11-4-2010

Digital Surveillance Based on Video CODEC System-on-a-Chip (SoC) Platforms

Wei Zhao
Florida International University, wzhao001@fiu.edu

FLORIDA INTERNATIONAL UNIVERSITY

Miami, Florida

DIGITAL SURVEILLANCE BASED ON VIDEO CODEC SYSTEM-ON-A-CHIP (SOC) PLATFORMS

A dissertation submitted in partial fulfillment of the

requirements for the degree of

DOCTOR OF PHILOSOPHY

in

ELECTRICAL ENGINEERING

by

Wei Zhao

2010

To: Dean Amir Mirmiran
College of Engineering and Computing

This dissertation, written by Wei Zhao, and entitled Digital Surveillance Based on Video CODEC System-on-a-Chip (SoC) Platforms, having been approved in respect to style and intellectual content, is referred to you for judgment.

We have read this dissertation and recommend that it be approved.

Armando Barreto

Deng Pan

Jean H. Andrian

Chunlei Wang

Jeffrey Fan, Major Professor

Date of Defense: November 4, 2010

The dissertation of Wei Zhao is approved.

Dean Amir Mirmiran
College of Engineering and Computing

Interim Dean Kevin O'Shea
University Graduate School

Florida International University, 2010

DEDICATION

I dedicate this dissertation to my family, especially to my beloved wife, Xiang, for making this work meaningful. I also dedicate this work to my dear parents for their unfailing love, support, and encouragement. Without their patience, understanding, support, and love, the completion of this endeavor would never have been possible.

ACKNOWLEDGMENTS

I wish to express my deepest appreciation to my advisor, Dr. Jeffrey Fan, for his great mentoring and guidance throughout the entire period of my graduate study. His passion and dedication are incomparable and always inspire me.

I would also like to express my deep gratitude to the other committee members, Dr. Armando Barreto, Dr. Deng Pan, Dr. Jean H. Andrian, and Dr. Chunlei (Peggy) Wang for giving their valuable time in advising and encouraging me throughout this research.

I am also thankful to our VLSI lab for giving me access to the equipment and software that allowed me to perform my research from the very first day of my graduate study. In addition, I thank all of the VLSI lab members, especially Raul Batista, Charles Castello, and Priyanka Mekala, for their assistance on different tasks related to my research.

ABSTRACT OF THE DISSERTATION

DIGITAL SURVEILLANCE BASED ON VIDEO CODEC SYSTEM-ON-A-CHIP (SOC) PLATFORMS

by

Wei Zhao

Florida International University, 2010

Miami, Florida

Professor Jeffrey Fan, Major Professor

Today, most conventional surveillance networks are based on analog system, which has a lot of constraints like manpower and high-bandwidth requirements. It becomes the barrier for today's surveillance network development. This dissertation describes a digital surveillance network architecture based on the H.264 coding/decoding (CODEC) System-on-a-Chip (SoC) platform. The proposed digital surveillance network architecture includes three major layers: software layer, hardware layer, and the network layer.

The following outlines the contributions to the proposed digital surveillance network architecture. (1) We implement an object recognition system and an object categorization system on the software layer by applying several Digital Image Processing (DIP) algorithms. (2) For better compression ratio and higher video quality transfer, we implement two new modules on the hardware layer of the H.264 CODEC core, i.e., the background elimination module and the Directional Discrete Cosine Transform (DDCT) module. (3) Furthermore, we introduce a Digital Signal Processor (DSP) sub-system on the main bus of H.264 SoC platforms as the major hardware support system for our software architecture. Thus we combine the software and hardware platforms to be an intelligent surveillance node.

Lab results show that the proposed surveillance node can dramatically save the network resources like bandwidth and storage capacity.

TABLE OF CONTENTS

CHAPTER PAGE

LIST OF TABLES

TABLE PAGE

LIST OF FIGURES

SYMBOLS AND ABBREVIATIONS

SYMBOLS	DEFINITION
CCTV	Closed-circuit television
RT	Real-Time
NRT	Non-Real-Time
SoC	System-on-a-Chip
DVD	Digital Versatile Disk
SD	Standard Definition
HD	High Definition
720p	720 progressively scanned lines
1080p	1080 progressively scanned lines
GIPS	Giga-Instructions per Second
GBPS	Giga-Bytes per Second
DMAC	Direct Memory Access Controller
PMU	Power Management Unit
DDCT	Directional Discrete Cosine Transform
2D	Two-Dimension
DCT	Discrete Cosine Transform
DC	Direct Current
LoG	Laplacian of Gaussian
MB	Marco Block
VB	Vector Bank
DSP	Digital Signal Processing

DC	Direct Current
AC	Alternating Current
PSNR	Peak Signal-to-Noise Ratio
ANN	Artificial Neural Network
BP	Backpropagation
MSE	Mean Square Error
DIP	Digital Image Processing

I INTRODUCTION

1.1 Motivation

People have been developing surveillance systems for centuries. In its early history, a surveillance system consisted exclusively of humans because no cameras were available. This situation did not change until the middle of the 20th century. The first technology-based surveillance system, a Closed-Circuit Television (CCTV) system, was installed for military purposes in 1942 during World War II by Siemens AG at Test Stand VII in Peenemünde, Germany. In the 1960s, officials in the United Kingdom began installing CCTV systems in public places to monitor crowds during rallies and the appearances of public figures. In the United States, the first CCTV system set up in a public building was installed in the New York City Municipal building in 1969. Thus, these “electrical eyes” began to serve public purposes for the first time. Half a century has passed since the first CCTV was installed in a public building, and surveillance systems have successfully made their way into banks, Automated Teller Machines (ATMs), grocery stores, gas stations–almost corner-to-corner in public areas.

Generally, today’s surveillance network systems can be divided into two categories: Real-Time (RT) systems and Non-Real-Time (NRT) systems [1]. An RT surveillance system usually has a “secure room” with one or more security guards inside it. The video data from every video camera is sent to this “secure room”. The security guards watch the monitors in an attempt to detect each potential threat. An NRT system does not have a secure room. Instead of live monitoring, it stores the video data in files for later references.

Compared to an RT system, an NRT system is not a complete surveillance system. It cannot detect potential threats but can only record them. However, because of their extremely low cost and ease of use, NRT systems are the most widely implemented surveillance systems in modern society. Nearly every ATM and grocery store is using an NRT system; however, very few (and important) facilities, such as government buildings, business centers, and airports are using RT systems.

