

國立交通大學

電機與控制工程學系

博士論文

MPEG-4/H.264 視訊壓縮標準於嵌入式系統之最佳化研究

The Study of Optimization Approaches for MPEG-4/H.264 Video Processing
Standards on Embedded Systems

研究生：彭信元

指導教授：吳炳飛 教授

中華民國九十七年五月

MPEG-4/H.264 視訊壓縮標準於嵌入式系統之最佳化研究
The Study of Optimization Approaches for MPEG-4/H.264 Video Processing
Standards on Embedded Systems

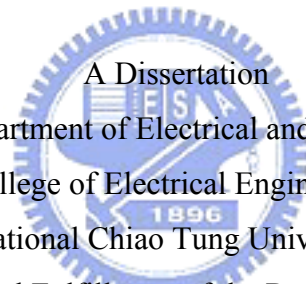
研究生：彭信元

Student : Hsin-Yuan Peng

指導教授：吳炳飛 教授

Advisor : Prof. Bing-Fei Wu

國立交通大學
電機與控制工程學系
博士論文



A Dissertation
Submitted to Department of Electrical and Control Engineering
College of Electrical Engineering
National Chiao Tung University
in partial Fulfillment of the Requirements
for the Degree of
Doctor of Philosophy
in

Electrical and Control Engineering

May 2008

Hsinchu, Taiwan, Republic of China

中華民國九十七年五月

MPEG-4/H.264 視訊壓縮標準於嵌入式系統之最佳化研究

學生：彭信元

指導教授：吳炳飛 教授

國立交通大學電機與控制工程學系博士班

摘 要

本論文中提出許多最佳化之方式將 MPEG-4 與 H.264 視訊壓縮技術分別實現在以 ARM、FPGA 搭配之 SoC 開發平台與以 ARM、DSP 為基礎之雙核心平台，透過硬體、韌體及軟體上在壓縮流程上的最佳化，將複雜的演算法實現在運算能力較低之嵌入式系統中，並得到良好的壓縮效果。

在 MPEG-4 硬體最佳化上，本論文提出一種階層式的動態估測演算法，透過數學的分析以及實驗的結果來決定哪些步驟對於壓縮品質之影響力較為廣泛，推導一快速、高品質且適合用硬體實現之演算法。且其運算複雜度僅為全域搜尋法之 3.91%。搭配此演算法，以 ARM 配合 FPGA 的實現方式來進行 MPEG-4 即時壓縮系統的研究。此系統可依據使用者要求動態調整輸入以及輸出影像參數，可有效提高其應用性。

而在 H.264 於 ARM/DSP 雙核心嵌入式平台的實現上，推導一適合影像式車用防盜系統之動態估測演算法，可用低複雜度的運算方式，有效克服車輛行駛中因外在光源變化等因素導致視訊壓縮品質下降的問題，並且針對車輛影像特性將動態估測流程作動態更新，能在重點區域保持良好壓縮品質，同時大幅增加運算速度。除此之外，本論文亦將 H.264 壓縮流程做最簡化，利用記憶體配置管理、雙核心 CPU 工作分配以及 DSP 特有指令等最佳化技巧，成功移植入低成本且高穩定之嵌入式系統中。

除了將演算法以硬體與軟體等不同的方式進行最佳化之外，本論文亦將實現之系統實際運用於由交通大學自製研發之 Taiwan ITS-1 智慧車中，搭配了 Zigbee 無線傳輸以及 3.5G/3G 手機程式設計，當車輛正在移動時，本系統可偵測駕駛員前方、後方以及兩側盲點區域之車道線以及車輛距離，並以直覺的方式提供資訊。而當車輛停妥後，若有歹徒強行闖入，系統則會即時發送警告訊息至使用者，車主即可利用手機隨時隨地瀏覽愛車情況，並可馬上提供警方歹徒之面貌。本論文之宗旨在利用各種最佳化方式，於嵌入式系統上實現動畫壓縮技術，並將其與智慧型車輛結合，讓駕駛員、乘客以及車輛都能有更安全的環境。

The Study of Optimization Approaches for MPEG-4/H.264 Video Processing Standards on Embedded Systems

Student : Hsin-Yuan Peng

Advisors : Prof. Bing-Fei Wu

Department of Electrical and Control Engineering
National Chiao Tung University

ABSTRACT

In this dissertation, several optimization approaches for implementing the MPEG-4 visual and the H.264 video processing standards on an ARM/FPGA Soc platform and an ARM/DSP dual-core embedded system are addressed. By the modification of the compression procedures in hardware, firmware and software, the complex algorithms are successfully ported to the low computing power embedded devices, and good encoding performance is obtained.

In the MPEG-4 hardware optimization, an efficient hierarchical motion estimation algorithm (HMEA) is proposed. With mathematical analyses and the experimental results, the search steps, which can dominate the image quality, are designed in order to develop a high-speed and good quality architecture that is suitable for hardware implementation. The operational complexity of HMEA is only 3.91% of the full search block matching algorithm, and with HMEA, a register-based platform-independent MPEG-4 co-processor (RPIMC) is designed. RPIMC can change the input and the output parameters of the bitstreams by adjusting the value of the registers dynamically to widen its applications.

For implementing the H.264 framework on an ARM/DSP dual-core embedded system, an adaptive motion estimation algorithm (AMEA) especially for the vehicle surveillance videos is developed. By easy manipulations, AMEA can overcome the coding quality degradation due to the changing environments, and it can update the search procedure adaptively according to the video characteristics in a moving object to maintain the performance in the region of interest. Moreover,

the platform-based acceleration techniques, such as the memory organization, the dual-core communication and the program modification, for H.264 are also presented.

In addition to algorithm optimization in hardware, firmware, and software, the developed system in this dissertation has been successfully tested in the intelligent vehicle, Taiwan *i*TS-1, developed in National Chiao Tung University, on the real road environment in Taiwan. Cooperating with the warning system, which can detect the vehicles and the lane marks in front/rear directions and blind spot areas, the Zigbee wireless communication and the 3.5G/3G mobile phone programming, the system can provide the distance information of the surrounding cars to the drivers through a friendly intuitive graph, and it will produce the warning signal when lane departure. After parking, if there exists intruders, the system will send a notification message to the users, and they can browse the real-time images of the vehicles by their mobile phones anytime and anywhere. The main contribution of dissertation is to provide a safer environment to drivers, passengers and vehicles at all time using several optimization approaches for implementing complex video processing standards on embedded systems.



Table of Contents

摘要.....	I
ABSTRACT.....	II
Table of Contents.....	IV
List of Figures.....	VII
List of Tables.....	IX
Chapter 1. Introduction.....	1
1.1 Motivation.....	1
1.2 The Proposed Optimization Approaches and Their Contributions.....	4
1.2.1. Efficient Hierarchical Motion Estimation Algorithm and Its VLSI Architecture.....	5
1.2.2. Register-Based Platform-Independent MPEG-4 Co-Processor.....	6
1.2.3. Adaptive Motion Estimation Algorithm for Vehicle Surveillance Videos.....	7
1.2.4. Real-Time Driving Assist and Surveillance System.....	8
1.3 Organization.....	9
Chapter 2. Efficient Hierarchical Motion Estimation Algorithm and Its VLSI Architecture.....	10
2.1. Introduction.....	10
2.2. Hierarchical Motion Estimation Algorithm.....	16
2.2.1. Downsampling Methods.....	16
2.2.2. Framework of HMEA.....	24
2.2.3. Experimental Results.....	30
2.3. Proposed VLSI Architecture.....	33
2.3.1. Overall Architecture.....	33
2.3.2. Downsampling.....	36
2.3.3. DAU.....	40
2.3.4. DAU Processing at Each Level.....	43
2.3.5. Half-Pel Search.....	48
2.4. Implementation Results.....	50
Chapter 3. Register-Based Platform Independent MPEG-4 Co-Processor and Its System-Level Design.....	53
3.1 Introduction.....	53
3.2 The Architecture of RPIMC.....	57
3.2.1. The Hardware Pipeline Scheduling.....	58
3.2.2. The Register Bank.....	61
3.2.3. Memory Organization.....	64

3.3	System-Level Design	66
3.3.1.	The Software Architecture of REMR	67
3.3.2.	The FPGA Prototype of REMR	69
3.4	Implementation Results	72
3.4.1.	The Chip Implementation Results	72
3.4.2.	The Performance Comparison	74
Chapter 4.	An Adaptive Motion Estimation Algorithm for Vehicle Surveillance Videos	76
4.1	Introduction	76
4.2	The Analysis of The Vehicle Surveillance Videos	80
4.2.1.	Lighting Effects	80
4.2.2.	The Coding Performance degradation due to the lighting effects	85
4.2.3.	The MV characteristics	88
4.3	Adaptive Motion Estimation Algorithm	90
4.3.1.	Histogram extension algorithm (HEA)	91
4.3.2.	AMEA in ROI	94
4.3.3.	AMEA out of ROI	100
4.3.4.	The complexity analysis of AMEA	101
4.4	Implementation Results	103
Chapter 5.	A Real-Time Driving Assist and Surveillance System	107
5.1	Introduction	107
5.2	The Innovative Functional Scheme of DASS	113
5.2.1.	Active Vision-based Warning Supports	114
5.2.2.	H.264 Driving Status Recorder	116
5.2.3.	Burglarproof with Mobile Surveillance	117
5.3	The Module and Architecture of DASS	118
5.3.1.	Lane Departure Warning Module	119
5.3.2.	Forward / Backward Collision Warning Module	119
5.3.3.	Blind Spot Warning Module	120
5.3.4.	The Software Architecture of VSS	121
5.4	H.264 Encoder with AMEA on VSS	122
5.4.1.	Cooperation of ARM and DSP	123
5.4.2.	H.264 Encoder Optimization	126
5.4.3.	The Mobile Phone Programming	130
5.5	Implementation Results	132
5.5.1.	Lane departure and vehicle detection algorithms	134
5.5.2.	Friendly Distance Display	136
5.5.3.	Experimental Results of H.264 Encoder	137

5.5.4.	The Performance and Functionalities of VSS	140
5.5.5.	Comparisons.....	143
Chapter 6.	Conclusions and Perspective.....	145
6.1	The hardware-based HMEA and RPIMC	145
6.2	The software-based AMEA and DASS	146
References		148



List of Figures

Fig. 2-1. The hierarchical frame structure	16
Fig. 2-2. Examples of 2D-DWT downsampling: (a) frequency bands after two-level DWT decomposition; (b) original Akiyo image; (c) reduce the Akiyo image 50% in both width and height; (d) reduce the Akiyo image 25% in both width and height	19
Fig. 2-3. The relationship between the 2D-DWT and the downsampled images	19
Fig. 2-4. The correlation coefficients between the adjacent images for averaging filter, left_top method, Haar DWT, and Antonini 9/7 DWT in (a) Flower garden (b) Stefan (c) Mobile	23
Fig. 2-5. The hierarchical search procedure.	25
Fig. 2-6. The overall architecture of HMEA.	35
Fig. 2-7. The timing diagram of HMEA for level 0	36
Fig. 2-8. Down sampling four rows of level 2.....	37
Fig. 2-9. The downsampling hardware block diagram for level 2.....	38
Fig. 2-10. The downsampling hardware block diagram for level 1.....	39
Fig. 2-11. The architecture of a 2D DAU.....	42
Fig. 2-12. The basic data flow of DAU.....	43
Fig. 2-13. The timing diagram of HMEA for level 1.....	44
Fig. 2-14. The timing diagram of HMEA for level 2.....	45
Fig. 2-15. Interpolation for half-pel search.....	49
Fig. 3-1 The overall architecture of RPIMC.....	58
Fig. 3-2 The intra MB scheduling	59
Fig. 3-3 The inter MB scheduling	61
Fig. 3-4 The flow chart of the controller	62
Fig. 3-5 The state of the external memory.....	66
Fig. 3-6 The system overview of REMR.....	67
Fig. 3-7 The software architecture of REMR	68
Fig. 3-8 The architecture of REMR in ARM Integrator.....	71
Fig. 3-9 The chip layout	73
Fig. 4-1. The vehicle surveillance image with ROI.....	80
Fig. 4-2. The tunnel which will cause lighting effects.	82
Fig. 4-3. (a)The brightest image, (b)The darkest image of V1.....	83
Fig. 4-4. The \bar{M}_k of each frame of V1	83
Fig. 4-5. (a)The brightest image, (b)The darkest image of V2.....	84
Fig. 4-6. The \bar{M}_k of each frame of V2	85
Fig. 4-7. The correlation coefficients between each frame pairs of V1.....	86
Fig. 4-8. The correlation coefficients between each frame pairs of V2.....	87

Fig. 4-9. (a) The \overline{M}_k of each frame ,(b)The correlation coefficients between each frame pairs of V0	88
Fig. 4-10. The manipulated MVs	89
Fig. 4-11. (a) depicts the probability of the magnitude of MVs in and (b) out of ROI for all frames of V0.	90
Fig. 4-12. (a) the histograms of the input frame before and (b) after the pre-processing.	94
Fig. 4-13. AMEA search diamonds SD_0 , SD_1 and SD_2	96
Fig. 4-14. Closed-loop adaptation process for AMEA.	97
Fig. 4-15 The correlation coefficients between each frame pairs of (a)V1 and (b)V2 before and after HEA.....	105
Fig. 4-16 The \overline{M}_k of each frame of (a)V1 and (b)V2 before and after HEA.....	106
Fig. 5-1 The detection zones of LVWS, BSWS and VSS	113
Fig. 5-2 The overall system block diagram of DASS.....	114
Fig. 5-3 The flowchart of the active real-time vehicle detection and lane departure warning system.....	116
Fig. 5-4 The flowchart of the full-time recorder.....	117
Fig. 5-5 The flowchart of the mobile surveillance system	118
Fig. 5-6 The estimated lateral offset.....	119
Fig. 5-7 The software architecture of VSS.....	122
Fig. 5-8 The proposed H.264 encoding data path.....	124
Fig. 5-9 The synchronization of audio and video encoding	125
Fig. 5-10 Communication between related threads	126
Fig. 5-11 The original H.264 encoding control flow.....	127
Fig. 5-12 The proposed H.264 encode faster control flow.....	128
Fig. 5-13 The state chart of server-push we addressed.....	131
Fig. 5-14 The processing flowchart of the program on a mobile phone.....	132
Fig. 5-15 TAIWAN ITS-1	133
Fig. 5-16 The monochromatic CCD mounted behind the (a) front (b) rear windshields.....	133
Fig. 5-17 The monochromatic CCD mounted on the (a) left (b) right side view mirrors.....	134
Fig. 5-18 The USB CCD mounted behind the right front windshield.....	134
Fig. 5-19 Detection results in front-rear views from LVWS.....	135
Fig. 5-20 Detection results in side views from BSWS	135
Fig. 5-21 Detection results in different conditions	136
Fig. 5-22 Distance and location on screen.....	137
Fig. 5-23 The experimental results of (a)applying deblocking filter and (b)not applying it	138
Fig. 5-24 The GUI developed on mobile phones.	142
Fig. 5-25 The screen of our program on Nokia mobile phone.	142
Fig. 5-26 The suspend mode.	142

List of Tables

Table 2-1	The comparison of the video quality between various downsampling methods for left-top, Haar's DWT, Atonini's 9/7 DWT, and the averaging filter in dB.	22
Table 2-2	The PSNR performance and the complexity analysis in different number of the MV candidates at the coarsest level.....	26
Table 2-3	The PSNR performance and the complexity analysis in different LSRs in the middle and the finest level.	27
Table 2-4	The comparison of the complexity, including half-pel search, between FSBMA, <i>n</i> SS, MRMC-4, MRMC5 and HMEA in different search ranges.....	32
Table 2-5	The PSNR comparisons of various fast-search algorithms in dB.	32
Table 2-6	The comparison between HMEA with other architectures for MMEA, FSBMA, and PMVFAST. The search range among each frameworks are different, the equivalent gate counts includes control and address generation overheads as well as the computational core (PEs), and the MB size is 16×16	52
Table 3-1	The register banks	62
Table 3-2	The chip specification of RPIMC	74
Table 3-3	The performance comparison between the software model and RPIMC in dB	74
Table 3-4	The performance comparison between other MPEG-4 chips and RPIMC.....	75
Table 4-1	The encoding quality comparisons of the different videos in dB.....	87
Table 4-2	The quality comparisons of the blocks in and out of ROI of different video sequences in dB.	104
Table 4-3	The normalized processing speed comparisons of the different videos	104
Table 5-1	PSNR and FPS evaluation for intra predication with different block sizes.....	129
Table 5-2	PSNR and FPS evaluation for intra predication with different directional predictions.....	129
Table 5-3	Performance comparison with deblocking filter	138
Table 5-4	PSNR and MCPS Comparison.....	139
Table 5-5	PSNR and comparison ratio comparison	140
Table 5-6	The specification of VSS.	141
Table 5-7	The comparisons between different systems.....	144

Chapter 1. Introduction

With the wide spread adoption of technologies such as digital televisions, internet streaming videos and DVD-Videos, video compression has become an essential component of broadcast and entertainment media. The MPEG-4 visual and the H.264 video coding standards are commonly used in internet and wireless transmission applications, and they can provide higher flexibility and compression ratio than the earlier MPEG-1, MPEG-2, and H.263 ones [1]-[5]. However, the computational complexity of these two superior frameworks, MPEG-4 visual and H.264, are also much larger. PCs with strong CPUs and inexhaustible power supplies can easily implement them, but the need of human beings can not accept that. People nowadays want to record the real-time videos to take notes or share with their friends using their handheld devices, such as portable video recorders, digital cameras, and mobile phones. These equipments are always using low-power CPUs in order to save their battery life. Therefore, the study of optimization approaches for MPEG-4/H.264 video compression standards on embedded systems is essential to port the complex algorithms on the consumer electronics while maintaining the image coding quality. In this dissertation, several algorithmic, practical and integrated methods in both the software and the hardware point of view, and the system-level design techniques for embedded platforms are presented.

1.1 Motivation

The last decade has shown the quick growth of multimedia applications and

services, with audiovisual information, playing an increasingly important role. Today's existence of tens of millions of digital audiovisual content users and consumers is tightly linked to the maturity of such technological areas as video and audio compression and digital electronics and to the timely availability of appropriate coding standards. These standards allow the industry to make major investments with confidence in new products and applications and users to experience easy consumption and exchange of content. In this environment, the Moving Picture Experts Group's (MPEG's) remit is to develop standards for compression, processing and representation of moving pictures. It has been responsible for a series of important standards starting with MPEG-1 (compression of video and audio for CD playback) and following on with the very successful MPEG-2 (storage and broadcasting of 'television-quality' video and audio). MPEG-4 is the latest standard that deals specially with audio-visual coding, and it can provide a more flexible and efficient update to the earlier MPEG-1 and MPEG-2 standards.

On the other hand, the Video Coding Experts Group (VCEG), a study group of the International Telecommunications Union (ITU), was also responsible for the first widely-used videotelephony standard (H.261) and its successor, H.263, and initiated the early development of the H.26L project [6]. The two groups set-up the collaborative Joint Video Team (JVT) to finalize the H.26L proposal and convert it into an international standard (H.264/MPEG-4 Part 10) published by both ISO/IEC and ITU-T.

With the widespread popularity of handheld multimedia devices, such as digital video recorders (DVs), digital cameras (DCs), personal digital assistants (PDAs), and mobile phones, in these days, people rely on the mobile technology more and more.

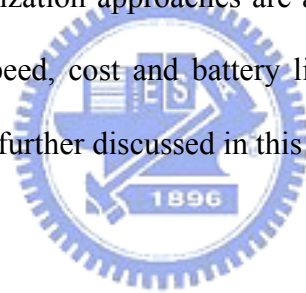
To integrate these consumer electronics into a multi-functionalities equipment is a world wide trend, and the need for capturing audio-visual information for various purposes is becoming a main stream. The MPEG-4 visual and the H.264 video processing standards can provide amazing compression ratio while maintaining excellent video quality, and the existence of them respond to the emerging requirement of mankind.

However, due to the cost, the size, the battery life, and many restrictions, these portable instruments are always implemented with CPUs with low operating frequencies or application-specific integrated circuits (ASICs) with small size and lesser power consumptions. Therefore, to development high performance video encoders with extremely tremendous computational complexities on these embedded platforms are great challenges that must overcome for new generation communications. Hence, the optimization approaches which can reduce the calculation load and increase the execution speed of these two frameworks, MPEG-4 visual and H.264, for the embedded systems are definitely necessary, and they can be separated into three major categories, one is the hardware, another is the software, and the other is the platform-based methods.

All of these three perspectives are efficient optimization approaches, they focus on the different implantations and applications, and they have distinct advantages. For general purpose video encoders, which have fixed parameters, such as input image resolution and framerate, using hardware accelerators is appropriate, since they can reduce the overall power consumption and cost. However, if the users want to have high performance in the changing environments, like vehicles or outdoors, the software video compressors can provide flexible inputs, and they can perform the data

pre-process to get better quality according to the various utilizations. Nevertheless, it requires a CPU which is power enough to execute the programs and the energy requirement will be also raised. Both hardware and software techniques need to be integrated into platforms to accomplish a system, so platform-based manners are also essential to reduce the system overhead. The off-chip memory organization, the usage of direct memory accelerator (DMA), and the communications between processors are all acceleration methods that depend on system-level designs.

A high performance video recorder has not only a fast compressor, implemented by hardware or software, but also a good system peripherals controller that coordinate the input and the output data scheduling well. Therefore, the hardware, the software, and the platform-based optimization approaches are all required to form an efficient system in terms of coding speed, cost and battery life, and the overall cooperation between these aspects will be further discussed in this dissertation.



1.2 The Proposed Optimization Approaches and Their Contributions

The most complex part of the video coding standards is the motion estimation (ME), it spends about 70% of the total encoding time, and it dominates the image quality. The straight forward method is full search block matching algorithm (FSBMA), but its computational complexity is too high to be implemented on embedded systems. Therefore, in order to simplify the search procedure while maintaining the image quality using different implementation frameworks, i.e. hardware or software, according to the system requirements is one of the main

contributions of this dissertation. They are briefly introduced in the following sub-sections.

1.2.1. Efficient Hierarchical Motion Estimation Algorithm and Its VLSI Architecture

As mentioned in Section 1.1, for general purpose video encoder, a hardware accelerator is suitable for reducing the cost and increasing the performance. Therefore, an efficient hierarchical ME algorithm (HMEA), and its high speed VLSI architecture are developed [7][8]. HMEA is a kind of multi-resolution ME algorithm, and it can simplify FSBMA by downsampling the original frame to three levels. At the smallest resolution, the least two motion vector (MV) candidates are selected using FSBMA. At the middle level, these two candidate MVs are employed as the center points for small range local searches. Then, at the original resolution, the final MV is obtained by performing a local search around the single candidate from the middle level. HMEA adopts an averaging filter, by analyzing several downsampling methods, to downsample the original image, which is the first step of the estimation progress. When superior quality is obtained in the anterior part of the ME, the refining procedures can be significantly shorter and the complexity can be reduced accordingly. HMEA can achieve almost the same coding performance as FSBMA in terms of peak signal-to-noise ratio (PSNR), but HMEA is faster. Moreover, A high-speed pipeline VLSI architecture with a reasonable chip area for HMEA is also addressed. It utilizes an efficient two dimensional (2D) processing element (PE) array to compute the sum of the absolute differences (SADs), and the search range can be doubled without adding any hardware. The architecture is suitable for VLSI

implementation because the number of the computations for each macro block (MB) is fixed. HMEA is faster and more area-efficient than numerous existing full-search and multi-resolution architectures.

1.2.2. Register-Based Platform-Independent MPEG-4 Co-Processor

When HMEA is successfully implemented, the rest manipulation blocks of the MPEG-4 video processing standard are started to develop. After all, a compressor needs not only to reduce the redundancy in spatial domain, but also in the frequency one. Moreover, the statistical encoding and the bitstream packaging functions are also necessary to encode raw data to the standard format. Therefore, a register-based platform-independent MPEG-4 co-processor (RPIMC) is designed to accentuate the efficiency of HMEA [9][10]. The goal of RPIMC is not merely to be a normal video encoder, and it tries to be as flexible as the software one using firmware/hardware hybrid methodology. RPIMC can transfer and receive the image data in all kinds of bus matrices with suitable wrappers for being easily integrated into other platform, and it can adjust the data types of the input and the output streams by modifying the relative registers. The main idea of RPIMC is to provide a programmable MPEG-4 encoder which can easily integrated into any platforms to satisfy the demand for various applications by modifying the input frame size, the input frame rate, and the output bitrate. The main controller of RPIMC will automatically calculate the internal loops of the pipelines for encoding, and will read the data in the corresponding memory with the correct image resolution based on these registers, respectively. Therefore, the manufacturers who need MPEG-4 encoders can easily integrate RPIMC into their platforms, and they can use RPIMC for various applications with

proper register settings.

1.2.3. Adaptive Motion Estimation Algorithm for Vehicle Surveillance Videos

Although RPIMC tries its best to be as flexible as the software coders, it still has limitations. If the input video is influenced by the noise, RPIMC can not pre-process that and will lead to poor performance. Besides, if the information in the dedicate part of the image are important, and clear results are required, RPIMC doesn't have this feature. Therefore, an adaptive motion estimation algorithm (AMEA) especially for the vehicle surveillance videos is developed to overcome the problems [11]. The total amounts of vehicles and the traffic accidents are increasing year by year, but the efficiency of finding the real reasons, which cause the tragedies, is still low. If a system, which can record the situations inside the cars, is adopted, the evidences can be provided to the police officers as soon as possible. In the real environments, the videos contain several problems, such as the different weathers, and the changing of the light sources, and these will lead to the heavy computation and the bad video quality. On one hand, in order to overcome the drawbacks, AMEA applies an easy histogram extension algorithm (HEA) to pre-process the images to keep the images stable and increase the estimation accuracy greatly. On the other hand, the purpose for the monitoring information is to provide clear faces of the intruders, so the images are separated into two parts, in and out of the region of interest (ROI). In ROI, AMEA can maintain the coding quality using the least numbers of manipulations, and meantime it can also enormously reduce the computational complexity for the blocks out of ROI.

1.2.4. Real-Time Driving Assist and Surveillance System

The environment inside vehicles has many restrictions, such as limited installation space, finite power supply when cars is parked and the noise effects due to the vibrations, so the stable and low power consumption embedded platform based on TI OMAP5912 is chosen to implement a dual-core DSP-based H.264 encoder (DDBE) with AMEA. DDBE combines an ARM9 RISC and a DSP core [12]. On one hand, ARM can perform the system control functions, such as image capturing, file transformation, user interface, process scheduler, and etc. On the other hand, DSP focuses on the H.264 algorithm, which requires more computing power than the peripheral controllers. Two kinds of optimization techniques, platform-independent algorithmic and memory optimizations, are applied to reduce the computational complexity. The advantage of DDBE is that it can execute the H.264 algorithms, and record the bitstream into a file or transmit them through the internet by the streaming server on ARM at the same time.

DDBE should be applied in the intelligent vehicles since thousands of people are killed or injured in the accidents which are mostly caused by the carelessness of humans. Therefore, a real-time driving assist and surveillance system (DASS) is integrated with a warning system, which can notify the driver of the approaching vehicles in all directions and the lane departure, and DDBE [13-16]. The driving status, such as the situation around the vehicle and the drivers' behaviors, can be recorded by DASS, so the police officers can reconstruct the scene easily. Furthermore, the increasing numbers of stolen cars is also an important issue so DASS can transmit the monitoring image inside the vehicle to users' mobile phones

to prevent thieves [17-20]. The goal of DASS is to provide a total solution which integrates the innovative functions in an embedded system to offer the users full-time protection, and DASS has been successfully implemented and fully tested by the real road environment in Taiwan.

1.3 Organization

The rest of this dissertation is organized as follows. Chapter 2 presents the efficient hierarchical motion estimation algorithm and its VLSI architecture. The register-based platform-independent MPEG-4 co-processor and its system-level design will be proposed in Chapter 3. In Chapter 4, the adaptive motion estimation algorithm for vehicle surveillance videos will be addressed. The real-time driving assist and surveillance system will be described in Chapter 5. Finally, some conclusions and future research perspectives will be stated in Chapter 6.

Chapter 2. Efficient Hierarchical Motion Estimation Algorithm and Its VLSI Architecture

This chapter addresses the development and hardware implementation HMEA using multi-resolution frames to reduce the computational complexity. Excellent estimation performance is ensured using an averaging filter to downsample the original image. At the smallest resolution, the least two MV candidates are selected using FSBMA. At the middle level, these two candidate MVs are employed as the center points for small range local searches. Then, at the original resolution, the final MV is obtained by performing a local search around the single candidate from the middle level. HMEA exhibits regular data flow and is suitable for hardware implementation. An efficient VLSI architecture that includes an averaging filter to down-sample the image and two 2D semi-systolic PE arrays to determine the SAD in pipeline is also presented. Simulation results indicate that HMEA is more area-efficient and faster than many full-search and multi-resolution architectures while maintaining high video quality. This architecture with 59K gates and 1,393 bytes of RAM is implemented for a search range of $[-16.0, +15.5]$.

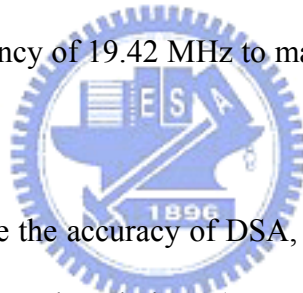
2.1. Introduction

The most complex part of popular video compression standards, including MPEG-4, MPEG-2, and MPEG-1, is ME [1][3][4]. The goal of ME is to remove the

temporal redundancies existing in adjacent frames, and the block-matching algorithm is used as a method for most of the video coding systems. It is used to find a block which is most similar to a current block within a pre-defined search area in a reference frame, and it dominates the encoded image quality, the compression ratio, and the computation time. The reference frame is a previously-encoded frame from the sequence and is before the current frame in the display order. The straight forward method to perform the operation is FSBMA, but it requires lots of manipulations due to its high complexity. Usually, FSBMA spends about 70% of the total encoding time, and this heavy computational load limits the performance of the encoder in terms of encoding speed and power consumption. Therefore, many VLSI architectures for FSBMA have been proposed for fast hardware implementation [21]-[29]. In these architectures, a result is observed that although FSBMA is easy to be implemented and can provide better compression quality, it has either large chip area or low speed. Traditionally, frameworks of FSBMA are block-level pipelined, where one reference block is considered at a time and the parameters are reset before starting another reference block. Compared with them, the frame-level pipelined FSBMA implementations can achieve nearly 100% fully pipelined computation by exploiting the explicit frame-level parallelism [30]. He et al. proposed a new two-level nested Do-loop FSBMA and a novel 2D array ME architecture [31]. However, its PE array size is fixed to N^2 , and will limit the capability. Therefore, they extend their design [32], and develop a scalable improved frame-level pipelined architecture, which reduce the internal FIFOs and increase the speed of [31]. It contains 1,024 PEs and can manipulate an MV in 256 cycles within a search range of [-16, +15].

To reduce the number of search steps of FSBMA in order to increase the overall speed is essential. The fast FSBMA, including the successive elimination algorithm

(SEA) [33]-[35], partial distortion elimination (PDE) [36], the winner-update algorithm [37], and the advanced diamond search algorithm (DSA) [38] are proposed to reduce the computational heavy load of FSBMA while maintaining its quality. However, the irregular data flow makes these algorithms suitable only for software implementation owing to their inability to determine exactly how many of SAD operations are required to calculate a single MV. Huang et al. proposed a new block matching algorithm called the global elimination algorithm (GEA), which is modified from SEA [39][40]. GEA has a more regular data flow than SEA. Moreover, the processing cycles are fixed, no initial guess is needed, and the conditional branch that applies when a candidate block cannot satisfy the criterion for early termination is removed. Although GEA is easily implemented and capable of providing good quality, it requires an operating frequency of 19.42 MHz to manipulate the MVs of CIF image in real-time.



Besides, in order to refine the accuracy of DSA, several new algorithms, such as motion vector field adaptive search technique (MVFAST) [41], predictive MVFAST (PMVFAST) [42], and enhanced predictive zonal search (EPZS) are proposed [43]. MVFAST improve DSA in both terms of visual quality and speed up by initially considering a small set of predictors. Unlike DSA where only a large moving diamond pattern was considered, MVFAST also introduced a smaller moving diamond. PMVFAST uses basically the same architecture and patterns as MVFAST does, but a significant difference of PMVFAST compared to MVFAST is the way the small versus the large diamond is selected. Dissimilar to MVFAST where motion was characterized as low, medium, or high by considering the largest motion vector candidate, in PMVFAST a different selection strategy, which can improve the overall speed of the algorithm by using the large diamond less often, is used. Furthermore,

EPZS that improves upon PMVFAST by considering several other additional predictors in the generalized predictor selection phase of PMVFAST. EPZS also selects a more robust and efficient adaptive threshold calculation where as, due to the high efficiency of prediction stage, the pattern of the search can be considerably simplified. However, the disorderly early termination of the search procedure still leads to the poor performance. An architecture, which combines PMVFAST and EPZS, is developed, and it can be configured to support different search patterns, and independent SAD computations [44]. The implementation results show that it requires 1,042 cycles to manipulate an MV, and it does not entirely complete the PMVFAST and EPZS due to their high complexity.

Another ME algorithm that can significantly reduce the computational complexity by decreasing the number of computations is the hierarchical motion vector search algorithms (HMVSA), including three-step search (3SS) [45], new three-step search [46], and four-step search [47], which separate the estimation process into several levels, and the numbers of levels is fixed. HMVSA has regular data flow, and the total execution time is constant, so HMVSA is suitable for hardware implementation. However, HMVSA suffers from a considerably lower PSNR than FSBMA, especially when the motion field is large and complex.

A particular HMVSA is developed to solve this problem, MMEA, whose basic idea is to make an initial coarse estimate and then refine it. Conventional MMEAs are usually implemented in two ways. One is to use a variable search area at each level [48]-[51], and the other is to apply a constant search area [52]-[54]. In the former, an MV candidate is obtained from a large search area at the coarse level and the candidate becomes the search center of the next level, which has a smaller search area.

A larger search area corresponds to a more accurate MV, but the extent of motion may increase with the search area. Therefore, the first MV candidate may not be a good estimate, and will yield an incorrect result at the next levels. Although the latter approach can partially solve this problem since the search area is constant at all levels, the MVs may be less robust against noise.

The above MMEAs that choose only one MV candidate fall easily into the local minimum, so numerous algorithms that combine the scheme with a multiple MV candidate search have been proposed [55]-[59]. However, these methods have a high computational cost to get the prediction performance close to that of FSBMA, because multiple MV candidates are required for local searches at each level. In these algorithms, the method for down-sampling the image is to select one of four pixels in a block. This method may be inappropriate if the block is the edge of a video object, and will influence the image quality, in terms of PSNR of the image. Accordingly, more MV candidates are required to yield an encoded image quality close to that of FSBMA. If the MV candidates are not only chosen by the basis of minimum SAD, such as by the neighborhood relaxation scheme in [58] or the four candidates, which correspond to four differently superblocks[51], the complexity will be increased.

Many hardware architectures for MMEA have been implemented [49], [60]-[63]. In [49], the framework is at the expense of a chip area because the on-chip memory is large. Each multi-resolution level in [60] and [61] has its own specific systolic array, which cannot commonly be applied among different levels, reducing the performance in terms of logic gate usage. [62] has a small chip area, but the reuse of data and the SAD computations are inefficient. Therefore, the overall speed is reduced, and it will limit its applications that require low operating frequency to save the power

consumption such as mobile phones and portable multimedia recorders. In [63], although the reuse of data is efficient, a large on-chip memory is required.

In this chapter, HMEA and its VLSI architecture are proposed. The main contributions of this chapter are to analyze several downsampling methods, discuss which method is suitable for hardware implementation, and derive a high speed pipeline VLSI architecture. HMEA adopts an averaging filter to downsample the original image, which is the first step of the estimation process. In these hardware frameworks for MMEA [49], [60]-[63], the downsampling methods are not addressed due to the hardware cost. However, when superior quality is obtained in the anterior part of the ME, the refining procedures can be significantly shorter and the complexity can be reduced accordingly. HMEA can achieve almost the same coding performance as FSBMA in terms of PSNR, but HMEA is faster. The MV will be more credible and the search speed is higher as the search area increases. Furthermore, the proposed HMEA limits the number of MV candidates to two at the coarse level, and sets the total number of levels to three to solve the significant problem of the local minimum. An averaging filter is employed, so a single candidate at the final level suffices to provide the desired performance.

A high-speed pipeline VLSI architecture with a reasonable chip area for HMEA is also addressed. It utilizes an efficient 2D PE array to compute the SADs, and the search range can be doubled without adding any hardware. The architecture is suitable for VLSI implementation because the number of the computations for each macro block (MB) is fixed. HMEA is faster and more area-efficient than numerous existing full-search and multi-resolution architectures.

2.2. Hierarchical Motion Estimation Algorithm

HMEA can be divided into two parts. One is the averaging filter for down-sampling, and the other is the MV search procedure. The complete algorithm is described as below.

2.2.1. Downsampling Methods

HMEA comprises three resolution levels, from zero to two. Level 0 is the top level, and level 2 is the lowest. The number of pixels at the next lower level is reduced to one quarter the number at the upper level. Figure 2-1 shows the hierarchical frame structure, and the W and H are the width and the height of the image, respectively. The MB size changes from 16×16 , through 8×8 , to 4×4 at levels 2, 1 and 0, respectively.

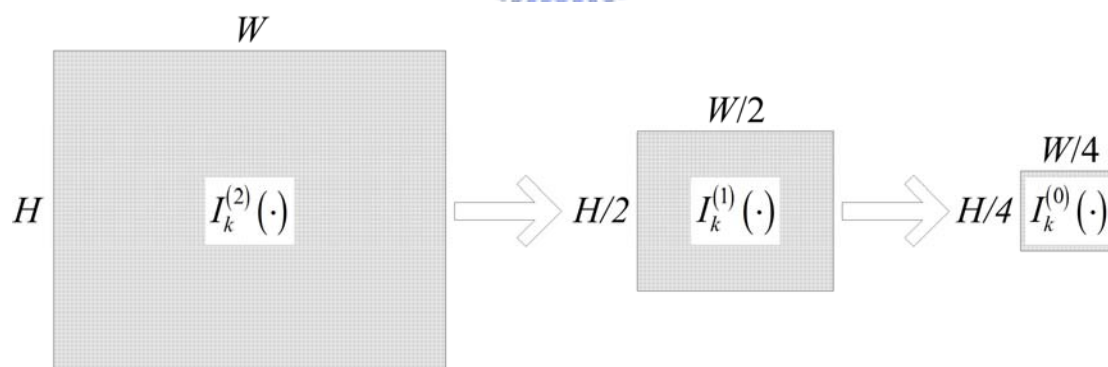


Fig. 2-1. The hierarchical frame structure

In block matching algorithm, SAD is an important procedure, and its value at level l can be defined as

$$SAD_{MB}^{(l)}(p, q) = \sum_{i=0}^{\lceil 16/2^{(2-l)} \rceil - 1} \sum_{j=0}^{\lceil 16/2^{(2-l)} \rceil - 1} \left| I_k^{(l)}(i, j) - I_{k-1}^{(l)}(i+p, j+q) \right|, \quad (2-1)$$

where $I_k^{(l)}$ is the k^{th} input image, and l is the level number and $l = 0, 1, 2$.

In (1), the computational complexity of the matching process can be seriously reduced. At level 1, the computational complexity is only one quarter than that at level 2, and that at level 0 is one quarter than that at level 1.

Numerous approaches are available to reduce an image. In this paper, three different methods, left-top, 2D discrete wavelet transform (2D-DWT), and averaging filter are adopted. The comparison of these methods in computational complexity, performance and hardware implementation are discussed. The bicubic interpolation that can provide better performance is not in consideration since its complexity is much more than the other methods. Besides, when it comes to reducing the image with 50% in both width and height, the quality is not as better as it of enlarging the frame.

A. Left-top method

The left-top method is one of the simplest approaches for subsampling an image. For the k^{th} input frame, $I_k^{(2)}(\cdot)$, the upper level images are computed by executing the following down-sampling:

$$I_k^{(l-1)}(i, j) = I_k^{(l)}(2i, 2j), \text{ for } l = 1, 2, \quad (2-2)$$

where $I_k^{(l-1)}(i, j)$ represents the gray level value at the position (i, j) of the k^{th}

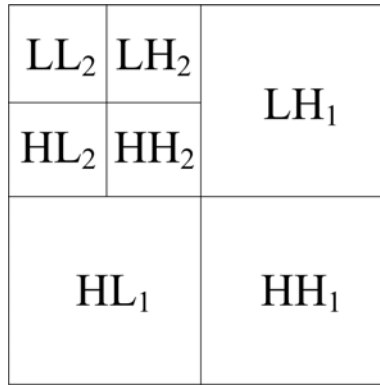
frame at level $l - 1$.

In the hardware implementation, the arithmetic operations are not necessary, and the output image can be generated by inputting the original one by specific order directly. The only cycles required are for moving data, and no extra hardware design is essential.

B. 2D-DWT

In image processing, most of the power associated with natural image signals tends to be in the low frequency band. Accordingly, the analysis of the low frequency band must be more extensive than that of the high frequency band. In practical applications, the low frequency band, decomposed from DWT, is further analyzed through second level DWT processing to yield more detail of the analysis signal at the lower frequency band. Such analysis is referred to as multi-resolution. Haar's and Antonini 9/7 Wavelet Transform is used to increase the speed of execution of the wavelet transform [64]. The 2D-DWT is applied as a one-dimensional DWT in the horizontal direction and then another in the vertical direction.

Figure 2-2(a) plots the corresponding locations of the images of the frequency bands decomposed by 2D-DWT. Fig. 2-2(c) and Fig. 2-2(d) shows the subsampled results obtained using the 'Akiyo' image, displayed in Fig. 2-2(b), after two levels of DWT processing. Fig. 2-2(c) and Fig. 2-2(d) truncate the values that are above 255 and below 0 for demonstration, but the value are retained in the evaluation progress. As shown in Fig. 2-3, for the k^{th} frame, $I_k^{(1)}(i, j)$ and $I_k^{(0)}(i, j)$ are the LL band of the first and the second order decomposition, respectively.



(a)

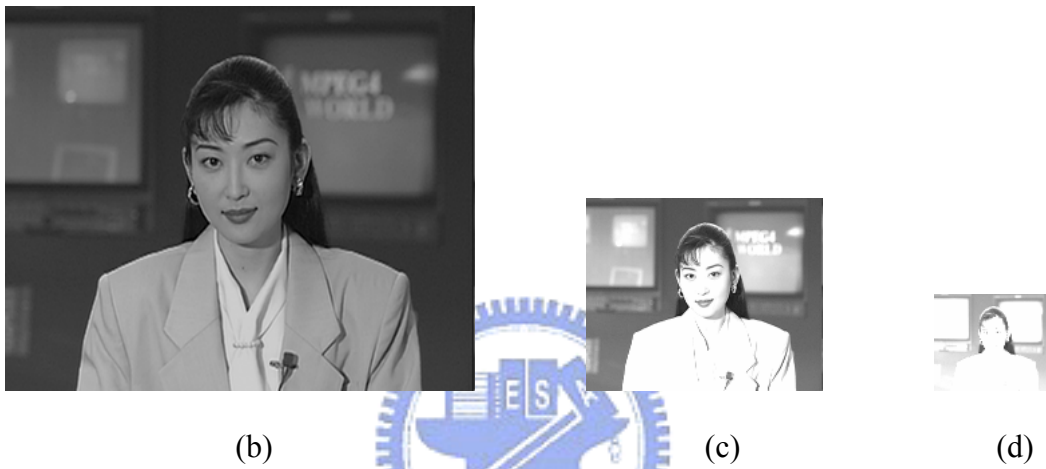


Fig. 2-2. Examples of 2D-DWT downsampling: (a) frequency bands after two-level DWT decomposition; (b) original Akiyo image; (c) reduce the Akiyo image 50% in both width and height; (d) reduce the Akiyo image 25% in both width and height

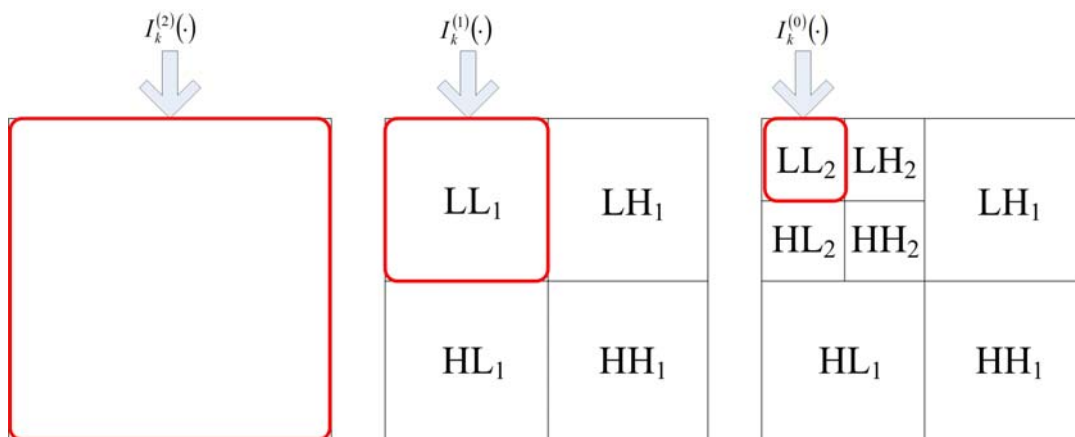


Fig. 2-3. The relationship between the 2D-DWT and the downsampled images

There are some problems existing in this downsampling method. Firstly, the

computational complexity is heavy because it requires more additions and more multiplications for subsampling one pixel. Secondly, in the normal hardware design, 8 bits are required to store the gray level value of the pixel from 0 to 255. However, the range of the downsampling pixel by 2D-DWT goes beyond 0 to 255, and more bits are necessary. Therefore, the memory bandwidth of the hardware architecture and the chip area will be increased. Although the range of the pixels can be normalized into 0 to 255 to reduce the bandwidth, the extra hardware for normalization is essential. The computational complexity and the die size will be also increased.

