – 651, May 1990.

- [29] J.-S. Kim and R.-H. Part, "A fast feature-based block matching algorithm using integral projections," *IEEE Journal Select. Areas Commun.*, vol. 10, no. 5, pp. 968–971, Jun. 1992.
- [30] K. Sauer and B. Schwartz, "Efficient block motion estimation using integral projections," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 6, no. 5, pp. 513–518, Oct. 1996.
- [31] S. Lee and S.-I. Chae, "Motion estimation algorithm using low-resolution quantization," *IEE Electronic Letters*, vol. 32, no. 7, pp. 647–648, Mar. 1996.
- [32] H.-W Cheng and L.-R. Dung, "EFBLA: A two-phase matching algorithm for fast motion estimation," in *Advances in Multimedia Information Processing - PCM 2002*, Dec. 2002, vol. 2532, pp. 112–119.
- [33] H.-W. Cheng and L.-R. Dung, "EFBLA: a two-phase matching algorithm for FAST motion estimation," *IEE Electronic Letters*, vol. 40, no. 11, pp. 660–661, May 2004.
- [34] W. Li and E. Salari, "Successive elimination algorithm for motion estimation," *IEEE Trans. Image Processing*, vol. 4, no. 1, pp. 105–107, Jan. 1995.
- [35] S. Lee, J.-M. Kim, and S.-I. Chae, "New motion estimation using low-resolution quantization for MPEG2 video encoding," in *Workshop on VLSI Signal Processing, IX*, Oct. 1996, pp. 428–437.
- [36] S. Lee and S.-I. Chae, "Two-step motion estimation algorithm using low-resolution quantization," in *International Conference on Image Processing*, Sep. 1996, pp. 795–798 vol.3.

- [37] R.-C. Gonzalez and R.-E. Woods, *Digital Image Processing*, Addison Wesley, Sep. 1993.
- [38] “<http://www.m4if.org/resources.php>,” .
- [39] H.-W Cheng and L.-R. Dung, “A vario-power ME architecture using content-based subsample algorithm,” *IEEE Trans. Consumer Electronics*, vol. 50, no. 1, pp. 349–354, Feb. 2004.
- [40] H.-W. Cheng and L.-R. Dung, “A power-aware motion estimation architecture using content-based subsampling,” *Journal of Information Science and Engineering*, (accepted).
- [41] H.-W. Cheng and L.-R. Dung, “A content-based methodology for power-aware motion estimation architecture,” *IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing*, (accepted).
- [42] M. Bhardwaj, R. Min, and A.-P. Chandrakasan, “Quantifying and enhancing power awareness of VLSI systems,” *IEEE Trans. VLSI Systems*, vol. 9, no. 6, pp. 757–772, December 2001.
- [43] J.-R. Jain and A.-K. Jain, “Displacement measurement and its application in interframe image coding,” *IEEE Trans. Commun.*, vol. COM-29, pp. 1799–1808, Dec. 1981.
- [44] C. Zhu, X. Lin, and L.-P. Chau, “Hexagon-based search pattern for fast block motion estimation,” *IEEE Trans. Circuits Syst. Video Technol.*, vol. 12, no. 5, pp. 349–355, May 2002.

- [45] J.-H. Luo, C.-N. Wang, and T. Chiang, "A novel all-binary motion estimation (ABME) with optimized hardware architectures," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 12, no. 8, pp. 700–712, Aug. 2002.
- [46] C.-L. Su and C.-W. Jen, "Motion estimation using msd-first processing," *IEE Proc-G Circuits, Devices and Systems*, vol. 150, no. 2, pp. 124–133, April 2003.
- [47] O.-S. Unsal and I. Koren, "System-level power-aware design techniques in real-time systems," *Proceedings of the IEEE*, vol. 91, no. 7, pp. 1055–1069, July 2003.
- [48] C.-K. Cheung and L.-M. Po, "Normalized partial distortion search algorithm for block motion estimation," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 10, no. 3, pp. 417–422, Apr. 2000.
- [49] D. Linden, *Handbook of Batteries, Second Edition*, McGraw-Hill, Inc., 1995.
- [50] *Mobile Pentium® III Processor in BGA2 and Micro-PGA2 Packages Datasheet*, p. 55, Intel Corporation.
- [51] Y.-L. Chan and W.-C. Siu, "New adaptive pixel decimation for block motion vector estimation," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 6, pp. 113–118, Feb. 1996.
- [52] P. Raghavan and C. Chakrabarti, "Battery-friendly design of signal processing algorithms," in *IEEE Workshop on Signal Processing Systems*, Aug. 2003, pp. 304–309.