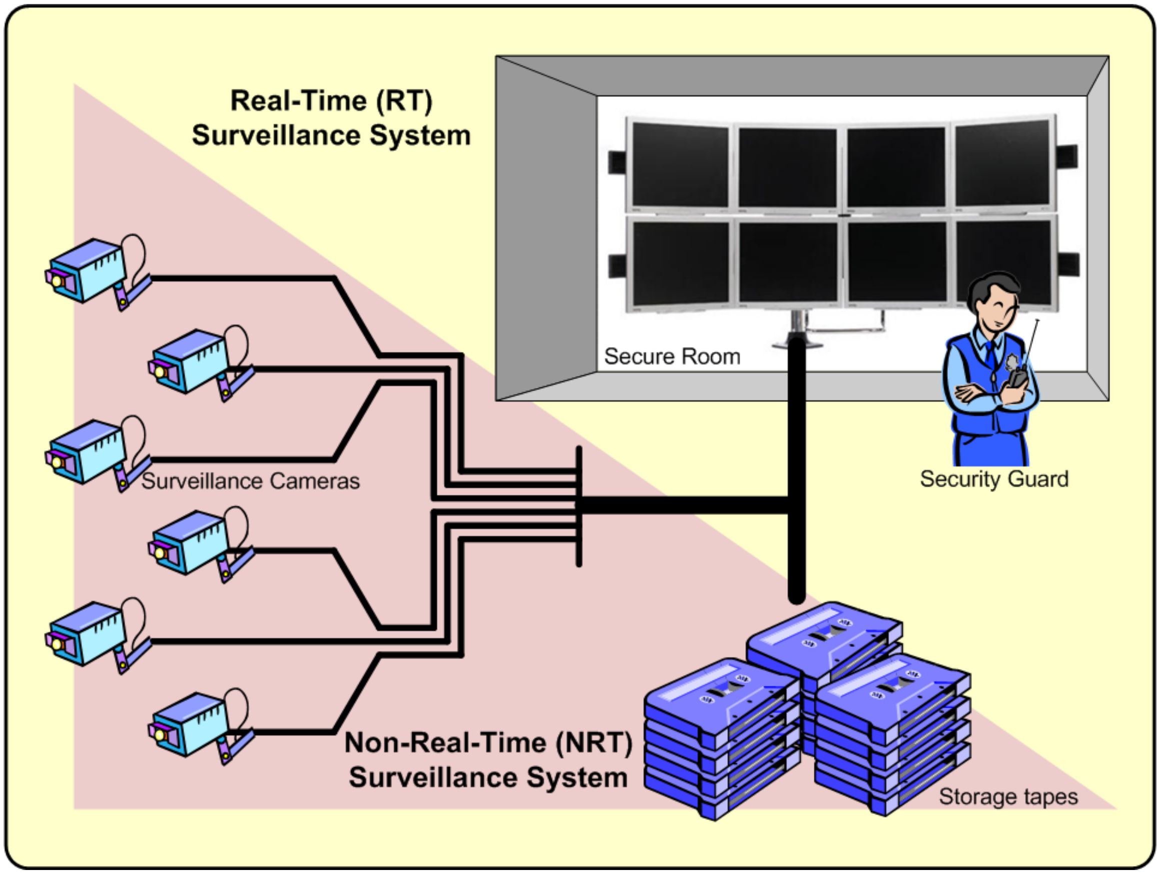


Figure I-1 RT surveillance system and NRT surveillance system

RT systems cannot be widely used because of their high cost and manpower requirement. Yet, the real reason for the high cost is the fact that the current video surveillance system is an analog system. This dissertation tries to demonstrate a purely digital surveillance network. With this network, we can build RT surveillance systems without the need for human guards, and the cost to set up this system will be even less than today's NRT surveillance system.

In this regard, what is an analog surveillance system? And what is the difference between an analog surveillance system and a digital surveillance system?

1.1.1 Conventional Analog Based Surveillance System

In an analog based surveillance system, the signals we mainly discuss are video signals. A typical Analog Surveillance System usually consists of

- analog video cameras
- analog video data transfer paths (e.g., video cables)
- analog video data storage equipment (e.g., cassette tapes)

For various reasons (e.g., supply shortage, etc.) some analog surveillance systems choose to store their analog video data digitally through the use of digital storage equipment (e.g., hard disks) instead of analog video data storage equipment (e.g., magnetic tapes). However, this does not change its attribute of being an analog surveillance system because its data transfer paths remain analog. Of the three previously-mentioned items, the "analog video data transfer path" is the most important, serving as an indicator of an analog surveillance system. Most of today's surveillance systems are analog surveillance systems.

1.1.2 Proposed Digital Surveillance System

The motivation of this dissertation is to build a digital surveillance system, which is a surveillance system based on digital signals. A typical digital surveillance system should consist of

- analog-to-digital video cameras
- digital data transfer paths (e.g., digital networks, fibers)
- digital data storage equipment (e.g., hard disks)

Just as with an analog surveillance network, the indicator of a digital surveillance network is the digital data transfer path.

1.1.3 Why Digital

Analog signals are everywhere in the real world. The only way to reproduce natural data (e.g., images, sounds) is to use analog sensors. The present visual surveillance network systems are primarily designed with analog technologies. Compared to a digital signal, the major disadvantage of an analog signal is the noise control. As an analog signal is copied and re-copied, or transmitted over long distances, it can be easily distorted by numerous random variations.

However, the real competition between a digital signal and analog signal is not the signal level. Up at the system level, the numerous advantages of a digital signal start to emerge. Digital systems allow for programmable integrated noise correction and automatic image processing. Digital data may be compressed and encrypted for higher security. Digital technologies allow for the use of wireless integration, which offers more cost-effective installation compared to long cable runs in analog systems. Wireless integration also allows easy distribution in most areas where cabling may be difficult.

Table I-1 shows the advantages of a digital signal system over an analog signal system. As it can be seen, a digital signal system has far more advantages than an analog signal system in surveillance systems.

Table I-1: Advantages of Digital Signal System

AREA	ANALOG SYSTEM	DIGITAL SYSTEM
Noise Control	weak	strong
	Digital signal uses only “0” and “1” to transfer data.	
Security	unsecure	secure
	With digital data encryption, a digital signal is more secure.	
Efficiency	redundant	efficient
	With digital data compression, digital data can be more efficient.	
Transfer	dedicated channel	digital data network
	Digital data network can be wireless or wired strategy network.	
Data Analysis	human only	computer intelligent aid
	Digital data can be analyzed by digital processing software.	
Storage	slow real-time redundancy	fast and efficient
	Storage strategy ensures digital data stored efficiently.	

As is the case with many other highly technical programs, surveillance systems were first introduced for military purposes. It was more than twenty years before they began to be used for civilian purposes. Today, fifty years later, the market continues to grow but only in civilian and commercial applications. Real-time surveillance system has never been used for residential purposes. The reason is simple: the need for a human guard. In an analog surveillance system, there is no way to analyze the video data without a human guard. If we can develop a digital surveillance system, we can monitor the video and analyze it using various digital image processing algorithms. Thus, we can finally change the entire market for surveillance systems. That is the true reason for us to go digital.