C. Averaging filter

This method is the same as the bilinear interpolation that rescale the image with 50% in both width and height. Therefore, the quality of the reduced image can be ensured. For the k^{th} input frame, $I_k^{(2)}(\cdot)$, the upper level images are computed by executing the following down-sampling:

$$I_k^{(l-1)}(i, j) = \frac{1}{4} \times \sum_{m=2i-1}^{2i+1} \sum_{n=2j-1}^{2j+1} I_k^{(l)}(m, n), \text{ for } l = 1, 2, \quad (2-3)$$

where $I_k^{(l-1)}(i, j)$ represents the value at position (i, j) of the k^{th} frame at level $l - 1$.

The hardware implementation for the averaging filter is simple, and only three additions and one bit shift operations are required for subsampling one pixel.

D. Comparisons

The main purpose of the ME is to eliminate the temporal redundancies existing in adjacent frames. Therefore, the quality will be increased when higher correlation coefficient exists between the successive images. The correlation coefficient, ρ , is defined as:

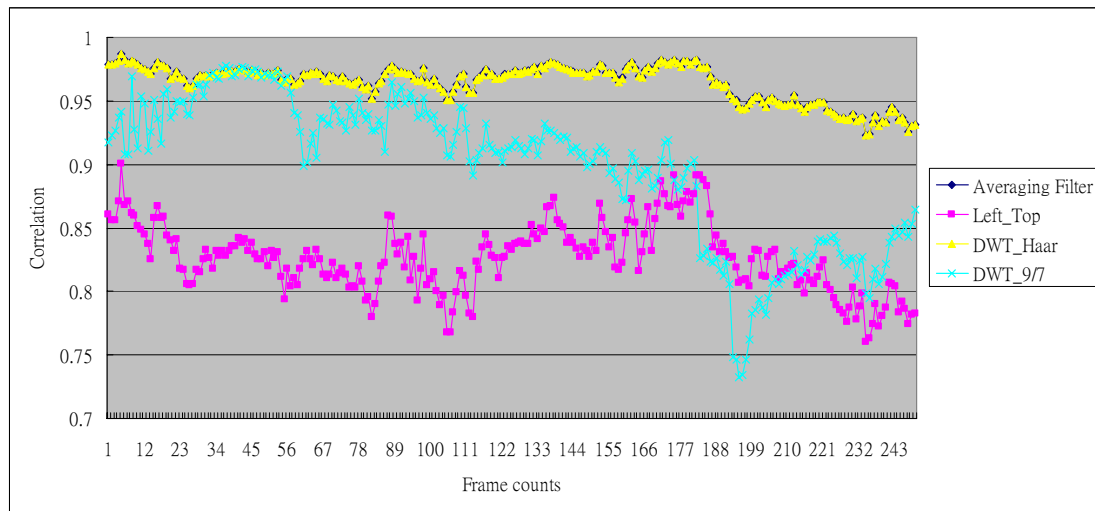
$$\rho = \frac{\sum_i \sum_j [(I_k^{(0)}(i, j) - m_1) \times (I_{k+1}^{(0)}(i, j) - m_2)]}{\sqrt{\sum_i \sum_j (I_k^{(0)}(i, j) - m_1)^2} \times \sqrt{\sum_i \sum_j (I_{k+1}^{(0)}(i, j) - m_2)^2}}, \quad (2-4)$$

where m_1 and m_2 are the means of $I_k^{(0)}(\cdot)$ and $I_{k+1}^{(0)}(\cdot)$, respectively.

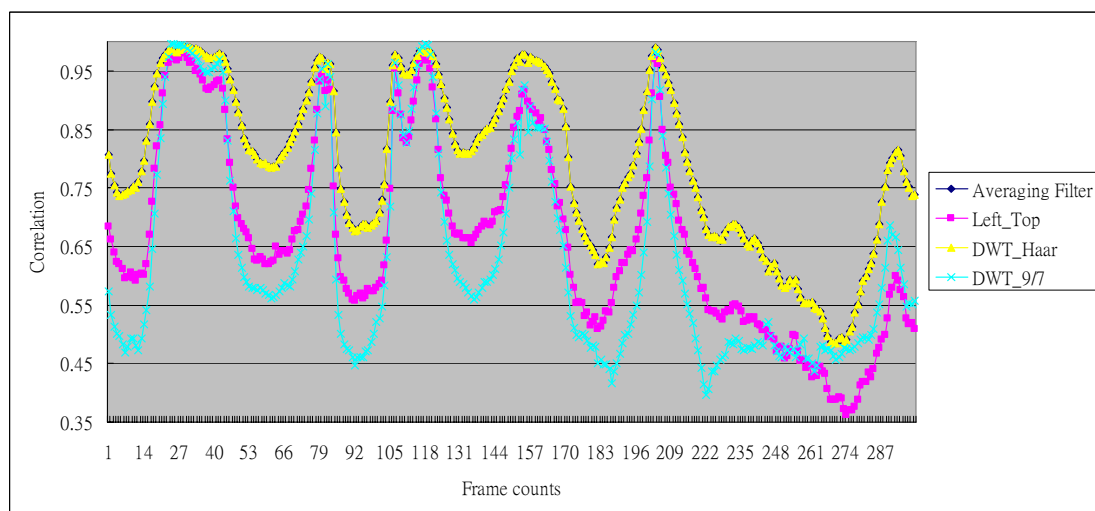
Figure 2-4 shows that the correlation coefficients between the downsampled images. It is observed that the correlation coefficients of the Haar DWT and the averaging filter are almost the same, and are much greater than the left-top and the Antonini 9/7 DWT methods, especially in the video sequences that their backgrounds are more complex. In Table 2-1, the estimation results are depicted, and it shows that the average PSNR of Haar DWT and the averaging filter are similar. Moreover, the image quality of the left-top method is bad, and Antonini 9/7 DWT even gets worse quality than it in some cases. The results indicate that downsampling method plays a very important role in MMEA. The estimation performance of adopting averaging filter significantly exceeds that of the method that considers only the left-top pixel, and can be used to design an efficient down-sampling hardware architecture.

Table 2-1 The comparison of the video quality between various downsampling methods for left-top, Haar's DWT, Atonini's 9/7 DWT, and the averaging filter in dB.

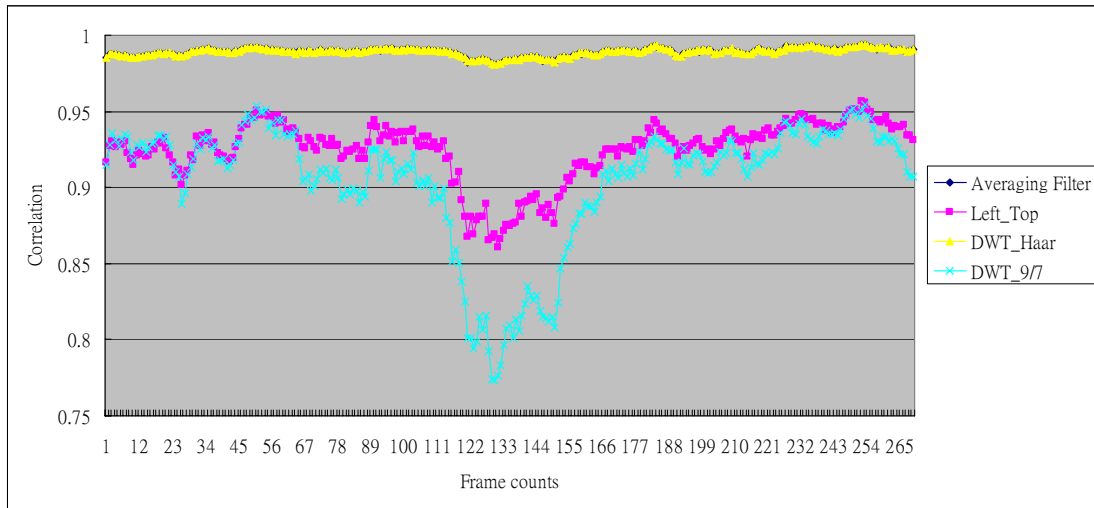
Video Sequence	Left-top	Antonini 9/7 DWT	Haar DWT	Averaging filter
	PSNR	PSNR	PSNR	PSNR
News	32.79	33.84	35.42	35.45
Flower garden	21.43	24.23	26.55	26.62
Foreman	30.53	28.74	33.14	33.18
Table tennis	31.02	32.17	33.05	33.07
Stefan	23.81	22.35	25.67	25.82
Mobile	23.07	21.42	24.49	24.53



(a)



(b)



(c)

Fig. 2-4. The correlation coefficients between the adjacent images for averaging filter, left_top method, Haar DWT, and Antonini 9/7 DWT in (a) Flower garden (b) Stefan (c) Mobile

The performance of the Antonini 9/7 DWT is worse than that of the Haar DWT is unexpected. Theoretically speaking, the Antonini 9/7 DWT can reserve more information in low frequency band since it adopts higher order filters. However, the statement stands only when the inverse transform, which is not executed in the downsampling procedures, is performed. Therefore, Antonini 9/7 DWT requires higher computational power, but it provides poor quality in the downsampling stage of HMEA. Moreover, if the scaling factor of Haar DWT is replaced by $1/2$, the results are exactly the same as the averaging filter, and can get rid of the dynamic range problem. The reason of the averaging filter outperforms the Haar DWT is that $1/\sqrt{2}$ is chosen as its scaling factor, and this will cause the inaccuracy of the values of downsampled pixels. Considering both the coding performance and the hardware design, the averaging filter is chosen to down-sample the image in HMEA.

2.2.2. Framework of HMEA

The overall searching process can be separated into three levels. As presented in Fig. 2-1, when level 2 receives an input image $I_k^{(2)}(\cdot)$, the image will be down-sampled to $I_k^{(1)}(\cdot)$ and $I_k^{(0)}(\cdot)$, where the resolutions of $I_k^{(1)}(\cdot)$ and $I_k^{(0)}(\cdot)$ are one quarter and one sixteenth of that of $I_k^{(2)}(\cdot)$, respectively. Let the entire search range at level 2, or $\Omega^{(2)}$, be $[-w, w-1]$. After the original image $I_k^{(2)}(\cdot)$ has been down-sampled, the search procedure, illustrated in Fig. 2-5, begins. Let $Cur^{(l)}$, $Pre^{(l)}$, $SA_k^{(l)}$ and $MV_n^{(l)}$ denote the current MB, the previous frame search area, the k^{th} search area and the n -th MV candidate at level l , respectively. The detailed process at each level will be described below.



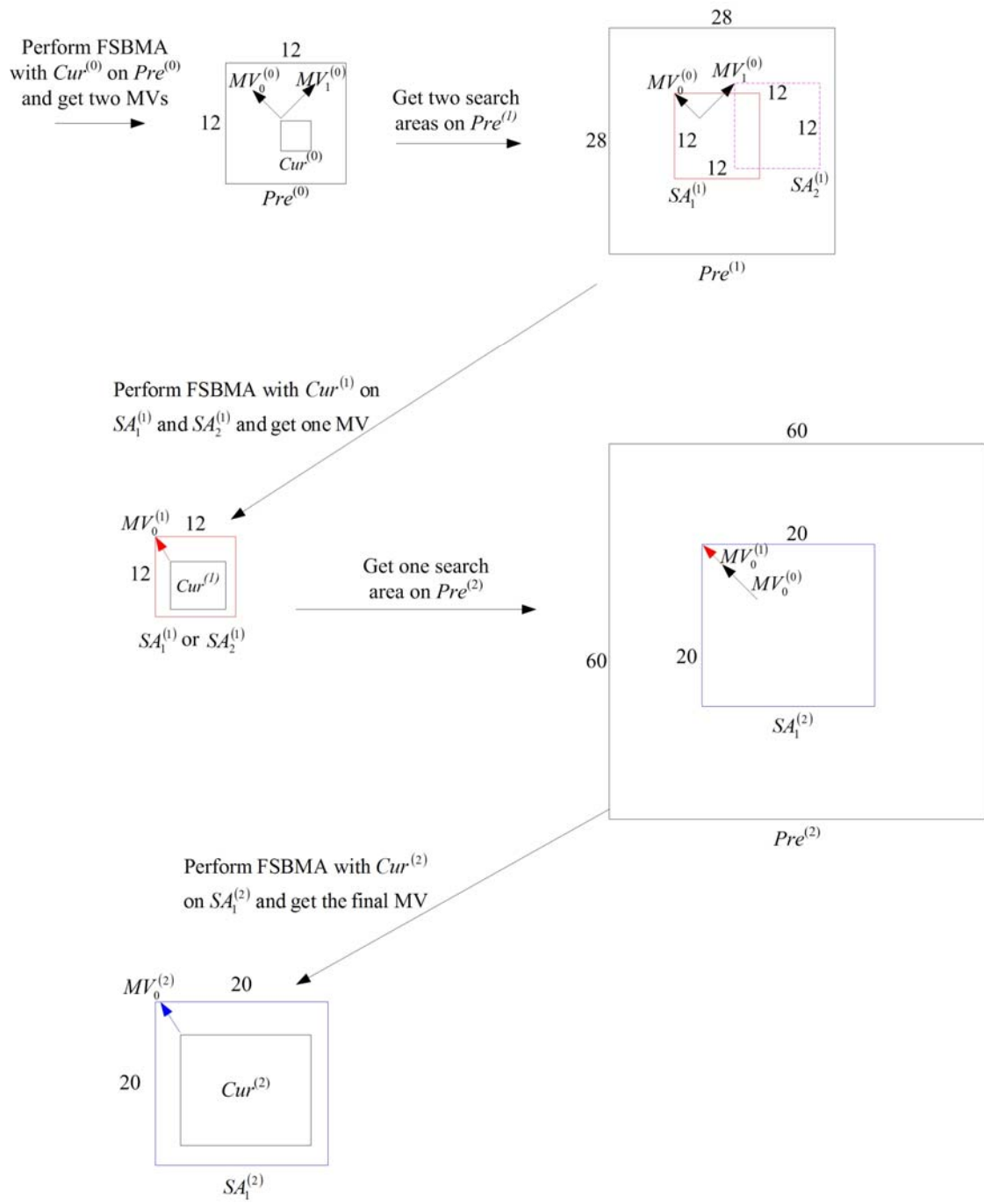


Fig. 2-5. The hierarchical search procedure.

A. Search at Level 0

Two MV candidates, $MV_1^{(0)}$ and $MV_2^{(0)}$, are manipulated at this level. $MV_1^{(0)}$ is defined as the least MV that obtained by a full search within a given search area, i.e.

$$MV_1^{(0)} = \min_{(p,q) \in \Omega^{(0)}} SAD_{MB}^{(0)}(p,q), \quad (2-5)$$

and $MV_2^{(0)}$ is the second least MV, i.e. $MV_1^{(0)} \leq MV_2^{(0)}$ over the area

$$\Omega^{(0)} = \{(p,q) | -w/4 \leq p \leq w/4, -w/4 \leq q \leq w/4\}.$$

According to Fig. 2-5, $Cur^{(0)}$ is a 4×4 block, and $Pre^{(0)}$ is a 12×12 previous frame search area. In level 0, the search range is only one quarter that at level 2, and the number of operations is much lower. The experimental results presented in Table 2-2 clearly reveal that if three least minimum MV candidates are selected, the performance only increases slightly. Hence, only 2 MV candidates are chosen. Accordingly, the computational complexity can be reduced, and the performance maintained.

Table 2-2 The PSNR performance and the complexity analysis in different number of the MV candidates at the coarsest level.

Video Sequence	1 candidate		2 candidates		3 candidates	
	PSNR	Complexity	PSNR	Complexity	PSNR	Complexity
News	34.96	0.85	35.45	1	35.57	1.15
Flower garden	25.41	0.85	26.62	1	26.83	1.15
Foreman	32.95	0.85	33.18	1	33.30	1.15
Table tennis	32.31	0.85	33.07	1	33.21	1.15
Stefan	25.01	0.85	25.82	1	26.02	1.15
Mobile	23.88	0.85	24.53	1	24.87	1.15

B. Search at Level 1

Local searches are performed around these two MV candidates from level 0. The local search range (LSR) is also an important issue for HMEA, and it is smaller than that at level 0 to reduce the number of operations. If LSR is too small, the estimation results may be incorrect when the movement of the MB is big. On the other hand, if

LSR is too large, the computational complexity will be increased greatly. Furthermore, the hardware design must be considerate. If LSR in the different levels are not the same, two hardware blocks are required, and the chip area will be increased. Thus, LSRs are the same in level 1 and level 2 of HMEA. Table 2-3 shows that the relationship between the LSR and the estimation performance. Although the PSNR increases when LSR extends, the complexity is also raised. The improvement of the PSNR for LSR from 2 to 3 is not as great as it from 1 to 2. Based on these test results, LSR equals to 2 is chosen.

Table 2-3 The PSNR performance and the complexity analysis in different LSRs in the middle and the finest level.

Video Sequence	LSR = 1		LSR = 2		LSR = 3	
	PSNR	Complexity	PSNR	Complexity	PSNR	Complexity
News	34.11	0.42	35.45	1	35.49	1.87
Flower garden	25.19	0.42	26.62	1	26.81	1.87
Foreman	31.86	0.42	33.18	1	33.26	1.87
Table tennis	31.71	0.42	33.07	1	33.15	1.87
Stefan	24.06	0.42	25.82	1	26.01	1.87
Mobile	23.12	0.42	24.53	1	24.82	1.87

In Fig. 2-5, $Cur^{(1)}$ is an 8×8 block, and $Pre^{(1)}$ is a 12×12 previous frame search area. $Cur^{(1)}$ performs two full local searches, whose LSR equals 2, over two search areas $\Omega_n^{(1)}$, $n=1, 2$, to refine the search results from level 0, and $(p_n^{(0)}, q_n^{(0)})$, denote the corresponding MV candidate from level 0. Following the local searches, an MV candidate, $MV_0^{(1)}$, can be determined,

$$MV_0^{(1)} = \min_{(p,q) \in \Omega^{(1)}} SAD_{MB}^{(1)}(p, q), \quad (2-6)$$

where $\Omega^{(1)} = \Omega_1^{(1)} \cup \Omega_2^{(1)}$, $\Omega_n^{(1)} = \{(p, q) | -2 \leq (p - 2 \cdot p_n^{(0)}) \leq 2, -2 \leq (q - 2 \cdot q_n^{(0)}) \leq 2\}$.

C. Search at Level 2

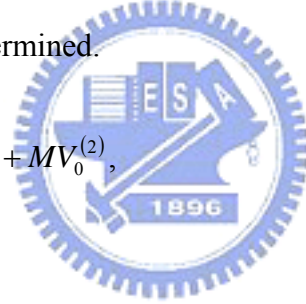
$MV_0^{(2)}$ is found by a local search around the MV candidates from level 1.

$$MV_0^{(2)} = \min_{(p,q) \in \Omega^{(2)}} SAD_{MB}^{(2)}(p, q), \quad (2-7)$$

where $\Omega^{(2)} = \{(p, q) | -2 \leq (p - 2 \cdot p^{(1)}) \leq 2, -2 \leq (q - 2 \cdot q^{(1)}) \leq 2\}$
 $(p^{(1)}, q^{(1)}) = MV_0^{(1)}$

As illustrated in Fig. 2-5, $Cur^{(2)}$ is a 16×16 block, and $Pre^{(2)}$ is a 20×20 previous frame search area. The MV searching process is completed when MV_{MB} , defined in (2-8), has been determined.

$$MV_{MB} = 4 \times MV_i^{(0)} + 2 \times MV_0^{(1)} + MV_0^{(2)}, \quad (2-8)$$



D. Half-Pel Search

After MV_{MB} is manipulated, the half-pel search is started. Therefore, the neighboring half accuracy pixels of the MV_{MB} have to be calculated, and a total of 833 pixels and 8 SADs are necessary. The complexity of the half-pel search, C_{half} is defined as (2-9), and it is combined with the pixels and the SAD operations.

$$C_{half} = (833 \times M + 8 \times N) \times \frac{W \times H}{16^2} \times R_f, \quad (2-9)$$

where M , N and R_f are the number of operations required to compute a half accuracy pixel, the SAD operation, and the frame rate, respectively. Fortunately, the

downsampling stage of HMEA has already calculated 144 pixels for half-pel search so the complexity of half-pel search for HMEA, C_{half_HMEA} , can be reduced as (2-10).

$$C_{half_HMEA} = (689 \times M + 8 \times N) \times \frac{W \times H}{16^2} \times R_f, \quad (2-10)$$

E. Complexity Analysis

The overall search procedure includes the downsampling stage, $C_{downsample}$, the integer-pel search, and the half-pel search, and $C_{downsample}$ is defined as (2-11). The half accuracy pixels only need one addition to manipulate where the pre-processed pixels for HMEA requires three of them, and the shift operation can be reduced by reading the higher bits of the pixel. Therefore, the cycles for downsampling a pixel are three times to them for the half accuracy pixels. During the overall search procedure, the search complexity is described as (2-12):

$$C_{downsample} = \left[\left(\frac{W \times H}{2^2} + \frac{W \times H}{4^2} \right) \times 3 \times M \right], \quad (2-11)$$

$$\begin{aligned} C_{HMEA} &= C_{downsample} + C^{(0)} + C^{(1)} + C^{(2)} + C_{half_HMEA} \\ &= \left\{ \left(\frac{W}{2} + 1 \right)^2 \times \left(\frac{1}{2^2} \right)^2 + 2 \times 5^2 \left(\frac{1}{2^1} \right)^2 + 5^2 \times \left(\frac{1}{2^0} \right)^2 \right\} \times N \times 16^2 \times \frac{W \times H}{16^2} \times R_f, \quad (2-12) \\ &\quad + C_{downsample} + C_{half_HMEA} \end{aligned}$$

where $C^{(l)}$ represents the search complexity in level l . In the case of FSBMA, computational complexity is given by (2-13).

$$C_{FSBMA} = (2w + 1)^2 \times N \times 16^2 \times \frac{W \times H}{16^2} \times R_f + C_{half}, \quad (2-13)$$

The SAD operation for a pixel, which is described in (2-1), needs 256 additions, and 256 subtractions, and the manipulation for a half accuracy pixel only requires one additions. Therefore, the relationship between M , and N can be illustrated as (2-14).

$$M = \frac{N}{256 + 256}, \quad (2-14)$$

From the equations (2-9) to (2-13), they demonstrate that the computational complexity of HMEA will be only 3.9% and 1.3% of that of it of FSBMA for w of 16 and 32, respectively.

2.2.3. Experimental Results

The MPEG test video sequences: “News,” “Foreman,” “Flower garden,” “Table tennis,” “Stefan,” and “Mobile” are used to evaluate the performance of HMEA.

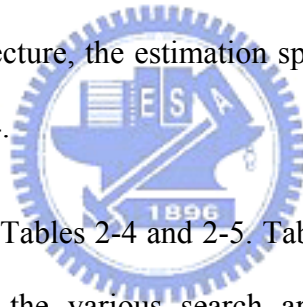
All the sequences consist of 300 frames; the frame rate is 30 frames per second (FPS), and the image size is CIF. The search range is defined as $[-w, w-1]$ where $w=16$. The PSNR is used for the measurement of performance, and the PSNR is defined as (2-15).

$$PSNR = 10 \log_{10} \frac{255^2}{\frac{1}{W \cdot H} \sum_{i=0}^{H-1} \sum_{j=0}^{W-1} [I_k(i, j) - \hat{I}_k(i, j)]^2}, \quad (2-15)$$

where $\hat{I}_k(\cdot)$ is the k^{th} motion compensated image, respectively.

The performance of HMEA is compared to that of two well-known algorithms:

FSBMA and n -step search (n SS) [65], and two MMEA algorithms, MRMC-m [59] and MRMCS [62]. n SS is a general version of the 3SS to cover the increased search ranges ($n = 3, 4, 5$ for $w = 8, 16, 32$, respectively). MRMC-m is a MMEA based on multiple candidates, and it has m -candidates at each resolution. MRMCS uses three MV candidates at level 1, and two of the MV candidates that are obtained on the basis of minimum matching error at level 0, and the other one is based on the spatial MV correlation. MRMC-m and MRMCS are both using left-top method for downsampling the images, and they also keep multiple winners at the top level. All algorithms are implemented by the standard C language to estimate the PSNR performance, and the input sequences are all the same, which can avoid the distinct quality due to the different fragment in the same test patterns. Since the software model can not represent the hardware architecture, the estimation speed and the memory usage will be discussed in the Section 2.4.



The results are shown in Tables 2-4 and 2-5. Table 2-4 describes the complexity of these four algorithms in the various search area, and Table 2-5 shows the performance in terms of PSNR. In order to truly reflect the MV accuracy, the estimation results are made by the MB and the corresponding MV_{MB} without the inflation of the error residuals. According to these tables, HMEA provides a prospective PSNR performance that is close to that of FSBMA, and a greater search range corresponds to a lower complexity. Although the averaging filter has higher computational complexity than the left-top method which MRMCS and MRMC-m adopted, the number of the MV candidates in level 1 of HMEA is less than these two MMEAs. Moreover, HMEA has not only lower SAD operations in the integer pixel search, but also smaller amount of pixel interpolations for the half-pel one. The reduced arithmetic can compensate the extra manipulations of the averaging filter.

Therefore, the overall complexity of HMEA is smaller than MRMCS and MRMC-m. In Table 2-4, *n*SS exhibits the lowest computational complexity with consistency that is proper for hardware implementation. However, it can be observed that *n*SS provides the lower PSNR especially for the sequences that have fast motion. Besides, although MRMC-m also needs a consistent computational complexity, it contributes the worse PSNR than MRMCS and HMEA for similar computational complexity. Meanwhile, the PSNR of HMEA is slightly less than MRMCS in the video sequences that contain high motions since MRMCS applies an MV candidate based on spatial correlation in an MV field. However, MRMCS needs many more cycles to manipulate the MV candidates. Based on the computational complexity results determined by the tests, HMEA is the most suitable algorithm for VLSI implementation.

Table 2-4 The comparison of the complexity, including half-pel search, between FSBMA, *n*SS, MRMC-4, MRMCS and HMEA in different search ranges.

Search Range	FSBMA	<i>n</i> SS	MRMC-4	MRMCS	HMEA
8	100	10.05	18.00	17.66	13.53
16	100	3.22	4.96	4.78	3.91
32	100	1.02	1.52	1.52	1.32

Table 2-5 The PSNR comparisons of various fast-search algorithms in dB.

Video Sequence	HMEA	FSBMA	4SS	MRMC-4	MRMCS
News	35.45	35.85	34.83	35.01	35.15
Flowergarden	26.62	27.22	26.39	26.57	26.71
Foreman	33.18	33.70	32.15	32.97	33.14
Table tennis	33.07	34.08	32.16	32.45	33.05
Stefan	25.82	26.43	25.13	25.41	25.98
Mobile	24.53	25.18	23.96	24.11	24.65

HMEA can reduce more SAD operations when the image resolution is larger from (2-12) and (2-13), and in SD and HD resolution, the reduced complexity is 3.4 and 9.1 times of that in CIF. Furthermore, the use of averaging filter improves the estimation accuracy of HMEA. This can be examined in Table 2-5 by comparing the

performance between MRMC-m and HMEA. By changing the downsampling method, HMEA can reduce the number of the MV candidates at level 1 and even obtain the better PSNR than MRMC-m. For video sequences having large motion, the effect becomes more noticeable. As a result, in HMEA, only one local search is performed at level 2, and two MV candidates are enough for maintaining the good quality. Therefore, the overall computational cost and data bandwidth burden of HMEA decrease.

2.3. Proposed VLSI Architecture

HMEA, described in Section 2.2, is mainly for low bit-rate video coding in MPEG-4. Therefore, a search range of $[-16.0, +15.5]$ is adopted. In this VLSI architecture, a 2D difference accumulation unit, DAU, is proposed for the VLSI architecture. Based on this 2D architecture, the image data can be input to the DAUs in pipeline, and the encoding speed can be greatly increased while maintaining a negligible degradation in the coding performance.

2.3.1. Overall Architecture

As stated in Section 2.2, HMEA comprises three levels, and the computation proceeds at each level with different MB sizes and search ranges. A basic computational component performs a 4×4 block FSBMA within the search range of $[-2, +2]$ at all levels. Therefore, a DAU that executes a ± 2 FSBMA for a 4×4 sub-block is introduced. Each DAU is divided into 5-stage pipelines, each of which

has five PEs, to compute the SADs.

Two DAUs are adopted to complete the process. Accordingly, the efficiency of data reuse can be markedly improved, and the encoding speed can be increased. Figure 2-6 shows the overall architecture. HMEA consists of two DAUs, an address generator, two comparators, 14 registers, and memory banks. As stated above, the block size and the search range are different at each level, and one DAU can compute an SAD of a 4×4 block. Therefore, when the computational proceeds to level 1 and level 2, SAD should be accumulated twice and eight times, respectively. Block 1 in Fig. 2-6 is the accumulator for level 1 and level 2. The advanced prediction mode should compare SADs of four 8×8 sub-blocks in a 16×16 MB, to predict an MV, and Block 2 is employed in this mode (8×8 prediction mode). The memory banks and the address generator provide a scheduled data flow to DAUs to calculate $SAD_{4 \times 4}^{(l)}$. The registers are used to delay the data input to fit the timing designed in Fig. 2-7, which is the timing diagram of level 0. The search window of the previous frame is divided into two parts, which are input to different DAUs. Thus, the image data can be reused more effectively, and selecting an MV takes only 56 cycles.

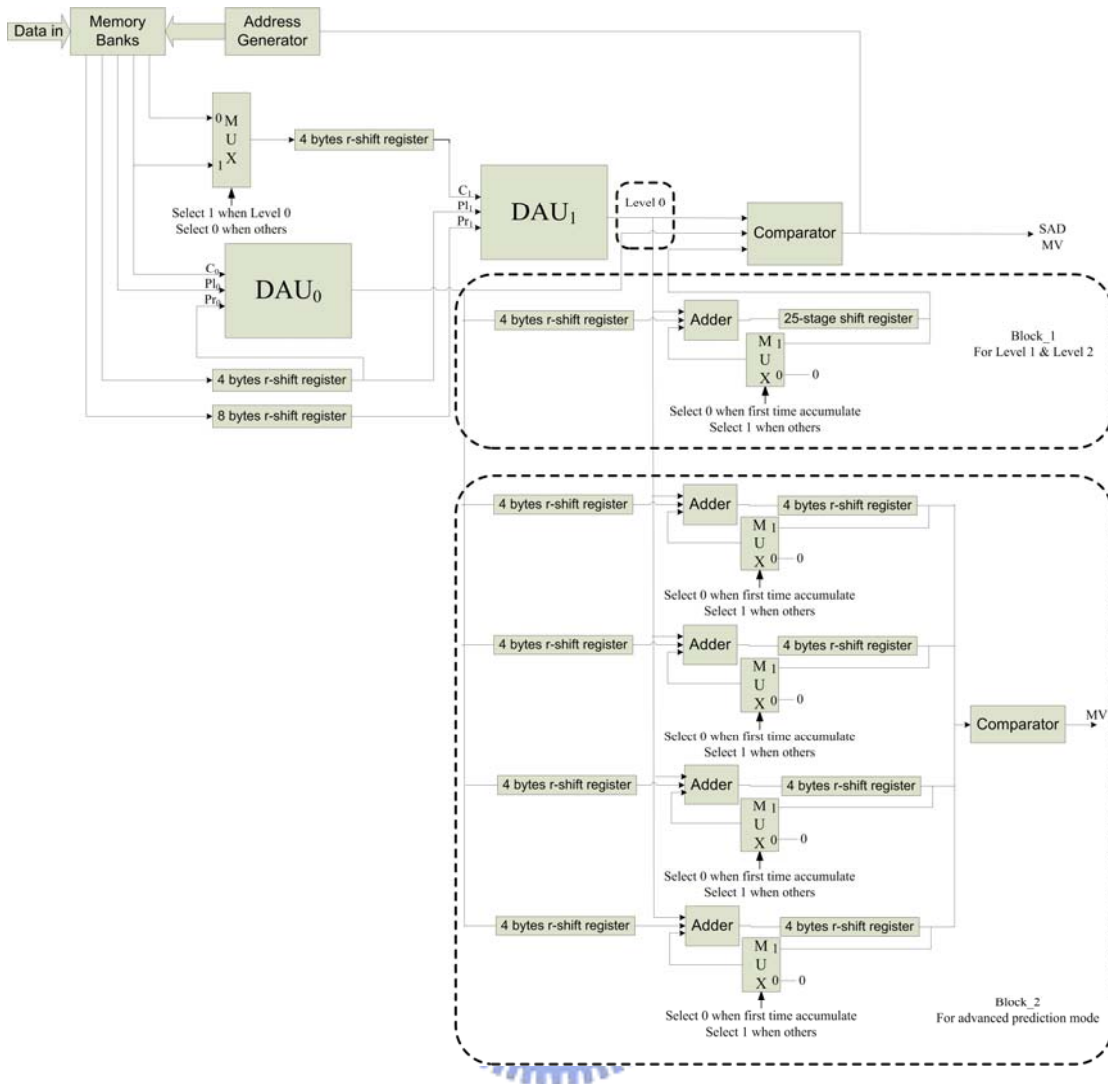


Fig. 2-6. The overall architecture of HMEA.

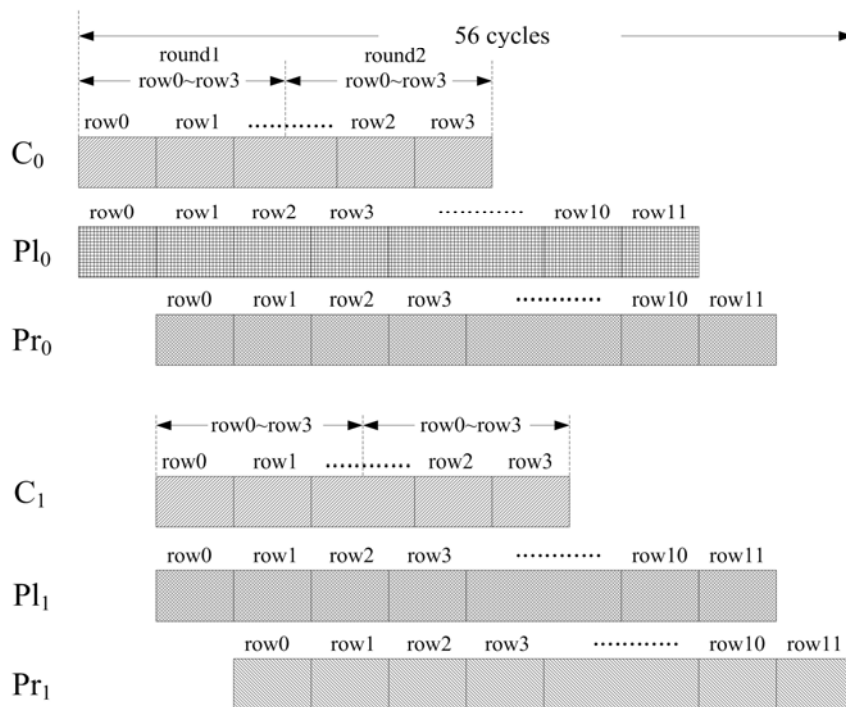
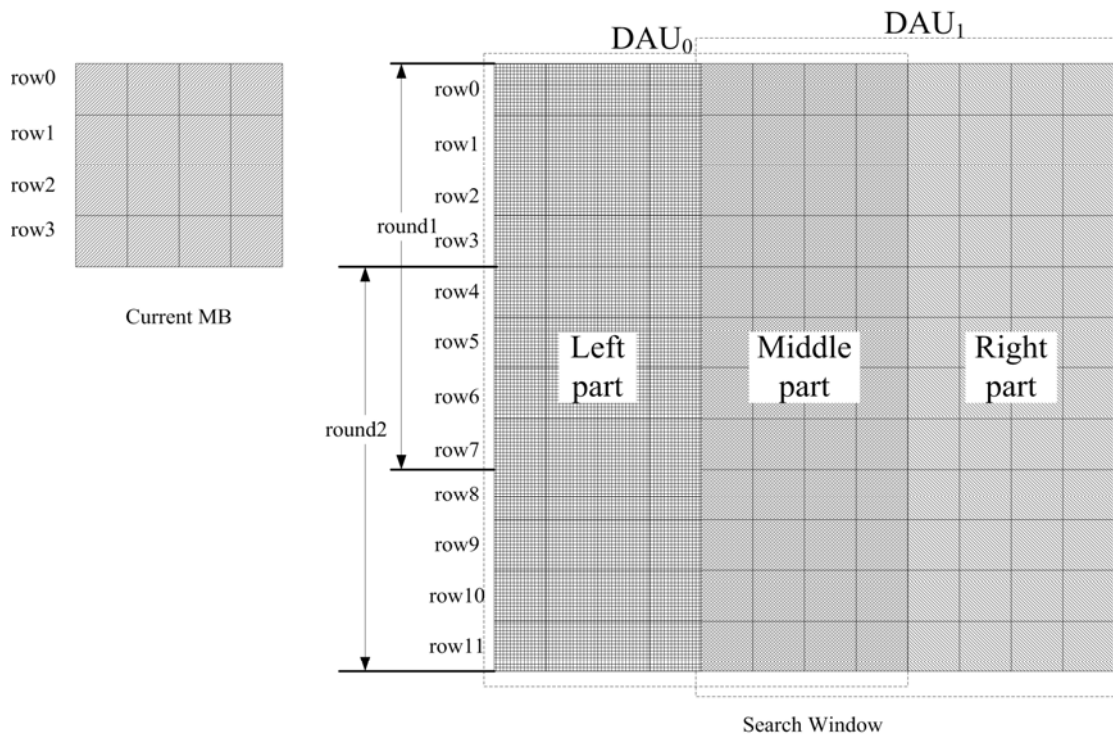


Fig. 2-7. The timing diagram of HMEA for level 0

2.3.2. Downsampling

Down-sampling is the pre-processing part of this ME architecture. A simple way

to down-sample an image is to input all of the image data to RAM, and then average four pixels as a single pixel. After numerous manipulations, an image that has one quarter of the original resolution is generated. This approach requires a large part of the available memory to store the original image, and this large amount of memory substantially increases the die size. A pipelined hardware is derived to downsample the images to fit the limitations on the memory and the die size. Figure 2-8 shows an example of down sampling four rows of level 2, where $P_{i,j}^l$ is the j -th pixel of the i -th row, row_i^l , at level l , and Figs. 2-9 and 2-10 illustrate the pipelined down-sampling hardware for levels 2 and 1, respectively. DP_k^l , RAM_i , and Sum_k are the data path of row_i^l , the on-chip memory of each DP, and the temporal summation registers. In Fig. 2-8, four rows of level 2 are subsampled to get two rows of level 1 and one row of level 0 each loop, and a total of 72 loops are required for one CIF image. The procedure is described step by step as follows.

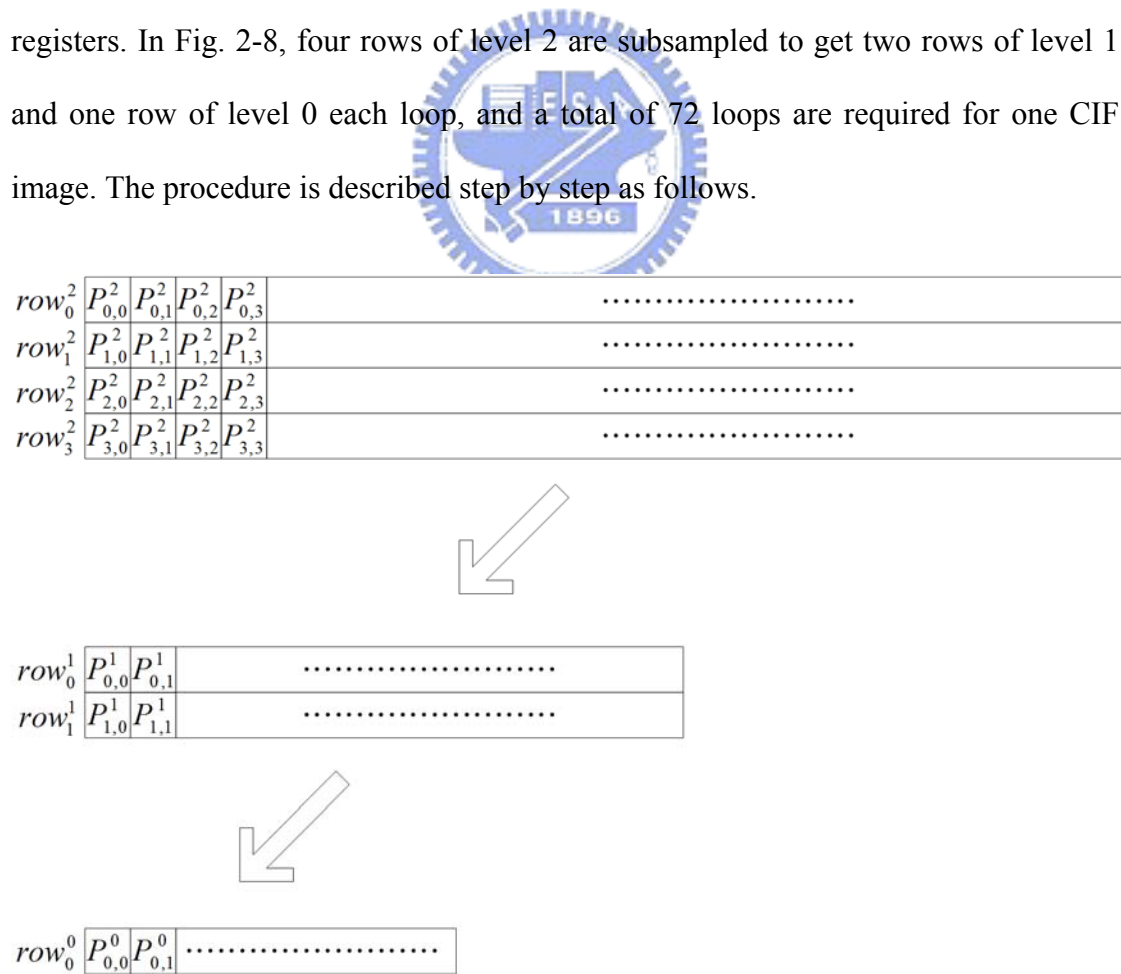


Fig. 2-8. Down sampling four rows of level 2.