- [53] S.-C. Cheng and H.-M. Hang, "A comparison of block-matching algorithms mapped to systolic-array implementation," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 7, no. 5, pp. 741–757, Oct. 1997.
- [54] H.-W. Cheng and L.-R. Dung;, "A novel vario-power architecture of motion estimation using a content-based subsample algorithm," in *IEEE Workshop on Signal Processing Systems*, Aug. 2003, pp. 201 – 206.
- [55] H.-W. Cheng and L.-R. Dung;, "A power-aware architecture for motion estimation," in *The 14th VLSI Design/CAD Symposium*, Taiwan, Aug. 2003, pp. 133–136.
- [56] H.-W. Cheng and L.-R. Dung;, "A power-aware ME architecture using subsample algorithm," in *IEEE International Symposium on Circuits and Systems*, May 2004, pp. III–821–4 Vol.3.
- [57] M.-B. Ahmad and T.-S Choi, "Edge detection-based block motion estimation," *IEE Electronic Letters*, vol. 37, no. 1, pp. 17–18, Jan. 2001.
- [58] L.-D Vos and M. Stegherr, "Parameterizable VLSI design for the motion compensation block-matching algorithm," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 36, no. 10, pp. 1317–1325, Oct. 1989.
- [59] Y.-K. Lai, Y.-L. Lai, Y.-C. Liu, P.-C. Wu, and L.-G. Chen, "VLSI implementation of the motion estimator with two-dimensional data-reuse," *IEEE Trans. Consumer Electronics*, vol. 44, no. 3, pp. 623–629, Aug. 1998.
- [60] J.-C. Tuan and C.-W. Jen, "An architecture of full-search block matching for minimim memory bandwidth requirement," *Proc. IEEE 8th Great Lake Symp. VLSI*, pp. 152–156, 1998.

- [61] S. Chang, J.-H. Hwang, and C.-W. Jen, "Scalable array architecture design for full search block matching," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 5, no. 4, pp. 332–343, Aug. 1995.
- [62] Y.-S. Jehng, L.-G. Chen, and T.-D. Chiueh, "An efficient and simple VLSI tree architecture for motion estimation algorithms," *IEEE Trans. on Signal Processing*, vol. 41, no. 2, pp. 889–900, Feb. 1993.
- [63] J.-N. Kim and T.-S. Choi, "A fast full-search motion-estimation algorithm using representative pixels and adaptive matching scan," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 10, no. 7, pp. 1040–1048, Oct. 2000.
- [64] T. Komarek and P. Pirsch, "Array architectures for block matching algorithms," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 36, no. 10, pp. 1302–1308, 1989.
- [65] B.-S. Kim and J.-D. Cho, "VLSI architecture for low power motion estimation using high data access reuse," in *ASICs, 1999. AP-ASIC '99. The First IEEE Asia Pacific Conference on*, Aug. 1999, pp. 162–165.
- [66] X. Lu and O.-C. Au, "Improved fast motion estimation using integral projection features for hardware implementation," in *IEEE Intel. Symp. Circuits and Syst.*, Hong Kong, Jun. 1997, pp. 1337–1340.
- [67] 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, no. 9, pp. 2355–2365, Sep. 2002.

- [68] C.-L. Su and C.-W. Jen, "MSD-first on-line arithmetic progressive processing implementation for motion estimation," *IEICE Trans. Inf. and Syst.*, vol. E86-D, no. 11, pp. 2433–2443, Nov. 2003.
- [69] B. Natarajan, V. Bhaskaran, and K. Konstantinides, "Low-complexity algorithm and architecture for block-based motion estimation via one-bit transforms," in *Acoustics, Speech, and Signal Processing, 1996. ICASSP-96. Conference Proceedings., 1996 IEEE International Conference on*, May 1996, pp. 3244–3247.
- [70] G. Privat and M. Renaudin, "Motion estimation VLSI architecture for image coding," in *Proc. Int. Conf. Comput. Design: VLSI Comput. Processors*, 1989, pp. 78–81.
- [71] Wujian Zhang, Runde Zhou, and T. Kondo, "Low-power motion-estimation architecture based on a novel early-jump-out technique," *The IEEE International Symposium on Circuits and Systems*, vol. 5, pp. 187–190, 2001.