1.2 Research Purpose and Difficulties

The aim of this research is to develop a hardware/software solution for a digital surveillance network. Today's analog surveillance networks are huge and sophisticated. They have numerous disadvantages compared to digital surveillance networks but are still the best solution because there are many difficulties associated with constructing a digital surveillance network:

- a digital data file's format is huge, making it hard to store and transfer
- digital video compression and decompression can result in a large amount of computation overhead
- Real-Time digital video analysis is hard to achieve

1.3 Significance of this Research

The system that we are proposing is an entire digital system, which means that the video we are capturing is digital, the compression technology is digital, the analysis and recognition process is digital, and the data to be stored will also be digital. The entire video surveillance system would be changed based on this change in the data format. The most significant improvement accompanying the digital data format, in comparison to analog, is the ability to analyze, compare, and finally extract information from streaming video by applying numerous types of digital algorithms to optimize it. This will increase the reliability of a large company's surveillance system and in the mean time, and make small (residential) surveillance systems possible.

Once we have the ability to analyze the video on-chip (on-camera), a "central control room" will no longer be needed. Theoretically, with these "intelligent" digital cameras, the whole visual surveillance network will become a decentralized video analysis system. Such a system could record video, analyze it, store it, and determine whether an alarm should be triggered. A remote distributed analysis and data storage system could significantly save precious system bandwidth, which is one of the known bottlenecks in hardware system development today.

1.4 Structure of Dissertation

This dissertation is structured in the following manner. Chapter 2 introduces the background information for this dissertation. An overview of the video data format, H.264 video coding standard, directional discrete cosine transform theory, and several surveillance-related image processing techniques will be introduced. Chapter 3 explains the major design details of the proposed hardware platform of the digital surveillance system architecture, from the very basic H.264 encoder core change to the video System-on-a-Chip (SoC) design, and finally, the whole architecture design for the surveillance system. Several different design candidates will be discussed. Chapter 4 demonstrates the software platform. Numerous implementations of digital video analysis algorithms will be illustrated. Chapter 5 describes the experimental platform and four sub-system designs: the Directional Discrete Cosine Transform system design, Vector Bank system design, Motion Detection system design, and Object Recognition system design. Finally, Chapter 6 concludes this dissertation and summarizes further plans and suggestions for this research.

II BACKGROUND

2.1 H.264 Video Coding Standard

2.1.1 Digital Video Coding Standard H.264

Electrical engineering technologies have developed rapidly over the last 50 years. High speed internet connections are commonplace, and the storage capacities of today's devices are huge. However, human beings have unfulfilled desires in technology. We want to watch movies at home over our internet connection. We want to store all of our movies on a single disk. Yet, at present, a single Digital Versatile Disk (DVD) can store only a few seconds of raw video at television-quality resolution and frame rate [2]. This is why video compression standards are popular today.

In 1995, the Moving Picture Experts Group (MPEG) developed the MPEG-2 video compression coding standard [3], [4], [5]. This famous standard has been widely used for over 15 years. It is the most popular standard for digital televisions (TVs), TV broadcasting, and movie DVDs.

With the development of new technology, the Standard Definition (SD) TV is being replaced by the new High Definition (HD) TV [6]. Typically, the 1080 progressively scanned lines (1080p) used for an HD video contain more than 6 times the pixels found in a 480p SD video. On the other hand, today's computer software/hardware is far more sophisticated than it was back in 1995. Obviously, the 15-year-old MPEG-2 standard is not efficient enough for today’s technology. This is why we have a new video coding standard. It is called H.264/MPEG-4 Part 10 AVC or just H.264 [7].

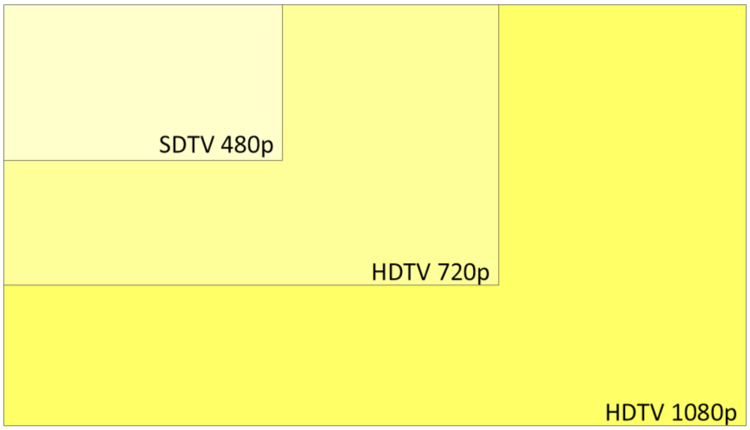


Figure II-1 Frame sizes of SDTV and HDTV

H.264 was developed by the Video Coding Experts Group (VCEG) and the Moving Picture Experts Group (MPEG). It has been proved that H.264 is at least two times more efficient than its predecessor–MPEG-2 and H.263 [8], [9]. This means H.264 needs only half the bandwidth required of MPEG-2 to transfer the same video, especially for HDTV and HD movies.

2.1.2 H.264 Coding and Decoding Algorithm

Video compression algorithms eliminate video redundancy to compress video. Video signals have two significant types of redundancy: time-domain redundancy and spatial domain redundancy. Some video compression algorithms (e.g., H.264) also apply arithmetic redundancy elimination to the coded data.

Figure II-2 shows the H.264 coder/decoder (codec) [2]. Generally, H.264 compresses video in Macro Blocks (MBs), rather than in frames. A Macro Block is a 16

× 16 pixel block. The H.264 encoder first splits the current video frame into numerous 16 × 16 Macro Blocks (MBs) and then processes them one by one.

A very important component in the Time Redundancy Removing Phase of H.264 is the Motion Estimation (ME) module [10], [11]. First, MBs from the current frame and several previous frames will be sent to the ME. After an enormous amount of computation and analysis, the ME will find the closest MB within the previous frames to represent the current MB, along with its relative position (Motion Vector) and absolute difference (Motion Residue). Then this information is sent to the next module, Motion Compensation (MC) [12], [13].