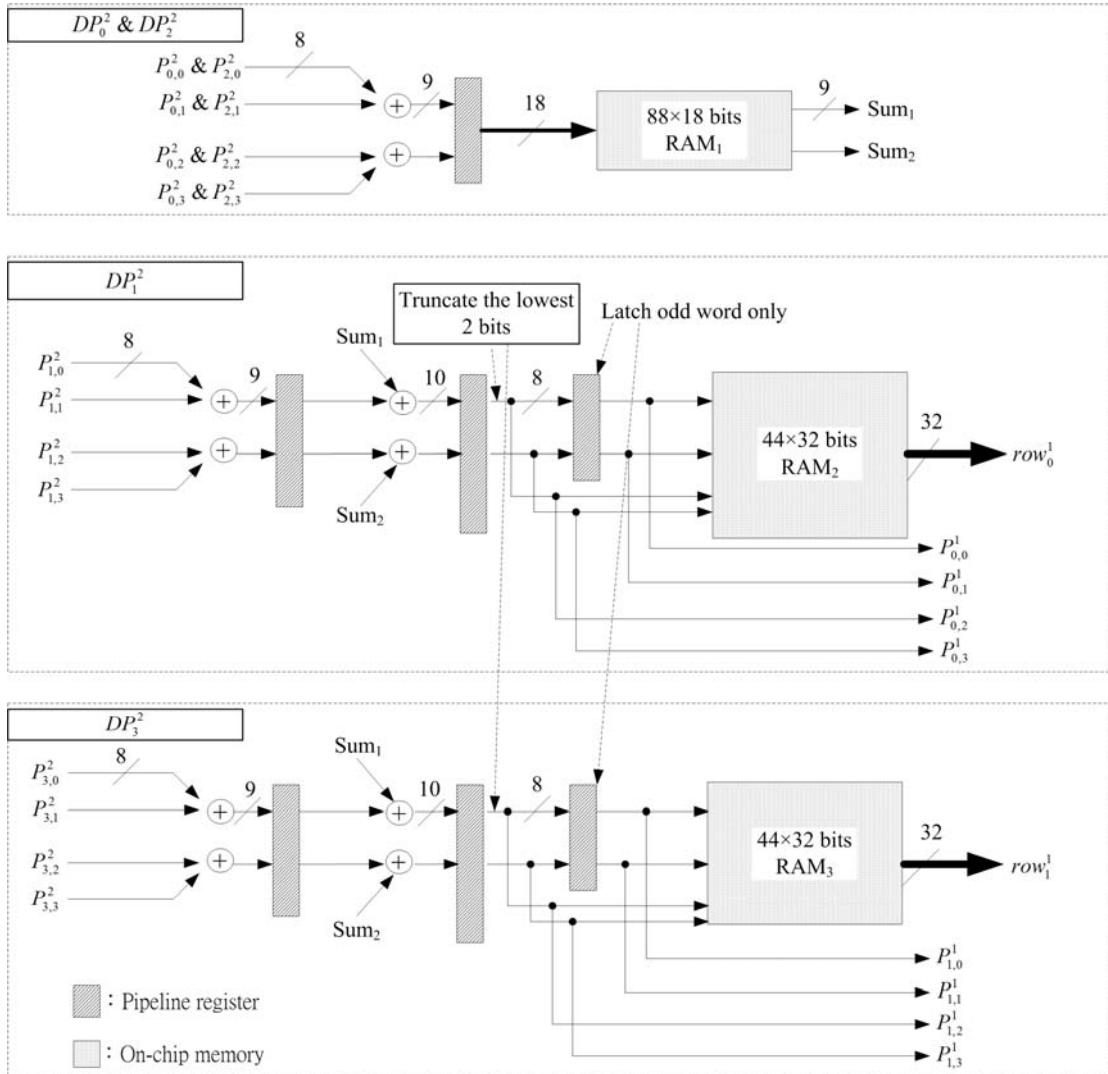


Fig. 2-9. The downsampling hardware block diagram for level 2.

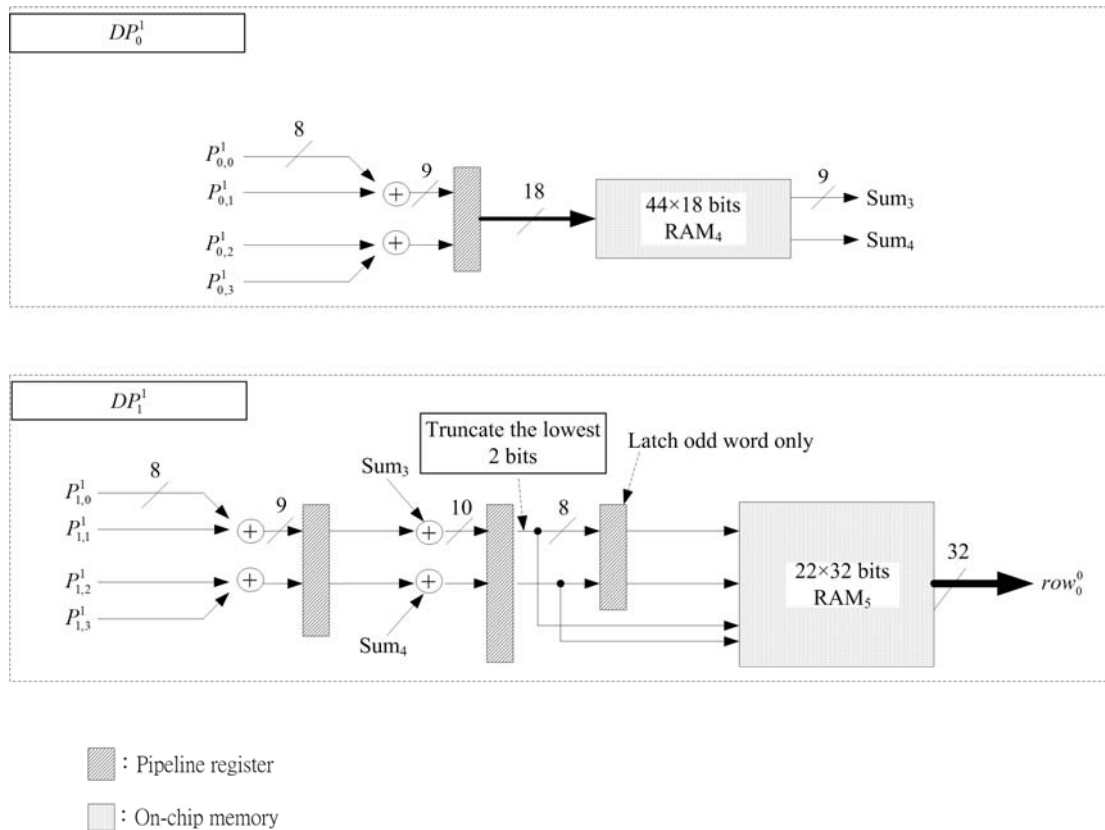


Fig. 2-10. The downsampling hardware block diagram for level 1.

- 1) As shown in Fig. 2-9, row_0^2 is input to DP_0^2 . After row_0^2 has been processed, the results of $P_{0,0}^2 + P_{0,1}^2$ and $P_{0,2}^2 + P_{0,3}^2$ are stored in RAM₁.
- 2) row_1^2 is input to DP_1^2 , and the result of row_0^1 is stored in RAM₂. At the same time, the data output by the second stage of the pipeline register in DP_1^2 , $P_{0,0}^1$, $P_{0,1}^1$, $P_{0,2}^1$, and $P_{0,3}^1$, are input to DP_0^1 , and the results of $P_{0,0}^1 + P_{0,1}^1$ and $P_{0,2}^1 + P_{0,3}^1$ are stored in RAM₄.
- 3) row_2^2 is input to DP_2^2 , and the results of $P_{2,0}^2 + P_{2,1}^2$ and $P_{2,2}^2 + P_{2,3}^2$ are stored in RAM₁.

- 4) row_3^2 is input to DP_3^2 . As in step 2, the result of row_1^1 is stored in RAM_3 , and $P_{1,0}^1$, $P_{1,1}^1$, $P_{1,2}^1$, and $P_{1,3}^1$ are input to DP_1^1 . Then, the result of row_0^0 will be stored in RAM_5 .
- 5) Finally, RAM_2 , RAM_3 , and RAM_5 contain the results of levels 1 and 0, respectively.
- 6) Repeat steps 1 to 5, a total of 71 times to complete the down-sampling procedure of a CIF image.

A row of CIF images has 352 pixels, and four bytes are input per cycle. Thus, 88 cycles are required to fetch a row of level 2, and 22 cycles to get a row of level 1. The pixels are down-sampled in pipeline, so the addition operation and the data input are generally performed simultaneously to significantly increase the overall speed. Nevertheless, 22 extra cycles are required for storing one row of level 0 into RAM_5 . In this architecture, four rows of level 2 and two rows of level 1 are input and one row of level 0 is stored in one turn. Hence, $88 \times 4 \times 72 + 22 \times 2 \times 72 + 22 \times 72 = 30,096$ cycles are needed for a CIF image. Therefore, 76 cycles are required for an MB.

2.3.3. DAU

HMEA works after the downsampling process. The number of PEs in DAU is decided by the operating frequency, the resolution of the images, and the search area. More PEs will increase the estimation speed but enlarge the chip area. Since the basic search range is ± 2 , five PEs will be a stage, and 5-stage 2D semi-systolic architecture is adopted as a DAU which is shown in Fig. 2-11. The Basic Search Unit,

BSU, is a one dimensional systolic PE array [62]. The reuse of data is not good enough, so the current block and previous block data must be loaded again in each loop. The architecture requires 40 mega clock cycles to complete the ME of 30 CIF images. If other system overhead, such as bus arbitration or software overhead, must be considered, real-time encoding will be difficult. Accordingly, an enhanced 2D semi-systolic DAU architecture with improved data reuse capability uses two DAUs to improve the processing speed. HMEA reduces the bandwidth requirement by 80% and only 5.88 mega clock cycles are required to accomplish the ME of 30 CIF images.

Figure 2-11 depicts the architecture of DAU, for a 4×4 current block FSBMA whose search range equals 2, and it consists of 25 PEs, flip flops (DFFs), multiplexers (MUXs), and simple logic for flow control of the input data. The basic data flow is shown in Fig. 2-12. The current MB is a 4×4 block and the search window is a 12×12 block; the search window is partitioned into two parts, "PI" and "Pr". The data are input to DAU from C, PI and Pr ports, as illustrated in Fig. 2-11. Starting from cycle 0, C and PI are sequentially read through each port, and after 4 cycles, the data is input into Pr. The current MB and the search window are input into 5-stage pipeline PEs with the corresponding timing, and 25 SADs in a search range of ± 2 can be calculated. The PE00 and PE10 will accumulate the SADs associated with the search positions $(-2, -2)$ and $(-1, -2)$, respectively. Similarly, the other PEs will compute the SADs of the other search positions. The main advantage of DAU is that the current MB and the search area only have to input once. Therefore, only 36 clock cycles are required to generate 25 SADs, where BSU in [62] needs 80 cycles.

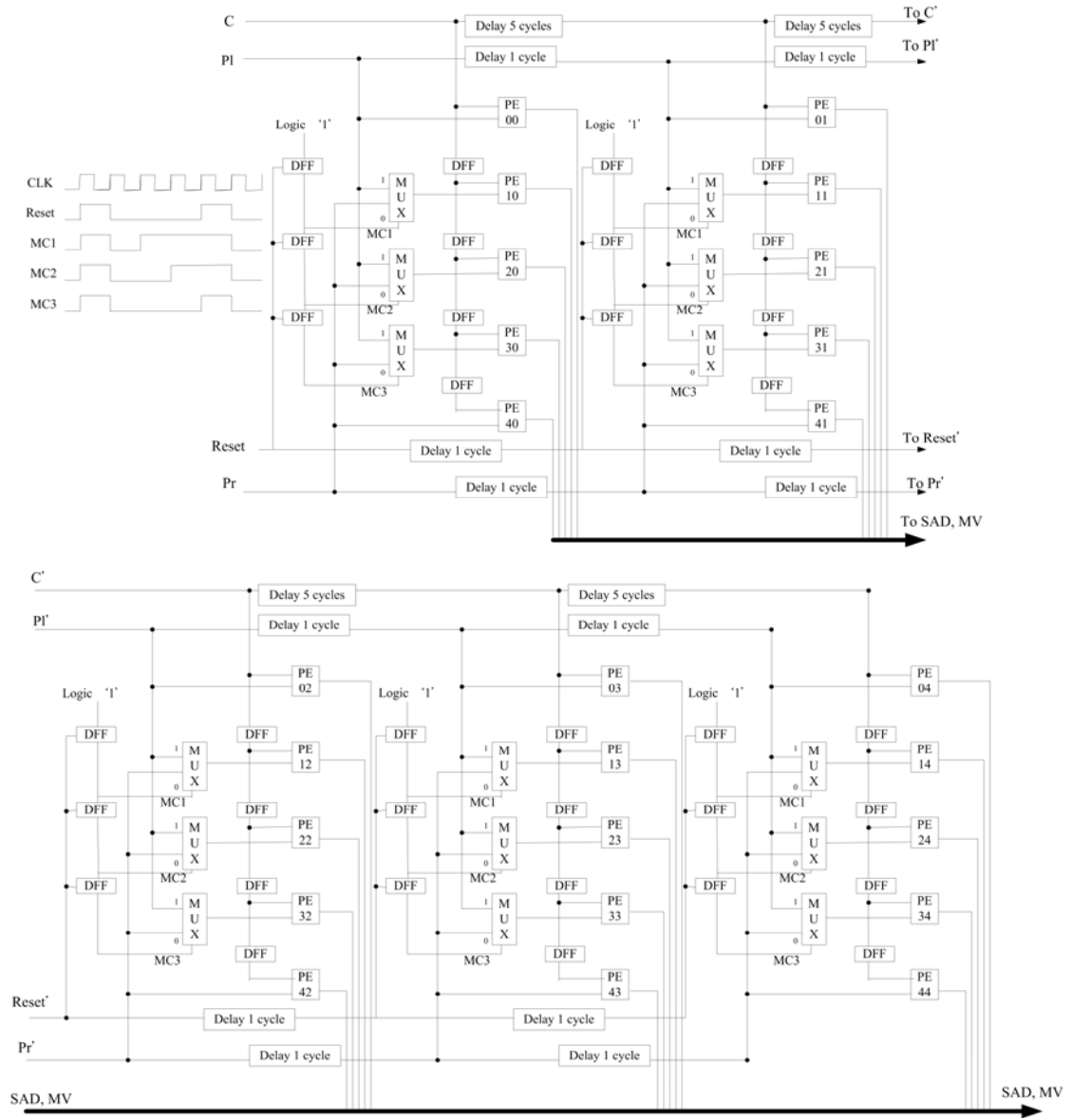


Fig. 2-11. The architecture of a 2D DAU.

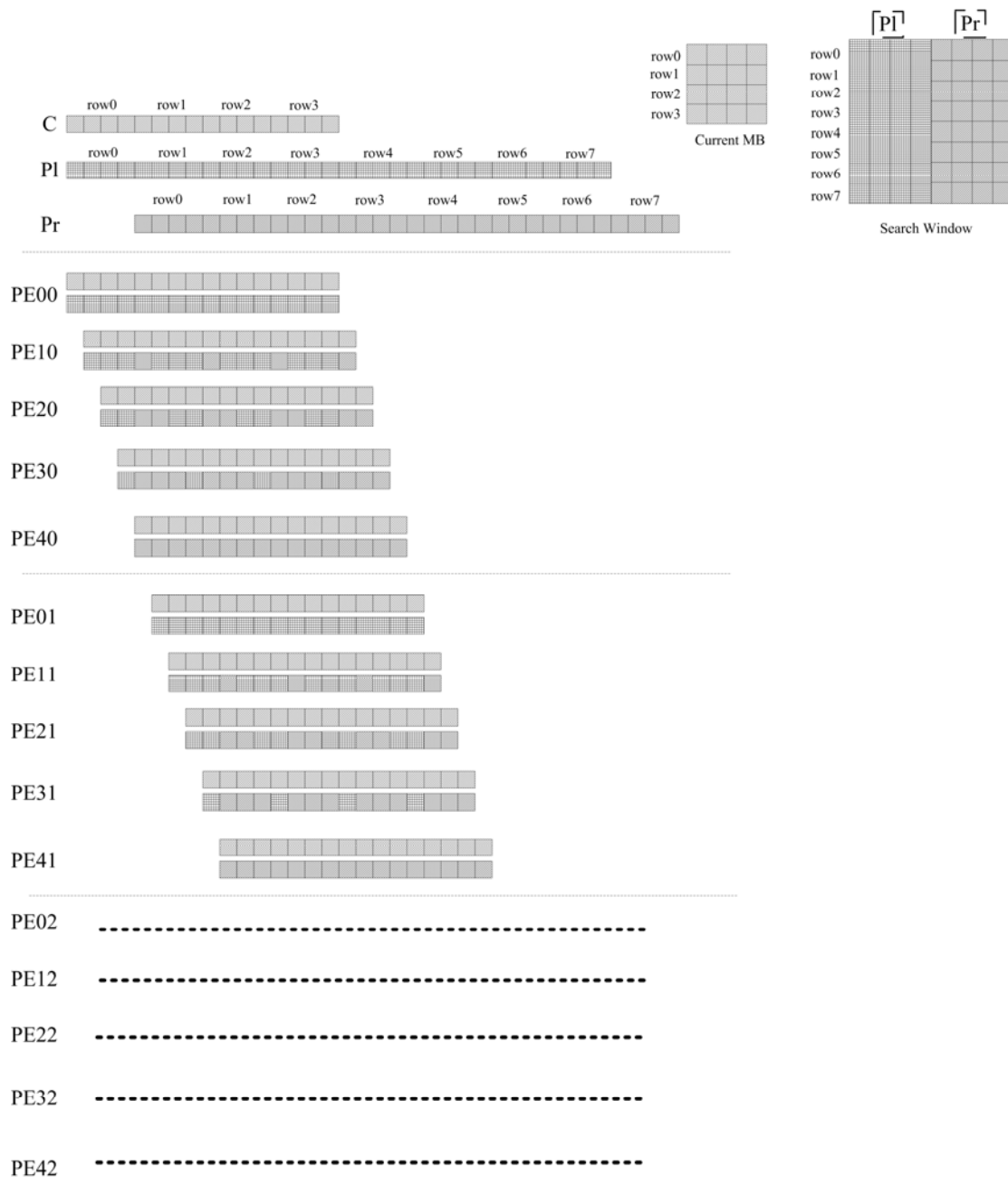


Fig. 2-12. The basic data flow of DAU.

2.3.4. DAU Processing at Each Level

Based on DAU architecture and the basic data flow, a ME architecture, shown in Fig. 2-6, where C_i , Pl_i , and Pr_i are the input ports of DAU_i ($i = 0, 1$), is developed. Two DAUs are employed to accelerate the ME and improve the reusability of data. This architecture is commonly used among different levels and can extend the search

range without adding additional hardware. The data flow at level 0, 1, and 2 are shown in Fig. 2-7, 2-13, and 2-14, respectively. The details of the processing at each level are as follows.

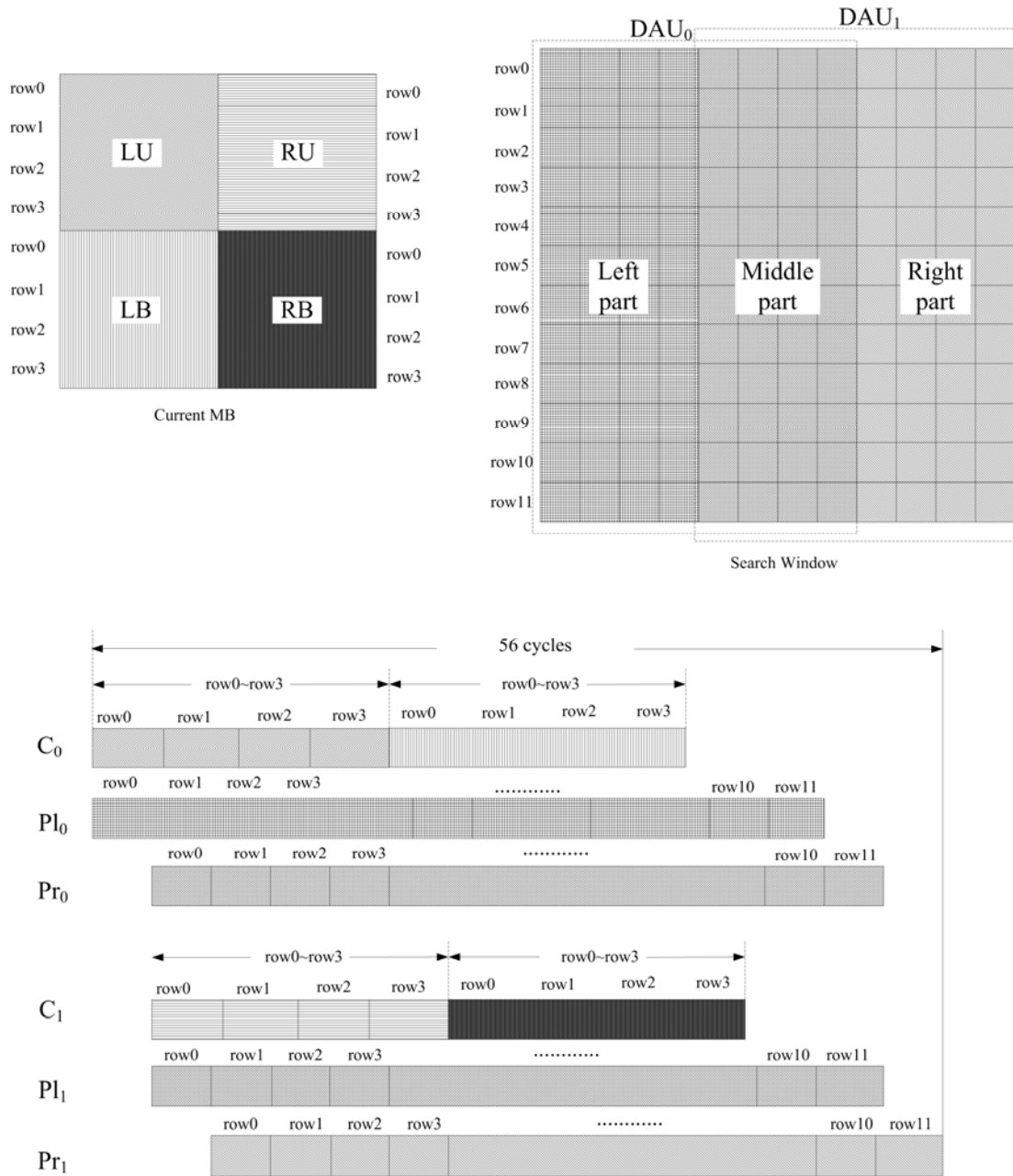


Fig. 2-13. The timing diagram of HMEA for level 1

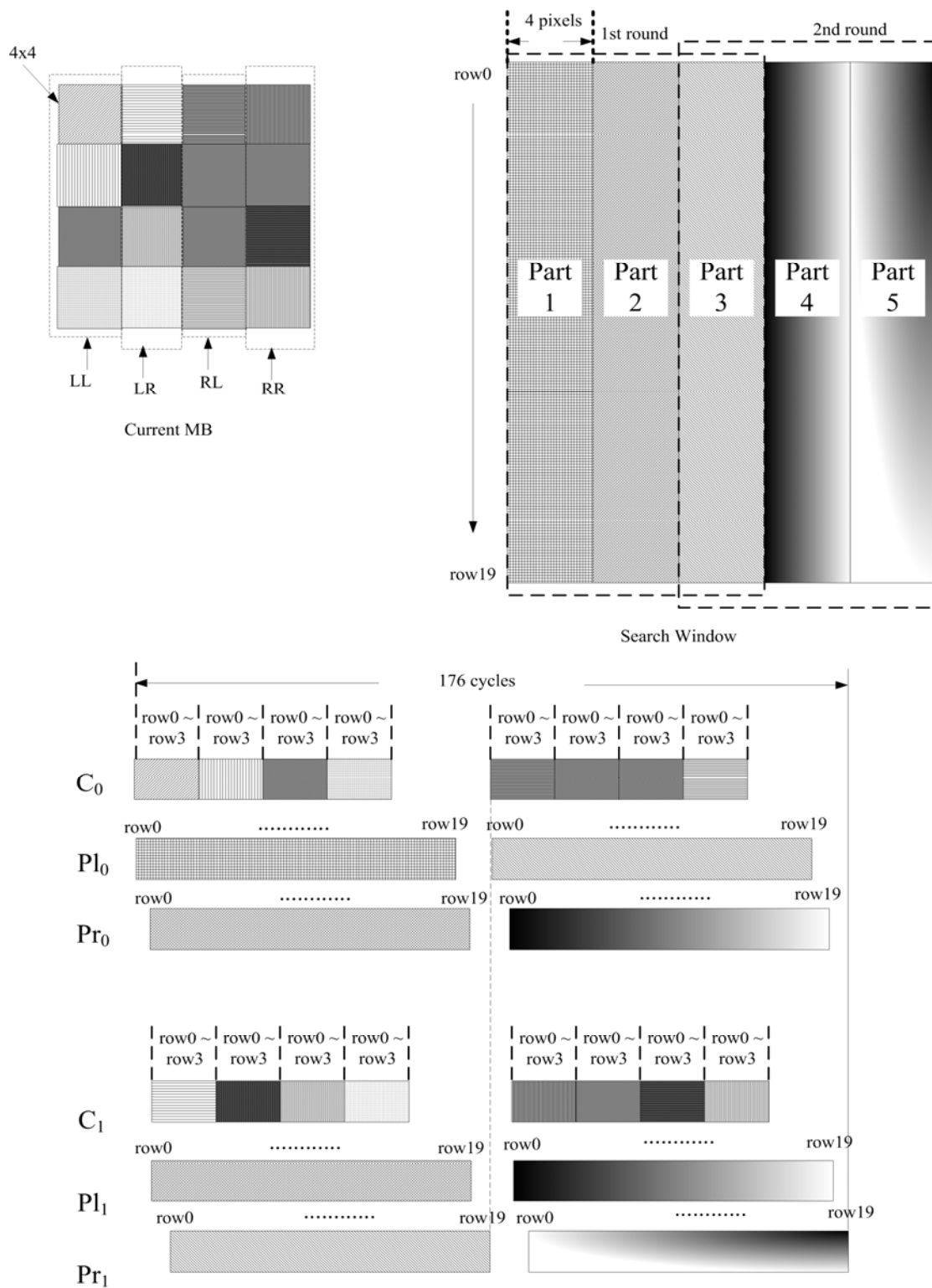


Fig. 2-14. The timing diagram of HMEA for level 2

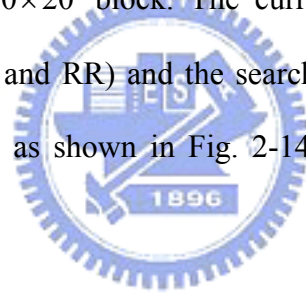
- 1) Level 0: The current MB is a 4×4 block and the search window is a 12×12 block. The search window is partitioned into two parts; one part is input to

DAU₀ and the other is for DAU₁. Figure 2-7 shows the data partition and the data flow. C₀ is fed twice by the current MB and C₁ receives the same data four cycles thereafter. The left part, the middle part, and the right part of the search window which is illustrated in Fig. 2-7 are put into Pl₀, Pr₀, Pl₁, and Pr₁ with the corresponding timing, respectively. After 17 cycles, the 1st SAD will be generated with the relative MV from DAU₀ and the other SADs and the MVs will also be available continuously from DAU₀ and DAU₁. The output of SADs from DAU₀ and DAU₁ are input to the comparator. The minimum SAD and its corresponding MV are retained in the comparator until the search of level 0 is completed. After 56 cycles, two MV candidates are obtained.

- 2) Level 1: The two MV candidates which are found in level 0 are used as the starting point of level 1. Then, FSBMA with a search range of 2 is used to refine MV. At level 1, the current MB is an 8×8 block and the search window is a 12×12 block. The current block is partitioned into four parts - left-upper, left-bottom, right-upper and right-bottom (LU, LB, RU, RB), and the search window is partitioned into three parts, as shown in Fig. 2-13. C₀ is fed by LU and LB of the current MB, and C₁ receives RU and RB four cycles later. In the search window, the condition is the same as that of the search at level 0. The left part, the middle part, and the right part of the search window which is depicts in Fig. 2-13 are put into Pl₀, Pr₀, Pl₁, and Pr₁ with the corresponding timing, respectively. After 17 cycles, the 25 SADs of LU and RU parts begin to be output from DAU₀ and DAU₁ sequentially, and these SADs are sent to “block 1” in Fig. 2-6, to accumulate with the SADs of LB and RB later. Following 56 cycles, the first SAD of level 1 with the relative MV are output from “25-stage shift register” in Fig. 2-6, and the remaining 24 SADs

with the MVs are also output sequentially. SADs of level 1 are sent into the comparator and the minimum SAD with the corresponding MV is retained in the comparator. After $56 + 25 = 81$ cycles, the first search of level 1 is finished. Level 0 has two MV candidates, so the search of level 1 must be performed twice, once for each of these two MV candidates, to determine which MV is the proper one. Therefore, searching for level 1 takes a total of 162 cycles per MB.

- 3) Level 2: The most suitable MV obtained in level 1 is the start point of the local search in level 2. FSBMA whose search range equals 2 is performed to refine MV found in level 1. At level 2, the current MB is a 16×16 block and the search window is a 20×20 block. The current block is partitioned into 4 portions (LL, LR, RL, and RR) and the search window is separated into five fragments, part 1 to 5, as shown in Fig. 2-14. The overall search procedure contains two rounds.



3.1) The first round: LL is input to C_0 , and LR is fetched by C_1 four cycles later. Part1, part 2 and part3 are put into Pl_0 , Pr_0 , Pl_1 and Pr_1 with the corresponding timing. At the 17th cycle, the first SAD with the relative MV is input to “25-stage shift register”, as shown in Fig. 2-6, to be accumulated later with the other SADs of the remaining parts. Following 88 cycles, these 25 SADs of LL and LR are stored in the “25-stage shift register”.

3.2) The second round: The second round is begun at the 89th cycle, RL begins to be input into the C_0 , and RR is fetched by C_1 four cycles later. In the search window, part3, part4 and part5 are put into Pl_0 , Pr_0 , Pl_1 , and Pr_1 with the corresponding timing. The 25 SADs are input to “block 1” in Fig. 2-6 and

accumulated with the 25 SADs, which were determined in the first round. At the 152th cycle, the first SAD of level 0 with the related MV is output from the “25 word circular buffer” and input to the comparator. The minimum SAD and its corresponding MV are retained in the comparator. At the 176th cycle, the first MV is output from the “25 word circular buffer”, and the search process in level 2 requires a total of $176 + 25 = 201$ cycles.

2.3.5. Half-Pel Search

After the final integer-pel MV is computed in Section 2.3.4, a half-pel search is to be performed around it. The search procedure is similar to it in level 2, and the current block is also partitioned into 4 partitions for DAUs processing. The main advantage of HMEA in half-pel search is that the averaging hardware and the on-chip memory for the half accuracy pixels are already existed. Moreover, some of the pixels are pre-processed during the downsampling stage described in Section 3.2. Since the downsampling images, $I_k^{(1)}(\cdot)$ and $I_k^{(0)}(\cdot)$, are no longer necessary, the on-chip memory can be re-used to store the interpolated image. As shown in Fig. 2-15, α denotes the 4-pixel average, and β and γ represent the vertical and the horizontal 2-pixel average, respectively. α' and β' are the pixels which can be directly read from RAM₁ to RAM₅, which is shown in Fig. 2-9 and Fig. 2-10, respectively, and other pixels can be generated from the downsampling hardware directly. First, β with 17×16 pixels are manipulated and be stored into the memory for the search area. Since 17×4 pixels of β are already existed in the downsampling stage, the number of the computations can be reduced. Then, the search steps which is analogous to them in the level 2 are started, and downsampling hardware begins to generate α with

17×17 pixels by the value of β in pipeline. When α is sent to estimate the MV, the values of γ can be calculated. A total of 8 SADs of the interpolated image are computed, and 404 cycles are needed in a half-pel search.

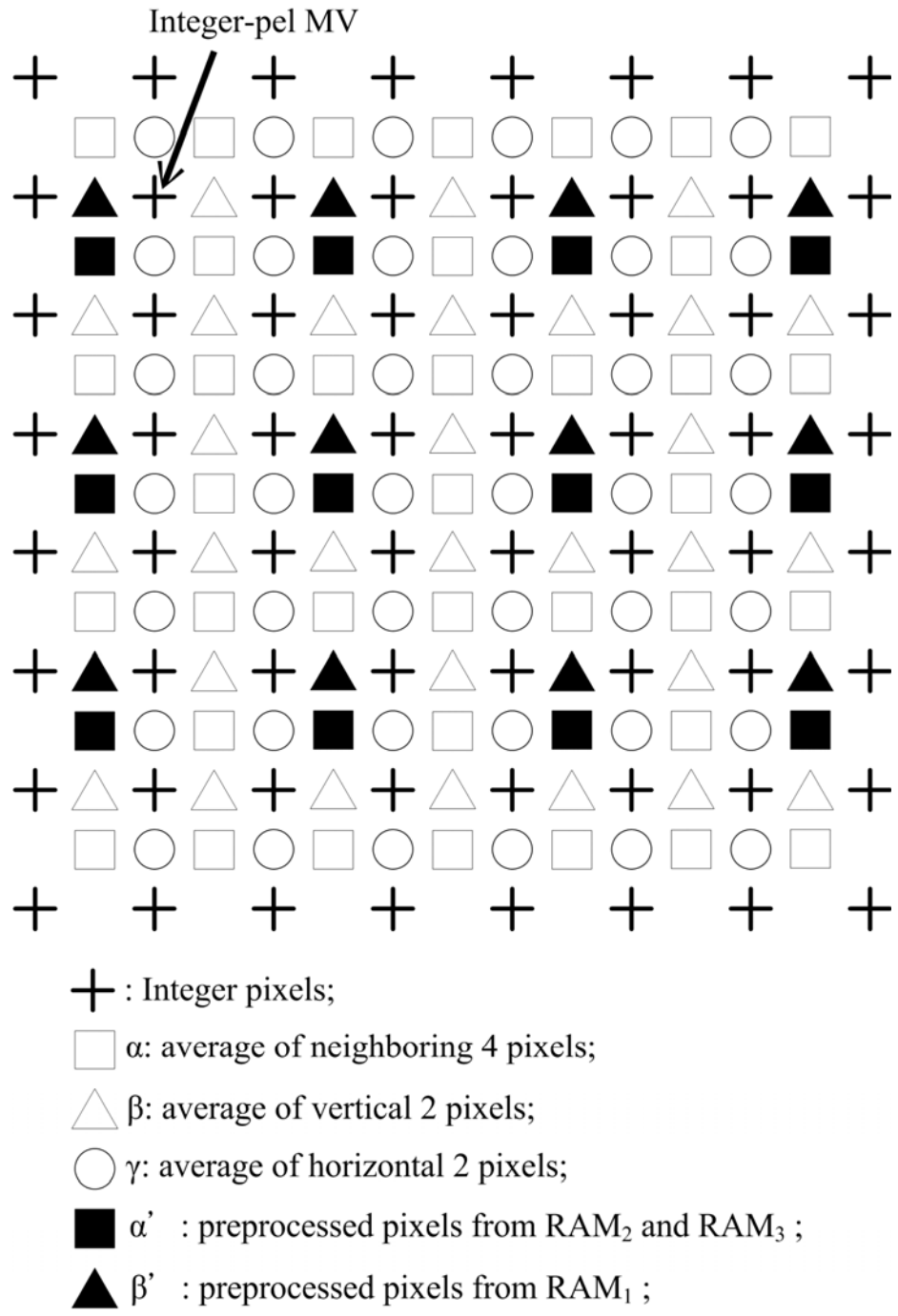


Fig. 2-15. Interpolation for half-pel search

2.4. Implementation Results

The hardware architecture, as described in Section 2.3, is successfully implemented. Additionally, the estimation performance is the same as the software model addressed in Section 2.2.3. If the ME module constantly accesses the external memory of the video coding system, then the system bus will become too busy. Accordingly, the on-chip SRAM is applied to store the current block data and the search area data. Data are reused in the overlapped search area between horizontally adjacent MBs to reduce the loading cycles. With the high video quality and the fast speed, HMEA requires 1393-byte on-chip single port SRAM which are used to store the downsampled images, the current MB, and the search area for estimation. Although the size of the current MB and the search area vary with level, the maximum values at level 2 are adopted. The number of cycles required computing an MV is the sum of the operations in downsampling stage and different levels, as described in Section 2.3, which is $76 + 56 + 162 + 201 = 495$ cycles, so an operating frequency of only 5.88 MHz is sufficient for real-time application for CIF images. As for the half-pel search, no more on-chip memory and logics are added, and a total of $495 + 404 = 899$ cycles are required for manipulating an MV.

The averaging filter, which is applied by HMEA, can increase the quality and speed, and it is suitable for half-pel search. In MMEA, the design of the hardware and the number of memories needed for down-sampling are not mentioned in [62] and [63]. The VLSI circuits for HMEA were described in VHDL and synthesized by SYNOPSIS Design Analyzer using UMC 0.18um CMOS standard cell library. In Table 2-6, HMEA is compared with three kinds of the hardware - MMEA, FSBMA, and PMVFAST. Most ME architectures will not work alone, and they will be

integrated into the video encoders. Consequently, the cycles for calculating an MV of a block and the required operating frequency for the real-time application for CIF images are estimated. The operating frequency of 30 MHz, which is quite enough for the video encoders with HMEA to compress the sequences in CIF format at real-time, is assumed. The comparison is illustrated in Table 2-6. Table 2-6 demonstrates that HMEA is the fastest architecture in MMEA, and has a reasonable chip area. The number of PEs dominates the size, so HMEA is a little larger size with ten times and three times as many PEs as in [62] and [63] respectively. In [63], the cycles and the area outperform that of HMEA, but it uses much more memory than HMEA. The memory will occupy a large die size and greatly increases the cost. Moreover, it can only support integer-pel search. In [62], the on-chip memory is lower than HMEA, so the reuse of data is inefficient. Hence, the required operating frequency for the real-time application for CIF images is too high. Meanwhile, the comparison of HMEA with FSBMA and PMVFAST indicates that the area-speed performance of HMEA is better. In Table 2-6, three architectures, HMEA, [62], and [28], support half-pel search, and HMEA is the fastest among them. Due to the design of the averaging filter, the equivalent gate counts and the on-chip memory can remain the same in the integer-pel search. With the low bandwidth requirement of the system bus, the fixed processing cycles, and the high video quality, HMEA can be easily integrated in video coding systems.

Table 2-6 The comparison between HMEA with other architectures for MMEA, FSBMA, and PMVFAST. The search range among each frameworks are different, the equivalent gate counts includes control and address generation overheads as well as the computational core (PEs), and the MB size is 16×16 .

Type	MMEA						
Architecture	HMEA		MRMCS [62]		QME [63]		
Technology (μm)	0.18		N/A		0.35		
Search range	[-16.0, +15.5]	[-16,+15]	[-16.0, +15.5]	[-16,+15]	[-16,+15]		
Number of PEs	50	50	5	5	16		
Cycles / block	899	495	3152	2640	615		
Required operating freq. for CIF @ 30 FPS (MHz)	10.68	5.88	37.45	31.36	7.31		
Throughput @ 30MHz (MVs per second)	33370	60606	9517	11363	48780		
Maximum operating freq. (MHz)	153		N/A		N/A		
Gate counts	59K	59K	25K	25K	24.8K		
On-chip Memory (bytes)	1393	1393	288	160	2623		
Type	FSBMA						PMV FAST
Architecture	Yeo [26]	Yeh [27]	Shen [28]		GEA [40]	He [32]	Li [44]
Technology (μm)	N/A	0.6	0.6		0.35	0.25	0.18
Search range	[-16, +15]	[-16,+15]	[-16.0, +15.5]	[-16, +15]	[-16,+15]	[-16,+15]	[-16,+15]
Number of PEs	1024	256	65	64	16	1024	9
Cycles / block	256	1024	6400	4096	1635	615	1042
Required operating freq. for CIF @ 30 FPS (MHz)	3.04	12.17	76.03	48.67	19.42	7.31	12.39
Throughput @ 30MHz	117187	29296	4687	7324	18348	48780	28790
Maximum operating freq. (MHz)	N/A	90	60		27.8	100	200
Gate counts	447K	87.9K	106K	N/A	23.1K	491 (PE only)	17.5K
On-chip Memory (bytes)	N/A	1726	N/A	N/A	N/A	2175 (PE only)	N/A

Chapter 3. Register-Based Platform

Independent MPEG-4 Co-Processor and Its System-Level Design

With the widespread popularity of portable devices, such as cellular phones and PDAs, in these days, people rely on the mobile technology more and more. To satisfy the low power, low cost, and high performance requirements for consumer electronics, the video encoders implemented by the VLSI architectures are much more suitable than the firmware or the software solutions. However, to integrate these ICs into a system is not easy, and these frameworks usually support dedicated frame size, frame rate and output bitrate, which will limit the utilities of the products. In order to overcome these drawbacks, RPIMC is proposed in this chapter, and it can transfer and receive the image data in all kinds of bus matrices with the suitable wrappers for being easily integrated into other platforms. RPIMC, which can be programmable to manipulate up to HD resolution and 30 FPS, with 204K gates and 6,462 bytes of RAM is implemented, and it can adjust the data types of the input and the output streams by modifying the relative registers. Moreover, the demonstration system-level design of RPIMC is also addressed, and a real-time audio/video synchronized streaming server by the software and the hardware co-design is successfully developed.

3.1 Introduction

Recently, the algorithms and the architectures for processing video and audio signals are improved significantly. They are employed in various applications, such as digital TV, video conferencing and mobile multimedia systems. The markets for mobile electronics equipments, like portable PCs, cellular phones, and PDAs, are currently growing rapidly. Moreover, the higher bandwidth for wireless telecommunications now is provided for transferring moving pictures in addition to speech and data. Therefore, multimedia processing will be an essential function in such mobile-equipment applications.

The improved coding efficiency and the advanced features of MPEG-4 come with much higher computational complexity compared with previous standards. Several MPEG-4 video encoders have been reported. To satisfy rich functionalities of the future multimedia, some are implemented in firmware based on the low power DSP platform [66]. They have the highest flexibility but the cost of the hardware is too expensive. Moreover, the low power DSPs are usually operate at lower frequency, so the image quality will be degraded due to the fast algorithms of ME and discrete cosine transform/ inverse discrete cosine transform (DCT/ IDCT). Therefore, the dedicated hardware methodologies are developed to achieve low power and low area cost, and it can encode the MPEG-4 video for CIF format at 15 FPS at 1.5V supply with 700K gates [67]. However, lack of potential for future modification of advanced algorithms and higher design effort are the disadvantages.

Hence, in order to compromise the performance and flexibility, the hybrid software/ hardware co-design is adopted [68]-[70]. A RISC-based platform with hardware accelerators is presented to implement MPEG-4 video encoding algorithms [68]. The optimization in both algorithm and architecture level is applied. However,

the operating frequency at 40 MHz is too high for portable devices. The architectures which are developed for reducing the power consumption usually provide lower encoding complexities. Another design with the same encoding complexity as [67] based on an ARM core and AMBA is introduced, but its power consumption is not suitable for consumer electronics [69]. Except for the single purpose video encoders, an multi-functionality videophone LSI is fabricated utilizing a 0.25-um CMOS triple-well quad-metal technology [70]. Three 16-bit multimedia-extended RISC processors, dedicated MPEG-4 hardware accelerators, and a 16-Mb embedded DRAM are integrated. Although it has reasonable chip area and power consumption, it can only encode the MPEG-4 video for QCIF format with 15 FPS.

These designs can be separated into two parts. One is the architectures that only can perform video encoding [67]-[69], and the other part is the frameworks which have more functionalities like videophone [70]. Generally speaking, the single purpose encoders often provide better coding performance in image size and frame rate. Moreover, the chip area and the power consumption are also less than that of the multi-purpose one. Therefore, to integrate the MPEG-4 encoder into a system is still a critical issue. Besides, these designs are developed only for their platforms, and the encoder parameters such as image resolution, output bitrate and input frame rate are fixed. Furthermore, to integrate these frameworks into other platform is difficult because the considerations, like the timing of fetching the image data, and the output packets formats of the encoded bitstream, are usually not compatible with other platforms. When the manufacturers own their RISC and relative peripherals, such as LCD, memory card, and USB controllers, all they need is an MPEG-4 encoder which can easily integrate into their system-on-chip (SoC) design for various applications. However, the dedicated and limited functionalities of these encoders usually force the

producers to establish their products by dual-chip. One is their RISC and relative peripheral controllers, and the other is an MPEG-4 encoder. The overall cost and the applications will be restricted.

In this chapter, RPIMC and its system-level design are proposed. The main ideas of this paper are to provide a programmable MPEG-4 encoder which can easily integrated into any platforms, and to illustrate its software/hardware SoC co-design. To satisfy the demand for various applications, RPIMC can modify the input frame size, input frame rate, and output bitrate by adjusting the relative registers. The main controller of RPIMC will automatically calculate the internal loops of the pipelines for encoding, and will read the data in the corresponding memory with the correct image resolution based on these registers, respectively. Therefore, the manufacturers who need MPEG-4 encoders can easily integrate RPIMC into their platforms, and they can use RPIMC for various applications with proper register settings. Not only the innovative features are developed, but also the algorithms which can fit the requirements are adopted. According to the computational complexity analysis report in [71] and [72], the dominating computation-intensive tasks in MPEG-4 core profile coding are motion estimation (ME) and shape coding, which together contribute more than 90% of the overall complexity. For simple profile without shape coding tools, ME becomes the most significant one. Hence, an efficient hierarchical motion estimation algorithm (HMEA) is applied [7][8]. HMEA using multi-resolution frames to reduce the computational complexity and excellent estimation performance is ensured using an averaging filter to down-sample the original image. When the image resolution is larger, HMEA can reduce more sum of absolute difference (SAD) operations.

In SoC, the system level software development is as important as the hardware architecture design, and it can greatly increase the flexibility and the functionality of the systems. In this chapter, RPIMC, an ARM920T core and the advanced microcontroller bus architecture (AMBA) are integrated together to become an embedded MPEG-4 audio and video compression system. A low-complexity MP3 algorithm [73], audio/video synchronization, and hardware controller are provided to enhance the functionalities of RPIMC. ARM core is responsible for the MP3 encoding and the synchronous audio/video bit stream packing. With collocation RPIMC, the real-time audio and video data can be encoded immediately, and the encoded bitstream can be transferred into internet or be saved in local storage devices. By integrating the flexible hardware and the versatile software, the SoC can be applied in more applications than the other MPEG-4 encoders. The main contribution of this paper are to design a high quality programmable video encoder that is suitable for every platform and every purpose with small gate counts to solve the changeless disadvantage of normal VLSI frameworks [67]-[70], and a demonstration SoC system is successfully designed to ensure its performance and innovative features.