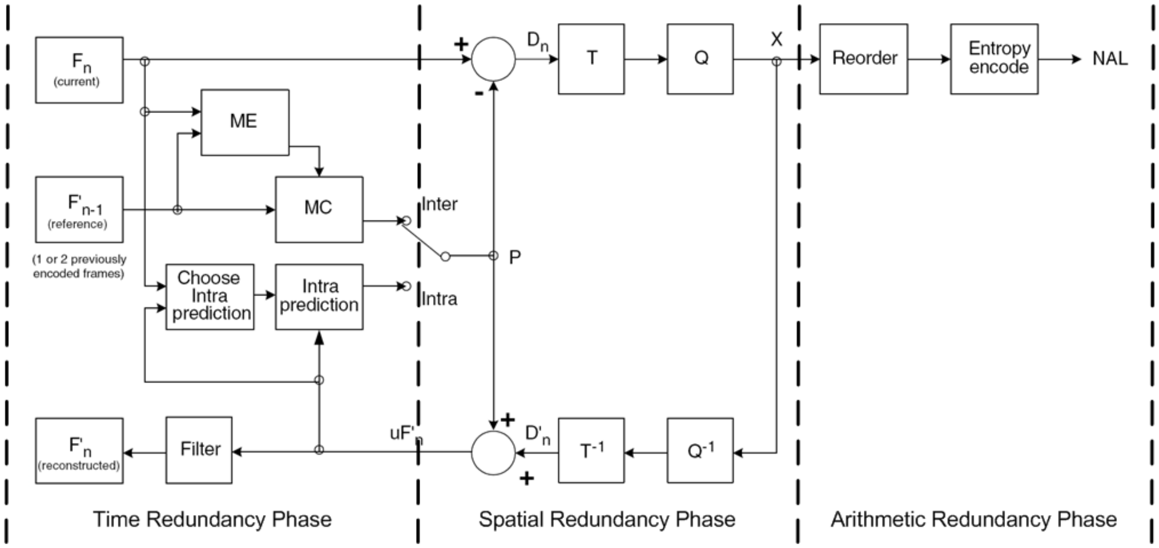


Figure II-2 H.264 Codec algorithm in phases

The Spatial Redundancy Removing Phase of the H.264 codec mainly uses the Discrete Cosine Transform (DCT) to separate the high and low frequency parameters of the residue of an MB. Most of the powers of these frequency parameters come from the low frequency domains. By applying a quantization map to the DCT results, the high frequency components are eliminated and the low ones are retained. Finally, H.264 uses arithmetic coding (e.g., Huffman coding) to eliminate arithmetic redundancy [14].

2.2 H.264 Based SoC Architecture

According to the instruction profiling of the HDTV1024P (High Definition TV with resolution of 2048 × 1024, 30 fps) specification [15], the H.264/AVC decoding process requires 83 Giga-Instructions per Second (GIPS) of computation and memory access rates of 70 Giga-Bytes per Second (GBPS). In the H.264/AVC encoder, up to 3,600 GIPS and 5,570 GBPS are required according to the HDTV720P (1280 × 720, 30 fps) specification .

Apparently, although many excellent H.264 integer motion estimation schemes have been proposed [16], [17] and [18], there is still a large need for the development of software to match the efficiency. A software approach is always the better solution (in consideration of the cost), in particular when ideally there is no limitation on computing time. However, realistically, applications exist with time constraints such as the need for real-time processing. This is true in our case because a visual surveillance sensor network needs to respond in real time, which can only be achieved based on a hardware solution.

Figure II-3 is a sample demonstration of an H.264 based System-on-a-Chip (SoC) architecture [19], [20], [21]. It contains one CPU as the management module of the entire system, one arbiter as the management module of the data/control bus, one Direct Memory Access Controller (DMAC), one main memory, one H.264 codec core, and several IO peripheral device interfaces such as a sensor interface and USB/Ethernet/Wi-Fi interface. The data flow is described below:

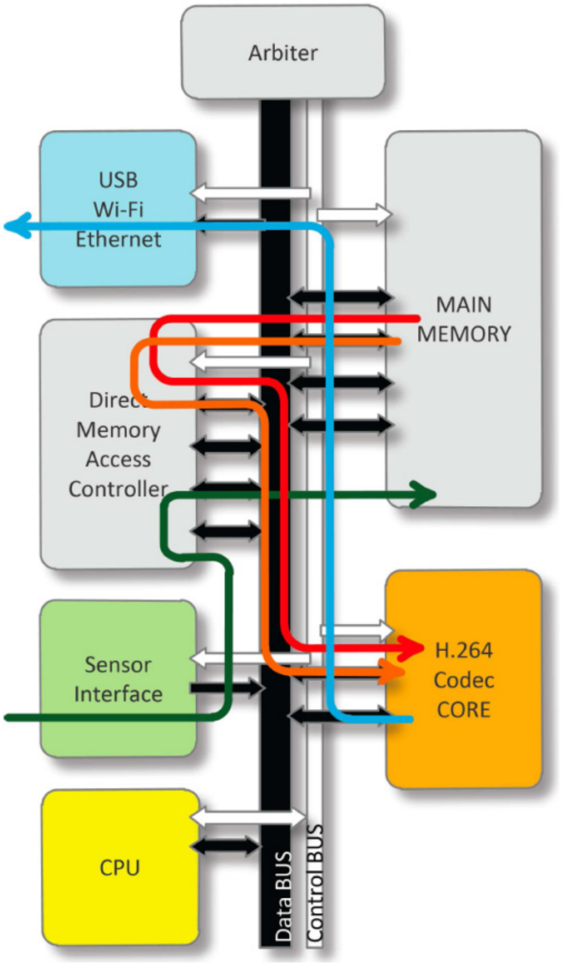


Figure II-3 General H.264 based System-on-a-Chip (SoC) design

- Green line: Video sensor captures video RAW stream data and sends it through the Sensor Interface module, data bus, and finally into the main memory through DMAC.
- Red and orange lines: These two lines represent the “current frame” and “reference frame” data, respectively. They will go through the DMAC again to reach the H.264 codec core.

- Blue line: Compressed data gets released from the H.264 and travels along the bus. It may stay in the memory for a while, but it will finally get out of the system through the USB/Wi-Fi/Ethernet system output interface.

A typical industry model generally consists of a Power Management Unit (PMU) and possibly several more I/O interfaces such as an LCD output interface, keyboard input interface, and so on. However, for research purposes only, we used this simplified SoC model to build our system.

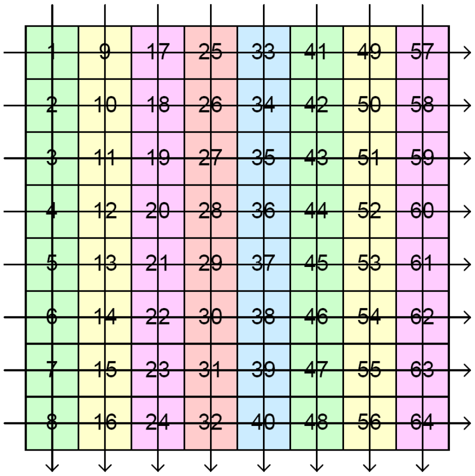
2.3 Direction Discrete Cosine Transform

Many image and video coding algorithms have been developed over the past 30 years, including sub-band/wavelet coding [22], [23], transform coding [24], vector quantization [25], and predictive coding [26], [27]. Among these coding technologies, the block-based transform approach has been recognized as the most successful, both for videos and images.

The Directional Discrete Cosine Transform (DDCT) [28] is a modified 2-Dimensional (2D) Discrete Cosine Transform (DCT) [29], [30], [31] technique used to optimize the block-based transform platform.

2.3.1 2-Dimensional Discrete Cosine Transform

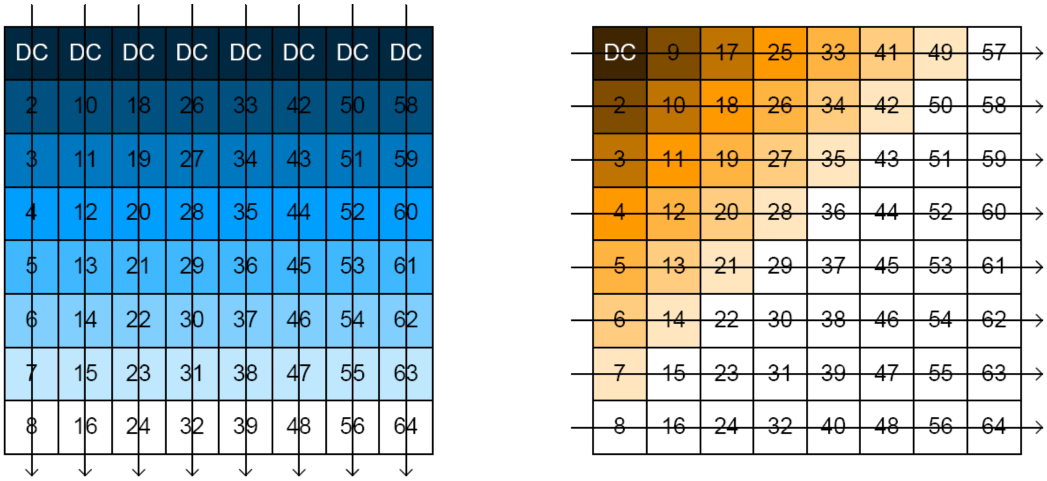
2D DCT is widely used in image compression algorithms (e.g., JPEG). As shown in Figure II-4, it is an 8 × 8 pixel image block. 2D DCT actually runs DCT 2 times, horizontally and vertically. By doing this, it separates the high frequency components from the low ones. The most important parameters come to the left top of the matrix.



1	9	17	25	33	41	49	57
2	10	18	26	34	42	50	58
3	11	19	27	35	43	51	59
4	12	20	28	36	44	52	60
5	13	21	29	37	45	53	61
6	14	22	30	38	46	54	62
7	15	23	31	39	47	55	63
8	16	24	32	40	48	56	64

Figure II-4 Typical 8 × 8 2D DCT Transform

Figure II-5 shows the split step of 2D DCT. It applies the DCT to all of the columns and then to the rows. The frequency components are separated: the darker ones represent lower frequencies and the lighter ones represent higher frequencies.



DC	DC	DC	DC	DC	DC	DC	DC
2	10	18	26	33	42	50	58
3	11	19	27	34	43	51	59
4	12	20	28	35	44	52	60
5	13	21	29	36	45	53	61
6	14	22	30	37	46	54	62
7	15	23	31	38	47	55	63
8	16	24	32	39	48	56	64

DC	9	17	25	33	41	49	57
2	10	18	26	34	42	50	58
3	11	19	27	35	43	51	59
4	12	20	28	36	44	52	60
5	13	21	29	37	45	53	61
6	14	22	30	38	46	54	62
7	15	23	31	39	47	55	63
8	16	24	32	40	48	56	64

Figure II-5 2D DCT transform

2.3.2 Directional Discrete Cosine Transform

The DDCT is a block-based transform algorithm that is based on 2D DCT image processing. In general, by applying the 2D DCT to an image block, a frequency distribution map can be generated that contains both the low frequency components to be kept and the high frequency components to be dumped. However, only the horizontal/vertical direction has been scanned. By adding several different modes, DDCT can process the original 8 × 8 pixel block in several different directions.

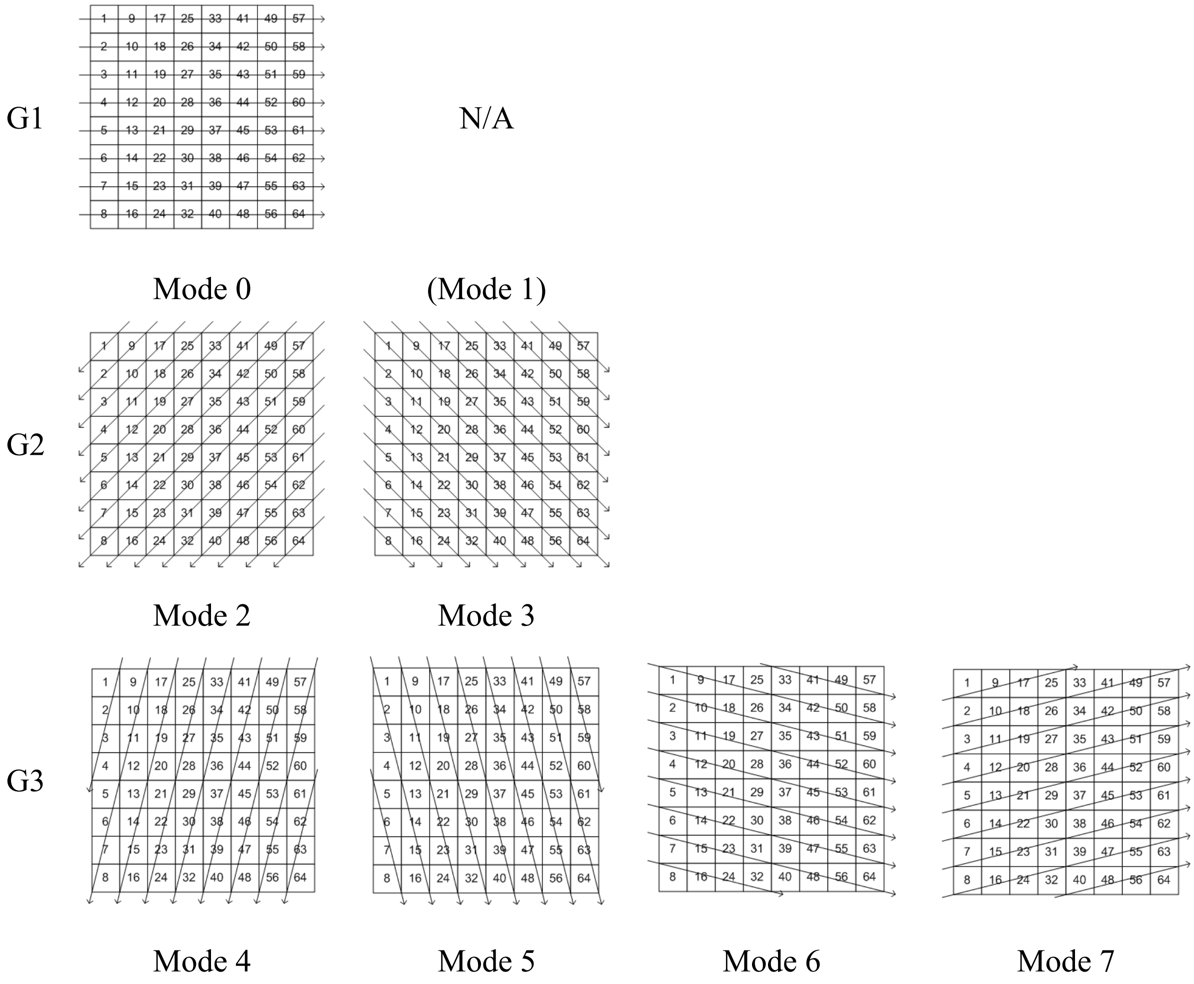


Figure II-6 DDCT modes in groups

As shown in Figure II-6, there are eight modes in DDCT, numbered 0 to 7, and these can be divided into 3 groups.