3.2 The Architecture of RPIMC

Figure 3-1 shows the overall architecture of RPIMC, and it mainly contains four parts, controller, MU, texture coding engine (TCE), and bitstream generator (BG). The controller will calculate the required inter loops for different input frame resolutions, and it is also responsible for macro block (MB) level hardware scheduling, and coding mode decision. Other hardware accelerators improve the system performance by parallel processing according to the parallelism of algorithms. MU

includes ME and motion compensation (MC), and can carries out ME with the search range from -16.0 to +15.5 pixel unit. Moreover, it interpolates pixels in reference frames into compensated MBs with the specific motion vector (MV). DCT, IDCT, quantization (Q), inverse Q (IQ), and AC/DC prediction on texture pixels in MBs are integrated in TCE. BG produces bitstream headers, motion information, and texture information in the format of variable length codes. The hardware pipeline scheduling and the register bank will be described below.

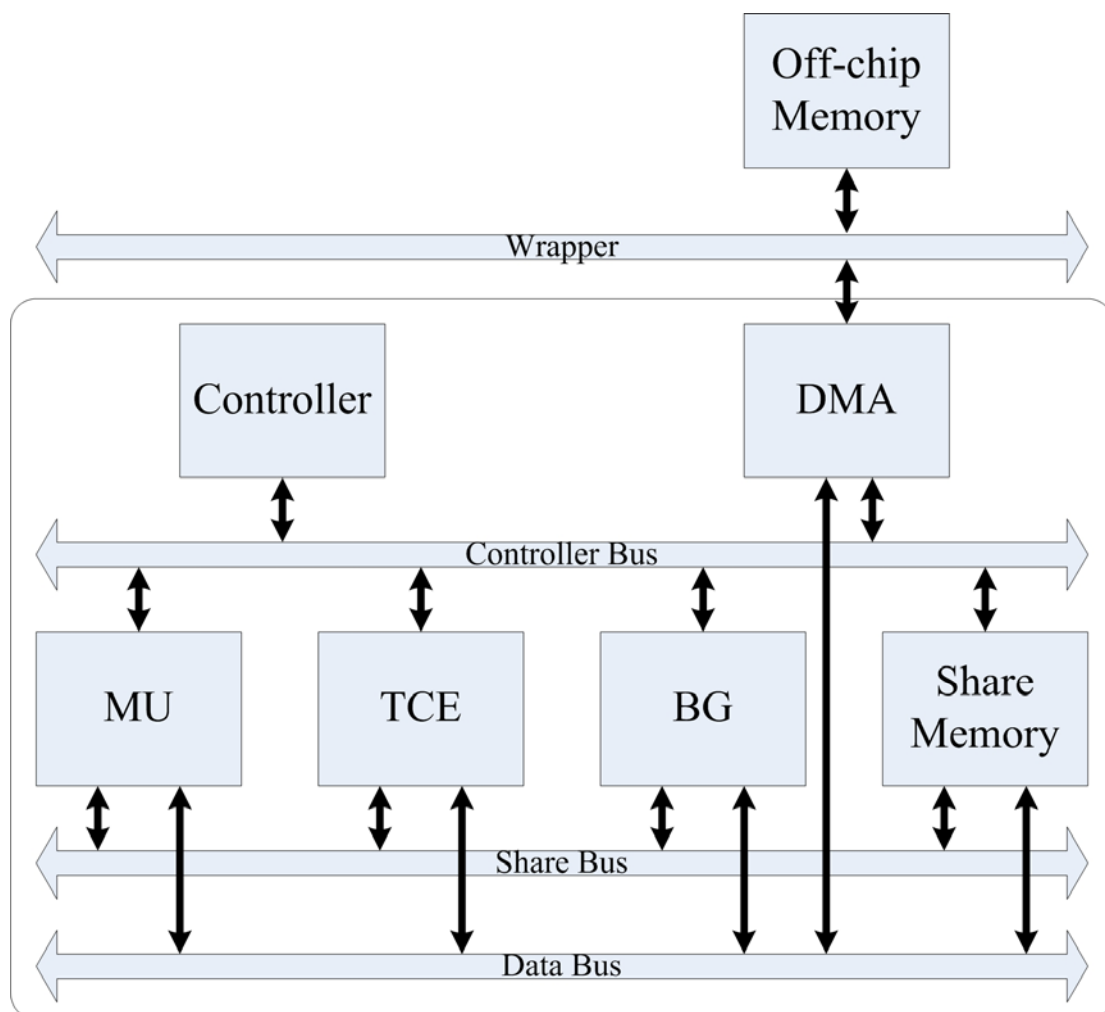


Fig. 3-1 The overall architecture of RPIMC

3.2.1. The Hardware Pipeline Scheduling

After analyzing the clock cycles needed for processing one macro block, three stages pipeline scheduling, which is divided by MU, TCE and BG, is applied. As shown in Fig. 3-2, RPIMC will fetch the input frame MB by MB, and the pipeline processing can be separated into the intra and the inter modes since MU is not activated in the latter mode. The duration for processing one MB is called one time slot (TS), which is the period between two vertical dotted lines, and MU_i , TCE_i , and BG_i denote the operation of the i -th MB of the image in the corresponding hardware accelerators.

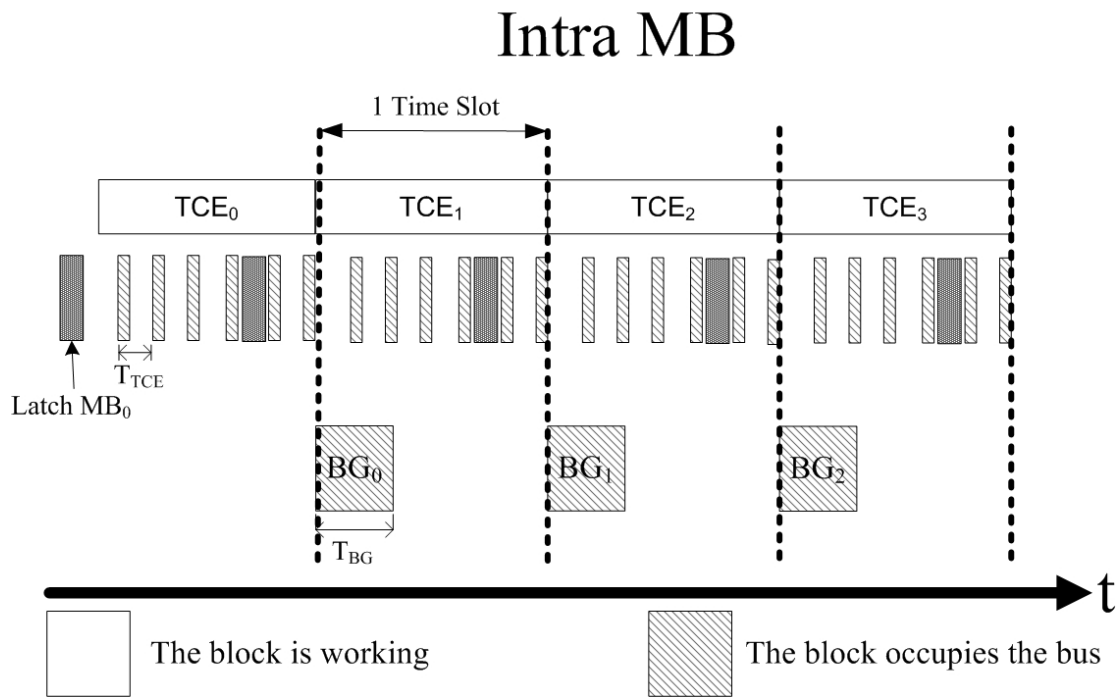
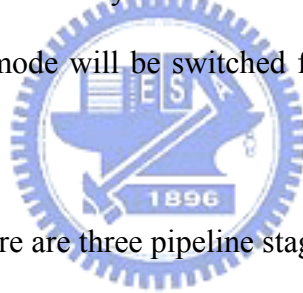


Fig. 3-2 The intra MB scheduling

In the first TS of the intra mode, only TCE is activated, and BG is in the suspend mode in order to reduce the power consumption. After TCE processing the first MB, the texture information, which are the quantization coefficients, will be input into BG, and TCE will start to manipulate the second MB in the second TS. In addition to the

scheduling of the accelerators, the usage of the bus matrix is also an important issue. In order to maximize the performance, the occupation of the bus matrix of each component should be separated as possible as they can. In RPIMC, the image data is input in the format of YUV420, and one 16×16 MB denotes four 8×8 Y blocks, one 8×8 U block, and one 8×8 V block. According to the design of TCE, these six blocks are manipulated and output in one TS with the same interval, T_{TCE} , which is shown in Fig. 3-2. The first two output data of TCE will be stored into the local buffer in order to let the BG to write out the bitstream since the duration of BG occupying the bus matrix, T_{BG} , is around $2 \times T_{TCE}$. After the fourth block producing from TCE, the bus matrix will be free for a while, and the controller will start to fetch the next MB from the external memory at this moment. When all MBs of the image are computed, the operation mode will be switched from the intra mode to the inter mode.



As shown in Fig. 3-3, there are three pipeline stages in the inter mode. In the first TS, only MU is enabled and processes the first MB. After that, TCE will compute the motion materials while the second MB is in MU, and BG will be activated to read the texture information at the third TS. In this mode, BG will still occupy the bus matrix at the beginning of each TS, and two FIFO buffers are adopted to store the temporal reconstructed MBs from MU since the bus is busy in the first half of TS. The controller will manage the usage of the bus matrix after BG finished, and ME will have higher priority than MC. If ME wants to fetch the next MB, MC will put the reconstructed MB into FIFO. Otherwise, MC will output the data to the external memory when ME is not using the bus. In this way, the pipeline scheduling and the usage of the bus matrix will be efficient to increase the overall processing speed. TS of each MB is 1,200 cycles in average, depending on the cycle time of BG occupying

the bus. If the latency of external memory is 5 cycles at the operating frequency of 20-MHz, which is 250 ns per word, RPIMC will encode the MPEG-4 videos for CIF format at 21-MHz for real-time applications.

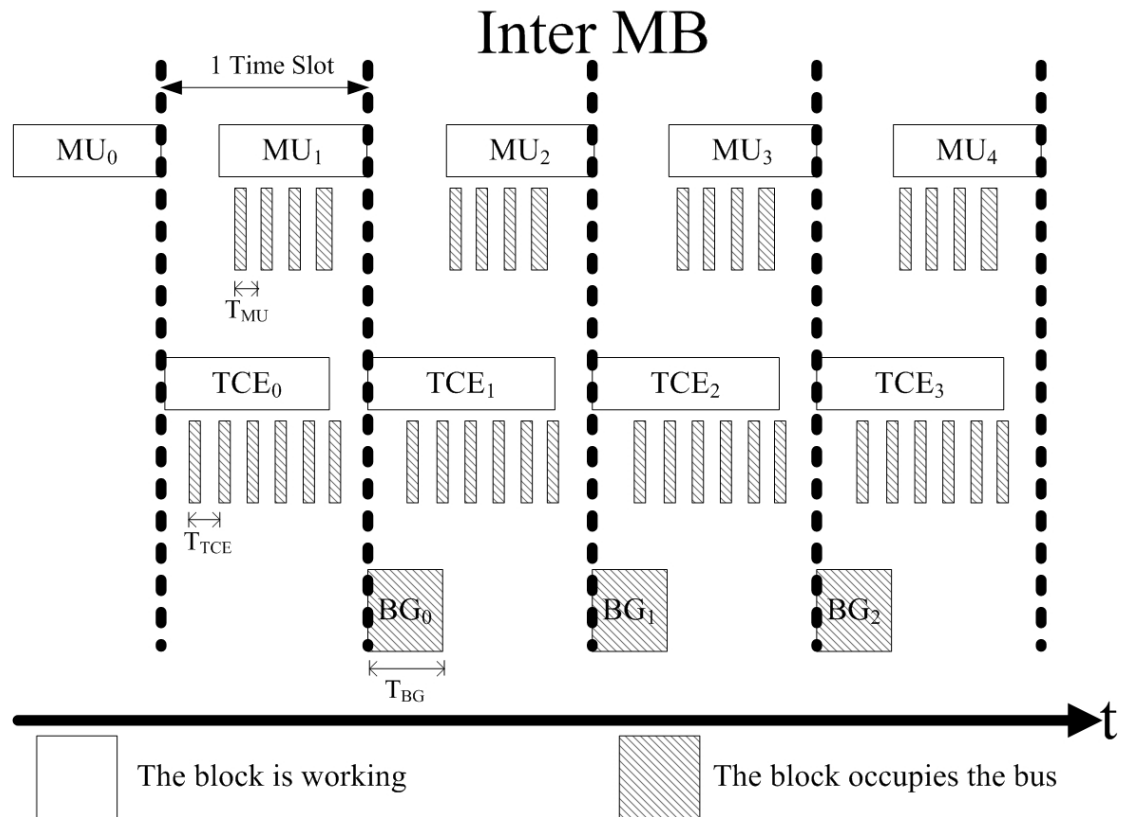


Fig. 3-3 The inter MB scheduling

3.2.2. The Register Bank

The design of the register bank makes RPIMC independent from the platforms, and it combines the control registers (CRs) and the status registers (SRs). The main feature of RPIMC is that it can program several system parameters to satisfy various applications, such as mobile video phones, digital video recorders, and high performance surveillance systems. In these utilizations, the required frame resolution, FPS, the output bitrate and the power consumption are different from each other. The manufacturers who have their own platforms can integrate RPIMC easily by setting

the corresponding CRs, and RPIMC will start to encode the video with the format they want. The working flow of RPIMC is shown in Fig. 3-4, and CRs and SRs with their definitions are listed in Table 3-1.

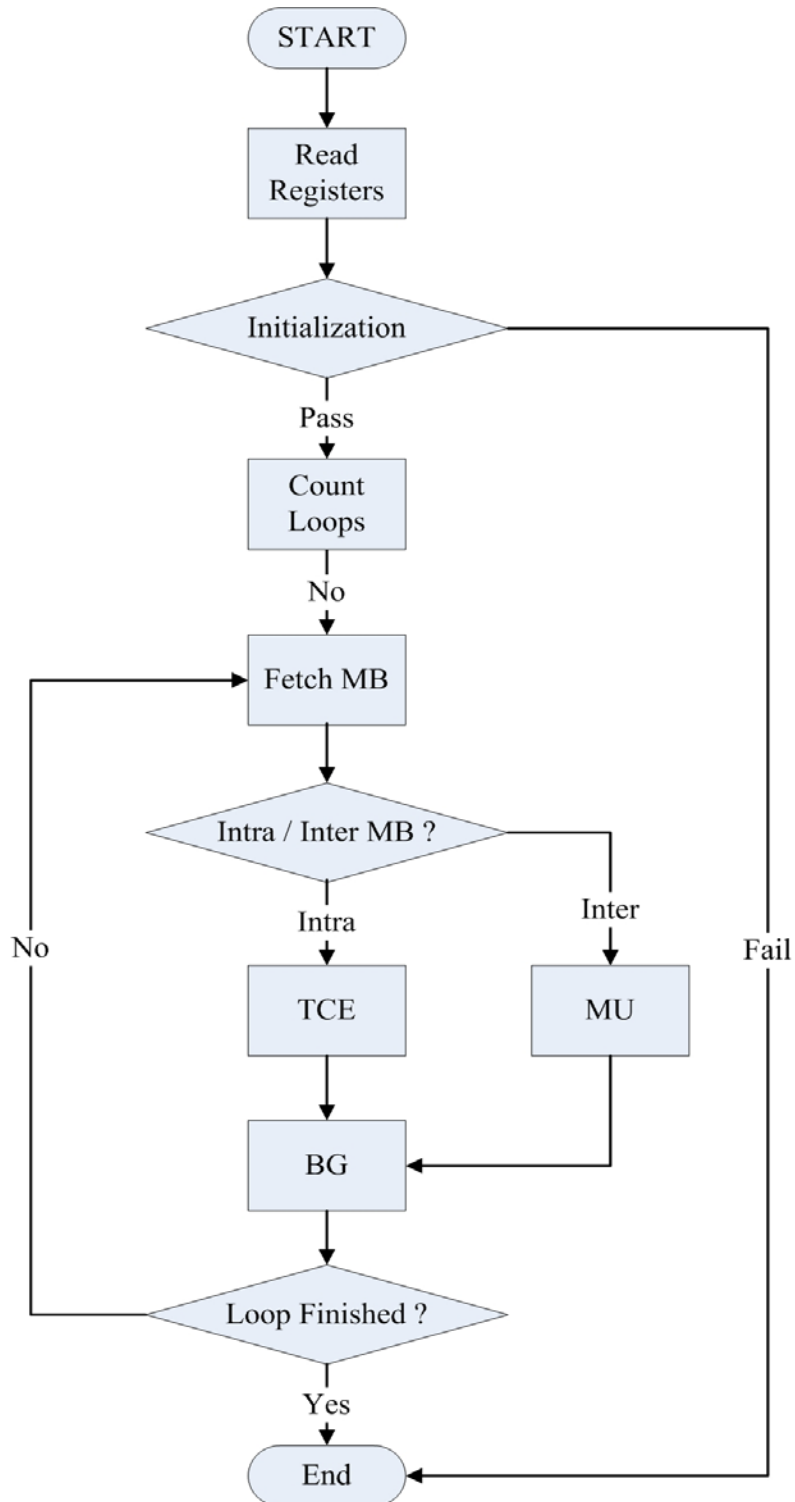


Fig. 3-4 The flow chart of the controller

Table 3-1 The register banks

Types	Name	Description
CR	W	The width of the input image
	H	The height of the input image
	I Size	The size of the input image
	I FPS	The input frame rate
	Bitrate	The output bit rate
	Clock	The input operating frequency
	MEM1	The start address of MEM ₁ for current frame
	MEM2	The start address of MEM ₂ for reconstructed frame
	O FPS	The output frame rate
	Out	The start address of output bitstream
SR	Status	The current state of RPIMC (Idle, Enable, Sleep or Finish)
	O Size	The size of the output bitstream
	Mode	The current operating mode of RPIMC (Intra / Inter)

To achieve the platform independent design, the controller has to verify the value of CRs in the initial state. At the beginning, it will fetch the values in the register bank, and check if the start memory address of the current frame, the reference frame, and the output data are overlapped by adding them with the size of each input picture which can be obtained in CRs. Since RPIMC can allow these input data be arranged in the discontinuous memory block to make the design of the platforms flexibly and efficiently, if one address of them is placed within the manipulated area of the others, RPIMC will not be ready and stays in the idle state. The decision equation is (3-1), where $Add_{(x)}$, img_{size} , and S represents the start place of (x) , the size of the image, and the set of memory address of the current frame, the reference frame, and the output data, respectively.

$$Add_{(A)} \in \left\{ \left[Add_{(B)}, Add_{(B)} + img_{size} \right] \cup \left[Add_{(C)}, Add_{(C)} + img_{size} \right] \right\},$$

where $A, B, C \in S$,

$$S = \left\{ Add_{(reference\ frame)}, Add_{(current\ frame)}, Add_{(output\ data)} \right\}, \quad (3-1)$$

and $A \neq B \neq C$,

Furthermore, because the encoding cycles for one 16×16 block of RPIMC are

fixed, the controller can estimate the computing ability according to the input operating frequency in CRs. If the requirement of the image resolution and the frame rate exceed the load of RPIMC, illustrated in (3-2), where C_{block} denotes the required cycles for encoding one block, the co-processor will not be enabled, too. On the contrary, while it has free time to wait for the image data, the controller will automatically gated the input clock and rewrite the SRs to save the power.

$$C_{block} \times \frac{img_size}{16 \times 16} \times fps > input\ frequency. \quad (3-2)$$

If the initialization procedure is passed with no errors, the controller will calculate the number of loops for encoding a frame by the desired image resolution. Then, the image data will be input one block by one block at the address assigned in the CRs, and the controller will determine whether the input block is intra or inter one. If it is an intra one, it will be sent to TCE, or else to MU. When the loops are finished, RPIMC will be back to the idle state to reduce the power consumption. In order to make the host easier to integrate with RPIMC, it will reflect the encoder conditions to the host through SRs. The mode of the encoder (intra MB or inter MB), the current working status (idle, enable, finish, or sleep), and the bitstream size of the encoded frame will be stored in SRs by the controller of RPIMC.

3.2.3. Memory Orgnization

RPIMC requires the off-chip memory (OFFM) and several on-chip memory (ONM) blocks to complete the whole MPEG-4 video encoding procedure. OFFM contains source frames and reconstructed frames, and ONM is used as local buffers to

reduce the bus bandwidth. In order to increase the speed of TCE for fitting the pipeline scheduling, the transformed coefficients for AC/DC prediction and the transpose memory for DCT/IDCT are integrated into ONM. Besides, for the data fetching performance and the information reuse efficiency, ONM allocates the space for storing the current MB and the search area for MU, and it also includes the input and output buffer for both TE and BG, respectively.

The input video source and the reference frames are stored in OFFM, and the direct memory access (DMA) plays an important role to control the memory interface to read data from or write them out to OFFM in a specified sequence after being initialized by the controller. In the OFFM design, two main parts, MEM₁ and MEM₂ are used to store two frames and they act as the ping-pong buffer to increase the encoding speed. The operations for MEM₁ and MEM₂ can be separated into two modes. First, in the intra frame mode, the input images are always stored in MEM₁, and the reconstructed frame produced from TCE will be saved in MEM₂. Second, in the inter frame mode, since the current frame will be the reference frame in the next encoding loop, these two parts will switching their status mutually until the next intra frame mode, and the scheme is illustrated in Fig. 3-5.

Frame Number and Frame Type	Mem ₁ Status	Mem ₂ Status
Intra Frame	Current Frame	Reference Frame
1 st Inter Frame	Current Frame	Reference Frame
2 nd Inter Frame	Reference Frame	Current Frame
3 rd Inter Frame	Current Frame	Reference Frame
⋮	⋮	⋮
Intra Frame	Current Frame	Reference Frame
1 st Inter Frame	Current Frame	Reference Frame
2 nd Inter Frame	Reference Frame	Current Frame
3 rd Inter Frame	Current Frame	Reference Frame
⋮	⋮	⋮

Fig. 3-5 The state of the external memory

3.3 System-Level Design

ARM integrator platform is used for functional verification of RPIMC and the development of a real-time embedded audio/video multimedia recorder (REMR). ARM is responsible for compressing the audio in MP3 format and merging it with the video bitstream which is generated by RPIMC with the packet form of the real-time transport protocol (RTP). The system overview is illustrated in Fig. 3-6. When the images and the audio are input from the outer devices, they will be transferred to the corresponding encoders. Finally, the compressed payloads will be integrated into files

or be transmitted through internet by the streaming server built in REMR. The files will be stored in the USB storage devices, such as flash memories or hard disks, and can be downloaded by the FTP or the HTTP server. In this chapter, the software architecture and the FPGA prototype system will be described.

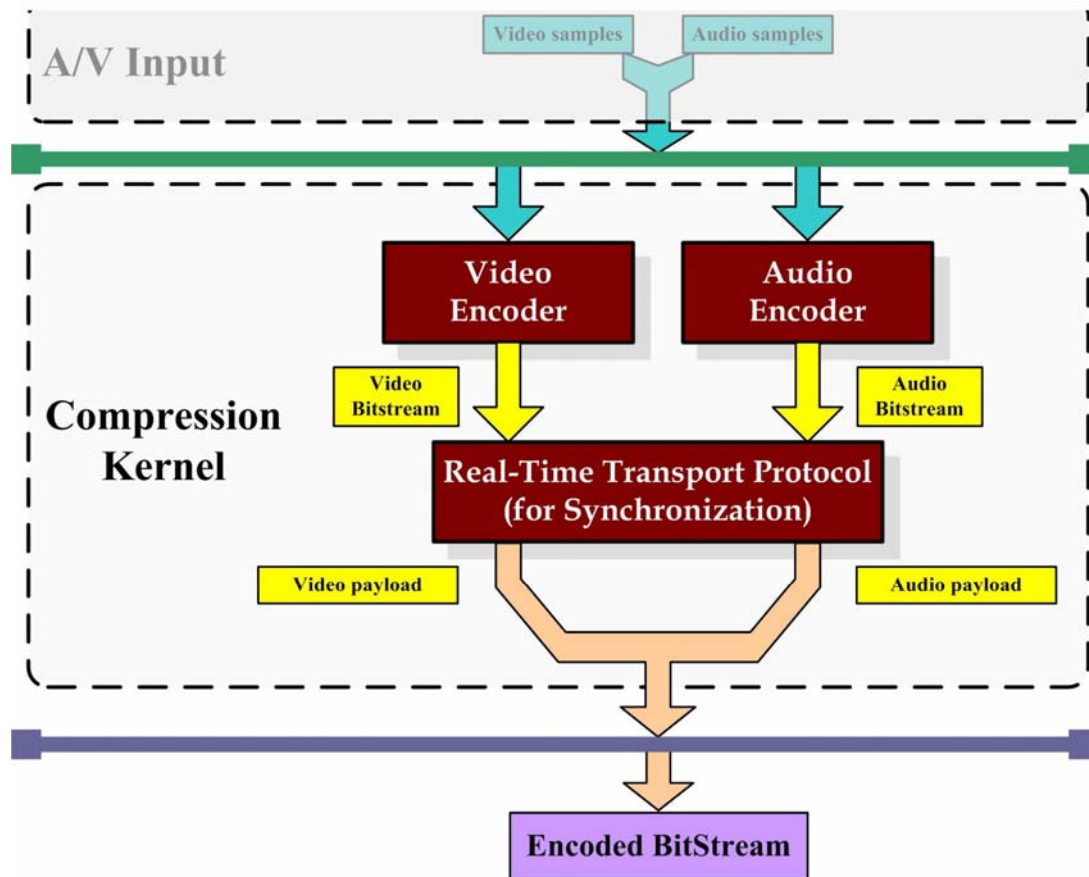


Fig. 3-6 The system overview of REMR

3.3.1. The Software Architecture of REMR

REMR integrates several software and hardware technologies, and the operating system (OS) is necessary for controlling the peripherals and managing the data flows. Embedded Linux is chosen as the OS for REMR since it is very stable and is suitable for the network transmission. The software architecture is shown in Fig. 3-7, and it is

divided into three layers, the application, the OS, and the hardware layer. The user programs, such as the data importer, the MP3 encoder, and the RTP packer, are in the same group, and OS will schedule their priority and allocate the resources they required.

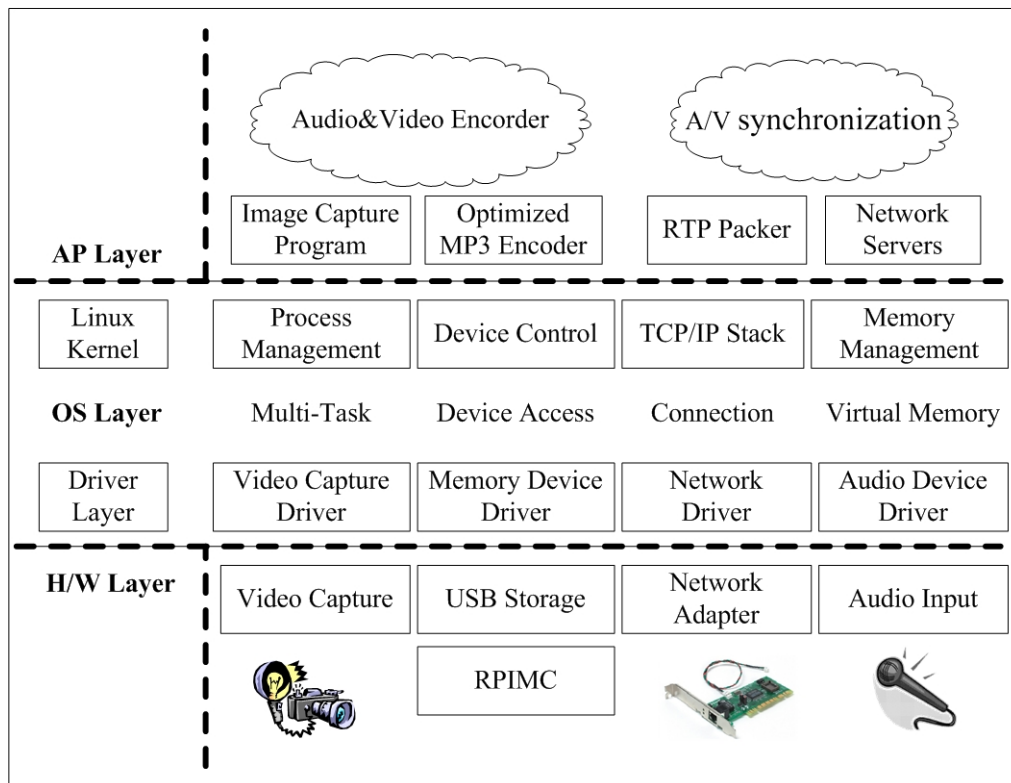
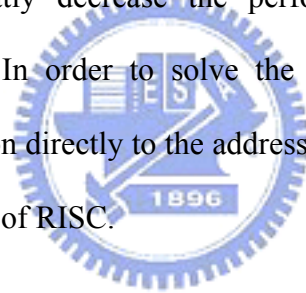


Fig. 3-7 The software architecture of REMR

The data importer is a platform dependent program which is responsible for input the pictures and the audio from the capture and the sound cards through PCI bus, respectively. When the image is ready, it will send a signal to RPIMC and start to compress it. On the other hand, the audio will be directly delivered to the MP3 encoder which is optimized for fixed-point RISC like ARM [73]. In the MP3 encoding algorithm, several approaches including the removal of psychoacoustic model, simplified iteration and fast rate control loop, and dynamic bit allocation

proportional to the energy of granules are adopted for optimization. Therefore, the ARM processor can perform the audio compression and other processes, such as RTP packer and several network servers at the same time.

The OS layer plays a very important role in REMR, and it is the bridge that connects the application and the hardware layers. In REMR, the hardware drivers are implemented in a special approach since the input data are not for general purpose. For example, the image capture driver is normally carried out to move the data to the memory area that is randomly allocated by OS, and then, the user applications can access them immediately. However, the hardware accelerator is always implemented to fetch the data at the specified address, so they need to be transferred again. It will consume lots of time, greatly decrease the performance, and even waste the computing power of RISC. In order to solve the serious problem, the driver is modified to put the information directly to the address that RPIMC desired to increase the speed and release the load of RISC.



3.3.2. The FPGA Prototype of REMR

ARM integrator is a platform that bases on a two-layer advanced micro-controller bus architecture, AMBA, and the hardware architecture of REMR is shown in Fig. 3-8, where $AMBA_i$ denotes the i -th layer of the bus matrix. $AMBA_1$ connects with ARM, SDRAM, and the PCI bus controller while $AMBA_2$ is built in FPGA and is combined with RPIMC and its AMBA-compatible wrapper, the SSRAM controller and an important direct memory access controller (DMAC). When REMR is working, SDRAM stores the output bitstream buffer and the whole software

including OS and the user applications, and the input frame from the capture card and the reconstructed frame are stored in SSRAM. The AMBA wrapper is the platform dependent hardware block, and as mentioned in Section 3.2, when the manufacturers want to port RPIMC into other platforms, they only have to modify the wrapper to fit their own communication bus and RPIMC will work fine. The two-layer bus matrix owns a major advantage which is that when the image data are ready for RPIMC in SSRAM, AMBA₁ will be free. ARM can fetch the audio data from the PCI bus, encodes it and put the bitstream into SDRAM through AMBA₁. As soon as DMAC moves the compressed video to SDRAM, ARM will combine the audio and the video information in the RTP format for storing in the local devices or transmitting through the internet.



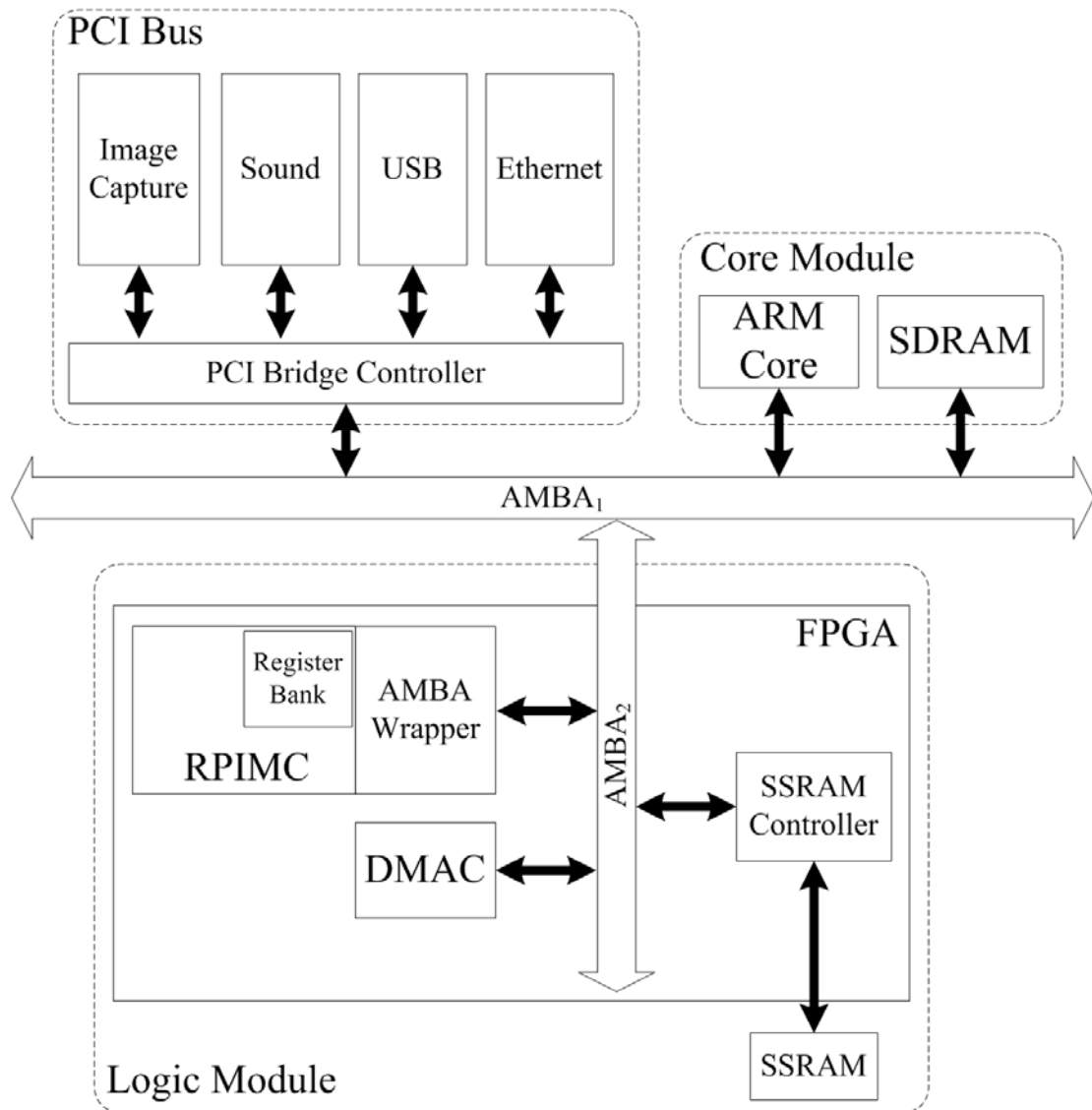


Fig. 3-8 The architecture of REMR in ARM Integrator

DMAC is designed to fit the specification of AMBA, and it can exchange the mass data over the bus. After the device driver delivering the real-time image to SSRAM, DMAC is waiting for the interrupt signal generated by RPIMC. DMAC will read the SRs to know the size of the output bitstream and moves them to SDRAM. When DMAC finishes this procedure, it will produce an interrupt signal to ARM. During the whole video encoding scheme, the input images and the compressed videos are processed automatically and put in the desired addresses to release the

heavy load of data exchanging for ARM. The optimized MP3 encoder and the RTP packer can be performed by a low operating frequency ARM to reduce the power consumption, so REMR is very suitable for consumer electronics.

3.4 Implementation Results

The hardware architecture of RPIMC, as described in Section 3.2, with the efficient HMEA [7][8], illustrated in Chapter 2, is successfully implemented. The VLSI circuits were described in VHDL and synthesized by SYNOPSIS Design Analyzer using UMC 0.18um CMOS standard cell library. The chip implementation results and the performance comparison will be depicted in this Section.



3.4.1. The Chip Implementation Results

The total gate counts of RPIMC are 204K gates, and it contains 6,462 bytes on-chip memory. Since the major features of RPIMC are that it is platform-independent and is programmable to encode videos with different resolutions and frame rate, RPIMC usually faces the situation of waiting the platforms. Therefore, RPIMC introduce two kinds of power saving techniques, sleep mode and clock gating. Sleep mode offers the greatest power savings to the user, and during this mode, RPIMC watches for a wake-up event which is asserted by the external pin. On the other hand, the clock gating can reduce a large percentage of the power since the logic activity in RPIMC is very high (~90%). Each clock of the blocks in RPIMC can be gated by the controller to manage the power down mode according to the

operations. The chip layout and the specification are shown in Fig. 3-9, and Table 3-2. When RPIMC is operating at 21 MHz to encode the images in CIF at 30 FPS, it consumes 262 mW. RPIMC has been confirmed that the maximum operating frequency of 140 MHz in a typical condition by the Shmoo Utility, and it means that RPIMC can perform the video compression up to the complexity of 1280x720 (HD) at 20 FPS which is enough for the usage of any consumer electronics and even surveillance systems.

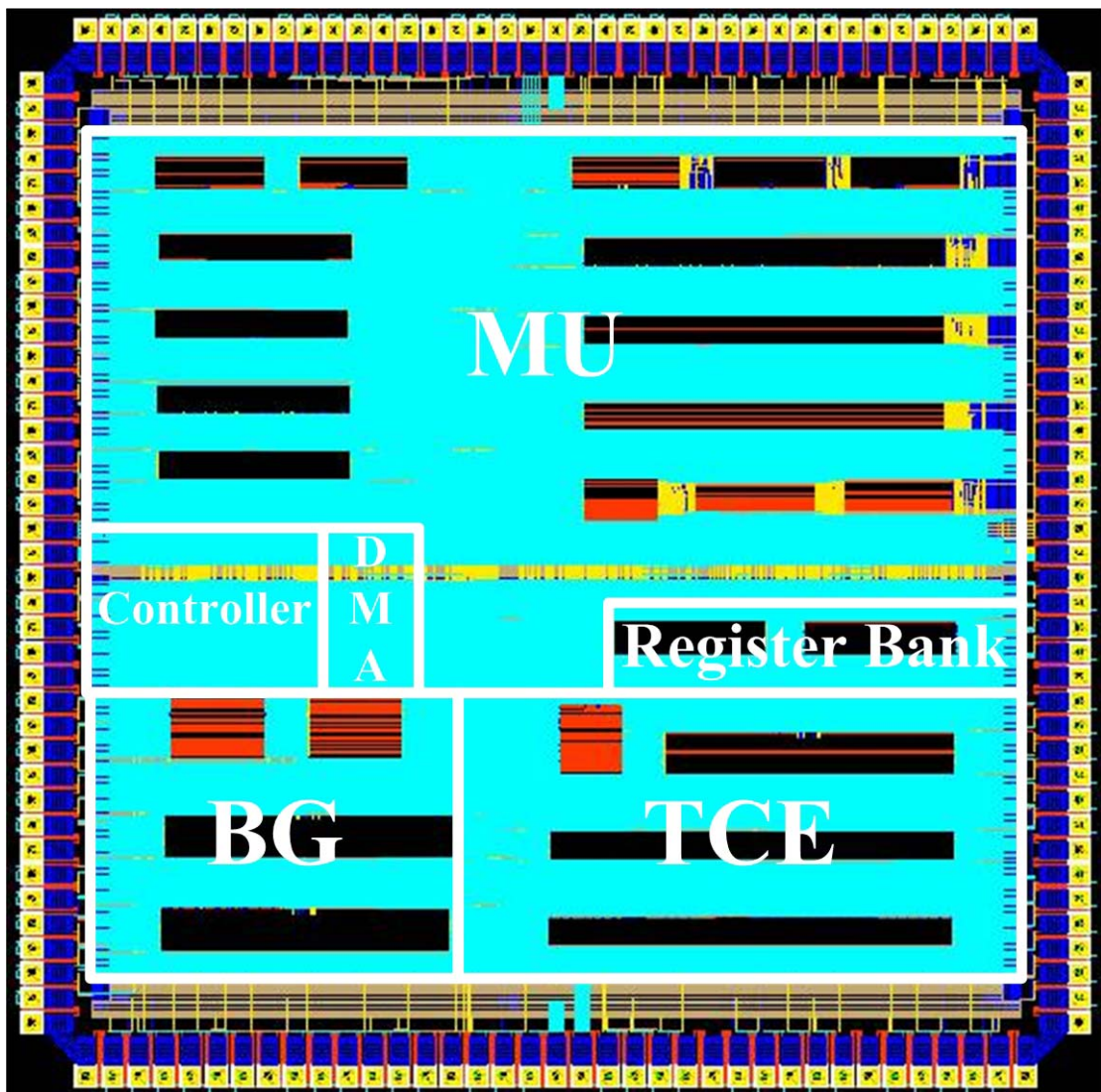


Fig. 3-9 The chip layout

Table 3-2 The chip specification of RPIMC

Supported image format	YUV 4:2:0
Supported VOP type	I frame & P frame
Supported image size	Programmable, Up to HD (1280 x 720)
Encoding frame rate	Programmable, Up to 30 frames/sec
Maximum operation frequency	140 MHz
Voltage	1.8 V
Power consumption	262 mW @ 21 MHz
Gate count	204K gates
On-chip memory	6462 bytes
ME Algorithm	HMEA, 4MV mode, Search range -16.0 to +15.5

3.4.2. The Performance Comparison

The encoding performance of RPIMC is compared with the XviD MPEG-4 software encoder using various test sequences provided by MPEG, and the result is shown in Table 3-3. It can be observed that the quality of RPIMC is a little less than XviD because the data precision of the hardware design is not as good as the software model, especially in the TCE block. However, the quality decrease of RPIMC is not easy to be distinguished by the human eyes.

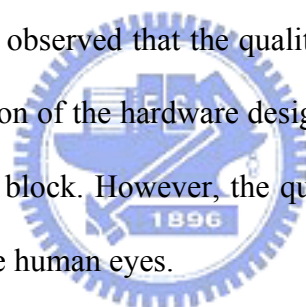


Table 3-3 The performance comparison between the software model and RPIMC in dB

Video sequences	XviD official version software encoder	RPIMC
Akiyo	42.38	41.45
Foreman	30.81	29.92
Table tennis	31.51	29.66
Mobile	21.31	20.93
Flower garden	22.25	21.88

Table 3-4 demonstrates the comparison between some MPEG-4 video encoders proposed before. In [67], it is full dedicated hardware video codec design, and it uses MVFAST for ME with search range equals to -16~+15.5 with extremely low operating frequency and power dissipation. However, the full dedicated design lacks of flexibility for future integration. The platform-based designs, including [74] and

[68]-[70], are in hardware/software co-design fashion with performance and flexibility. In [70], it adopts 3SS for ME with the search range of -32~+31.5, and several LSI, such as the logic for H.223, the speech DSP, and an 16-Mb embedded DRAM, are also integrated. Although it contains many functions, its die size is too large and it can only encode the images of QCIF at 15 FPS. As for [69] choose a coarse ME with search range -8~+7.5, and it contains an ARM embedded microprocessor and it consumes 500 mW to compute the images of CIF at 15 FPS. In the above designs, the encoding complexities are too low, and their gate counts are too high for consumer electronics. In [68], the cost-efficient video encoder SoC consumes 256.8 mW at 40 MHz and achieves real-time encoding of CIF at 30 FPS. All of these designs before only works for specific resolution of the input images, and are not easy to integrate to other platforms. RPIMC with the programmable register bank and the efficient HMEA [7][8] which can manipulate various image sizes, up to HD, and it is designed to be a platform-independent co-processor. In the viewpoint of video encoder part, RPIMC has the richest functionalities, the highest encoding complexity and the lowest cost.

Table 3-4 The performance comparison between other MPEG-4 chips and RPIMC

Designer	[67]	[68]	[69]	[70]	[74]	RPIMC
Encoding Complexity	CIF, 15FPS	CIF, 30 FPS	CIF, 15FPS	QCIF, 15 FPS	CIF, 30FPS	Programmable Up to HD, 20FPS
Operating Frequency (MHz)	13.5	40	27	60	36	Maximum 140
Power (mW)	29	256.8	500	240	N/A	262 @ 21 MHz
Gate Counts (K)	700	278	1,700	6,800 (DRAM)	170	204
Process (μm)	0.18	0.35	0.35	0.25	0.18	0.18

Chapter 4. An Adaptive Motion Estimation Algorithm for Vehicle Surveillance Videos

This chapter addresses the development of AMEA especially for the vehicle surveillance videos. In the real environments, the videos contain several problems, such as the different weathers, and the changing of the light sources, and these will lead to the heavy computation and the bad video quality. On one hand, in order to overcome the drawbacks, AMEA applies an easy HEA to pre-process the images to keep the images stable and increase the estimation accuracy greatly. On the other hand, the purpose for the monitoring information is to provide clear faces of the intruders, so the images are separated into two parts, in and out of ROI. In ROI, AMEA can maintain the coding quality using the least numbers of manipulations, and meantime it can also enormously reduce the computational complexity for the blocks out of ROI.