Just like the conventional 2D DCT, each mode in DDCT will run DCT twice, perpendicularly. To better illustrate this, Figure II-6 only shows the first DCT direction.

The first group contains mode 0 and mode 1. However, mode 0 and mode 1 are the same (the only difference is that mode 0 runs DCT horizontally first, while mode 1 runs DCT vertically first.). Thus, in DDCT, only mode 0 counts, and it is identical to the conventional 2D DCT. The second group contains mode 2 and mode 3. These are two diagonal modes. The third group is made up of modes 4, 5, 6, and 7, which are four symmetric semi-diagonal modes.

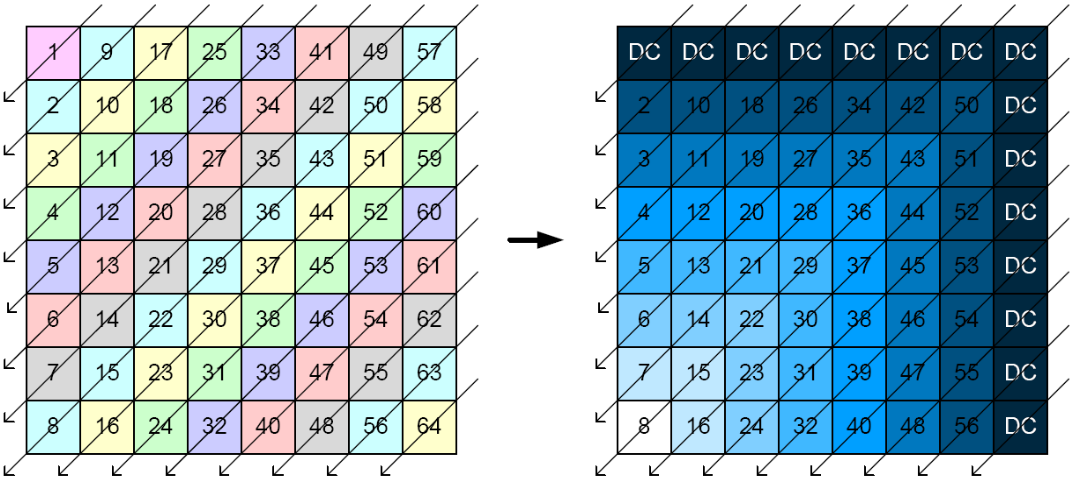


Figure II-7 First DCT of DDCT mode 2

Figure II-7 shows how the first DCT transform is performed by mode 2 of DDCT. As shown, the first DCT direction is not horizontal or vertical, but diagonal. In this 8 × 8 image pixel block, 15 Direct Current (DC) values will be generated, and we need to “reorganize” the pixel block first before performing another DCT on it.

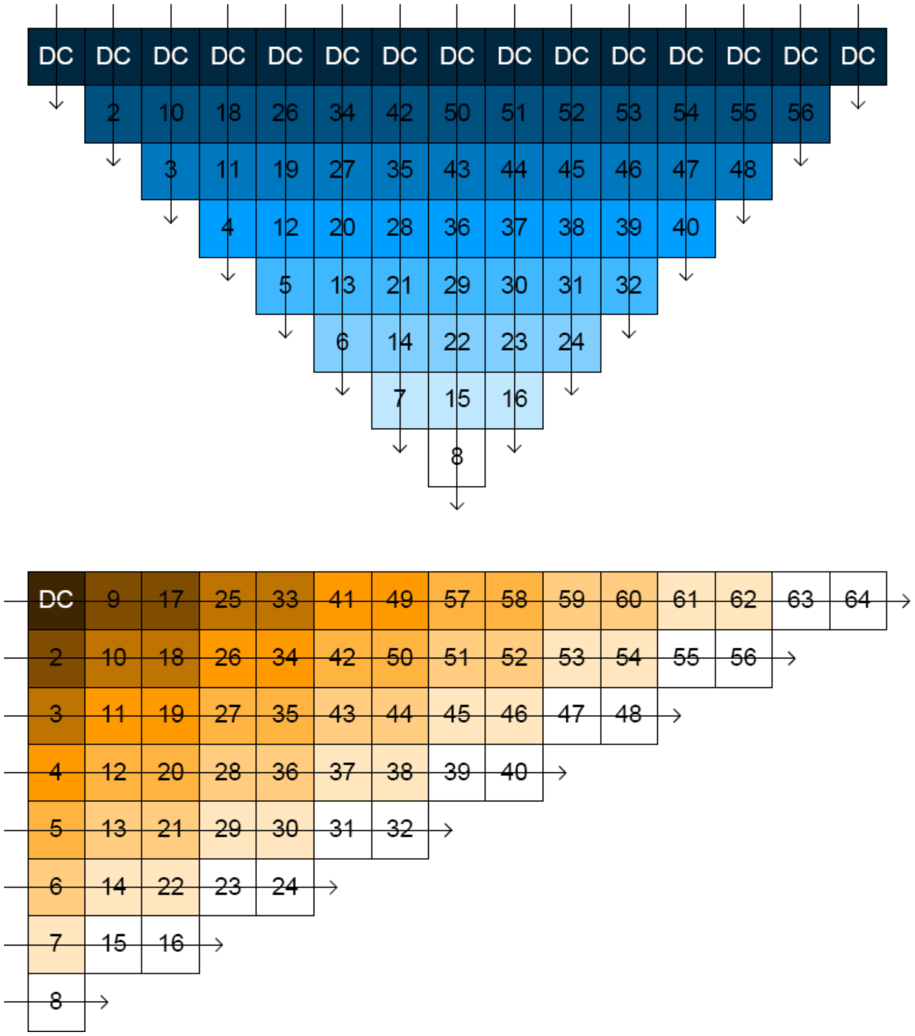


Figure II-8 Reformatted pixel block and second DCT of DDCT mode 2

As shown in Figure II-8, the way to reorganize the results of the first DCT of DDCT mode 2 is to group the corresponding frequency components together along the direction of the DCT performed. It then becomes a right triangle. Then, another DCT will be performed after the reorganization, and it becomes a right-angled triangle. Finally, we get the result of 2D DCT: a 2D Direct Current (DC) value and distributed frequency parameter map.