4.1 Introduction

The most complex part of popular video compression standards, including MPEG-1/2/4 [1][3][4] and H.261/3/4 [2][5][6], is ME. The goal of ME is to remove the temporal redundancies existing in adjacent frames, and the block-matching algorithm is used as a method for most of the video coding systems. It is used to find a block which is most similar to a current block within a pre-defined search area in a reference frame, and it dominates the encoded image quality, the compression ratio, and the computation time. The most straight forward method, known as the FSBMA

of obtaining MVs, is to search all possible locations within a given area. Since FSBMA uses an exhaustive search to locate the minimum block-distortion measure (BDM) for each candidate block, it provides optimal performance but at the expense of very high computation. Usually, FSBMA spends about 70% of the total encoding time and this is the reason why FSBMA is not used in real-time systems. Indeed, ME is the major bottleneck in video coding applications, hence the need for faster algorithms.

To reduce the number of search steps of FSBMA in order to increase the overall speed is essential. The fast FSBMA, including the 2-D logarithmic search (2DLOG) [75], the three-step search (TSS) [45], the new three-step search (NTSS) [46], the advanced center biased search [76], the four-step search (4SS) [47], the cross-search [77], the prediction search [78], the successive elimination algorithm (SEA) [33]-[35], partial distortion elimination (PDE) [36], the winner-update algorithm [37], the advanced diamond search algorithm (DSA) [38], and the hexagon-based search (HEXBS) [79], [80], are proposed to reduce the computational heavy load of FSBMA while maintaining its quality. DSA has achieved a significant speed gain by considering diamond-shaped search patterns instead of the conventional square ones with a view to approximate the optimal (but unrealizable) circular shape as closely as possible. Recently, HEXBS algorithm has surpassed the speed of DSA by using hexagon-shaped search patterns. Moreover, in order to refine the accuracy of DSA, several new algorithms, such as motion vector field adaptive search technique (MVFAST) [41], predictive MVFAST (PMVFAST) [42], and enhanced predictive zonal search (EPZS) are proposed [43]. MVFAST improves DSA in both terms of visual quality and speed up by initially considering a small set of predictors. Unlike DSA where only a large moving diamond pattern was considered, MVFAST also introduced a smaller moving diamond. PMVFAST uses basically the same

architecture and patterns as MVFAST does, but a significant difference of PMVFAST compared to MVFAST is the way the small versus the large diamond is selected. Dissimilar to MVFAST where motion was characterized as low, medium, or high by considering the largest motion vector candidate, in PMVFAST a different selection strategy, which can improve the overall speed of the algorithm by using the large diamond less often, is used. Furthermore, EPZS that improves upon PMVFAST by considering several other additional predictors in the generalized predictor selection phase of PMVFAST. EPZS also selects a more robust and efficient adaptive threshold calculation where as, due to the high efficiency of prediction stage, the pattern of the search can be considerably simplified. In addition, an efficient hierarchical motion estimation algorithm adopting the averaging filter to downsample the image in order to reduce the computational complexity greatly while maintaining the good coding quality are also proposed, but it is a hardware architecture, which can not gain the same performance using the software implementation [7][8].

All of these fast algorithms [33]-[38][41]-[43][45]-[47][75]-[80] assume that either the error surface is unimodal over the entire search area (i.e., there is only one global minimum) or MV is center-biased. These hypotheses essentially require that either BDM increases monotonically as the search point moves away from the global minimum position or MV exists in a small range. However, they are generally invalid for many real video sequences because the highly nonstationary characteristics of the signals. Moreover, despite their respective differences, these fast search algorithms all have one common feature that none of them has been designed to provide flexibility in controlling the performance in terms of predicted picture quality and processing time. Golam et al. propose a fully adaptive distance-dependent thresholding search (FADTS) [81], which can adjust the required threshold control parameter via an

adaptive process which uses the information from previous frames to achieve specified image quality and manipulation speed. However, it needs to be modified when dealing with the vehicle surveillance videos.

As illustrated in Fig. 4-1, the images of the vehicle surveillance videos have some key features. First, the important information, like the faces of the drivers and the judgments of the existence of the intruders, usually locates in a specific area, ROI, and other parts of the image, out of ROI, often do not contain significant data. The coding results in ROI can be as clear as possible in order to provide evidences for the police department, and the complexity also should be kept as low as it can. Second, since the vehicles are frequently in the moving state, and the surrounding environments keep changing, the parameters of the images, such as brightness and the light source, are varied case by case. The gray level of the pixels will be influenced and the accuracy of the MVs will be forced to be lower due to the undesired environment effects. Third, the postures of the motormen do not move wildly when they are driving. Therefore, the ME framework for the specific application should be modified to increase the performance in terms of the coding speed and the image quality.

The main contributions of this chapter are to develop AMEA especially for the surveillance videos in vehicles. In ROI, AMEA can do its best to maintain the encoding quality, using the least number of operations, and it can adaptively change the search procedures to overcome the big motions and the changes of the environments. Therefore, a HEA is adopted to pre-process the blocks in order to overcome the lighting effects. Furthermore, since the MVs of these out-of-ROI blocks are usually not large, and the importances of them are not high, AMEA can

manipulate them as fast as possible by optimizing the search procedure according to the MV characteristic. The experimental results show that AMEA can even provide better quality than FSBMA does when the video is seriously influenced by the environmental changes.



Fig. 4-1. The vehicle surveillance image with ROI.

4.2 The Analysis of The Vehicle Surveillance Videos

The special features of the vehicle surveillance videos will be addressed in this Section. One is the lighting effects, and the other is the characteristics of MVs in and out of ROI. As shown in Fig. 4-1, the camera is installed in the front-right corner of the vehicle since not only the face of the driver but also the scene outside the window, which can help the police to recognize the exact location, is important. The detail analyses are described as below.

4.2.1. Lighting Effects

Lighting conditions are one of the most important problems to be solved in real tests. As shown in Fig. 4-2, the vehicle is passing through a long tunnel, which has rectangular windows on one side, so the sunlight will influence the brightness of the images periodically. When passing the middle of the window, outer lights directly illuminate the driver, increasing the pixel levels quickly. Once the car has left, the pixel values decline immediately. Fig. 4-3(a) and Fig. 4-3(b) illustrate the brightest and the darkest images, respectively, and the time difference between them is only 360 milliseconds. In these two pictures, it can be observed that the driver's posture only changes a little, but the pixel values of them are quite different. Another method to exam the effects is to compare the mean, \bar{M}_k , defined in (4-1), of the k^{th} frame in the whole video,

$$\bar{M}_k = \frac{1}{W \times H} \sum_{i=0}^{W-1} \sum_{j=0}^{H-1} I_k(i, j), \quad (4-1)$$



where W, H are the width and the height of the image $I_k(\cdot)$, and $I_k(i, j)$ represents the value at position (i, j) of the k^{th} frame, respectively. The mean of the each frame in this video (V1) is depicted in Fig. 4-4, and the lighting effects can be easily observed. Fig. 4-5(a) and Fig. 4-5(b) also shows the most luminous and the obscurest frames of another video (V2), respectively, and the position of the vehicle is closer to the windows than it of V1. Fig. 4-6 depicts the average value of all pixels of each frame in V2, and it is higher than it of V1 due to the location of the automobile. The lighting effect appears to be smaller than it of V1, but the trend of the diagram is almost the same. The serious problem exists and needs to be solved.



Fig. 4-2. The tunnel which will cause lighting effects.



(a)



(b)

Fig. 4-3. (a)The brightest image, (b)The darkest image of V1.

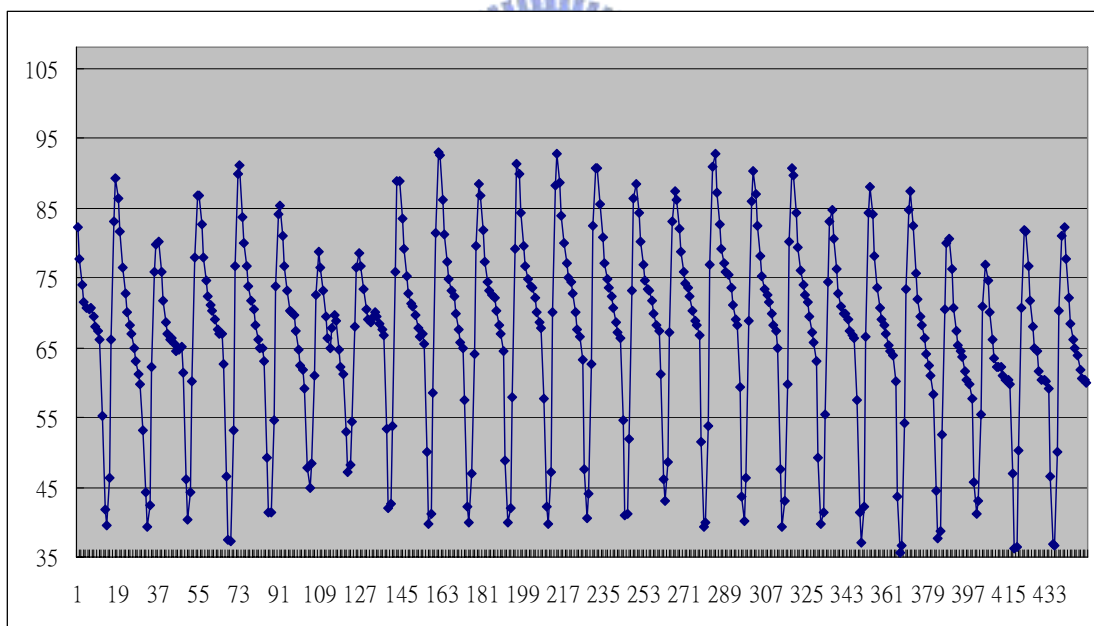


Fig. 4-4. The \bar{M}_k of each frame of V1



(a)



(b)

Fig. 4-5. (a)The brightest image, (b)The darkest image of V2.

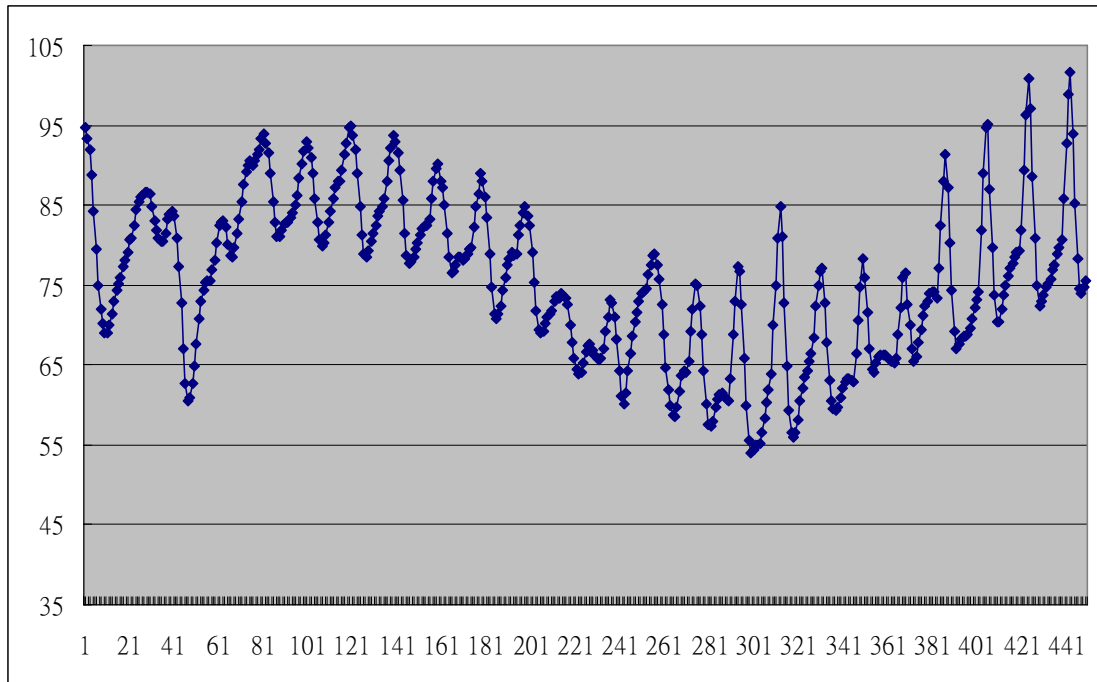


Fig. 4-6. The \bar{M}_k of each frame of V2

4.2.2. The Coding Performance Degradation due to the lighting effects

As described in Section 4.1, when using the block-matching algorithm, MVs are obtained by locating the minimum BDM, which is usually measured by the sum of absolute difference (SAD), for candidate blocks. However, the lighting effect will cause the pixel values to change greatly even though the motion of driver is small. Thus, the MVs manipulated by the minimum SAD will be wrong and the quality will be decreased. The effect can also be observed in the correlation coefficient, ρ' , which is defined as

$$\rho' = \frac{\sum_i \sum_j [(I_k(i, j) - \bar{M}_k) \times (I_{k+1}(i, j) - \bar{M}_{k+1})]}{\sqrt{\sum_i \sum_j (I_k(i, j) - \bar{M}_k)^2} \times \sqrt{\sum_i \sum_j (I_{k+1}(i, j) - \bar{M}_{k+1})^2}}, \quad (4-2)$$

where \bar{M}_k and \bar{M}_{k+1} are the means of the image pairs, $I_k(\cdot)$ and $I_{k+1}(\cdot)$, and

$I_k(i, j)$ represents the value at position (i, j) of the k^{th} frame, respectively. Fig. 4-7 and Fig. 4-8 show the correlation coefficients between the adjacent frames of V1 and V2, and the results are highly related to Fig. 4-4 and Fig. 4-6, respectively. The quality will be increased when higher correlation coefficient exists between the successive images.

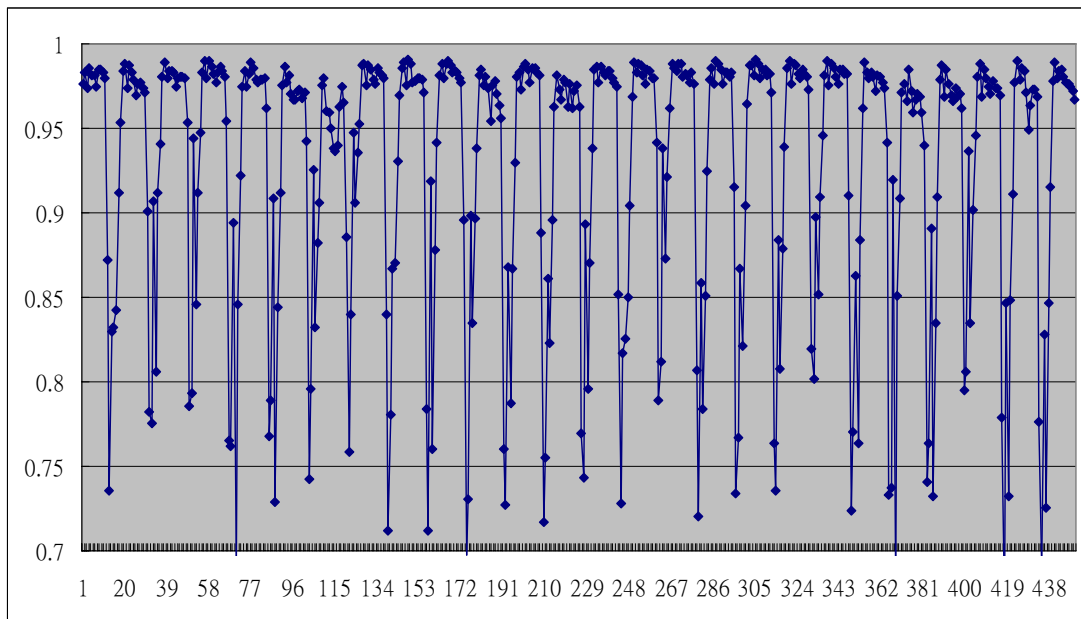


Fig. 4-7. The correlation coefficients between each frame pairs of V1

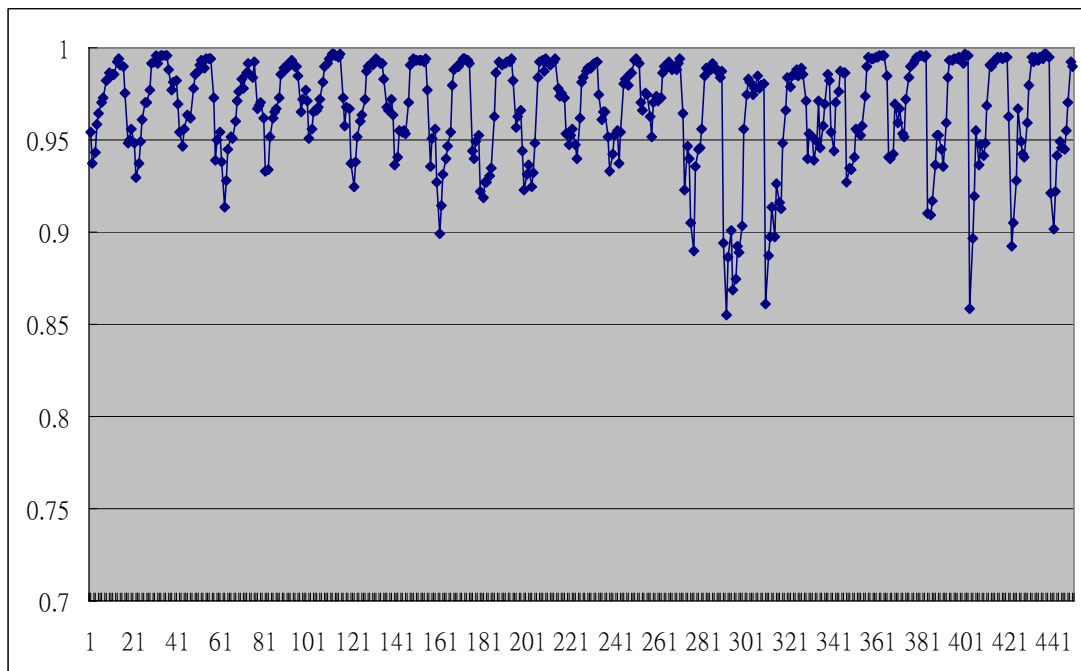
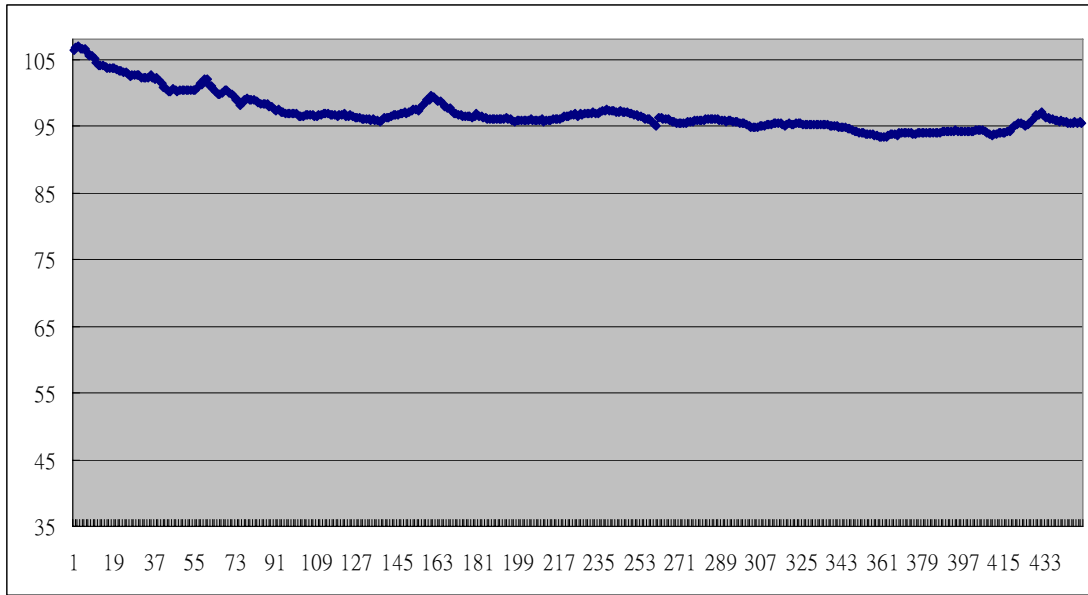


Fig. 4-8. The correlation coefficients between each frame pairs of V2

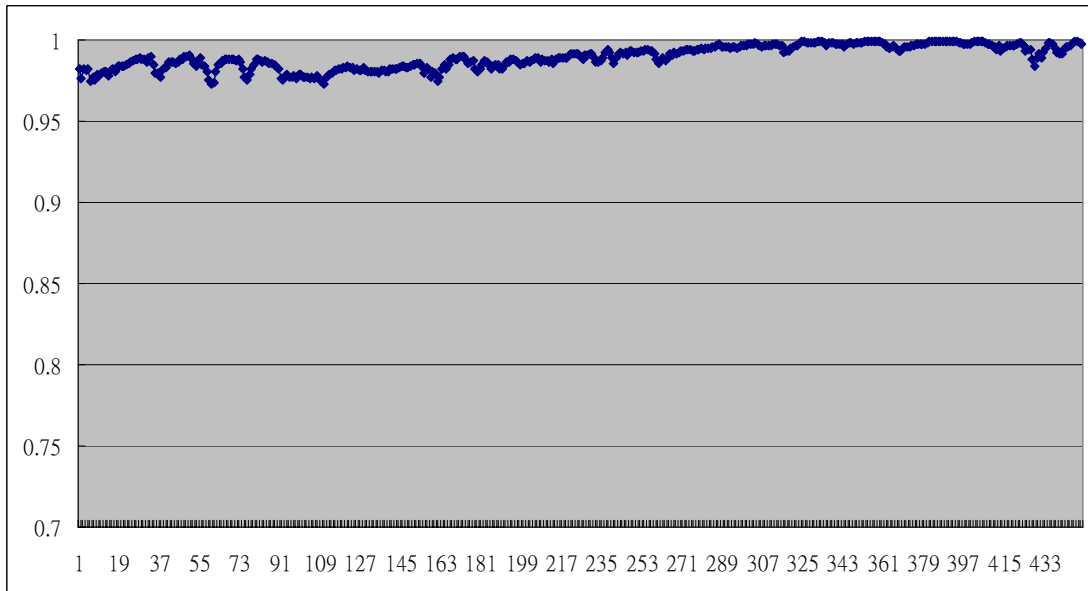
If the low correlation coefficient exists in a neighboring frame pair, it means that the distortion between them is large, and will lead to poor quality and compression ration. The PSNR is used for the measurement of performance, and it is defined as (2-15). Table 4-1 illustrates the PSNR of these videos, where V0 is a surveillance video when driving on a normal road without the lighting effects. The ME method is FSBMA, the block size is 16×16 and the search area is $[-16, +15]$. These videos, V0, V1, and V2, are all in CIF format, and a total of 450 frames. Fig. 4-9(a) and Fig. 4-9(b) are the mean and the correlation coefficients of V0, respectively. The diagram shows that a normal motorist has quite small motion during driving and PSNR is much higher than those of V1 and V2. However, V0, V1, and V2 are filmed at the same angle, with the same driver, in the same vehicle, and on the same day with only five minutes time interval each, and the behavior of the same person will not change enormously in a short time. Therefore, from Fig. 4-4 to Fig. 4-9 and Table 4-1, the lighting effect indeed results the bad estimation performance, and this problem needs to be solved.

Table 4-1 The encoding quality comparisons of the different videos in dB.

Video Sequence	PSNR
V0	34.09
V1	25.02
V2	30.15



(a)



(b)

Fig. 4-9. (a) The \bar{M}_k of each frame ,(b)The correlation coefficients between each frame pairs of V0

4.2.3. The MV characteristics

In Section 4.2.2, the hypothesis of the drivers' behavior is made, and some statistical results will support the assumption. First of all, because the camera of the vehicle surveillance system is firmly fixed in the car, and the capture area will be also

bounded, ROI can be pre-defined. ROI should cover almost the moving region of the driver's head to recognize the intruder's face, so the ROI like Fig. 4-1 is set artificially. Fig. 4-10 shows the manipulated MVs, which is drawn in the shape of arrows, on the reference frame and the area besieged by the green dotted line is ROI. Fig. 4-11(a) and Fig. 4-11(b) depict the probability of the magnitude of MVs in and out of ROI for all frames of V0, respectively. It can be observed that there exists high chance of short MVs when its corresponding block is out of ROI, so the ME algorithm for these blocks can be modified based on the results. On the other hand, the drivers' heads will move to all directions in order to check different angles, and their faces will turn left and right to see the side view. Therefore, BDM in ROI will be larger than it out of ROI, and hence the MVs in ROI will have high probability to be longer. To maintain the coding quality of the blocks in ROI, which will have much more situations than those out of ROI, is essential since the main goal of the surveillance videos is to provide clear evidences.

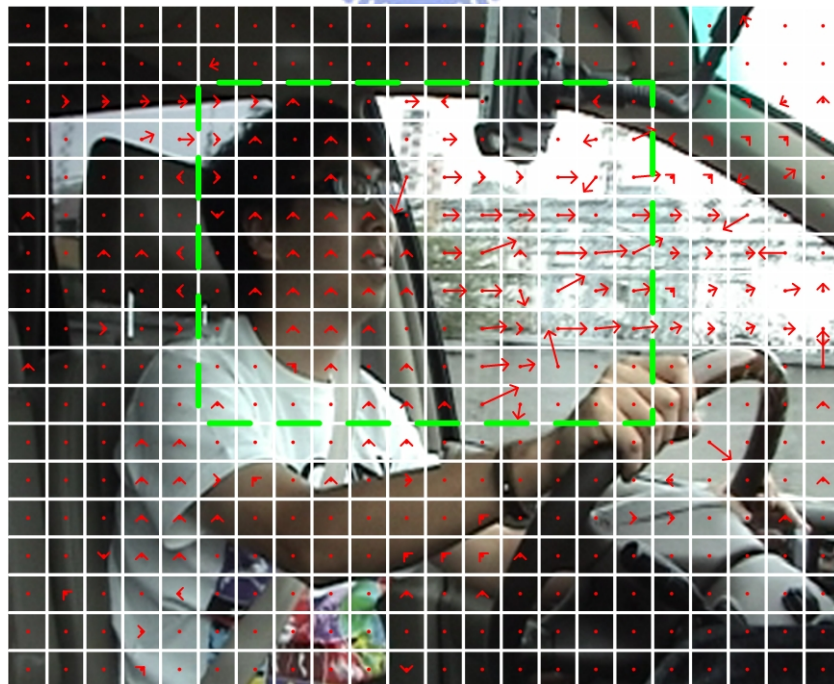
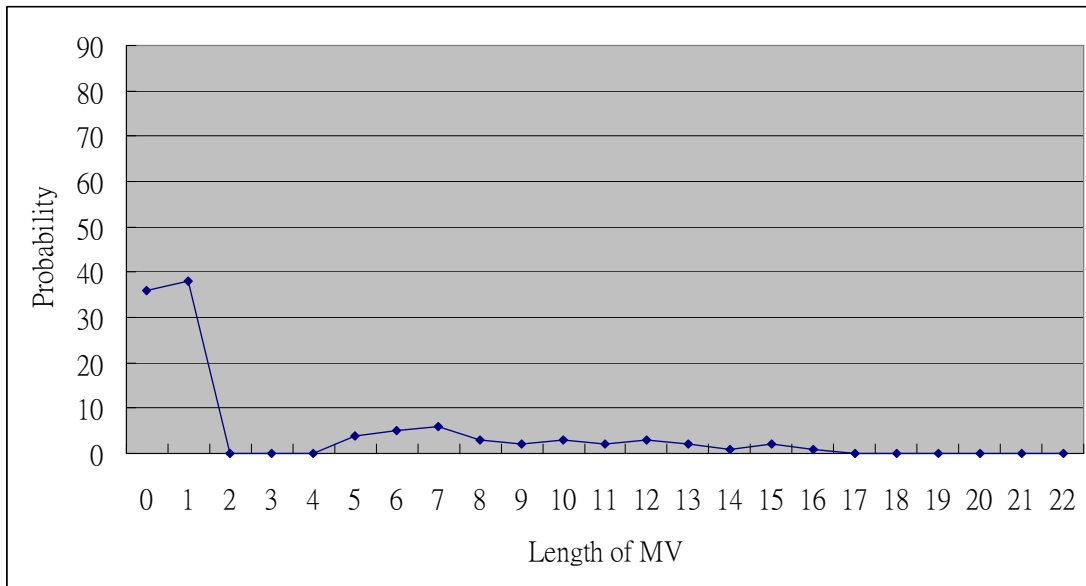
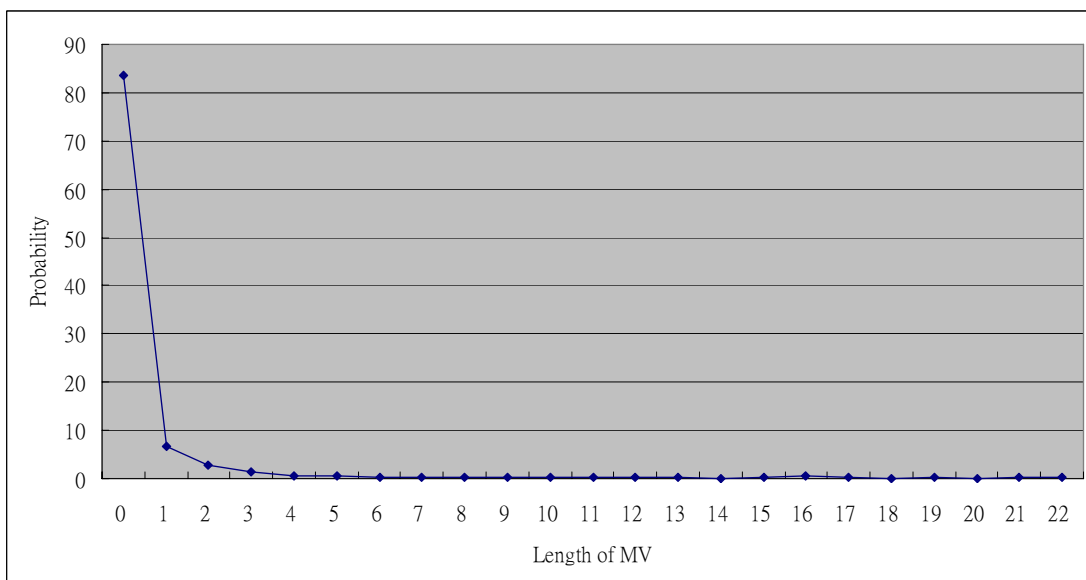


Fig. 4-10. The manipulated MVs



(a)



(b)

Fig. 4-11. (a) depicts the probability of the magnitude of MVs in and (b) out of ROI for all frames of V0.

4.3 Adaptive Motion Estimation Algorithm

AMEA can be divided into three parts. One is HEA for pre-processing the image, one is the adaptive ME framework for the blocks in ROI to uphold the image quality,

and the other is the simplified MV search procedure for those out of ROI. The complete algorithm is described as below.

4.3.1. Histogram extension algorithm (HEA)

The lighting effect is a serious problem for the image processing for vehicular technology. Bergasa et al. propose a real-time system for monitoring driver vigilance [82], and they are facing the same issue. In order to minimize the interference from light sources beyond the IR light emitted by the LEDs in [82], a narrow bandpass filter centered at the LED wavelength has been attached between the CCD and the lens. However, the solution is not suitable for the video encoding system since the types of input devices are quite different. For minimizing the BDM due to the numerous reasons except from the motion, the brightness preserving algorithm should be adopted. Several methods, including histogram equalization [83], Brightness preserving bi-histogram equalization [84], equal area dualistic sub-image histogram equalization [85], minimum mean brightness error bi-histogram equalization [86], and brightness preserving histogram equalization with maximum entropy [87] are proposed to reduce the artifacts. These frameworks are very powerful to overcome the different light sources at any angle, and are quite complex. Since EMVSS is put in the real vehicle and works whenever the car is started, it must be a low-power embedded system. Therefore, a simple HEA is derived to fit the request and release more time for CPU to processing the H.264 video compression.

HEA pre-processing is addressed to adjust the images which are under different luminance to the ones with a similar luminance condition. A color image can be

separated into three dimensions, Y, U, and V, and ME only calculates the MVs for Y plane. Therefore, only the histogram of the Y component needs to be extended, and the linear normalization with mean shift (LNMS) is applied to modify the images of different conditions to a similar one.

As illustrated in Fig. 4-12(a) and Fig. 4-12(b), x_i and x'_i are the intensity of a pixel value of the gray-level image, $I_k(\cdot)$, before and after the HEA manipulation, where i is from 0 to 255, s_i is the total amounts of the x_i -th level and $a, b, c, d, e,$ and f are random numbers with the relationship of $a < b < c < d < e < f$, respectively. Fig. 4-12(a) and Fig. 4-12(b) are the histograms of the input frame before and after the pre-processing. Suppose the mean of Fig. 4-12(a), \bar{M}_k , which is calculated by (4-1), is located on $x_{\bar{M}_k}$ with $s_{\bar{M}_k}$ pixels, and HEA will shift the value of all these $s_{\bar{M}_k}$ points to 128. In another word, $x_{\bar{M}_k}$ is moved to x_{128} , and the remaining pixels will be linearly shifted to the two sides of x_{128} . Two transform parameters, α and β , are used to move the x_i to x'_i , and they are defined as

$$\alpha = \frac{128}{x_{\bar{M}_k}}, \quad (4-3)$$

and

$$\beta = \frac{128}{255 - x_{\bar{M}_k}}. \quad (4-4)$$

The transfer function from x_i to x'_i is defined as

$$\begin{cases} x'_i = \alpha \times x_i, & 0 \leq i < \overline{M}_k \\ x'_i = \beta \times x_i, & \overline{M}_k < i \leq 255. \\ x'_i = 128, & i = \overline{M}_k \end{cases} \quad (4-5)$$

By (4-5), \overline{M}_k will be shifted to around 128, x_i , $0 \leq i < \overline{M}_k$, will be modified as x'_i , $0 \leq i < 128$, and x_i , $\overline{M}_k < i \leq 255$, will be substituted as x'_i , $128 < i \leq 255$.

Therefore, the average value of each input image, which can be influence by the light source, will be around 128, and the lighting effect problem can be solved. Moreover, the MV search procedure can avoid the wrong minimum BDM, and can reduce the search points to increase the speed.



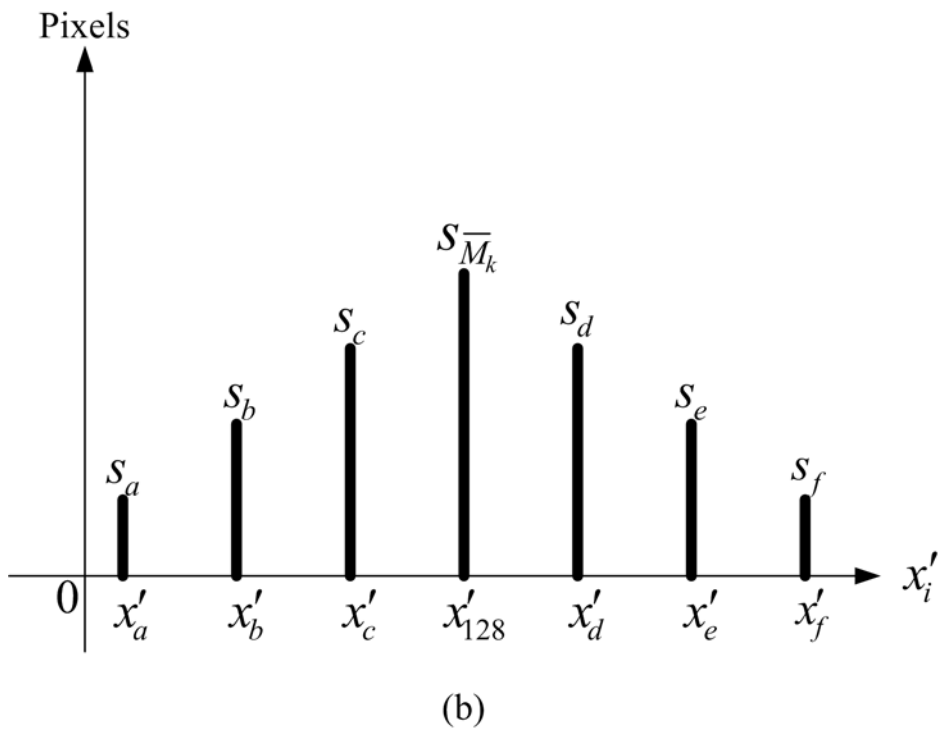
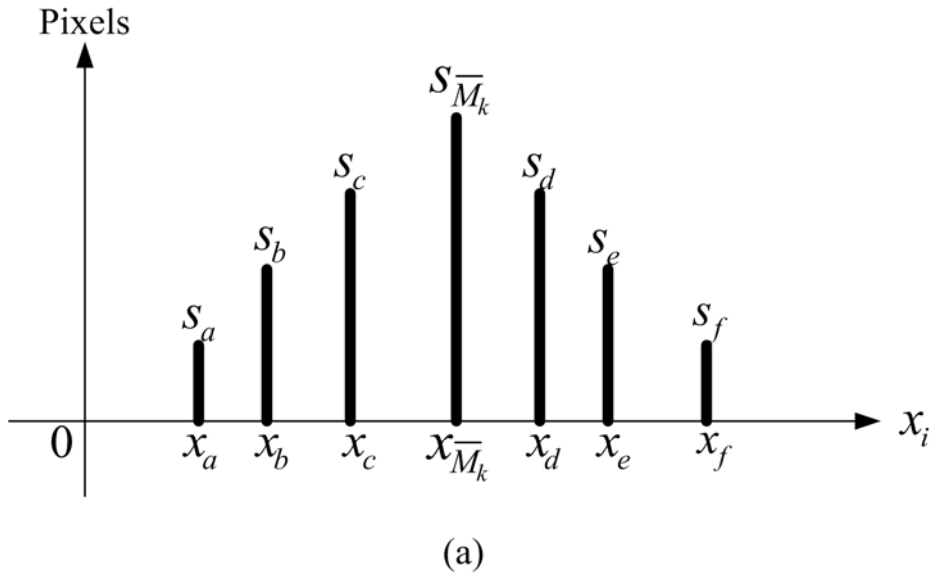


Fig. 4-12. (a) the histograms of the input frame before and (b) after the pre-processing.

4.3.2. AMEA in ROI

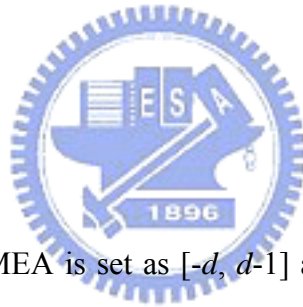
In performance-management ME, given a target prediction image quality in terms of average SAD per pixel, the motion search algorithm tries to achieve it using as few search checking points as possible. Performance-management ME also

assumes real time constraint, which allows very limited number of passes per MB. Without such a constraint, trivial trial and error technique with a very high number of passes would suffice the adaptation. Without any loss of generality, this paper assumes the strictest constraint where only one ME pass is performed per MB.

A. AMEA framework

The concept of AMEA is not linked to any specific search pattern shape. In the wake of improved speed gain by nonsquare search patterns, AMEA has been implemented using search diamonds, SD_τ , as shown in Fig. 4-13 where the number of checking points in SD_τ is

$$\begin{cases} 1 & , \quad \tau = 0 \\ 4\tau & , \quad \tau = 1, 2, \dots, 2d \end{cases} \quad (4-6)$$



where the search range of AMEA is set as $[-d, d-1]$ and SD_τ represents the MV of length in the range of $[\tau/\sqrt{2}, \tau]$. Some of the checking points in the search diamond fall outside the search windows that are obviously ignored.

Like all block-base ME search techniques, AMEA starts at the center of the search space. The search then progresses outwards by using SD_τ in order while monitoring the current minimum SAD. A parametric thresholding function, r , defined as

$$r(\tau, Th_m) = Th_m \times \tau, \quad (4-7)$$

is used to determine the various thresholds to be used in the search involving each SD_τ where Th_m is an adaptive index for r of the m -th frame (the adaptation of Th_m

and the definition of m will be addressed in the next paragraph), and it is set at the start of each search and acts as a control parameter. After searching each SD_τ , the current minimum SAD is compared against the threshold value, $r(\tau, Th_m)$, of that specific search procedure in SD_τ and the search is terminated if this SAD value is not higher than that threshold value.

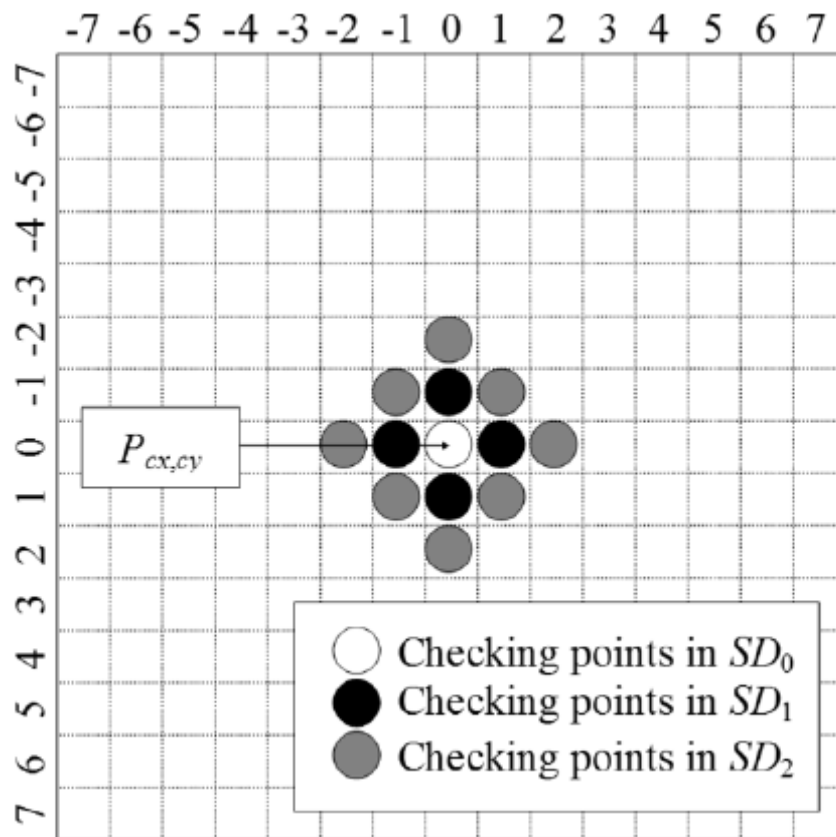


Fig. 4-13. AMEA search diamonds SD_0 , SD_1 and SD_2 .

B. AMEA closed-loop adaptation model

AMEA works sequentially on the frames of an input video sequence. Although the consecutive frames are considered to be highly correlated, the input video signal can be considered time variable or nonstationary from the adaptation point of view. Therefore, a closed-loop adaptation model, as shown in Fig. 4-14, is presented for

AMEA, and it has the following four modules.

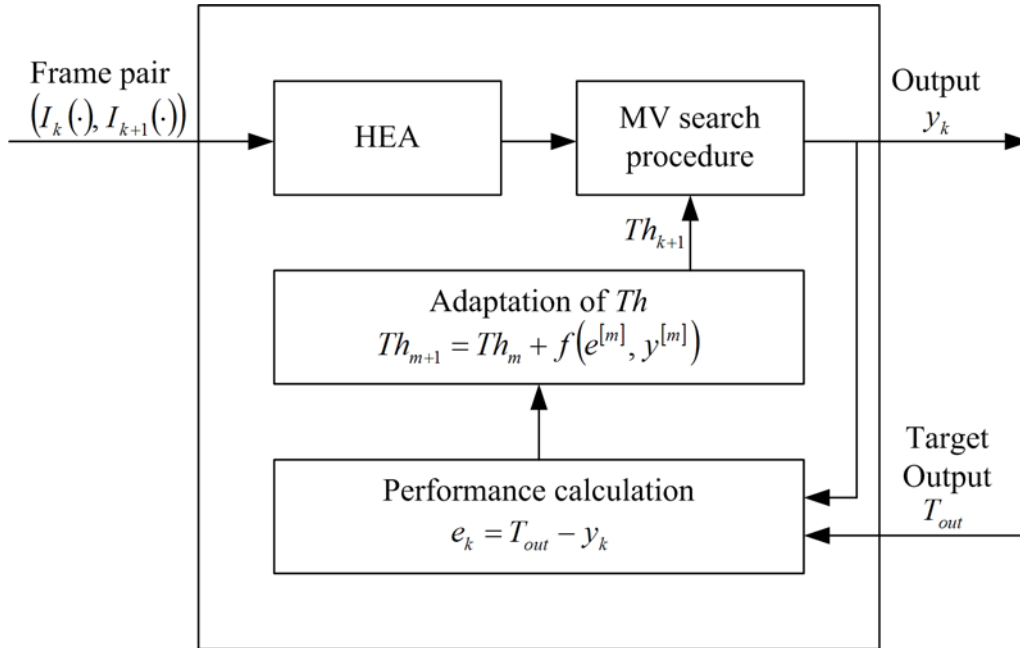


Fig. 4-14. Closed-loop adaptation process for AMEA.