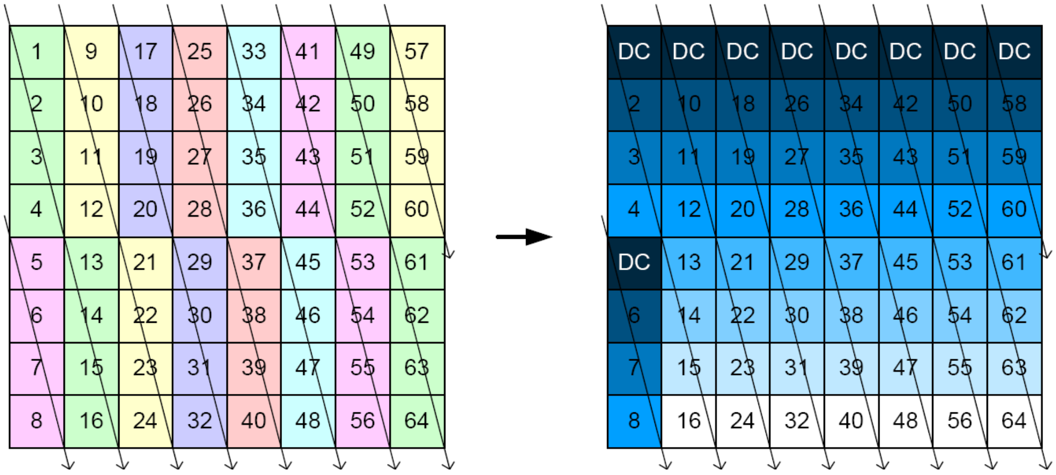


Figure II-9 First DCT of DDCT mode 5

Figure II-9 and Figure II-10 show another example from DDCT mode 5. Apparently, different methods are required to perform DCT transforms for different modes.

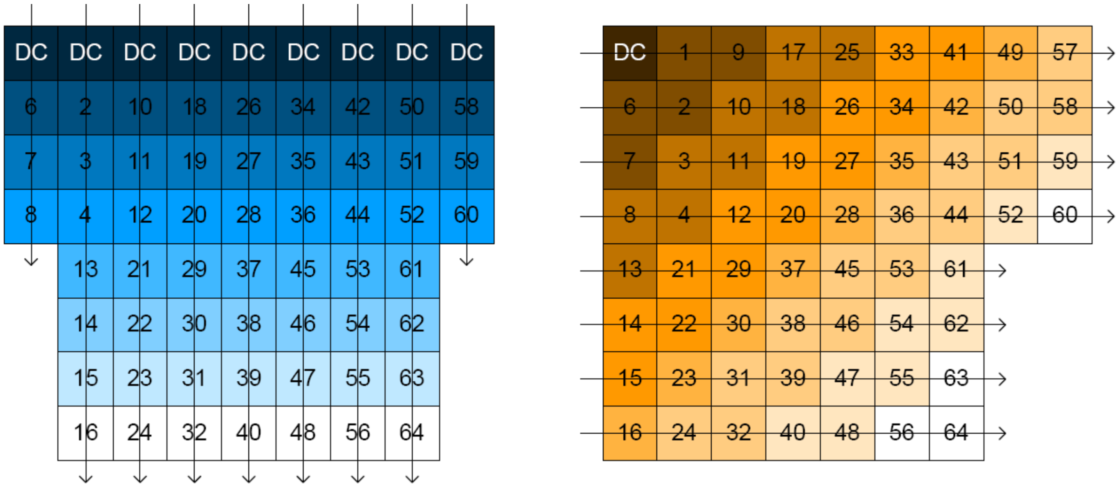


Figure II-10 Reformatted pixel block and second DCT of DDCT mode 5

Just like the original 2D DCT, the previous DDCT figures show that all of the corresponding components will be processed in the same column, which indicates that all of the DC values will be processed together to obtain the DC value of the whole map, 2nd derivative frequency values, 3rd derivative frequency values, and so on. Yet, because this

DC value is the average of the results for different numbers of pixels, as shown in Figure II-8 and Figure II-10, we could get a so-called mean weighting defect [24] if we only perform the DCT without any appropriate corrections.

There are generally two ways to deal with the mean weighting defect [21]. In our research, we use the simplest method to modify the weighting factors. If we define α as the original $N \times N$ block and A as the conventional 2D DCT result, the conventional 2D DCT's coefficient block can be expressed as:

$$A = \left[A_{u,v}\right]_{N\times N} = C_{N\times N} \cdot a \cdot C_{N\times N}^{T} \qquad \text{II (1)}$$

where

$$C_{N\times N} = \left[C_{i,j}\right]_{N\times N}, c_{i,j} = \alpha_i Cos\left(\frac{(2j+1)}{2N} i\pi\right)$$
$$\alpha_i = \begin{cases} \sqrt{1/N}\,, i = 0 \\ \sqrt{2/N}\,, i \neq 0 \end{cases} \qquad \text{II (2)}$$

By changing the weighting factor, a_i, we can use

$$\hat{\alpha}_i = \begin{cases} 1/N_k\,, & i = 0 \\ \sqrt{2}/N_k\,, & i \neq 0 \end{cases} \quad k = 0,1,\ldots,2N-2 \qquad \text{II (3)}$$

to correct the DC value of mode 2.

For the rest of the DDCT modes, we repeat the process, but in different directions, as shown in mode 5 in Figure II-10.

After DDCT obtains all of the result candidates from the 7 internal modes, the selection is made based on the "zero" counts. Generally, an image with a higher frequency of zeros after quantization has lower correlation from pixel to pixel in this direction, which will theoretically lead to a more efficient compression ratio.

2.4 Background Elimination Algorithm

Foreground detection algorithms (e.g., motion detection, object recognition, etc.) are based on an analysis of continuous object colors or patterns [32], [33]. These patterns can easily be recognized if there are no background distractions. For still images, a background subtraction algorithm [34], [35] can be very hard to use. Yet, a reference background image makes it much easier.

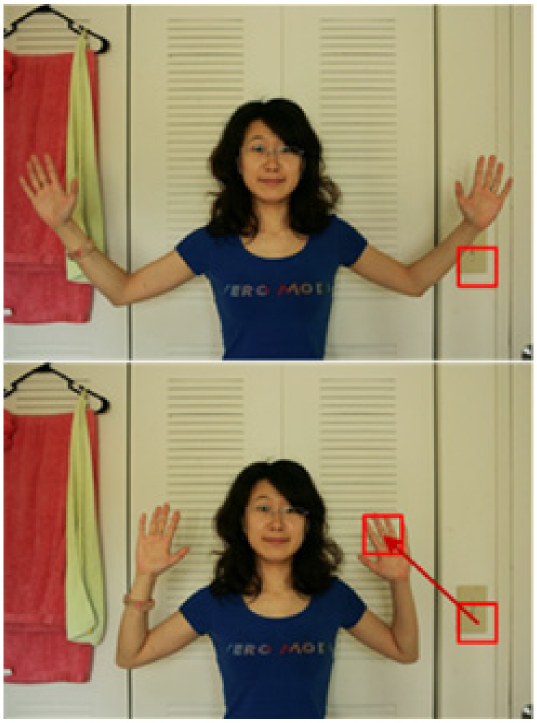


Figure II-11 Foreground recognition failure without background elimination

Figure II-11 shows that in a video sequence, because of fast movement and rotation, the color/pattern correlations of a moving object between the current frame and previous frame(s) begin to decrease. If the background is not eliminated, this could lead to an object/motion detection failure.