- Adaptation of Th_m : Since the adjacent frames have high correlation coefficients, the number of updating the Th_m can be reduced. In other words, the Th_m will maintain the same value during L frames, and the iteration index m are changing from 1, 1+ L to $\lfloor (N-1)/L \rfloor L$ for a video sequence with N frames. The value of Th_{m+1} for the next iteration are defined as

$$Th_{m+1} = Th_m + f(e^{[m]}, y^{[m]}), \quad (4-8)$$

where $e^{[m]}$ and $y^{[m]}$ are the average value of each e_k and y_k , which are the error signal and the average SAD output of the k^{th} frame, in a total of L frames, respectively.

- HEA: This module will manipulate the histogram of the input image, and then shift the histogram using HEA, which is described in Section 4.3.1. The results

of HEA will directly replace the original frame to reduce the memory usage. Two blocks of the memory, which are acted as a ping-pong buffer, Mem₁, and Mem₂, are allocated for AMEA. First, in the intra frame mode, the input images are always stored in Mem₁, and the reconstructed frame will be saved in Mem₂. Second, in the inter frame mode, since the current frame will be the reference frame in the next encoding loop, these two parts will switching their status mutually until the next intra frame mode, and the scheme is illustrated in Fig. 3-5.

- MV search procedure: This module calculates MVs using the AMEA framework, which is depicted in Section 4.3.2.A. The inputs of the module are the video frame pair $(I_k(\cdot), I_{k+1}(\cdot))$ and the control parameter Th_m . The output of the model y_k is the predicted image quality in terms of average SAD and it is a monotonically increasing function.
- Performance calculation: The performance of the adaptive system are estimated by calculating the error signal, e_k , as

$$e_k = T_{out} - y_k, \quad (4-9)$$

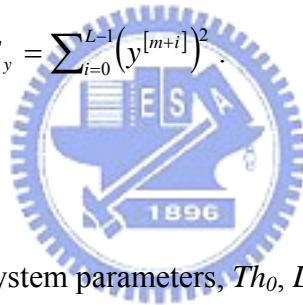
where T_{out} is the target output, defined by the users. The value of e_k must be minimized as the adaptation process progresses.

The performance of an adaptive system largely depends on how the function $f(e^{[m]}, y^{[m]})$ is defined. A few gradient search algorithms [88]–[92] exist that can adapt a system in searching for the optimal parameter to minimize error signal in (4-9). Among these, the least mean square (LMS) is the most well-known and popular method for its computational simplicity, robustness, and relatively easy

implementation for online estimation of time-varying system parameters. A number of variants on the LMS theme have been conceived in order to ratify potential problems of the original LMS algorithm such as the need to guess the best value of step size, slow convergence, and numerical instability. The normalized block LMS (NBLMS) [92] is considered as the best option for automatically adjusting the control parameter Th in order to achieve a target average SAD while coding a video sequence, where this sequence can be considered as a time varying nonstationary input to the adaptation system. Based on NBLMS, the threshold control parameter is updated as

$$Th_{m+1} = Th_m + ue^{[m]} \frac{\frac{1}{L} \sum_{i=0}^{L-1} y^{[m+i]}}{E_y}, \quad (4-10)$$

where u is the step size and $E_y = \sum_{i=0}^{L-1} (y^{[m+i]})^2$.

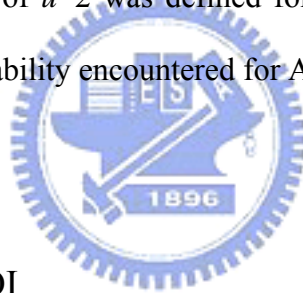


C. The initializations of the system parameters, Th_0 , L , and u

In FADTS, the initialization of Th_0 is chosen by estimating the first frame pair using a normal method, like FSBMA, DSA, or etc., and performs the MV search procedure again after Th_0 is decided. The flow is achievable when the powerful PC is used to develop the algorithm when the processing framerate is higher than 60 frame per second. Thus, Th_0 is appropriate for the video sequence since the frame pair for estimation is still the same. However, when the low computing power embedded system is chosen in order to save the consuming energy, Th_0 will be calculated after a period of time, and it may not suitable for the current frame pair. Therefore, Th_0 is directly set to 0, and it will be updated to increase the speed after the first L frames. The parameter L for AMEA is defined as 4 since the most serious lighting effects

usually take 15 to 20 frames to change from the darkest image to the brightest one. Although lower L also provided similar performance in satisfying the targets, according to (4-8), it increases the overhead computational cost for the adaptation process. Conversely, higher L can be considered in order to reduce the overhead cost, though the block length of L in the NBLMS algorithm cannot be too high if it is assumed that the content of a video sequence may be unstable.

With regard to the parameter u , Meghriche et al. [93] highlighted that there is no universal solution for finding the optimal value of u . In [92], the NLMS algorithm considers a step size range of ($0 < u < 2$) for signal processing applications. The lower the value of u , the slower the convergence rate while a high step size can lead to system instability. The value of $u=2$ was defined for all the various standard video sequences tested, with no instability encountered for AMEA.



4.3.3. AMEA out of ROI

As mentioned in Section 4.2.3, the probability of the MVs whose lengths are less than 2 is much higher than the MVs with other magnitudes, so the ME algorithm can be greatly modified based on these results. Moreover, since the instruction memory of EMVSS is quite small and the computing power of the embedded processor is low, the software optimization for reducing the code size and increasing the performance is necessary. Therefore, the programs for manipulating the MVs out of ROI should re-use the functions in Section 4.3.2 well to achieve the needs. In the whole searching procedure, the most frequently used part is the diamond-shaped SAD comparator, hence it is written by assembly codes efficiently. No matter the incoming block is in

or out of ROI, the high speed calculations are performed according to the system parameters. The main differences between AMEA in and out of ROI are the stop conditions and the search ranges. Because the importance of the blocks out of ROI is not as high as those in ROI, the maintenance of the estimation quality is no longer essential. Thus, the adaptations of the encoding parameters can be removed and the stop condition can be the same as DSA to decrease the complexity. Furthermore, based on the analysis results in Section 4.2.3, the search range of these out-of-ROI blocks can be shrunk to $[-2, +2]$. With these prior techniques, the computational overhead can be much lower.

4.3.4. The complexity analysis of AMEA

Consider an ME system with the following parameters: frame size = $W \cdot H$ pixels, MB size = $M \cdot M$, and the search range is $[-d, d-1]$. The overall search procedure includes the HEA, and the different methods for the blocks in and out of ROI. Therefore, the complexity of AMEA can be defined as

$$C_{AMEA} = C_{HEA} + MB_{in} \times C_{in} + MB_{out} \times C_{out}, \quad (4-11)$$

where C_{HEA} , C_{in} , and C_{out} represent the complexity of HEA, the estimation algorithm for the in-ROI and the out-of-ROI blocks, and MB_{in} and MB_{out} are the number of the blocks in and out of ROI, respectively. C_{HEA} contains the operations for calculating $Mean_k$, which requires $W \cdot H$ additions and one division, and shift each input pixel to a new value due to the corresponding formulas with one multiplication and one division. Therefore, C_{HEA} can be defined as

$$C_{HEA} = (W \cdot H + 1) + (W \cdot H) + (W \cdot H) = 3(W \cdot H) + 1. \quad (4-12)$$

If there are Γ numbers of operations required for the SAD calculation of one search checking point, then FSBMA needs $\Gamma(2d)^2$ manipulations per MB using integer-pel accuracy. AMEA requires extra d operations to compare the current minimum SAD with the predefined threshold. Therefore, the maximum computational bound of C_{in} is

$$C_{in}^{\max} = \Gamma(2d)^2 + d \quad (4-13)$$

operations per MB. Conversely, by using a very high threshold value, when only the corresponding center of the search space is checked and only one calculation is required to compare SAD. Hence, the minimum bound of C_{in} is

$$C_{in}^{\min} = \Gamma + 1 \quad (4-14)$$

operations per MB. As for the out-of-ROI MBs, the search range is reduced to 2, and the algorithm is the same as DSA. Thus, the upper and the lower bound of C_{out} are defined as

$$\begin{cases} C_{out}^{\max} = 16\Gamma \\ C_{out}^{\min} = \Gamma \end{cases}. \quad (4-15)$$

For example, consider AMEA for a typical CIF video sequence with $W = 352$, $H = 288$, $M = 16$, $d = 16$ and $\Gamma = 3M^2 = 768$. The pre-defined ROIs of V0, V1, and V2 are all set to 108 blocks, and the number of the blocks, which are out of ROI, is 288. Therefore, the largest and the smallest numbers of calculations for AMEA per frame are $C_{AMEA}^{\max} = 88.78$ million and $C_{AMEA}^{\min} = 0.61$ million, respectively while $C_{FSBMA} = 311.43$ million. After testing several vehicle surveillance videos, C_{AMEA} is

usually around $0.1 \times C_{FSBMA}$ since the probability of the short MVs is much higher than those of the long ones.

4.4 Implementation Results

The test video sequences: “V0,” “V1,” and “V2” are used to evaluate the performance of AMEA. The degrees of the influences of the lighting sources to these three videos are “almost zero”, “very serious”, and “middle”, respectively. All the sequences consist of 450 frames; the frame rate is 30 FPS, and the image size is CIF. The search range is defined as $[-d, d-1]$ where $d=16$. The performance of AMEA is compared to that of four well-known algorithms: FSBMA, NTSS, DSA, and HEXBS. In addition to these non adaptive algorithms, FADTS is also implemented by the standard C language on a normal PC with Pentium 4 processor at 3.0GHz and a total of 1G byte DDR400 SDRAM. The input sequences are all the same, which can avoid the distinct quality due to the different fragment in the same test patterns, and the target output PSNR of FADTS and AMEA is set to 30 dB.

Tables 4-2 and 4-3 present the results. Table 4-2 describes the estimation quality of these six algorithms in terms of PSNR, and Table 4-3 shows the comparisons of the normalized processing speed. In order to truly reflect the MV accuracy, the estimation results are made by the MB and the corresponding MV without the inflation of the error residuals. In Table 4-2, AMEA can maintain the pre-defined PSNR by the adaptive threshold, and it even outperforms the FSBMA when the input video sequences have lighting effects. The HEA adopted by AMEA can reduce the influences from the light sources, and it can provide higher correlation coefficients

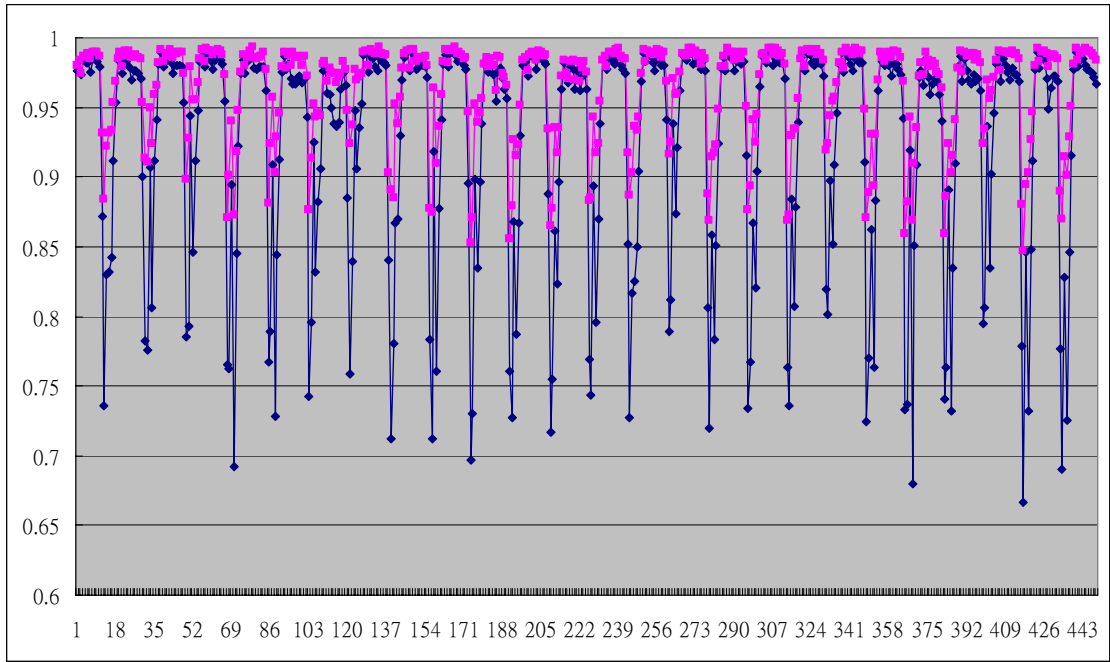
between the adjacent frames, as shown in Fig. 4-15 and Fig. 4-16. Therefore, the performance of AMEA in V1 and V2 is the highest among these frameworks. Since AMEA adopts the simple DSA with small search range for the blocks out of ROI, the quality of V0 is worse than other algorithms. However, the difference is negligible. Furthermore, the processing speed of AMEA is as fast as HEXBS and DSA while providing better quality. It can be observed that FADTS is almost as slow as FSBMA if the target output PSNR is unachievable. From these experimental results, AMEA can solve the major problems of the vehicle surveillance videos, and it is suitable for the embedded platform due to its lower complexity.

Table 4-2 The quality comparisons of the blocks in and out of ROI of different video sequences in dB.

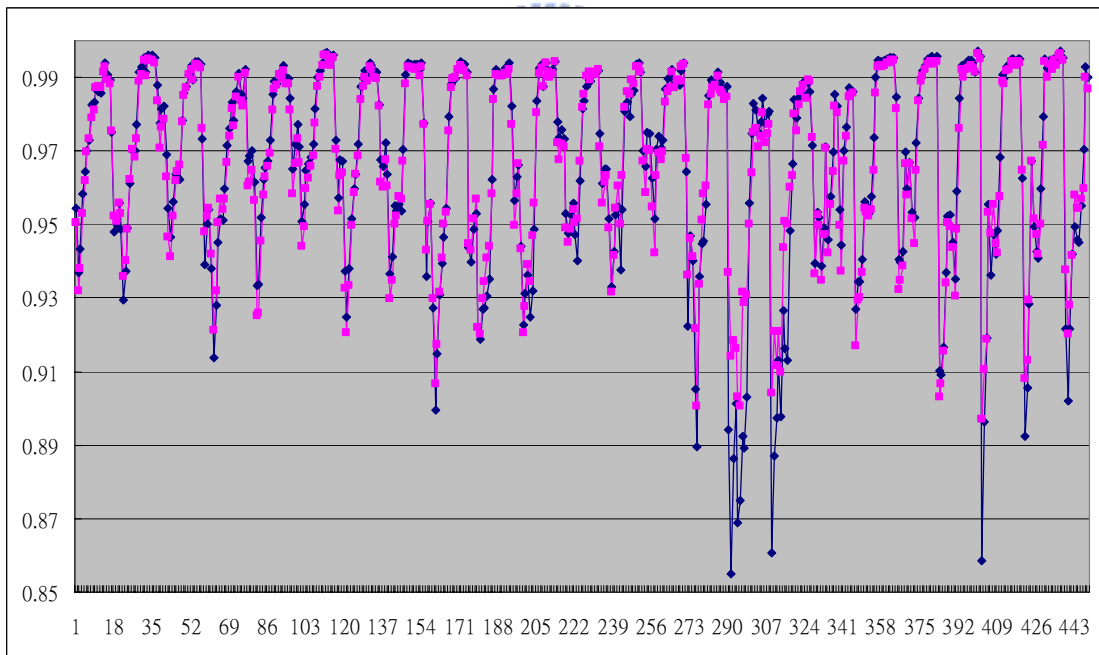
Video Sequence	AMEA (Ours)		FSBMA		FADTS	
	In	Out	In	Out	In	Out
V0	30.37	34.62	32.27	35.38	30.75	30.86
V1	29.86	31.35	24.73	25.91	24.67	25.79
V2	31.12	32.47	29.24	30.47	29.16	30.42
Video Sequence	HEXBS		DSA		NTSS	
	In	Out	In	Out	In	Out
V0	31.87	35.13	31.95	35.17	30.55	34.87
V1	23.82	25.67	23.89	25.74	22.78	23.19
V2	28.91	30.03	29.11	30.12	28.07	29.55

Table 4-3 The normalized processing speed comparisons of the different videos

Video Sequence	AMEA (Ours)	FSBMA	FADTS
V0	0.07	1	0.25
V1	0.15	1	0.89
V2	0.14	1	0.77
Video Sequence	HEXBS	DSA	NTSS
V0	0.11	0.12	0.07
V1	0.21	0.22	0.07
V2	0.16	0.17	0.07

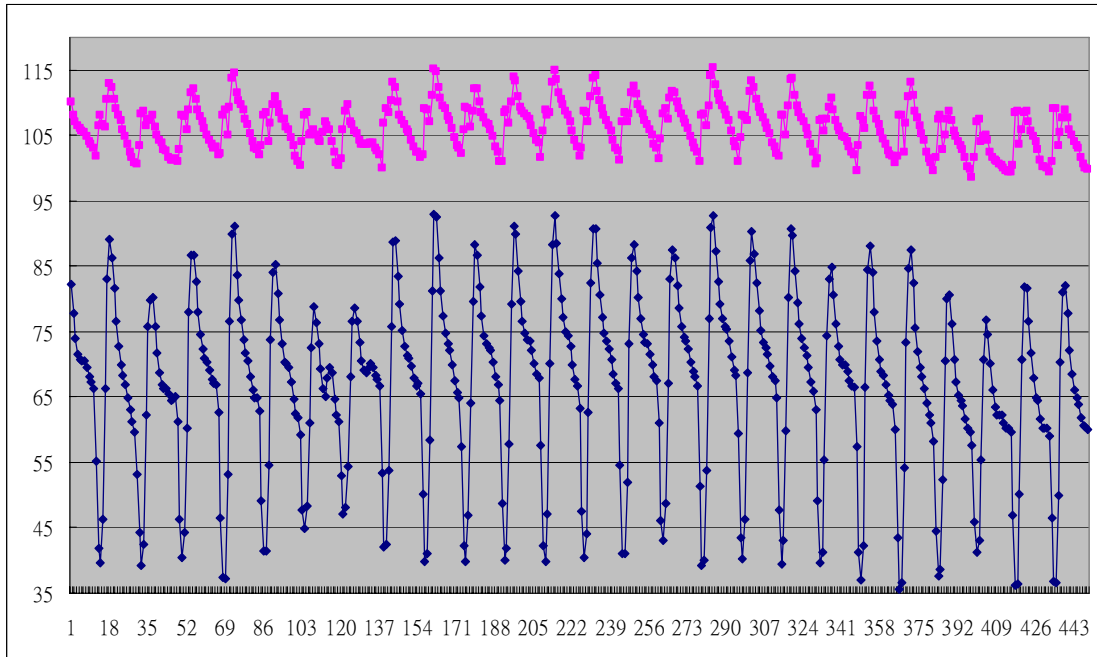


(a)

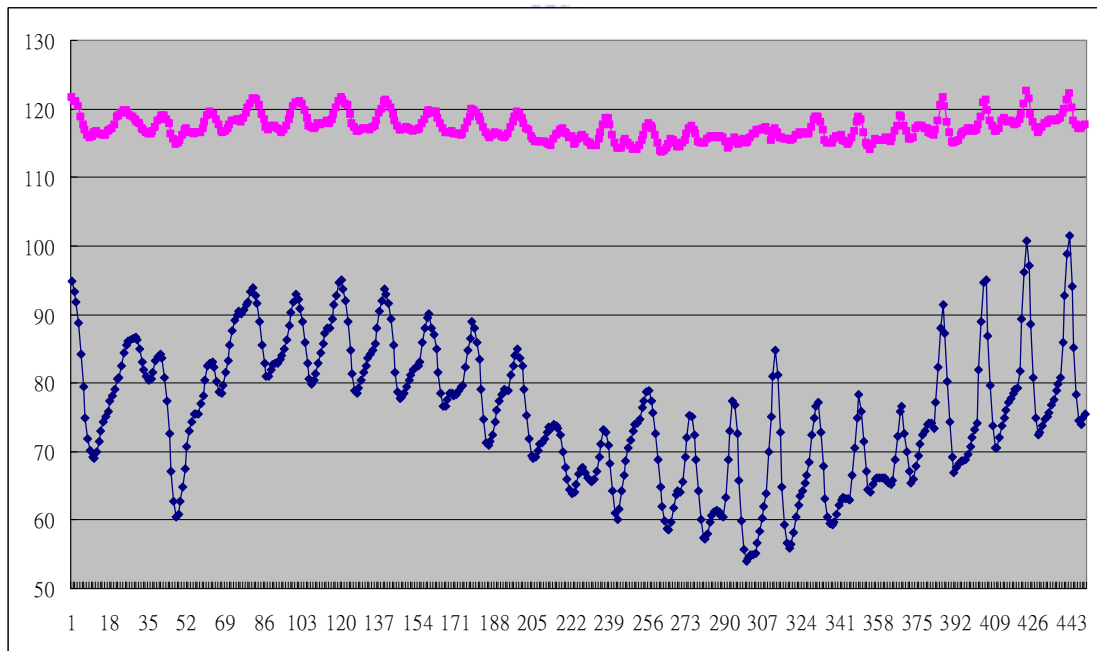


(b)

Fig. 4-15 The correlation coefficients between each frame pairs of (a)V1 and (b)V2 before and after HEA



(a)



(b)

Fig. 4-16 The \bar{M}_k of each frame of (a)V1 and (b)V2 before and after HEA

Chapter 5. A Real-Time Driving Assist and Surveillance System

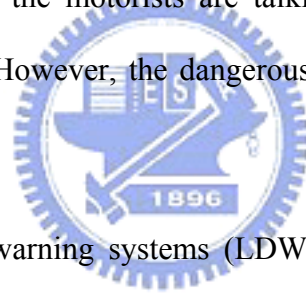
Thousands of people are killed or injured in the vehicle accidents which are mostly caused by the carelessness of humans. Therefore, if there exists a warning system which can notify the driver of the approaching vehicles and the lane departure, the terrible tragedies can be greatly avoided. Moreover, if the driving status, such as the situation around the vehicle and the drivers' behaviors, can be recorded in videos, not only in texts, the police officers can reconstruct the scene easily. Furthermore, the increasing numbers of stolen cars is an important issue so the protection of the vehicles is essential. In this chapter, a real-time DSP-based driving assistance and surveillance system is proposed. It can manipulate the images to detect the lane and the vehicles in front/rear directions, and the blind spot area and the full-time driving status can also be recorded in H.264. The estimated distances and a friendly intuitive graph are shown on the screen for drivers to examine the distance between other vehicles. After parking, the monitor images inside the vehicle can be transmitted to users' mobile phones to prevent the thieves. The contribution of this chapter is to provide an embedded system to improve the security at all-time, and the drivers, the passengers, and the vehicles can be much safer. The proposed system has been successfully implemented and fully tested by the real road environment in Taiwan so the stability and the reliability can be ensured.

5.1 Introduction

Recently, the death rate of traffic accidents has a huge increasing, and about 800,000 people died in world-wide road related accidents in 1999[94]. According to the report by the U.S. National Highway Traffic Safety Administration (NHTSA), falling in asleep while driving is responsible for at least 100,000 automobile troubles annually [95]. In 2000, the situation even got worse, and 1.26 million people are killed in the tragedies [96]. From the above statistical results, the terrible disasters have been increased dramatically, and the driving behavior is the main reason. The misfortunes can be avoided if the drivers are notified with the real-time warning messages by the driving assistance system (DAS) in a jiffy. Thus, many researchers focus on the developments of DAS to solve these huge problems [97]-[101]. One kind of these systems need several instruments except cameras to deal with the input information [97]-[99]. Wijesoma et al. use radar to scanning the lane status [97]. If the driving conditions are classified as the irregular cases, the system will trigger the warning messages. Besides, the car accidents are not only caused by the lane departure, but also by the inappropriate distances between vehicles. DDAS, proposed by Sheu et al., is able to provide useful surrounding information, such as directions and distances to nearby vehicles, to drivers, so unnecessary collisions could be avoided, especially in cases of changing lanes, crossing intersections, and making turns [98]. The main idea of [98] is to install DDAS in all cars, so the vehicles can detect each other by the smart antennas to exchange the data to maintain the proper behaviors. Moreover, in the advanced safety vehicles (AVS) project of Toyota, the driver must wear a wristband in order to measure his heart rate, and Mitsubishi has reported the use of steering wheel sensors to measure the vehicles' behavior to detect if the drivers drowse or not in their AVS system [99]. However, these systems may not be practical since they need more equipments, which will lead to increase the cost

and the limitation compared to the general image-based processing system.

Other methods are based on the employment of the machine vision, and they only require several cameras and the process units to compute the driving safety assist algorithms. Hayami et al. use eye's status to adjudge the drivers' behavior to distinguish whether they are drowsiness or not [100]. In the algorithm of [100], few parameters are roughly calculated so the precision is not good enough to provide the safety. In order to increasing the degree of accuracy and the robustness, more complex equations are calculated in [101]. Although the performance has been promoted, its computational complexity also increases comparatively. These methods can mainly solve the problems that the drivers fall in asleep, but not the irregular driving behaviors caution. If the motorists are talking on the phones or having a daydream, they won't doze. However, the dangerous actions such as lane departure will be produced.



Several lane departure warning systems (LDWS) based on the variety of the computer vision techniques have been proposed [102]-[106]. The main principles of these different technologies are to detect whether the vehicles are in some risky conditions. When the systems find that the cars are in some irregular statuses, it will alarm the drivers by generating warning signals like making a huge sound or a blinking light. Most algorithms only can run on personal computers or work stations due to the complexity, and it is not acceptable in real application such as buses, trucks and movable cars because of the limited space and the restricted power supplement. In order to avoid these drawbacks, the modified schemes for the embedded systems are proposed [107]-[109]. In [108], an ARM-based system is provided to act as an LDWS, and it has a major advantage of the cost and the power consumption. However, its

computational load is very weak, 128x128 at 22 FPS. For the huge input image information, a system adds field programmable gate array (FPGA) device to increase processing speed [107]-[109]. The quasi-system based on Power PC is also proposed [110], and it uses a high performance core and a small machine box to form a mini-system. Although these solutions can overcome the complicated computations and finish the particular works, the system is hard to change if users or programmers want to modify or add some functions in the future. For example, the system needs to modify the algorithms for the signs of velocity-limitation which are different in U.S.A. and Asia.

There are many vision-based algorithms for detecting the lane markings on the road. M. Bertozzi et al. developed a system, Generic Obstacle and Lane Detection system (GOLD), which performs both lane and obstacle detection in front of the vehicle [111]. The contour-based motion estimation and object tracking system was developed in detecting approaching vehicles in [112]. However, the above systems only consider the situation of approaching vehicles in front-rear direction. The cars approaching from the blind spot area are also possible to make accidents. Besides, most algorithms only can run on personal computers or work stations due to the algorithm complexity, and it is not acceptable for real application such as buses, trucks and cars because of the limited space and the restricted power supplement.

The above works are used to prevent the accidents, but they can not provide useful information for users and police officers when the misfortunes happen. The vehicle box, proposed by Yeh et al., is designed to keep track of the vehicles and the driving information, which is analogous to the flight data recorder used in aircrafts [113]. Since it takes down the GPS data, it can reconstruct the car's location, speed,

and orientation. However, it can not reveal the exact causes.

All of these works lacks the real-time notification which can greatly help the drivers, and the driving status that can analyze the reasons of the accidents [97]-[113]. Moreover, these systems can only protect the users and the vehicles when they are moving. When the vehicles are parked, the users still have to face the opportunity of missing them. In this chapter, a real-time digital signal processor (DSP) based driving assistance and surveillance system (DASS) is implemented. For the considerations in both flexibility and computing power, the DSPs with strong calculation, addition and multiplication, are good choices for the vehicular electronics. The system is combined by three independent DSP sub-systems, which are the Lane departure and Vehicle detection Warning System (LVWS), the Blind Spot Warning System (BSWS) and the Vehicle Surveillance System (VSS). The detection zones of the three sub-systems are shown in Fig. 5-1. LVWS and BSWS are realized by DSP, TI DM642, while VSS is implemented by TI OMAP5912. The algorithms on LVWS works not only for the front view but also the rear direction, and can manipulate the real-time images from these two channels simultaneously in CIF at 30 FPS. BSWS is utilized to detect the lane departure and the vehicles approaching from the blind spot of the car. Combined with these two sub-systems, the system has a complete ability of lane departure and vehicles detection in omni-direction. VSS is responsible for recording the full-time driving status with H.264 video compression algorithm, and it also provides the monitoring functionalities using cell phones. Furthermore, the estimated distance calculated by LVWS and BSWS will be transferred to VSS via ZigBee, and the screen will show an intuitive graph, which is designed to display the distance between the host vehicle and the others. When the vehicle is driving away from the road or is too close to the other cars, LVWS or BSWS will generate a warning signal and

transmit it to VSS by the Zigbee wireless communication. The main advantages of using Zigbee are to simplify the installation of DASS and save the space which is strongly restricted in the vehicles. As soon as LVWS or BSWS receive the caution information, they will produce short messages to the mobile phones of the related people, such as their families or companies. Then, they can browse the real-time images from VSS with their mobile phones anytime and anywhere and caution the driver or request the police for help in order to avoid a possible accident. In addition, if the motorist is in distress, the video recorded by VSS can help the police to recover the original condition. Furthermore, if the vehicle happened to an accident, such as dropping down the mountain valley or being capsizing, VSS will call for help automatically to reduce the waiting time and greatly increase the survival rate. Moreover, when the vehicles are parked, the owners can receive the images from VSS to protect their properties. The main contribution of DASS is to provide a total solution which integrates the innovative functions in an embedded system to furnish the users full-time protection. No matter the drivers, the passengers, and the vehicles are moving or not, their safety are ensured.

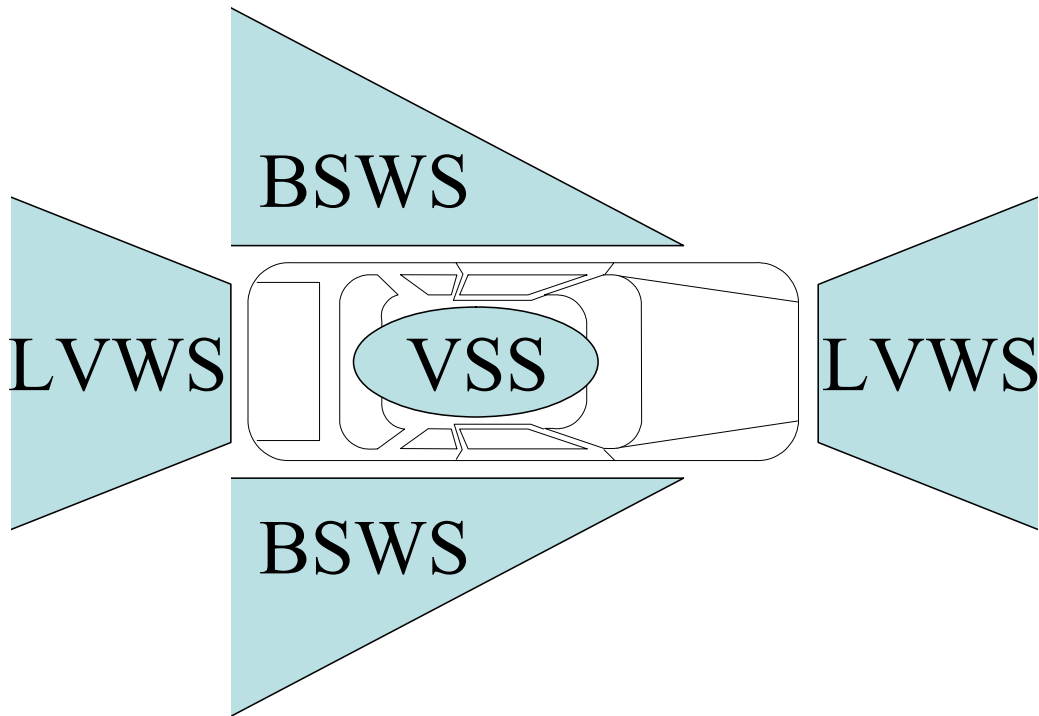


Fig. 5-1 The detection zones of LVWS, BSWS and VSS

5.2 The Innovative Functional Scheme of DASS

An overall system block diagram of DASS is depicted in Fig. 5-2. LVWS combines two cameras, a DM642 platform, and a Zigbee device. The cameras are responsible for capturing the real-time information and send them into DSP, and when the manipulation results determine that the vehicle is in danger, the warning signal will be transmit to VSS via the Zigbee device. The major function of LVWS is to detect the lanes and the vehicles in both front and rear views while BSWS processes with the side views. VSS includes a touch screen module, a USB camera, a CF storage card, and a 3G/GPRS module. The camera will input the images in the vehicle, and the OMAP processor will compress them with the format of H.264 in the CF storage card. When VSS receives the warning signal from LVWS and BSWS, it will send the warning short message to the mobile phones that set by the users. As soon as

the cautions are received, the cell phone can browse the real-time image in the vehicle anytime and anywhere. The functional schemes of DASS are described as follows.

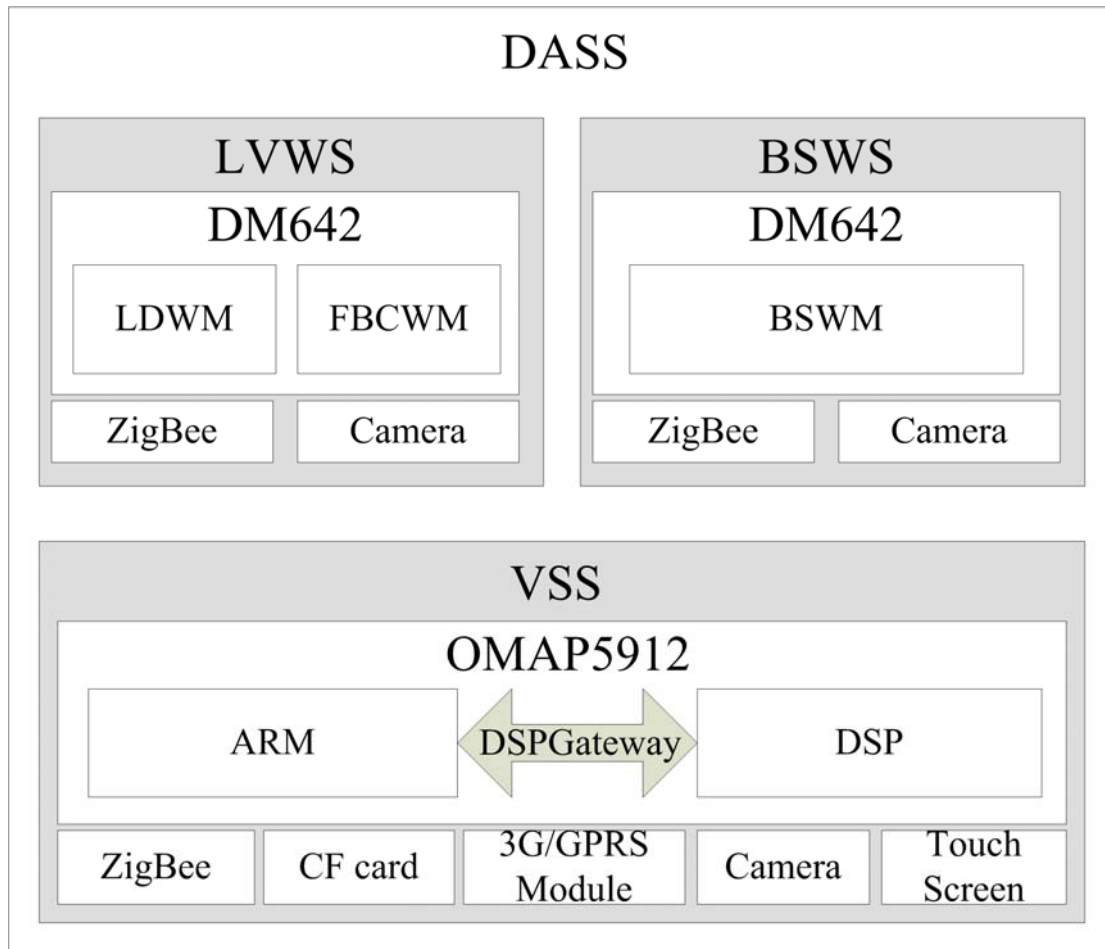
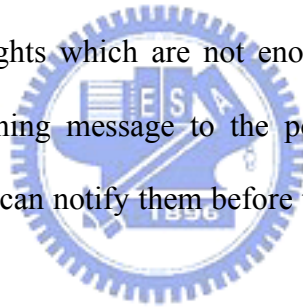


Fig. 5-2 The overall system block diagram of DASS

5.2.1. Active Vision-based Warning Supports

With the platform-dependent optimizations of the memory and the enhanced direct memory access (EDMA) management, the complex algorithms that can perform real-time vehicle detection and lane departure warning for both front and rear views are implemented. Combined with two sub-systems, LVWS and BSWS, DASS can provide three major modules for safety, which are Lane Departure Warning Module (LDWM), Front/Backward Collision Warning Module (FBCWM) and Blind

Spot Warning Module (BSWM). LDWM can determine whether the vehicle is deviating from the road or not and make sounds to warn the driver. FBCWM can estimate the distance between the host vehicle and the others in front and rear sides to notify the driver the possible danger. On the other hand, BSWM can detect the vehicles within the blind spot area and inform the driver when changing lanes. Furthermore, the estimated distance information is shown for displaying the relation between the host vehicle and other vehicles surrounding by a friendly GUI. Therefore, the vehicle has a complete omni-direction protection with these modules. In the different conditions, such as sunny, cloudy, rainy days and night, the information of lanes and the vehicles can be successfully obtained. The main reason for the vehicle accidents is the irregular driving behaviors, and the normal warning system can only produce sound or blinking lights which are not enough for protection. LVWS and BSWS can transmit the warning message to the police department or the family people of the driver, and they can notify them before the tragedies. The whole flow is shown in Fig. 5-3.



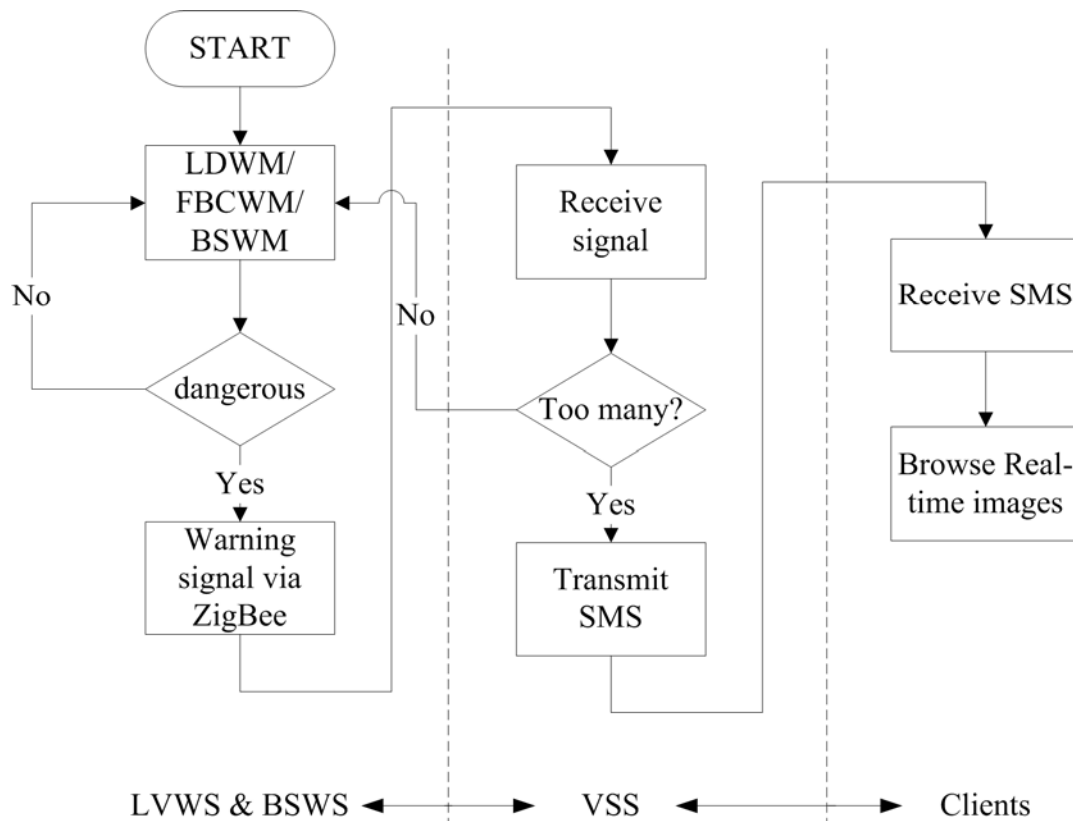


Fig. 5-3 The flowchart of the active real-time vehicle detection and lane departure warning system

5.2.2. H.264 Driving Status Recorder

If the drivers are in danger, the reasons that cause the accidents are the most important part. The driving status recorders are not popular nowadays, and they can only provide the data of throttle and the brake which are not able to reconstruct the situation. MSR can record the full-time driving status in H.264, which is suitable for a stand-alone system since its video quality and its compression ratio are very high, in the CompactFlash (CF) storage card, and the video recorded by VSS can help the police or the related departments for reference. The state chart of the system is illustrated in Fig. 5-4.

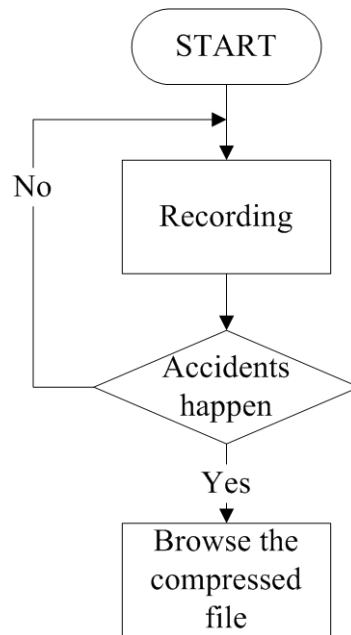


Fig. 5-4 The flowchart of the full-time recorder

5.2.3. Burglarproof with Mobile Surveillance

The recent vehicle surveillance systems, such as On Star [114] in USA, TOBE [115] in Taiwan, can provide the latest information of your car by short text messages. However, in the current applications, no matter the short message or the GPS navigation, both can not provide the intruders' pictures. The users can only know there is something happened in their vehicles, but they can not know what exactly the situation is. Maybe there really exists thieves or just a heavy truck goes by. Therefore, a burglarproof system with mobile surveillance and friendly user interface for mobile devices are developed to clear the above drawbacks. It can access the internet by 3G/GPRS so that the server can be installed in the vehicles, not restricted by the wired-internet environment. When the intruders try to steal your vehicles, VSS will transmit the warning short messages to users' mobile phones. As soon as the users receive the cautions, they can browse the real-time images in the vehicle anytime and

anywhere. In addition, the software in the mobile clients is very easy to use. The users simply press one button to browse the continuous pictures. Thus, it can supply the all service functions through 3G/GPRS communication to help the vehicle owner understanding their vehicle conditions, and the program flowchart is shown in Fig. 5-5.

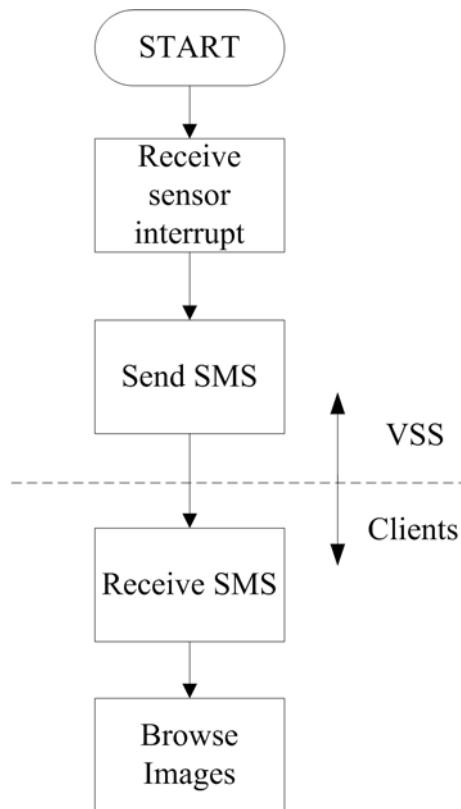


Fig. 5-5 The flowchart of the mobile surveillance system

5.3 The Module and Architecture of DASS

In this section, the related algorithms including three modules, LDWM, FBCWM and BSWM are addressed. The software architecture of VSS is also illustrated in this part. Since the features of real-time notification for the mobile surveillance require the network protocol stack, embedded Linux is chosen as the

Operating System (OS) in VSS.

5.3.1. Lane Departure Warning Module

Lane detection and position estimation are two key processes of LDWM. The lane detection algorithm utilizes a two-stage process with two detection modes. The first mode is for images when no previous information of lane marking pixels and lane trends exist. The second one works on the image when the lane marking pixels and lane trends are detected in the previous image. When it appears that the lane cannot be identified successfully, the mode will be switched to the first one again. The detected marking pixels are approached by a second order polynomial and its curvature is taken as the lane trend information on the image plane. The lateral offset of the host vehicle is estimated from the image shown in Fig. 5-6 and the warning signal will be enabled once the lateral offset is wider than a quarter lane width.

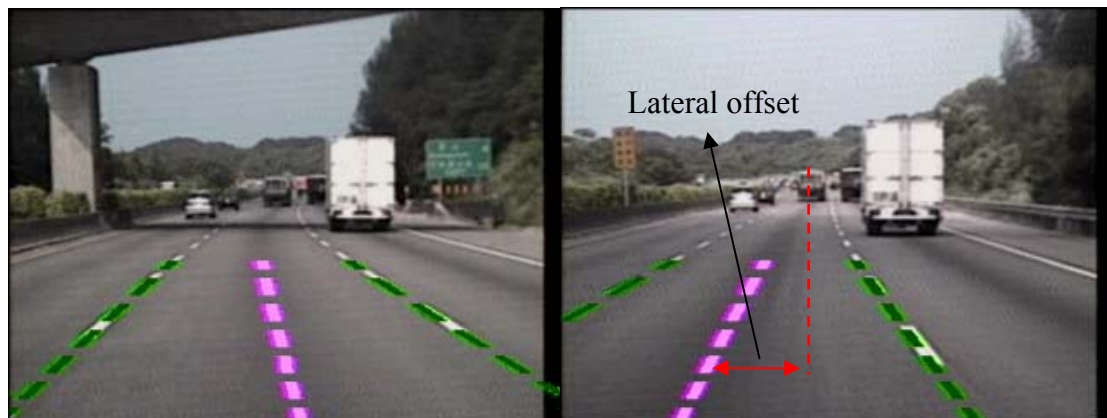


Fig. 5-6 The estimated lateral offset

5.3.2. Forward / Backward Collision Warning Module

In FBCWM, vehicles are the interesting objects on the images. The lane information from LDWM is applied in this module. The candidates of the vehicle are searched in the host lane by different features for the weather conditions. Before the vehicle detection, a threshold of the variance between the objects and the road surface is identified automatically by the histogram distribution so that the vehicle detection algorithm adapts for different weather conditions. In sunny and cloudy conditions, the shadow under the vehicle is interested for the detection. The horizontal and vertical edge information is applied to avoid the fault detection resulted from the long shadows. The tail-lights of the preceding car are the important features at night for the detection. However, the influence of the street lights and the head lights of the vehicle in neighbor lanes exist. Sobel edge information is combined with the tail-lights to correct the vehicle position so that the accurate distance is estimated. To detect the vehicle from the real-time image in rainy days is a big challenge. The images are blurred by the raindrops on the windshield, so the gray level histogram is analyzed to separate the dark vehicles, light color vehicles, the road surface and the wipers. The classifications will be recognized from the histogram distribution in the host lane.

5.3.3. Blind Spot Warning Module

Blind spot is another important issue for drivers while changing lanes. In order to avoid the possible collision within blind spot zones, BSWM is developed to conquer this drawback to improve the safety. The information of the near lane marking is used to estimate the far one. Once the both lane markings are detected, the detected zone for the vehicles in the blind spot area is bounded. The candidates of the vehicle are searched from the bottom to the top in this area by detecting the Sobel horizontal edge.

After the most possible candidate is decided, the estimated distance between the host vehicle and the approaching vehicle is calculated. The information will be used to trigger the alarm to warn the driver when the distance is too close

5.3.4. The Software Architecture of VSS

With the rich functionalities, the software architecture of VSS is illustrated in Fig. 5-7. The embedded OS is responsible for handling the schedule of the software and the control of the hardware. The software of architecture of VSS is separated into three layers, the application, the OS, and the hardware layers, and each layer has its dedicated works. The ZigBee receiver will acquire the data from LVWS and BSWS, and VSS can process the data to distinguish whether the vehicles are safe or in danger. The GSM controller can send the warning short messages and connect to the internet via 3G/GPRS to pass the images input by the USB camera to users' mobile phones through the GSM module. Furthermore, the H.264 video encoding algorithm is manipulated by DSP. When ARM grabs a picture from the USB camera via the device driver, it will generate an interrupt to DSP to enable it for encoding. As soon as DSP starts to compress, ARM will capture the next image to reduce the time for data fetching, and perform the H.264 framework in pipeline. Moreover, the network servers can stream out the bitstream on the fly, and can let the users download the files to reconstruct the driving status by the general internet browsers easily.

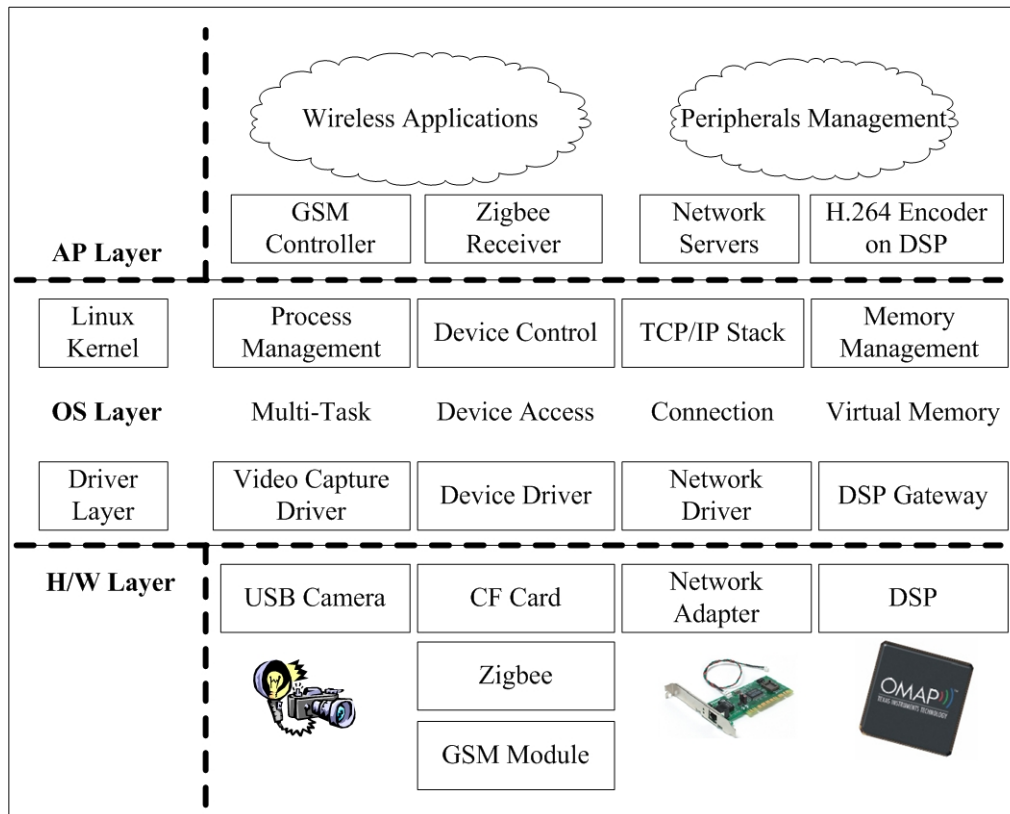


Fig. 5-7 The software architecture of VSS

5.4 H.264 Encoder with AMEA on VSS

With the popularity of mobile communication in recent years, people always have a mobile phone or a PDA. When the systems can transmit the vehicle security information to the owners' mobile devices, more properties can be protected by real-time alarm messages. Moreover, if the guard system of the vehicle can capture the images which are in or out of the vehicle anywhere, the vehicle owners can handle the situation of the vehicle more precisely. A preliminary version of this kind of system has been presented in [17]. Moreover, in the system level design, VSS has greatly enhanced the functionalities of those in [17]. The 3.5G/3G network can provide much higher bandwidth than it of GPRS, and H.264 video encoding has better compression ratio than JPEG. Hence, VSS can offer a real-time videos inside the

vehicles to the users at anytime and anywhere. Since VSS should work whether the vehicle is started or not, it has to be as low power consumption as it can. Therefore, a dual-core embedded platform, TI OMAP5912, is chosen, and the algorithms for VSS are all modified to be much simpler than their original versions in order to save the energy. VSS can be integrated with the signal triggers, such as open door, start engine, and etc., of the existing burglarproof systems, and it will automatically enter to the suspend mode when nothing happened for a period of time. The hardware and software architectures are shown in Fig. 5-2, and Fig. 5-7, respectively, and all programs, including the embedded Linux operating system, and the peripheral controllers, except for the H.264 encoder are executed by ARM.

5.4.1. Cooperation of ARM and DSP




Figure 5-8 describes the efficient cross work between the ARM and the DSP cores, operating in parallel when the H.264 encoding function is enabled. At first, when the function is called, ARM initializes the system, reads frames in the storage devices and then writes into the external memory of DSP. When the source is ready, ARM activates the H.264 encoding routine of DSP. Then, the next frame will input into the buffer whenever the side information of fame is processed. After encoding of the current proceeding frame, the routine restores the necessary information and transits it to the next frame. In this configuration, ARM performs a part of the H.264 encoding process in addition to system control. It should be noted that ARM generally has the better architecture for the frame buffering than DSP.

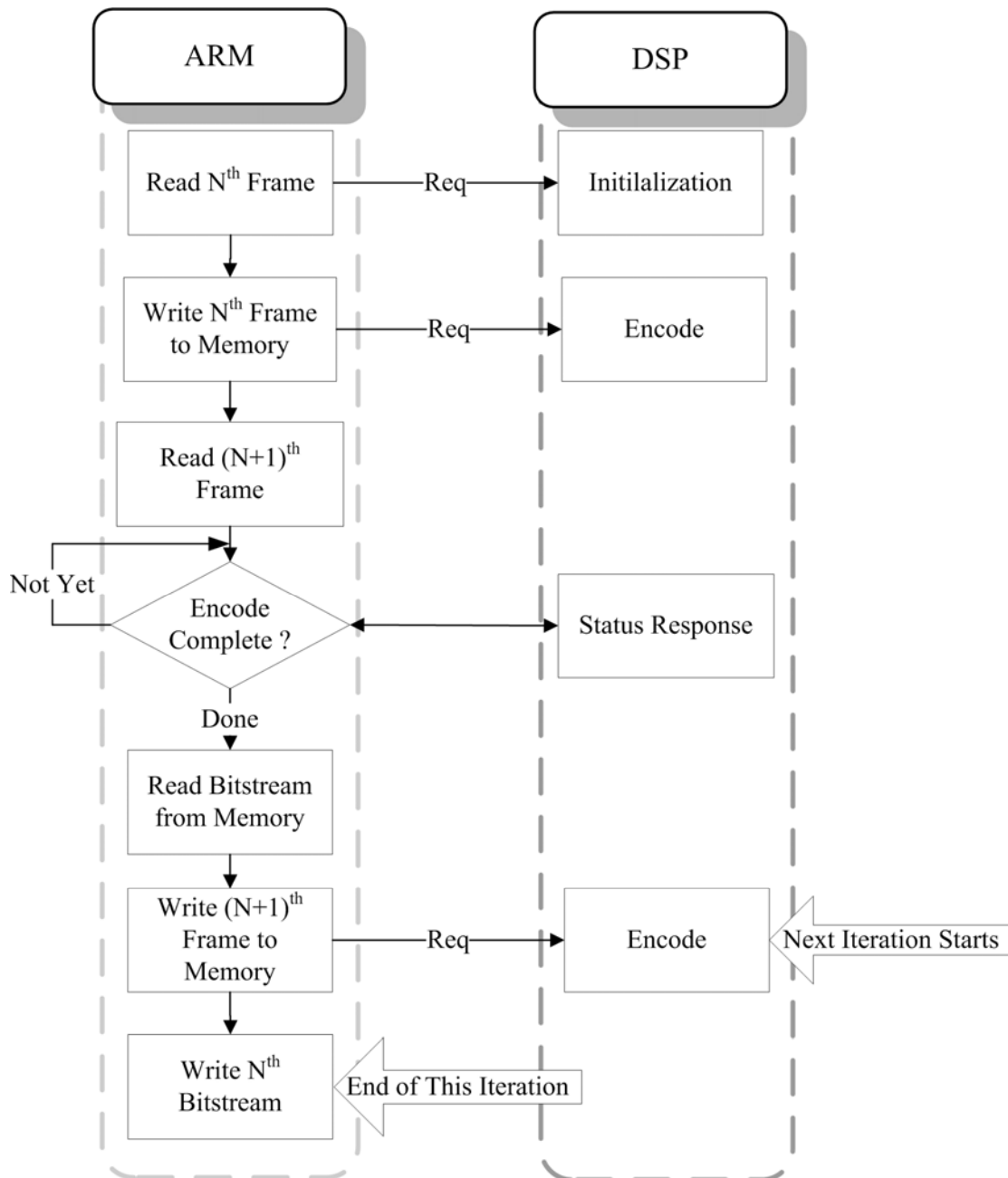


Fig. 5-8 The proposed H.264 encoding data path

Figure 5-9 shows the synchronization of audio and video encoding of both CPU cores in OMAP5912. Three threads, video, packaging, and streaming thread (VT, PT, and ST), are created at the ARM side. VT fetches the video data from the USB camera and sends to DSP for video encoding. Once the data are acquired, a timestamp is recorded for synchronization. The communication between ARM and DSP is

solved by DSPGateway[116]. After DSP completes encoding a video frame, it sends a notifying signal through the mailbox to ARM, and VT starts receiving the bitstream and the related information from DSP. After the receiving process is done, VT also sends the new information and a starting signal through mailbox to DSP immediately to ensure that there is a minimum latency between ARM and DSP. PT takes charge of packaging the video bitstream into H.264 file format, and adds the necessary header to make it compatible with RTP. ST takes charge of gathering necessary information for the streaming server. Furthermore, it transfers the data in UDP for remote clients to receive them via 3.5G/3G network. Figure 5-10 shows that when switching the threads, PT and ST acquire timestamps and bitstream created from VT. If the data are dropped due to the heavy network traffic, the frames which fall behind are dropped in order to keep in phase and they will not re-send again according to UDP.

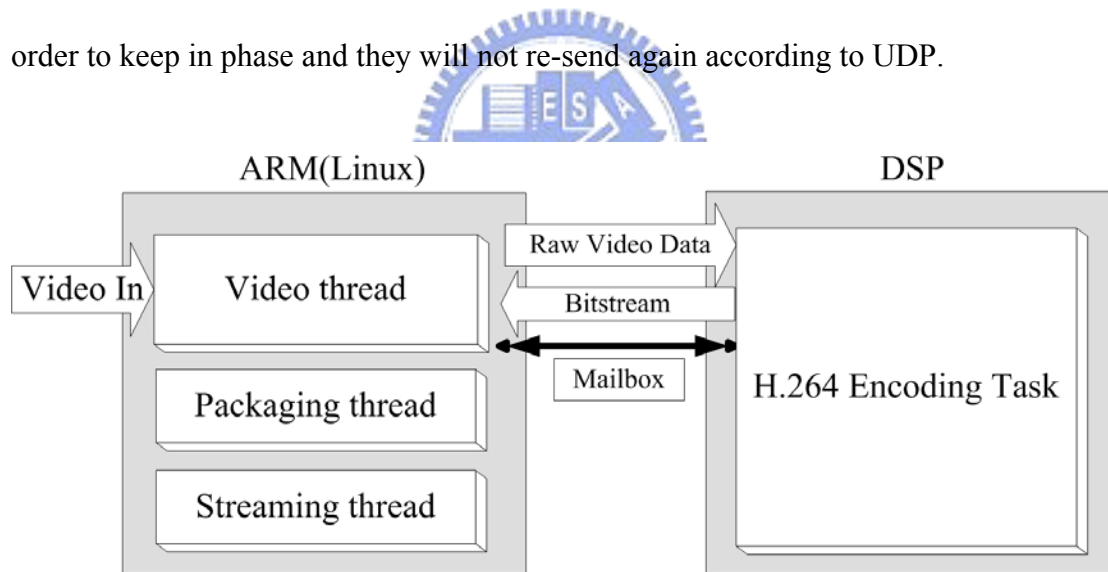


Fig. 5-9 The synchronization of audio and video encoding

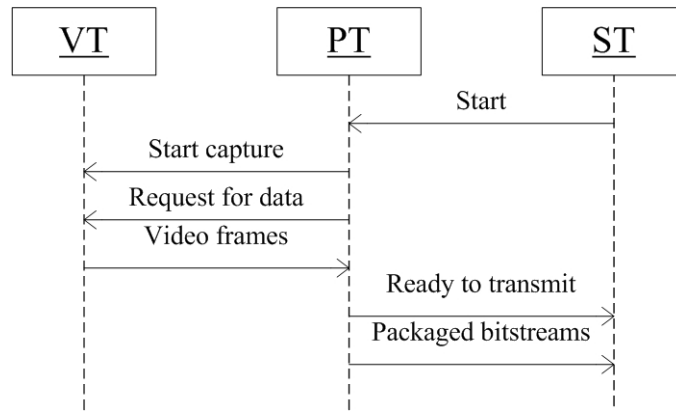


Fig. 5-10 Communication between related threads

5.4.2. H.264 Encoder Optimization

The proposed encoder is developed based on the open source x264 encoder. The reason why we choose it as the proposed encoder is its excellent performance. But it has some disadvantages to be used for embedded devices. The computational complexity and the memory usage are huge, and its target platform and program flow is designed for x86. Therefore, a whole program optimization should be done, and the platform feature must be involved to the design.

Optimization techniques employed to reduce the computational complexity of modules are grouped into two categories: platform-independent algorithmic optimizations and memory optimizations.

A. Platform-independent algorithmic optimizations

To reduce the complexity of H.264 encoder, algorithms are modified in high level language. Apart from implementing optimal algorithms, optimization techniques like loop unrolling, loop distribution and loop interchange are used. Algorithmic

optimizations presented below are platform independent and can be used on any platform.

B. Memory optimizations

As the memory access of encoder is a lot, many cycles are spent by DSP core while waiting for that ARM moves data from external memory to DSP memory. In this encoder, the data is moved in an efficient way. Due to the dual-core feature, much time is saved in the parallel processing.

C. Rearrange H.264 Encoding Control Flow

The original H.264 encoding control flow is shown in Fig. 5-11

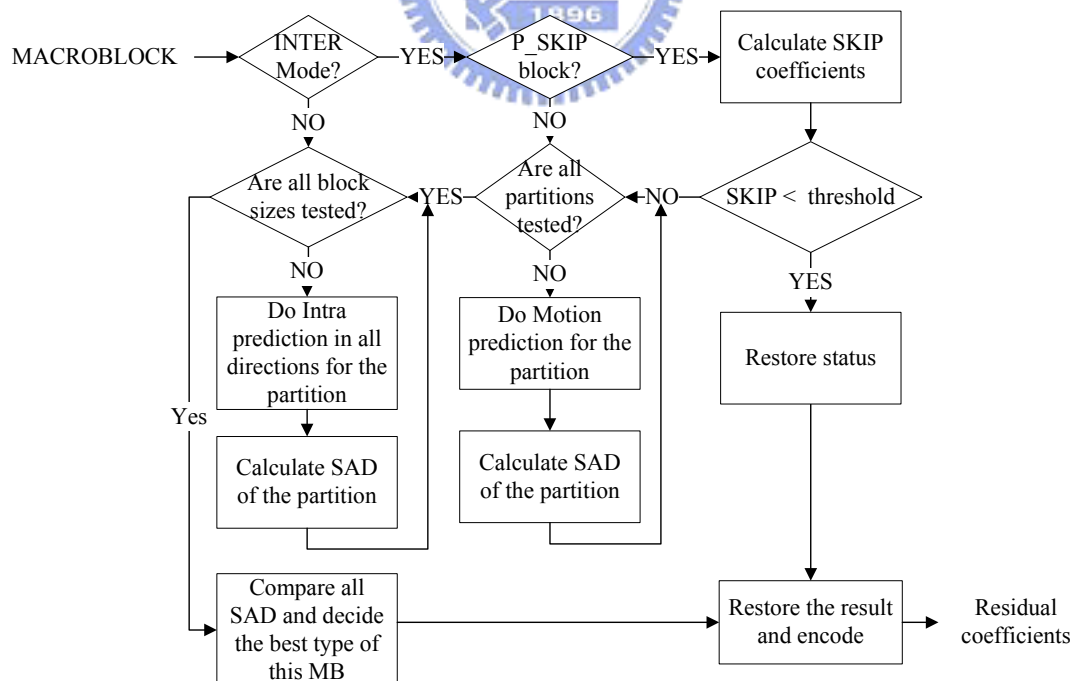


Fig. 5-11 The original H.264 encoding control flow

As shown in Fig. 5-11, the encoding flow of the encoder is too complicated to implement for an embedded system. For the purpose of accelerating the encoding speed, processing fewer modes used will save a lot of time to encode a frame.

The DSP core has a useful feature, zero-overhead loop. If the loop codes are kept small enough to enable the compiler to make use of the native local repeat instruction, the compiler will generate local repeat for small loops that do not contain any control flow structures other than forward conditionals. Local repeat loops consume less power than other looping constructs. In addition, to generate a hardware loop, the compiler would need to emit code that would determine the number of loop iterations at run time. This code would require an integer division. Since this is computationally expensive, the compiler will not generate such code and will not generate a hardware loop.

The block size is fixed and the intra predictions loop and the inter predictions loop are canceled, because the block decision loop can be compiled to hardware loop, and more overhead will be consumed here. The optimized control flow is summarized in Fig. 5-12.

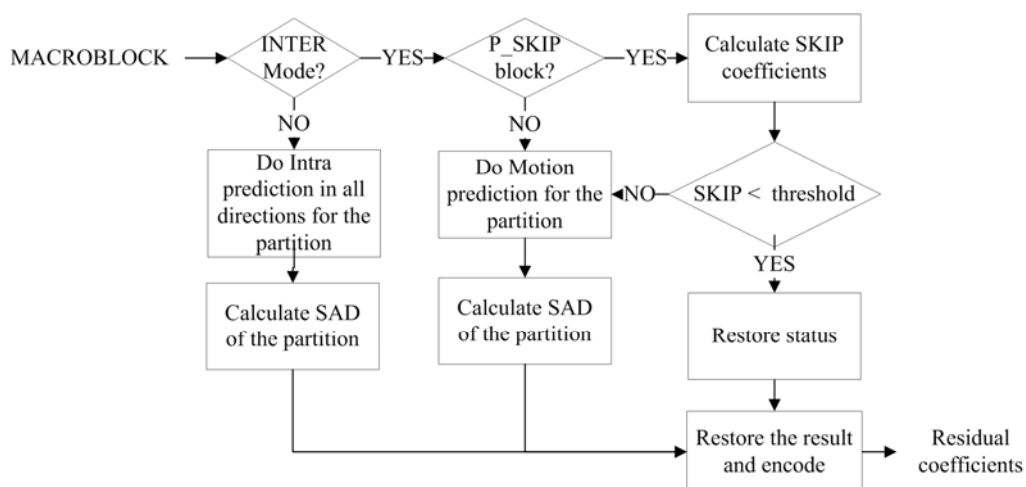


Fig. 5-12 The proposed H.264 encode faster control flow

The proposed faster H.264 control flow disables block decision in order to reduce computational complexity. However, this method will decrease PSNR. The impact of reducing three different block sizes into one for intra prediction is evaluated on x86 CPU first. Then, reducing several directional prediction modes for intra prediction is also evaluated. Our evaluation platform is Pentium 4 2.4GHz and the size of test sequence is QCIF and the results are shown in Table 5-1 and Table 5-2.

Table 5-1 PSNR and FPS evaluation for intra predication with different block sizes

Sequences	16x16 8x8 4x4		16x16 4x4		16x16	
	PSNR	FPS	PSNR	FPS	PSNR	FPS
foreman	38.63	32.38	38.57	36.71	38.54	39.03
Stefan	37.35	30.43	37.27	34.29	37.15	37.65
Claire	42.07	71.11	41.98	76.18	41.95	78.70
container	39.37	51.48	39.23	58.35	39.23	63.99
News	40.13	45.39	40.10	50.79	39.96	56.15
suzie	40.36	37.50	40.24	42.10	40.21	45.50

Table 5-2 PSNR and FPS evaluation for intra predication with different directional predictions

Directions	7		6		5		4	
	PSNR	FPS	PSNR	FPS	PSNR	FPS	PSNR	FPS
foreman	38.54	39.03	38.53	39.67	38.52	40.42	38.53	41.29
stefan	37.15	37.65	37.15	38.32	37.14	39.03	37.14	39.92
claire	41.95	78.70	41.93	80.67	41.88	81.02	41.91	83.13
container	39.23	63.99	39.23	65.53	39.23	65.75	39.20	66.67
news	39.96	56.15	39.93	57.14	39.88	58.17	39.88	58.72
suzie	40.21	45.50	40.19	46.15	40.19	47.29	40.20	48.99

Therefore, the algorithm simplification includes two steps: reduce different block size prediction modes and different directional prediction modes.

First, there are three different block sizes for original intra prediction procedure. Removing them one by one makes it easier to have a closer look at the result. As shown in Table 5-1, removing block sizes of 8x8 and 4x4 contribute to FPS raise and only a little PSNR loss. Apparently, this simplification with only 16x16 blocks

increases the encoding efficiency for intra prediction.

Second, the results of reducing total seven directional prediction modes for intra prediction into four are shown in Table 5-2. Same as above, removing them one by one makes it easier to have a closer look at the result. As shown in Table 5-2, removing the number of the directional mode increases FPS, and only incurs a little PSNR loss as result. It also helps us simplify the computational complexity without much loss of the video quality.

5.4.3. The Mobile Phone Programming

Since more and more mobile phones support Java programming, the J2ME Java version is addressed for developing applications on mobile devices. But the API of J2ME is not as complete as that of J2SE. There are still many restrictions in developing the Java program. The socket programming is used to establish the connection between VSS and the mobile phone. The server-push method is applied in the data transmission between them. That is, the mobile phone will not download the files until the server is ready for the downloading. Figure 5-13 shows the state chart of this method and Fig. 5-14 is the processing flowchart of this program on a mobile phone.

Users can download the continuous real-time images to their mobile phones from VSS anywhere if this Java applet has been embedded in their mobile phones. However, the surveillance videos are very private information, so the authentications are required. The computing power and memory size of the mobile phone are both much lower than other peripherals so that software decoding of the received H.264

videos relies on H.264 hardware decoder on mobile phones. Therefore, a cellular phone with H.264 de-compressor, such as Nokia N95, is necessary.

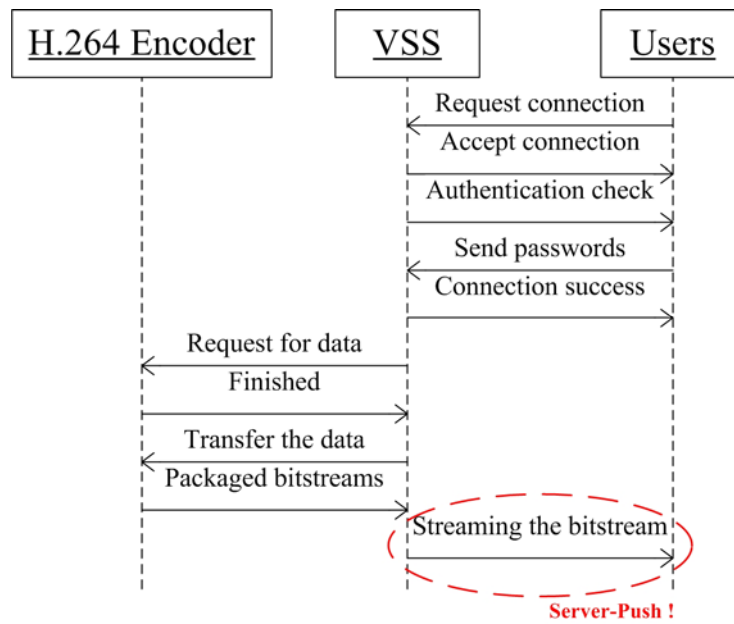
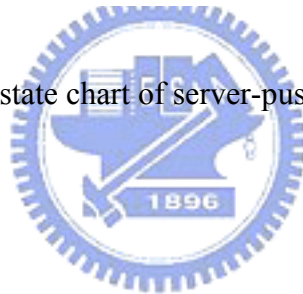


Fig. 5-13 The state chart of server-push we addressed.



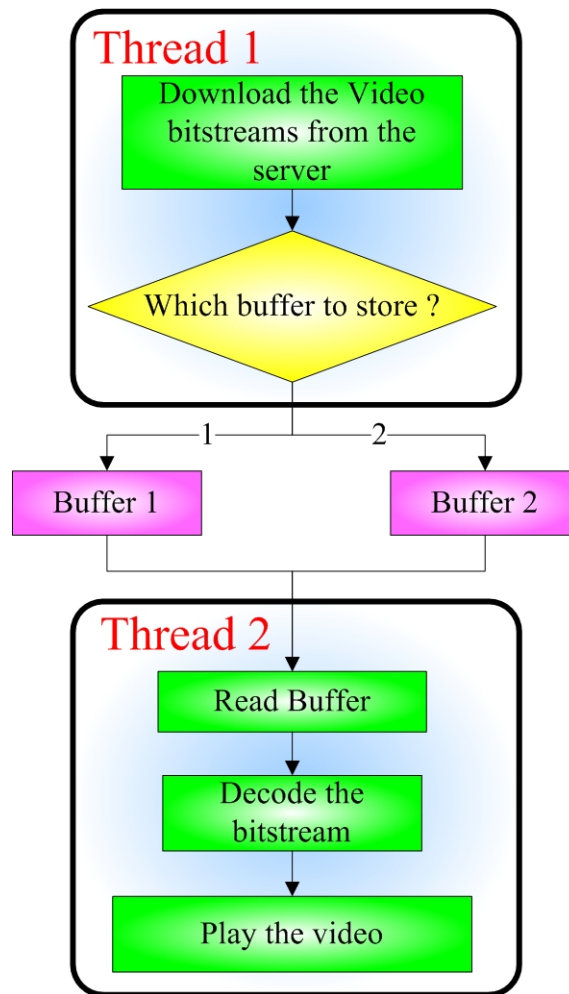


Fig. 5-14 The processing flowchart of the program on a mobile phone

5.5 Implementation Results

All functionalities and the system of DASS are successfully implemented, and they are tested by the real road environments in Taiwan. DASS is well installed in the vehicle, which is called “TAIWAN iTS-1” in Fig. 5-15, and the cameras of LVWS are placed in front and back of the vehicle, shown in Fig. 5-16 while those of BSWS is shown in Fig. 5-17. On the other hand, the USB camera of VSS is also planted in the right-front of the vehicle in order to capture the images of the driving status, and the position is illustrated in Fig. 5-18. The testing results include series of the

conditions such as general cases, rainy days, complicated roads, and the interference of the windshield wipers and the lights. The experimental results of each function are described below.



Fig. 5-15 TAIWAN iTS-1



(a)

(b)

Fig. 5-16 The monochromatic CCD mounted behind the (a) front (b) rear windshields



(a)

(b)

Fig. 5-17 The monochromatic CCD mounted on the (a) left (b) right side view mirrors



Fig. 5-18 The USB CCD mounted behind the right front windshield

5.5.1. Lane departure and vehicle detection algorithms

The lane departure and the vehicle detection algorithms are verified, and the

results of lane markings and the vehicles in both front and rear direction, which are shown in Fig. 5-19. Fig. 5-20 demonstrates the detection results from blind spot views. Both LVWS and BSWS are tested under all kinds of condition, such as rainy days, complex roads, curved roads, and night as shown in Fig. 5-21. The results show that the algorithms in LVWS and BSWS are robust and suitable for the real world.

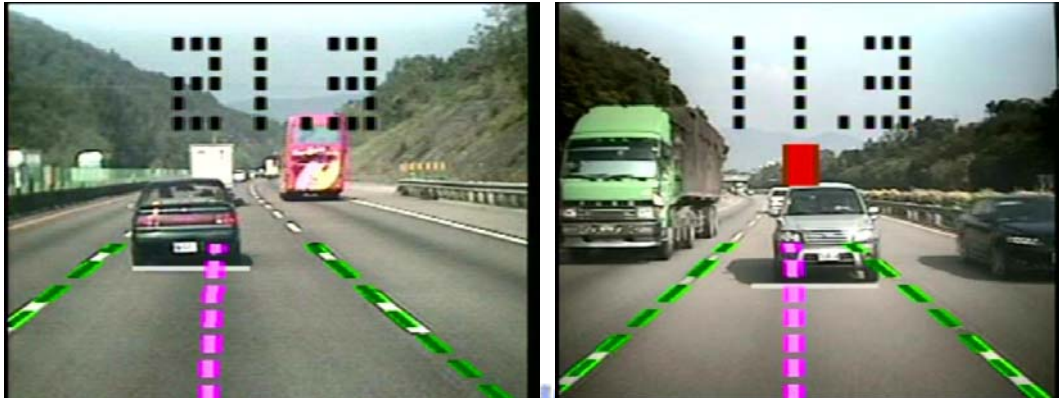


Fig. 5-19 Detection results in front-rear views from LVWS

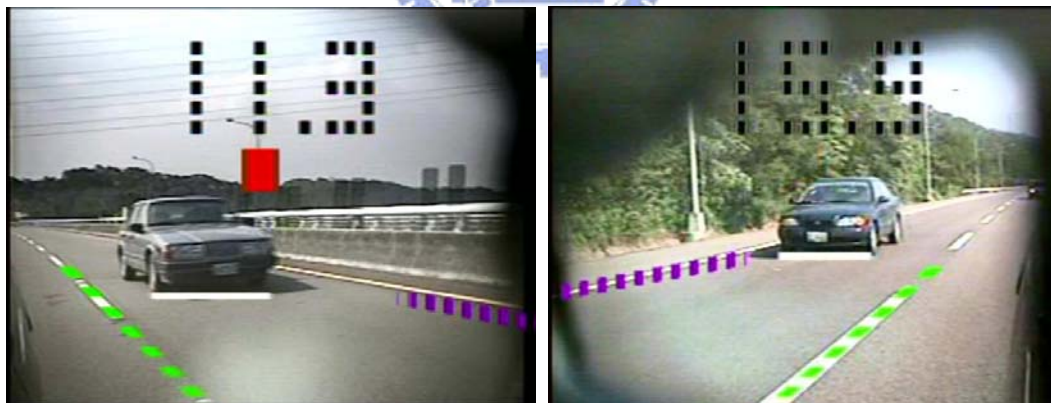


Fig. 5-20 Detection results in side views from BSWS



Fig. 5-21 Detection results in different conditions

5.5.2. Friendly Distance Display

If the approaching vehicle is detected, the system estimates the distance between the vehicle and the host vehicle. The data of the distance in four detected direction is encoded and transferred to the OMAP via ZigBee. In Fig. 5-22, a friendly GUI is designed to assist the driver to easily know the distance with other vehicles. With this friendly design, the driver can identify the location of other vehicles more quickly and avoid the possible collision more easily.

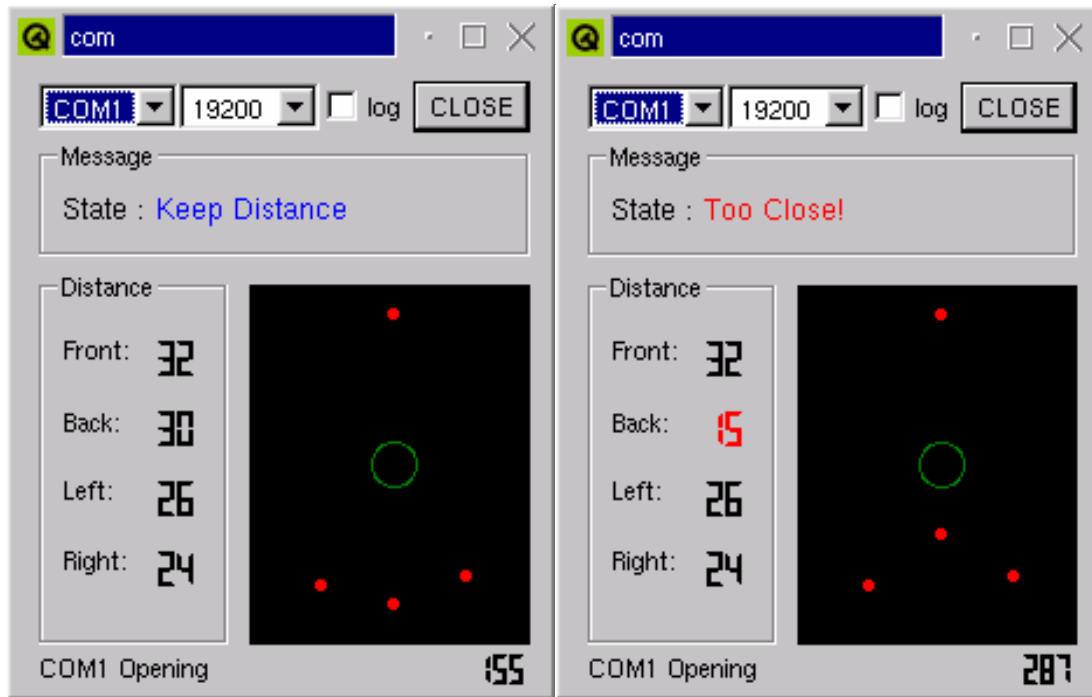


Fig. 5-22 Distance and location on screen

5.5.3. Experimental Results of H.264 Encoder

Table 5-3 shows the experimental results of H.264 encoder, the bitrate of the processed streams is at 32kbps, the quality of each stream in PSNR, separated in Y, Cb, and Cr, is listed, and the test sequences include Akiyo, Grandma, Salesman, Bridge, News, Mother and daughter, Container, and Claire. Figure 5-23(a) illustrates the result of mother and daughter that does not apply the deblocking filter, and Fig. 5-23(b) depicts the conditions of that does. The difference of the effect of the applying deblocking filter will contribute the higher quality with the cost of longer encoding time.

Table 5-3 Performance comparison with deblocking filter

Sequence	VSS @ 32kbps Without deblocking filter				VSS @ 32kbps With deblocking filter			
	Y	Cb	Cr	MCPS	Y	Cb	Cr	MCPS
Akiyo	35.73	40.90	41.62	7.91	36.30	41.32	42.45	8.62
Grandma	33.29	38.74	39.36	9.95	33.62	39.21	39.75	10.60
Salesman	30.83	38.11	38.78	9.14	31.13	38.74	39.33	9.86
Bridge	31.61	36.50	37.28	14.6	31.72	36.73	37.50	15.37
News	26.22	34.90	36.35	8.96	26.35	35.41	36.92	9.73
Mthr_dot	32.69	40.35	41.32	9.54	33.49	40.81	41.82	10.31
Container	29.75	38.69	38.32	9.78	29.93	38.91	38.56	10.55
Claire	25.45	35.58	37.81	9.87	25.66	36.01	38.44	10.72

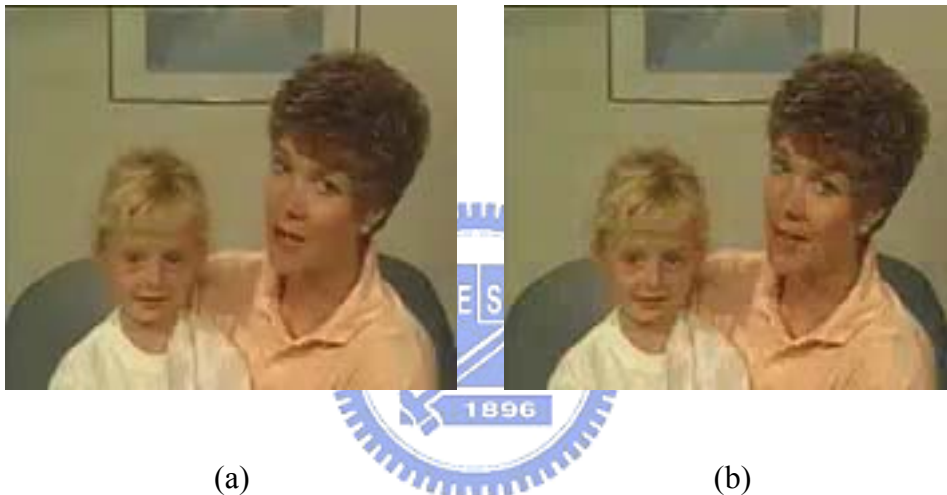


Fig. 5-23 The experimental results of (a) applying deblocking filter and (b) not applying it

From [117], the processor used for the H.264 encoder in their platform is TMS320C55X, which is the same DSP processor as the proposed system. The image size of test sequences is QCIF. As shown in Table 5-4, the performance of the proposed system has higher PSNR and lower Mega Cycles Per Second (MCPS) than [117]. Without using the prediction mode like [117], which could cause less appropriate prediction making and more PSNR loss, more efforts are spent on increasing the computing efficiency of H.264 encoding. For example, there are totally four prediction modes for a 16×16 macroblock which has upper and left neighborhood

in the 16x16 intra prediction. By statistical analysis, reducing four modes into one can increase computing speed and there is only 0.01~0.02dB video quality loss in PSNR, as shown in Table 5-1 and Table 5-2. Therefore, reducing the number of intra prediction modes for each 16x16 macroblock into one helps us increase encoding efficiency and maintain the video quality.

Table 5-4 PSNR and MCPS Comparison

Sequence	[117] @ 32kbps				VSS @ 32kbps			
	Y	Cb	Cr	MCPS	Y	Cb	Cr	MCPS
Akiyo	31.4	36.39	38.16	12.50	36.30	41.32	42.45	8.62
Grandma	30.94	36.4	37.09	12.88	33.62	39.21	39.75	10.60
Salesman	27.24	36.52	36.93	15.50	31.13	38.74	39.33	9.86
Bridge	28.41	36.44	36.95	14.96	31.72	36.73	37.50	15.37
News	26.3	34.42	35.87	13.54	26.35	35.41	36.92	9.73
Mthr_dot	28.84	37.02	36.94	13.71	33.49	40.81	41.82	10.31
Container	28	37.01	36.55	14.59	29.93	38.91	38.56	10.55
Claire	34.25	37.11	39.24	11.30	25.66	36.01	38.44	10.72

In [118], the processor used in their platform is TMS320C6416, which is more powerful than the proposed system, and the image size of the test sequences is also QCIF. Due to the limited computing power for the proposed system, reducing computational complexity is the critical point. Without using different block sizes, like 16x16, 16x8, 8x16 and 8x8 for inter prediction, only one block size 16x16 is used. Therefore, the encoding speed can be greatly increased with some loss of video quality and compression ratio. The comparison results are shown in Table 5-5.

Table 5-5 PSNR and comparison ratio comparison

[118] @ Qp=28					
Sequence	SNR(Y)	SNR(U)	SNR(V)	Compression Ratio	MCPS
Foreman	36.19 dB	40.70 dB	41.97 dB	50.73	24.50
Container	36.34 dB	41.25 dB	41.25 dB	102.04	22.53
Susie	37.14 dB	43.65 dB	43.32 dB	71.89	23.34
News	36.81 dB	39.85 dB	40.61 dB	70.40	22.20
VSS @ Qp=28					
Foreman	34.47 dB	40.06 dB	41.40 dB	25.37	15.46
Container	34.71 dB	41.76 dB	41.62 dB	103.13	9.74
Susie	36.15 dB	44.29 dB	44.10 dB	60.53	13.51
News	33.01 dB	41.33 dB	41.88 dB	66.29	9.25

Moreover, TMS320C55X is known for its power saving advantage, which is more suitable for handheld devices such as mobile phones and PDAs than 64x. Table 5-5 shows that not only our algorithm has made H.264 encoder on DSP an effective one, but also made such encoder more practical for consumer electronics. From above comparison results, it is proved that the optimized algorithm of the proposed system has brought better results in both video quality and encoding speed.

5.5.4. The Performance and Functionalities of VSS

With the tight cooperation between ARM and DSP, the system parameters are shown in Table 5-6. If the images are always ready for processing, the H.264 encoder with the efficient AMEA can calculate 7 FPS. However, after adding the system overhead, such as the dual-core intercommunication, the frame grabbing latency, the memory collision hazard, the 3.5G/3G internet transmission delay, and etc., the actual frame rate of the surveillance video, which is decoded and shown on the screen of the mobile phone, is around 4 FPS, which is enough for the checking the status of the vehicles and recognizing the intruders. Moreover, VSS will be activated only when

the triggers from the sensors and the users, or it will stay in the suspend mode to greatly save the battery energy when the car is not moving and the fuel-to-electricity procedure is disable. The power consumption of VSS is quite small no matter if it is enable or not, and VSS can wake up from the suspend mode in a very short time to provide the users real-time notifications. Therefore, VSS is very suitable for developing the vehicle applications.

The GUI developed on mobile phones is shown in Fig. 5-24. The top of the screen in Fig. 5-25 displays the date and time of the captured image, and the two buttons on the bottom are “Option” in Chinese and “Stop”, respectively. The “Stop” button can suspend the image transmission, as illustrated in Fig. 5-26, and resume transmission by pressing the “Resume” button.

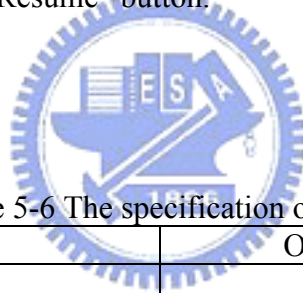


Table 5-6 The specification of VSS.

Processor		OMAP5912 @ 192 MHz	
Profile		H.264 baseline	
Frame rate	Only encode	7 frames/second	
	Overall	4 frames/second	
Average compression ratio		91.78	
Image size		QCIF	
Power consumption		Active	5W
		Suspend	20 μ W
Transmission method		3.5G / 3G	
Storage		Compact Flash Card	



Fig. 5-24 The GUI developed on mobile phones.



Fig. 5-25 The screen of our program on Nokia mobile phone.



Fig. 5-26 The suspend mode.

5.5.5. Comparisons

Finally, the comparisons in functionalities between several different systems are shown in Table 5-7. [110] has the highest performance in the detection of the lane departure warning, but the platform is too expensive. Furthermore, it can only detect the lane markings and the vehicles in the front view. [108] is an embedded processor-based system which can greatly reduce the cost and the power consumption. However, they can only locate the straight roads which are not robust enough to provide better safety assist. [97] can detect the vehicles in both front and rear directions, but it has to be installed in at least two cars to take effect. [114] is equipped with a GPS receiver to obtain vehicle's location, velocity and orientation periodically. Nevertheless, it can not show the real causes of the casualty. The proposed system can manipulate the image in CIF format of 30 FPS in omni-direction, and it is a low-power and stable DSP-based system, which is very suitable for installing in the vehicles. It can adapt the conditions of curved roads, complex roads, rainy days, cloudy days, and night. Besides, DASS can film the situation to provide the facts to the users and the police officers, and they can easily and quickly understand the reasons. Furthermore, DASS provides the unique function – mobile surveillance, which is useful for keeping the thieves away from users' properties. With these full-time safety assist features, DASS can establish a safer driving and parking environment.

Table 5-7 The comparisons between different systems.

Systems	Platforms	Frame size/ Frame rate	Road type	Vehicle detection	Mobile notification/ Vehicle surveillance	Status Recorder	Directions
DASS	DSP-based	352×240 / 30 FPS	Curved	Yes	Yes	Yes	Front, rear and side
[108]	ARM-based	128×128 / 22 FPS	Straight only	No	No	No	Only front
[110]	ASIC / power PC	640×480 / 30 FPS	Curved	Yes	No	No	Only front
[97]	PC-Based	X	X	Yes	No	No	Front and rear
[114]	PC-Based	X	X	No	No	Yes	X



Chapter 6. Conclusions and Perspective

In this chapter, a brief conclusion of this dissertation is stated, and some useful directions for future works on the research studies are also presented.

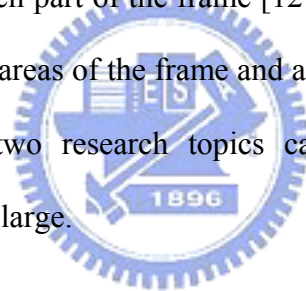
6.1 The hardware-based HMEA and RPIMC

This dissertation has addressed the design and implementation of HMEA, which efficiently uses an averaging filter for down-sampling and multiple MV candidates for fast search. HMEA exhibits its advantages by providing not only a PSNR performance close to that of FSBMA, but also a simple hardware architecture that is appropriate for VLSI implementation. The developed VLSI architecture includes an averaging filter to down-sample the images and 2D semi-systolic PE arrays to compute the SAD in pipeline. It is implemented with reasonable area, 59K with 1393 bytes of on-chip RAM, and a low required operating frequency, 5.88 MHz for the search area of $[-16, +15]$, and 10.68 MHz for $[-16.0, +15.5]$, in the real-time application for CIF images. Furthermore, the area-speed performance of HMEA is better than many existing architectures based on FSBMA or MMEA.

Moreover, this dissertation has also addressed the implementation of RPIMC, which can encode various format of the input image by setting the corresponding register bank, and its system-level design. With the programmable controller and the efficient HMEA, RPIMC can compress the real-time video at low operating frequency, 21 MHz, for the real-time application for CIF images to reduce the power consumption, compared with other MPEG-4 chips. RPIMC is designed to be easily

integrated into other platforms by modifying the wrapper to achieve the platform-independent purpose to wider its applications. Furthermore, the system-level design of REMR is also developed to illustrate the simple usage of RPIMC, and the software/hardware co-design makes REMR can process the real-time images and audio at the same time.

To achieve more effective and satisfactory performance in hardware video coder issues, the further research directions of this topic may be: (1) to develop the multi-frame ME algorithm, it enables the encoder to search for the best ‘match’ for the current MB from a wider set of pictures than just the previously encoded frame [119][120]. (2) To implement the multi-blocksize ME algorithm, the encoder selects the ‘best’ partition size for each part of the frame [121][122]. A large partition size is appropriate for homogeneous areas of the frame and a small one may be beneficial for detailed areas. Both these two research topics can improve the image quality especially when the motion is large.

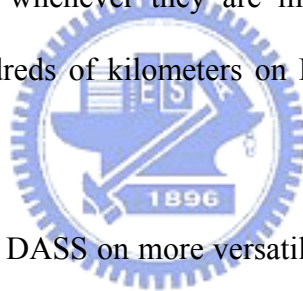


6.2 The software-based AMEA and DASS

AMEA has overcome the video encoding problems, such as lighting effects, outside environmental influences, and the quality issues of the in-ROI part, in the vehicle surveillance videos. Moreover, the design and implementation of AMEA, which adopts an efficient HEA to pre-process the images to reduce the luminance changing due to the different environments, are proposed. An adaptive and low complexity MV search procedure in and out of ROI is developed to increase the calculation speed and the encoding quality, and be integrated into VSS, which is

suitable for telematics because of its low power consumption and stability. VSS can capture the real-time images inside the car, compress it with H.264 standard, and transfer it to the mobile phones. The programs with friendly GUI are completed to make the users to browse the instant videos through 3.5G/3G network to determine whether their properties are stolen or not anytime and anywhere.

Moreover, DASS with the lane and vehicle detection in omni-direction of several conditions, such as general cases, rainy days, complex roads, curved roads, and nights, are implemented. Unlike other systems that can only provide single function, DASS also develops three main safety assist functions, lane departure and collision warning, full-time driving status recorder, and mobile surveillance system. DASS can protect the drivers and the vehicles whenever they are moving or not, and it has been successfully verified for hundreds of kilometers on Highway No.3 and Expressway No.68 in Taiwan.



To enhance AMEA and DASS on more versatile applications, future work may focus on: (1) to adjust the ROI adaptively [123][124]. Since the ROI will be varied car by car due to the installation, face detection should be applied to automatically focus on the important area that will provide more useful information. (2) To change the compression ratio according to the network bandwidth [125][126]. VSS will transmit the bitstream to users' mobile phone wirelessly, so the ME and the texture coding blocks should modify their accuracy based on the feedback of the rate control engine. (3) To simplify the encoding procedure [127][128]. There have two approaches to solve this issue: one is the simplification of H.264, and the other is to make the current MB pass the processing steps. Therefore, all-zero block detection will be a good method to achieve extra computational saving.

References

- [1] *Information Technology – Coding of Audio-Visual Objects – Part 2: Visual*, ISO/IEC 14496-2. MPEG-4, 2001.
- [2] *Joint Video Team (JVT) of ISO MPEG and ITU-T VCEG, JVT-G050*, ITU-T Rec. H.264/ISO/ICE 14496-10 AVC, 2003.
- [3] *Information Technology - Coding of Moving Pictures and Associated Audio for Digital Storage Media at Up to About 1.5Mbit/s*, ISO/IEC 11182. MPEG-1, 1993.
- [4] *Information Technology: Generic Coding of Moving Pictures and Associated Audio Information: Video*, ISO/IEC 13818. MPEG-2, 1995.
- [5] *Video Coding for Low Bitrate Communication*, ITU-T Rec. H.263, 2000.
- [6] *Video Codec for Audiovisual Services at p*64 kbit/s*, ITU-T Rec. H.261, 1993.
- [7] B. F. Wu, H. Y. Peng, C. J. Chen and T. L. Yu, "An Efficient Hierarchical Motion Estimation Algorithm and Its VLSI Architecture," *Proc. of the 7th IASTED International Conference on Signal and Image Processing*, SIP2005, pp. 64-69, Aug., 15-17, 2005, Honolulu, Hawaii, USA.
- [8] B. F. Wu, H. Y. Peng and T. L. Yu, "Efficient Hierarchical Motion Estimation Algorithm and Its VLSI Architecture," accepted by *IEEE Transactions on Very Large Scale Integration Systems*, Sep., 2007.
- [9] B. F. Wu and H. Y. Peng, "Platform Independent MPEG-4 Co-processor," *Proc. of the National Computer Symposium 2007*, NCS2007, Dec., 20-21, 2007, Taichung, Taiwan, ROC.
- [10] B. F. Wu, and H. Y. Peng, "Platform Independent MPEG-4 Co-processor and Its System-level Design," submitted for publication in *Journal of Information Science and Engineering*, Nov., 2007.
- [11] B. F. Wu, and H. Y. Peng, "An Adaptive Motion Estimation Algorithm for Embedded Mobile Vehicle Surveillance Systems," submitted for publication in *IEEE Transactions on Vehicular Technology*, Apr., 2008.
- [12] B. F. Wu, H. Y. Peng, H. Y. Huang, and C. Y. Hsu, "A New Dual-core DSP Based H.264/AAC Encoder Design," submitted for publication in *IEEE Transactions on Consumer Electronics*, Nov., 2007.
- [13] B. F. Wu, H. Y. Peng, C. J. Chen and Y. H. Chen, "A Real-time DSP-based Driving Assistance and Surveillance System," *Proc. of the National Computer Symposium 2007*, NCS2007, Dec. 20-21, 2007, Taichung, Taiwan, ROC.
- [14] B. F. Wu, H. Y. Peng, C. J. Chen and Y. H. Chen, "A Real-time DSP-based Driving Assistance and Surveillance System," revised to *Journal of Information*

Science and Engineering, May, 2008.

- [15] B. F. Wu, H. Y. Peng, Y. P. Hsu, M. W. Chung, P. T. Tsai and C. J. Chen, "An Advanced Driver Safety Assist System," *Electronic Engineering & Product World*, Vol. 224, pp. 56-61, Feb., 2007.
- [16] B. F. Wu, Y. H. Chen, H. Y. Peng and C. J. Chen, "A Vision-Based Vehicle Safety Assist System," to be submitted to *IEEE Transactions on Vehicular Technology*.
- [17] B. F. Wu, H. Y. Peng, C. J. Chen and Y. H. Chan, "An Encrypted Mobile Embedded Surveillance System," *Proc. of 2005 IEEE Intelligent Vehicles Symposium*, IV05, pp. 502-507, Jun., 6-8, 2005, Las Vegas, Nevada, USA.
- [18] B. F. Wu, H. Y. Peng and C. J. Chen, "A Practical Home Security System via Mobile Phones," *WSEAS Transactions on Communications*, Issue 6, Vol. 5, pp. 1061-1066, Jun., 2006.
- [19] B. F. Wu, C. J. Chen and H. Y. Peng, "An Innovative Video Phone System in DHCP and Firewall Network Environment," *WSEAS Transactions on Signal Processing*, Issue 5, Vol. 2, pp. 768-773, May, 2006.
- [20] B. F. Wu, C. J. Chen, H. H. Chiang, H. Y. Peng, J. W. Perng, L. S. Ma, and T. T. Lee, "The Design of Intelligent Real-Time Autonomous Vehicle, TAIWAN iTS-1," *Journal of the Chinese Institute of Engineers*, Vol. 30, No. 5, pp. 829-842, 2007.
- [21] L. De Vos and M. Stegherr, "Parameterizable VLSI architectures for the full-search block-matching algorithm," *IEEE Trans. Circuits Syst.*, Vol. 36, pp. 1309-1316, Oct. 1989.
- [22] T. Komarek and P. Pirsh, "Array architecture for block matching algorithms," *IEEE Trans. Circuits Syst.*, Vol. 36, pp 1301-1308, Oct. 1989.
- [23] 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.*, Vol. 36, pp. 1317-1325, Oct. 1989.
- [24] C. H. Hsieh and T. P. Lin, "VLSI architecture for block-matching motion estimation algorithm," *IEEE Trans. Circuits Syst. Video Technol.*, Vol. 2, pp. 169-175, June 1992.
- [25] 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.
- [26] 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, pp. 407-416, Oct. 1995.
- [27] Y. H. Yeh and C. Y. Lee, "Cost-Effective VLSI Architectures and Buffer Size

- Optimization for Full-Search Block –Matching Algorithms,” *IEEE Trans. VLSI Systems*, Vol. 7, No. 3, pp. 345-358, Sep. 1999.
- [28] 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, July 2001.
- [29] 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, pp. 61-72, 2002.
- [30] L. C. Liu, J. C. Chien, Y. H. Chuang, and C. C. Li, “A frame-level FSBM motion estimation architecture with large search range,” *Proc. of IEEE Conference on Advanced Video and Signal Based Surveillance*, Miami, USA, pp. 327-333, 2003.
- [31] W. F. He, Y. L. Bi, and Z. G. Mao, “Efficient frame-level pipelined array architecture for full-search block matching motion estimation,” *Proc. of IEEE Internal Symposium on Circuits and Systems*, Kobe, Japan, pp. 2887-2890, 2005.
- [32] W. F. He, M. L. Zhao, C. Y. Tsui, and Z. G. Mao, “A Scalable Frame-Level Pipelined Architecture for FSBM Motion Estimation,” *Proc. of IEEE Internal Conference on VLSI Design*, pp. 830-835, 2007.
- [33] W. Li and E. Salari, “Successive elimination algorithm for motion estimation,” *IEEE Trans. Image Processing*, Vol. 4, pp. 231-236, June 1995.
- [34] X. Q. Gao, C. J. Duanmu, and C. R. Zou, “A multilevel successive elimination algorithm for block matching motion estimation,” *IEEE Trans. Image Processing*, Vol. 9, pp. 501-504, Mar. 2000.
- [35] M. Brunig and W. Niehsen, “Fast full-search block matching,” *IEEE Trans. Circuits Syst. Video Technol.*, Vol. 11, pp. 241-247, Feb. 2001.
- [36] Digital Video Coding Group, *ITU-T Recommendation H.263 Software Implementation*, Telenor R&D, 1995.
- [37] Y. S. Chen, Y. P. Huang, and C. S. Fuh, “Fast block matching algorithm based on the winner-update strategy,” *IEEE Trans. Image Processing*, Vol. 10, pp. 1212-1222, Aug. 2001.
- [38] 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, Feb. 2000.
- [39] Y. W. Huang, S. Y. Chien, B. Y. Hsieh, and L. G. Chen, “An Efficient and Low Power Architecture Design for Motion Estimation Using Global Elimination Algorithm,” *Proc. of IEEE International Conference on Acoustics, Speech, and Signal Processing*, Vol. 3, May 2002.
- [40] Y. W. Huang, S. Y. Chien, B. Y. Hsieh, and L. G. Chen, “Global Elimination

- Algorithm and Architecture Design for Fast Block Matching,” *IEEE Trans. Circuits Syst. Video Technol.*, Vol. 14, Issue 6, pp. 898-907, June 2004.
- [41] P. I. Hosur and K. K. Ma, “Motion Vector Field Adaptive Fast Motion Estimation,” *Proc. of Second International Conference on Information, Communications and Signal Processing*, Singapore, Dec. 1999.
- [42] A. M. Tourapis, O. C. Au, and M. L. Liou, “Predictive Motion Vector Field Adaptive Search Technique (PMVFAST) – Enhancing Block Based Motion Estimation,” *Proc. of Visual Communications and Image Processing*, San Jose, CA, Jan. 2001.
- [43] A. M. Tourapis, “Enhanced Predictive Zonal Search for Single and Multiple Frame Motion Estimation,” *Proc. of Visual Communications and Image Processing*, San Jose, CA, Jan. 2002.
- [44] T. Li, S. Li, and C. Shen, “A Novel Configurable Motion Estimation Architecture for High-Efficiency MPEG-4/H.264 Encoding,” *Proc. of the Asia and South Pacific Design Automation Conference*, Vol. 2, pp. 1264-1267, Jan. 2005.
- [45] T. Koga, K. Iinuma, A. Hirano, Y. Iijima, and T. Ishiguro, “Motion-compensated interframe coding for video conferencing,” *Proc. of National Telecommunication Conference*, pp. C9.6.1-9.6.5, New Orleans, LA, Nov./Dec. 1981.
- [46] 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, pp. 438-422, Aug. 1994.
- [47] 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, pp. 313-317, June 1996.
- [48] M. Bierling, “Displacement estimation by hierarchical block matching,” *Proc. of International Symposium on Visual Communications & Image Processing*, Vol. SPIE-1001, pp. 942-951, 1988.
- [49] G. Gupta and C. Chakrabarti, “Architectures for hierarchical and other block matching algorithm,” *IEEE Trans. Circuits Syst. Video Technol.*, Vol. 5, pp. 477-489, Dec. 1995.
- [50] T. Onoye et al., “A VLSI architecture for MPEG2 MP@HL real time motion estimator,” *Proc. of IEEE International Symposium on Circuits and Systems*, 1996, pp. 664-667.
- [51] M. Uz, M. Vetterli, and D. Le Gall, “Interpolative multiresolution coding of advanced TV and compatible subchannels,” *IEEE Trans. Circuits Syst. Video Technol.*, Vol. 1, pp. 86-99, Mar. 1991.
- [52] J. Chalidabhongse and C.-C. J. Kuo, “Fast motion vector estimation using multiresolution-spatio-temporal correlations,” *IEEE Trans. Circuits Syst. Video*

Technol., Vol. 7, pp. 477-488, June 1997.

- [53] S. Zafar, Y.-Q. Zhang, and B. Jabbari, "Multiscale video representation using multiresolution motion compensation and wavelet decomposition," *IEEE J. Select. Areas Commun.*, Vol. 11, pp. 24-35, Jan. 1993.
- [54] J. Li, X. Lin, and Y. Wu, "Multiresolution tree architecture with its application in video sequence coding: A new result," *Proc. of International Symposium on Visual Communications & Image Processing*, Vol. 2094, pp. 730-741, 1993.
- [55] K. M. Nam, J. S. Kim, R. H. Park, and Y. S. Shim, "A fast hierarchical motion vector estimation algorithm using mean pyramid," *IEEE Trans. Circuits Syst. Video Technol.*, Vol. 5, pp. 344-351, Aug. 1995.
- [56] K. W. Chun and J. B. Ra, "An improved block matching algorithm based on successive refinements of motion vector candidates," *Signal Processing: Image Commun.*, Vol. 6, pp. 115-122, 1993.
- [57] K. W. Cheng and S. C. Chan, "Fast block matching algorithms for motion estimation," *Proc. of IEEE International Conference on Acoustics, Speech, and Signal Processing*, pp. 2311-2314, 1996.
- [58] J. C.-H. Ju, Y.-K. Chen, and S. Y. Kung, "A fast rate-optimized motion estimation algorithm for low-bit-rate video coding," *IEEE Trans. Circuit Syst. Video Technol.*, Vol. 9, pp. 994-1002, Oct. 1999.
- [59] X. Song, T. Chiang, X. Lee, and Y.-Q. Zhang, "New fast binary pyramid motion estimation for MPEG2 and HDTV encoding," *IEEE Trans. Circuits Syst. Video Technol.*, Vol. 10, pp. 1015-1028, Oct. 2000.
- [60] B.-M. Wang, J.-C. Yen, and S. Chang, "Zero waiting-cycle hierarchical block matching algorithm and its array architecture," *IEEE Trans. Circuits Syst. Video Technol.*, Vol. 4, pp. 18-28, Feb. 1994.
- [61] L. De Vos, "VLSI architectures for the hierarchical block-matching algorithms for HDTV applications," *Proc. of International Symposium on Visual Communications & Image Processing*, pp. 398-409, 1990.
- [62] J. H. Lee, K. W. Lim, B. C. Song, and J. B. Ra, "A Fast Multi-resolution Block Matching Algorithm and its LSI Architecture for Low Bit-Rate Video Coding," *IEEE Trans. Circuits Syst. Video Technol.*, Vol. 11, No. 12, Dec. 2001.
- [63] K. B. Lee, H. Y. Chin, H. C. Hsu, and C. W. Jen, "QME: an efficient subsampling-based block matching algorithm for motion estimation," *Proc. of International Symposium on Compound Semiconductors*, Vol. 2, pp. 23-26, May 2004.
- [64] B. E. Usevitch, "A Tutorial on Modern Lossy Wavelet Image Compression: Foundations of JPEG2000," *IEEE Signal Processing Magazine*, Vol. 18, No. 5, pp. 22-35, 2001.
- [65] J. Lu and M. L. Liou, "A simple efficient search algorithm for block matching

- motion estimation,” *IEEE Trans. Circuits Syst. Video Technol.*, vol. 7, Apr. 1997, 429-433.
- [66] A. Hatabu, T. Miyazaki, and I. Kuroda, “QVGA/CIF resolution MPEG-4 video codec based on a low-power and general-purpose DSP,” in *IEEE Workshop on Signal Processing Systems (SIPS)*, 2002, pp. 15-20.
- [67] H. Nakayama, T. Yoshitake, H. Komazaki, Y. Watanabe, H. Araki, K. Morioka, J. Li, L. Prilin, S. Lee, H. Kubosawa, and Y. Otobe, “An MPEG-4 Video LSI with an Error-Resilient Codec Core Based on a Fast Motion Estimation Algorithm,” in *IEEE internal Solid-State Circuits Conference (ISSCC)*, 2002, Vol. 1, pp. 368-370.
- [68] Y. C. Chang, W. M. Chao and L. G. Chen, “Platform-based MPEG-4 Video Encoder SoC Design,” in *IEEE Workshop on Signal Processing Systems (SIPS)*, 2004, pp. 251-256.
- [69] J. H. Park, I. K. Kim, S. M. Kim, S. M. Park, B. T. Koo, K. S. Shin, K. B. Seo, and J. J. Cha, “MPEG-4 Video Codec on an ARM Core and AMBA,” in *Workshop and Exhibition on MPEG-4*, 2001, pp. 95-98.
- [70] M. Takahashi, T. Nishikawa, M. Hamada, T. Takayanagi, H. Arakida, N. Machida, H. Yamamoto, T. Fujiyoshi, Y. Ohashi, O. Yamagishi, T. Samata, A. Asano, T. Terazawa, K. Ohmori, Y. Watanabe, H. Nakamura, S. Minami, T. Kuroda, and T. Furuyama, “A 60-MHz 240-mW MPEG-4 Videophone LSI with 16-Mb Embedded DRAM,” *IEEE Journal of Solid-State Circuit*, Vol. 35, No. 11, pp. 1713-1721, Nov. 2000.
- [71] P. M. Kuhn and W. Stechele, “Complexity analysis of the emerging MPEG-4 standard as a basis for VLSI implementation,” in *International Conference on Visual Communications and Image Processing*, 1998.
- [72] H. C. Chang, L. G. Chen, M. Y. Hsu, and Y. C. Chang, “Performance analysis and architecture evaluation of MPEG-4 video codec system,” in *IEEE International Symposium on Circuits and Systems (ISCAS)*, 2000, Vol. 2, pp. 449-452.
- [73] C. H. Yen, Y. S. Lin, and B. F. Wu, “A low-complexity MP3 algorithm that uses a new rate control and a fast dequantization,” *IEEE Transaction on Consumer Electronics*, Vol. 51, Issue 2, May 2005, pp. 571-579.
- [74] Amphion CS6701 is available on <http://www.amphion.com/>
- [75] J. R. Jain and A. K. Jain, “Displacement measurement and its application in inter frame image coding,” *IEEE Trans. Commun.*, vol. COM-29, no. 12, pp. 1799-1808, Dec. 1981.
- [76] H. Nisar and T.-S Choi, “An advanced center biased search algorithm for motion estimation,” in *Proc. IEEE Int. Conf. Image Process.*, 2000, vol. 1, pp. 832-835.
- [77] M. Ghanbari, “The cross-search algorithm for motion estimation (image

- coding),” *IEEE Trans. Commun.*, vol. 38, no. 7, pp. 950-953, Jul. 1990.
- [78] L. Luo, C. Zou, X. Gao, and H. Zhenya, “A new prediction search algorithm for block motion estimation in video coding,” *IEEE Trans. Consumer Electron.*, vol. 42, no. 1, pp. 56-61, Feb. 1997.
- [79] C. Zhu, K.-P. Chau, and X. Lin, “Hexagon-based search pattern for fast block motion estimation,” *IEEE Trans. Circuits Syst. Video Technol.*, vol. 12, no. 5, pp. 349-355, May 2002.
- [80] C. Zhu, X. Lin, L. Chau, and L.-M. Po, “Enhanced hexagonal search for fast block motion estimation,” *IEEE Trans. Circuits Syst. Video Technol.*, vol. 14, no. 10, pp. 1210-1214, Oct. 2004.
- [81] G. Sorwar, M. Murshed, and L. S. Doolwey, “A Fully Adaptive Distance-Dependent Thresholding Search (FADTS) Algorithm for Performance-Management Motion Estimation,” *IEEE Trans. Circuits Syst. Video Technol.*, vol. 17, no. 4, Apr. 2007.
- [82] L. M. Bergasa, J. Nuevo, M. A. Sotelo, R. Barea, and M. E. Lopez, “Real-Time for Monitoring Driver Vigilance,” *IEEE Trans. Intelligent Transportation Syst.*, vol. 7, no. 1, pp. 63-77, Mar. 2006.
- [83] R. C. Gonzalez and R. E. Woods, “Digital Image Processing,” 2nd Edition, *Prentice Hall*, 2002.
- [84] Y.-T Kim, “Contrast Enhancement Using Brightness Preserving Bi-Histogram Equalization,” *IEEE Trans. Consum. Electr.*, vol. 7, no. 4, pp. 304-312, 1988.
- [85] Y. Wang, Q. Chen, and B. M. Zhang, “Image Enhancement based on Equal Area Dualistic Sub-image Histogram Equalization Method,” *IEEE Trans. Consum. Electr.*, vol. 45, no. 1, pp. 68-75, 1999.
- [86] S.-D. Chen and A. R. Ramli, “Minimum Mean Brightness Error Bi-Histogram Equalization in Contrast Enhancement,” *IEEE Trans. Consum. Electr.*, vol. 49, no. 4, pp. 1310-1319, 2003.
- [87] C. W. and Z. Ye, “Brightness Preserving Histogram Equalization with Maximum Entropy: A Variational Perspective,” *IEEE Trans. Consum. Electr.*, vol. 51, no. 4, pp. 1326-1334, 2005.
- [88] G. B. Thomas, *Calculus and Analytic Geometry*, 4th ed., New York: Addison-Wesley, 1968.
- [89] B. Widrow and S. D. Stearns, *Adaptive Signal Processing*, Engle-wood Cliffs, NJ: Prentice-Hall, 1985.
- [90] J. Y. Stein, *Digital Signal Processing: A Computer Science Perspective*, New York: Wiley, 2000.
- [91] E. Eweda and O. Macchi, “Convergence of the RLS and LMS adaptive filters,” *IEEE Trans. Circuits Syst.*, vol. 34, no. 7, pp. 799-803, Jul. 1987.

- [92] S. C. Douglas, "A family of normalized LMS algorithms," *IEEE Signal Process. Lett.*, vol. 1, no. 3, pp. 49-51, Mar. 1994.
- [93] K. Meghriche, S. Femmam, B. Derras, and M. Haddadi, "A non linear adaptive filter for digital data communication," in *Proc. 6th Int. Sump. Signal Process. Appl.*, Piscataway, NJ, 2001, vol. 1, pp. 304-306.
- [94] G. Jacobs, A. Aeron-Thomas, and A. Astrop, "Estimating global road fatalities," Australian National University, Transport Research Laboratory, Technical Report TRL 445, 1999.
- [95] D. Royal, "Volume I – Finding report; national survey on distracted and driving attitudes and behaviors, 2002," The Group Organization, Washington, D.C., Tech. Rep. DOT HS 809 566, Mar. 2003.
- [96] J. Hagen., "Road safety crisis," UN Chronicle Online Edition., 2004, http://www.un.org/Pubs/chronicle/2004/webArticles/012204_road_safety.asp
- [97] W. S. Wijesoma, K. R. S. Kodagoda, and A. P. Balasuriya, "Road-boundary detection and tracking using ladar sensing," *IEEE trans. on Robotics and Automation*, Vol. 20, Issue 3, pp. 456-464, June 2004.
- [98] S. T. Sheu, J. S. Wu, C. H. Huang, Y. C. Cheng and L. W. Chen, "DDAS: Distance and Direction Awareness System for Intelligent Vehicles," *Journal of Information Science and Engineering*, Vol. 23, pp. 709-722, 2007.
- [99] A. Kircher, M. Uddman, and J. Sandin, "vehicle control and drowsiness," Swedish National Road and Transport Research Institute, Linkoping, Sweden, Tech. Rep. VTI-922A, 2002.
- [100] T. Hayami, K. Matsunaga, K. Shidoji, and Y. Matsuki, "Detecting drowsiness while driving by measuring eye movement - a pilot study," *Proc. of IEEE Intelligent Vehicles Symposium*, pp. 156-161, Sept. 2002.
- [101] L. M. Bergasa, J. Nuevo, M. A. Sotelo, R. Barea, and M. E. Lopez, "Real-time system for monitoring driver vigilance," *IEEE trans. on Intelligent Transportation System*, Vol. 7, Issue 1, pp. 63-77, Mar. 2006.
- [102] V. Kastinaki, M. Zervakis, and K. Kalaitzakis, "A survey of video processing techniques for traffic applications," *Image and Vision Computing*, vol. 21, no. 4, pp. 359-381, April 2003.
- [103] J. W. Lee, C. D. Kee, and U. K. Yi, "A new approach for lane departure identification," *Proc. of IEEE Intelligent Vehicles Symposium*, Columbus, OH, June 2003, pp. 100-105.
- [104] S. Y. Kim and S. Y. Oh, "A adaptive lane departure warning system based on image processing and a fuzzy evolutionary technique," *Proc. of IEEE Intelligent Vehicles Symposium*, Columbus, OH, June 2003, pp. 361-365.
- [105] K. Weiss, N. Kaempchen, and A. Kirchner, "Multi-model tracking for the detection of lane change maneuvers," *Proc. of IEEE Intelligent Vehicles*

- Symposium*, Parma, Italy, 2004, pp. 937-942.
- [106]C. R. Jung and C. R. Kelber, "A lane departure warning system based on a linear-parabolic lane model," *Proc. of IEEE Intelligent Vehicles Symposium*, Parma, Italy, 2004, pp. 891-895.
- [107]J. Kaszubiak, M. Tornow, R. W. Kuhn, B. Michaelis, and C. Knoepfel, "Real-time vehicle and lane detection with embedded hardware," *Proc. of IEEE Intelligent Vehicles Symposium*, June 2005, pp. 619-624.
- [108]P. Y. Hsiao, C. W. Yeh, "A Portable Real-Time Lane Departure Warning System based on Embedded Calculating Technique," *Proc. of IEEE Vehicular Technology Conference*, 2006, pp. 2982-2986.
- [109]C. H. Yeh and Y. H. Chen, "Development of vision-based lane and vehicle detecting systems via the implementation with a dual-core DPS," *Proc. of IEEE Intelligent Vehicles Symposium*, Sep. 2006, pp. 17-20
- [110]<http://www.mobileye.com/asp4.shtml>
- [111]M. Bertozzi and A. Broggi, "GOLD: A parallel real-time stereo vision system for generic obstacle and lane detection," *IEEE Trans. on Image Processing*, vol. 7, pp. 62-81, January 1998.
- [112]A. Techmer, "Contour-Based Motion Estimation and Object Tracking for Real-Time Applications," *Proc. of IEEE Image Processing*, vol. 3, pp. 648-651, October 2001.
- [113]H. C. Yeh, S. Y. Chang, T. S. Chu, C. W. Chen and C. S. Shih, "Vehicle Information Systems Integration Framework," *Journal of Information Science and Engineering*, Vol. 23, pp. 681-695, 2007.
- [114][http:// www.onstar.com](http://www.onstar.com)
- [115][http:// www.tobe.com.tw](http://www.tobe.com.tw)
- [116]DSPGateway. URL:<http://dspgateway.sourceforge.net/pub/index.php>
- [117]T. Chattopadhyay, S. Banerjee, and A. Pal, "Enhancements of H.264 Encoder performance for video conferencing and videophone applications in TMS320C55X", *Proc. of IEEE Tenth International Symposium on Consumer Electronics*, June 2006.
- [118]Z. Wei and C. Cai, "Realization and Optimization of DSP Based H.264 Encoder", *Proc. of IEEE International Symposium on Circuits and Systems*, May 2006.
- [119]Y. W. Huang, B. Y. Hsieh, S. Y. Chien, S. Y. Ma and L. G. Chen, "Analysis and complexity reduction of multiple reference frames motion estimation in H.264/AVC," *IEEE Trans. Circuits Syst. Video Technol.*, Vol. 16, Issue 4, pp. 507-522, Apr., 2006.

- [120] Y. Su and M. T. Sun, "Fast multiple reference frame motion estimation for H.264/AVC," *IEEE Trans. Circuits Syst. Video Technol.*, Vol. 16, Issue 3, pp. 447-452, Mar., 2006.
- [121] T. Y. Kuo and C. H. Chan, "Fast Variable Block Size Motion Estimation for H.264 Using Likelihood and Correlation of Motion Field," *IEEE Trans. Circuits Syst. Video Technol.*, Vol. 16, Issue 10, pp. 1185-1195, Oct., 2006.
- [122] E.A.A. Qaralleh and T.-S. Chang, "Fast Variable Block Size Motion Estimation by Adaptive Early Termination," *IEEE Trans. Circuits Syst. Video Technol.*, Vol. 16, Issue 8, pp. 1021-1026, Aug., 2006.
- [123] K. Kollreider, H. Fronthaler, M. I. Faraj and J. Bigun, "Real-Time Face Detection and Motion Analysis With Application in "Liveness" Assessment," *IEEE Transactions on Information Forensics and Security*, Vol. 2, Issue 3, Part 2, pp. 548-558, Sep., 2007.
- [124] J. Wu, S. C. Brubaker, M. D. Mullin and J. M. Rehg, "Fast Asymmetric Learning for Cascade Face Detection," *IEEE Trans. Pattern Anal.*, Vol. 30, Issue 3, pp. 369-382, Mar., 2008.
- [125] Y. Liu, Z. G. Li and Y. C. Soh, "Rate Control of H.264/AVC Scalable Extension," *IEEE Trans. Circuits Syst. Video Technol.*, Vol. 18, Issue 1, pp. 116-121, Jan., 2008.
- [126] H. Wang and S. Kwong, "Rate-Distortion Optimization of Rate Control for H.264 With Adaptive Initial Quantization Parameter Determination," *IEEE Trans. Circuits Syst. Video Technol.*, Vol. 18, Issue 1, pp. 140-144, Jan., 2008.
- [127] Y. H. Moon, G. Y. Kim and J. H. Kim, "An improved early detection algorithm for all-zero blocks in H.264 video encoding," *IEEE Trans. Circuits Syst. Video Technol.*, Vol. 15, Issue 8, pp. 1053-1057, Aug., 2005.
- [128] Z. Xie, Y. Liu, J. Liu and T. Yang, "A General Method for Detecting All-Zero Blocks Prior to DCT and Quantization," *IEEE Trans. Circuits Syst. Video Technol.*, Vol. 17, Issue 2, pp. 237-241, Feb., 2007.

Curriculum Vitae



博 士 生：彭信元(Hsin-Yuan Peng)

指導教授：吳炳飛 教授(Prof. Bing-Fei Wu)

論文題目：MPEG-4/H.264 視訊壓縮標準於嵌入式系統之最佳化研究 (The Study of Optimization Approaches for MPEG-4/H.264 Video Processing Standards on Embedded Systems)

Educations

- 84 年 9 月～87 年 6 月 國立台東高級中學
- 87 年 9 月～91 年 6 月 國立交通大學電機與控制工程學系
- 91 年 9 月～now 國立交通大學電機與控制工程學系博士班

Publications

Referred Journal Papers:

- Bing-Fei Wu, Hsin-Yuan Peng and Tung-Lung Yu, “**Efficient Hierarchical Motion Estimation Algorithm and Its VLSI Architecture**,” accepted by *IEEE Transactions on Very Large Scale Integration Systems*, Sep., 2007.
- Bing-Fei Wu, Chao-Jung Chen, Hsin-han Chiang, Hsin-Yuan Peng, Jau-Woei Perng, Li-Shian Ma, and Tsu-Tian Lee, “**The Design of Intelligent Real-Time Autonomous Vehicle, TAIWAN iTS-1**,” *Journal of the Chinese Institute of Engineers*, Vol. 30, No. 5, pp. 829-842, 2007.
- Bing-Fei Wu, Hsin-Yuan Peng, Yi-Pin Hsu, Ming-Wei Chung, Ping-Tsung Tsai and Chao-Jung Chen, “**An Advanced Driver Safety Assist System**,” *Electronic Engineering & Product World*, Vol. 224, pp. 56-61, Feb., 2007.
- Bing-Fei Wu, Hsin-Yuan Peng and Chao-Jung Chen, “**A Practical Home Security System via Mobile Phones**,” *WSEAS Transactions on Communications*, Issue 6, Vol. 5, pp. 1061-1066, Jun., 2006.
- Bing-Fei Wu, Chao-Jung Chen and Hsin-Yuan Peng, “**An Innovative Video Phone System in DHCP and Firewall Network Environment**,” *WSEAS Transactions on Signal Processing*, Issue 5, Vol. 2, pp. 768-773, May, 2006.

Submitted Journal Papers:

- Bing-Fei Wu, Hsin-Yuan Peng, Chao-Jung Chen and Ying-Han Chen, “A

Real-time DSP-based Driving Assistance and Surveillance System,” revised to *Journal of Information Science and Engineering*, May, 2008.

- Bing-Fei Wu, and Hsin-Yuan Peng, “**An Adaptive Motion Estimation Algorithm for Embedded Mobile Vehicle Surveillance Systems,**” submitted for publication in *IEEE Transactions on Vehicular Technology*, Apr., 2008.
- Bing-Fei Wu, Hsin-Yuan Peng, Hao-Yu Huang, and Chin-Yuan Hsu, “**A New Dual-core DSP Based H.264/AAC Encoder Design,**” submitted for publication in *IEEE Transactions on Consumer Electronics*, Nov., 2007.
- Bing-Fei Wu, and Hsin-Yuan Peng, “**Platform Independent MPEG-4 Co-processor and Its System-level Design,**” submitted for publication in *Journal of Information Science and Engineering*, Nov., 2007.

To be submitted Papers:

- Bing-Fei Wu, Ying-Han Chen, Hsin-Yuan Peng and Chao-Jung Chen, “**A Vision-Based Vehicle Safety Assist System,**” to be submitted to *IEEE Transactions on Vehicular Technology*.
- Bing-Fei Wu, Hao-Yu Huang and Hsin-Yuan Peng, “**MPEG-2/4 Low-Complexity Advanced Audio Coding Optimization and Implementation on DSP,**” to be submitted to *IEEE Transactions on Consumer Electronics*.

Conference Papers:

- Bing-Fei Wu, Ying-Han Chen, Hsin-Yuan Peng, and Chao-Jung Chen, “**A Real-Time Vision-Based Safety Assist System,**” accepted by *IEEE International Conference on Systems, Man, and Cybernetics 2008*, SMC2008, Oct. 12-15, 2008, Singapore.
- Bing-Fei Wu and Hsin-Yuan Peng, “**Platform Independent MPEG-4 Co-processor,**” *Proceedings of the National Computer Symposium 2007*, NCS2007, Dec. 20-21, 2007, Taichung, Taiwan, ROC.
- Bing-Fei Wu, Hsin-Yuan Peng, Chao-Jung Chen and Ying-Han Chen, “**A Real-time DSP-based Driving Assistance and Surveillance System,**” *Proceedings of the National Computer Symposium 2007*, NCS2007, Dec. 20-21, 2007, Taichung, Taiwan, ROC.
- Bing-Fei Wu, Chao-Jung Chen and Hsin-Yuan Peng, “**An Innovative Video Phone System in DHCP Network Environment,**” *Proceedings of the International Conference on Consumer Electronics 2006*, ICCE2006, pp. 61-62, Jan., 7-11, 2006, Las Vegas, Nevada, USA.
- Bing-Fei Wu, Hsin-Yuan Peng, Chao-Jung Chen and Tung-Lung Yu, “**An Efficient Hierarchical Motion Estimation Algorithm and Its VLSI Architecture,**” *Proceedings of the 7th IASTED International Conference on Signal and Image Processing*, SIP2005, pp. 64-69, Aug., 15-17, 2005, Honolulu, Hawaii, USA.
- Bing-Fei Wu, Hsin-Yuan Peng, Chao-Jung Chen and Yi-Huan Chan, “**An Encrypted Mobile Embedded Surveillance System,**” *Proceedings of 2005 IEEE Intelligent Vehicles Symposium*, IV05, pp. 502-507, Jun., 6-8, 2005, Las Vegas, Nevada, USA.

Book Chapter:

- 吳炳飛、彭信元、游東龍、洪堯俊、陳昭榮, "MPEG-4 視訊壓縮技術", 全華科技圖書股份有限公司, 2006 年 11 月初版.

Patents:

- 吳炳飛、陳昭榮、彭信元, "用於車輛之影音記錄器", 中華民國發明專利第 I282942 號, 專利權期間 2007 年 6 月至 2025 年 5 月.
- 吳炳飛、瞿忠正、陳昭榮、彭信元、游東龍, "動態影像估測之裝置及方法", 中華民國發明專利第 I246323 號, 專利權期間 2005 年 12 月至 2024 年 7 月.
- 吳炳飛、陳昭榮、彭信元, "用於防盜之監視系統", 中華民國發明專利第 I233066 號, 專利權期間 2005 年 5 月至 2023 年 10 月.
- Bing-Fei Wu, Chao-Jung Chen, Hsin-Yuan Peng, "Audio-Video Recorder for Vehicles", applied for United States Patent, 11/273,109, Nov., 2005.
- Bing-Fei Wu, Chao-Jung Chen, Hsin-Yuan Peng, "Burglarproof Security System", applied for United States Patent, 10/867,724, Jun., 2004.

Experiences

- "嵌入式系統設計與實做", 經濟部工業局主辦, 工業技術研究院承辦, 交通大學電機與控制工程學系執行之中短期在職人才培訓班講師, 2007/7/2 ~ 2007/8/2
- "MPEG-4 視訊壓縮技術", 經濟部工業局主辦, 工業技術研究院承辦, 交通大學電機與控制工程學系執行之中短期在職人才培訓班講師, 2007/7/3 ~ 2007/7/20
- 88 學年度交通大學天文社社長
- 89 學年度交通大學天文社設備組長
- 88 至 89 學年度教育部"大專社團帶動中小學社團發展活動計劃"交大天文社負責人

Honors/Awards

2007	第二屆龍騰微笑競賽	參獎
	第七屆旺宏金砂獎	優等獎
	2007 亞洲區校際車輛創新設計大獎	表現優異獎
2006	德州儀器 2006 TI 大陸—台灣兩岸 DSP 大獎賽	一等獎
	德州儀器 亞洲區 DSP 競賽—台灣分區競賽	優等獎
2005	2005 機動車輛創新設計競賽	銅質獎
	教育部 94 學年度微電腦應用系統設計製作競賽	第三名
	交通大學電機與控制工程學系博士研究生	成績優異獎學金
2004	第六屆 TIC100 科技創新事業競賽冬令營	樹苗獎
	第四屆旺宏金砂獎	優勝獎

	第五屆台灣工業銀行創業大賽	佳作
2003	第五屆 TIC100 科技創新競賽	銀質獎
	教育部 92 學年度嵌入式軟體製作競賽	佳作
	91 學年度 第二學期 電機與控制工程學系研究生	書卷獎
	交通大學電機與控制工程學系博士研究生	成績優異獎學金
2002	91 學年度 交通大學 電機資訊學院 微電腦與嵌入式 應用系統設計製作競賽	優等獎
	91 學年度 第一學期 電機與控制工程學系研究生	書卷獎
2000	89 學年度 第一學期 電機與控制工程學系	書卷獎