In contrast to general purpose cameras, surveillance cameras are usually positioned at static positions. Therefore, if we simply add a background reference frame to the camera, it should be easy to remove the background from the current frame. Thus, the object/motion detection result will be more reliable. Generally, the lighting condition is the key to background elimination. For a stable lighting condition (e.g., laboratory, classroom, etc.), background elimination is relatively easy. However, in an outdoor parking lot, the lighting condition is constantly changing over time. Background elimination may therefore require the assistance of several algorithms such as exposure compensation to normalize the light, as well as a luminance ripple filter to even out an unstable lighting condition for better results.

2.5 Edge Detection Operator

Motion object detection and tracking techniques have been studied for years in relation to numerous areas of interest such as authentication systems, machine-human interfaces, and, most commonly, video surveillance. There are a number of different techniques used in the motion tracking fields today [36]-[39]. Most of these utilize the frame differences to detect object movement.

Edge Detection [40]-[42] is another very popular and important technology in Computer Vision. It is well developed and widely used in Digital Image Processing (DIP). Edge Detection also utilizes many individual algorithms, which can be categorized as first-derivative operators and second-derivative operators. First-derivative operators such as Roberts, Prewitt, and Sobel can detect the edge of an image in one dimension (either horizontal or vertical), while second-derivative operators such as the Laplacian operator can detect it in both dimensions at the same time [30]. In this paper, we will use the two most famous, which are the “Sobel” and “Laplacian of Gaussian” (LoG) operators [43].

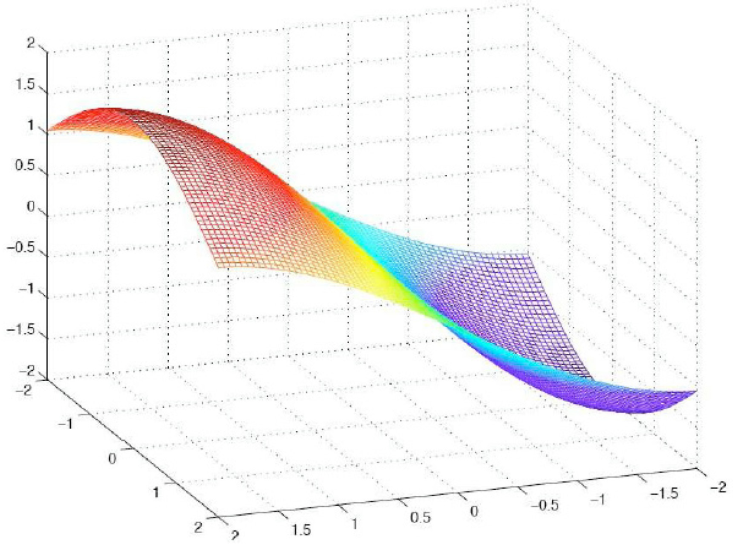


Figure II-12 Sobel operator 3D plot in MatLab

The Sobel operator is a first-derivative edge detection operator. It is simple and easy to realize in most cases. In order to detect a 2D image edge, we need to run Sobel twice, in different directions. Figure II-12 shows a typical Sobel bi-directional kernel:

$$G_y = \begin{bmatrix} +1 & +2 & +1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix} \qquad \text{II (4)}$$

and

$$G_x = \begin{bmatrix} +1 & 0 & -1 \\ +2 & 0 & -2 \\ +1 & 0 & -1 \end{bmatrix} \tag{II (5)}$$

G_x and G_y can be combined to get the absolute magnitude of the gradient:

$$|G| = \sqrt{G_x^2 + G_y^2} \tag{II (6)}$$

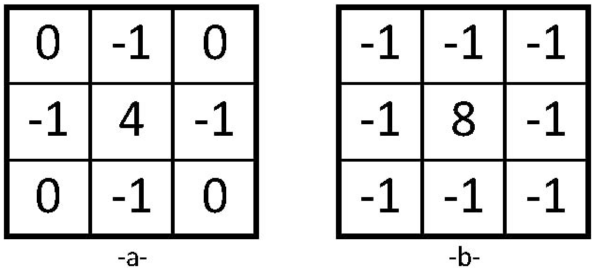
For fast computation, the magnitude can also be approximated as

$$|G| = G_x + G_y \tag{II (7)}$$

Thus, the approximate kernel for the 2D Sobel detection operator is

$$\begin{aligned} |G| = {} & |(z_1 + 2 \times z_2 + z_3) - (z_7 + 2 \times z_8 + z_9)| \\ & +|(z_3 + 2 \times z_6 + z_9) - (z_1 + 2 \times z_4 + z_7)| \end{aligned} \tag{II (8)}$$

In contrast to the Sobel operator, the Laplacian of Gaussian (LoG) operator is a second-derivative edge operator.



0	-1	0
-1	4	-1
0	-1	0

-a-

-1	-1	-1
-1	8	-1
-1	-1	-1

-b-

Figure II-13 Laplacian digital approximations

Figure II-13 shows two typical Laplacian operators. Its mathematical equation is expressed as

$$\nabla^2 f = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2} \tag{II (9)}$$

A typical Gaussian Kernel with width σ is

$$G_\sigma = \frac{1}{\sqrt{2\pi\sigma}} e^{-\frac{x^2+y^2}{2\sigma^2}} \qquad \text{II (10)}$$

Thus, the Laplacian of Gaussian will be

$$\nabla^2 G_\sigma = \frac{\partial^2 G_\sigma}{\partial x^2} + \frac{\partial^2 G_\sigma}{\partial y^2} \qquad \text{II (11)}$$

Variables x and y are equal in this equation. We determine the x part first:

$$\frac{\partial^2 G_\sigma}{\partial x^2} = \frac{x^2 - \sigma^2}{\sigma^4} e^{-\frac{x^2+y^2}{2\sigma^2}} \qquad \text{II (12)}$$

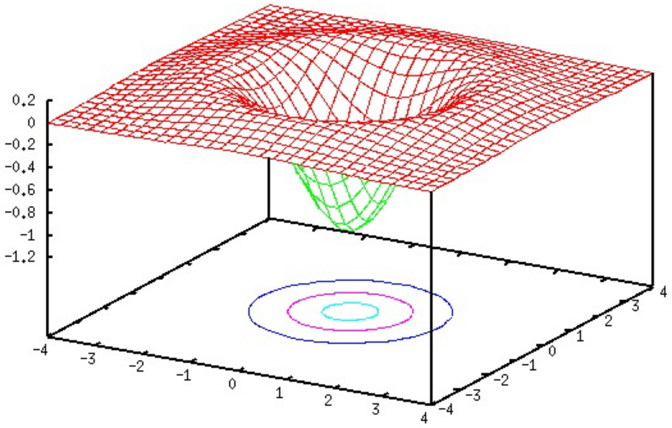


Figure II-14 LoG kernel 3D plot in MatLab

Let $x^2 + y^2 = r^2$, and put x and y back together in the equation:

$$\nabla^2 G_\sigma = \frac{x^2 + y^2 - 2\sigma^2}{\sigma^4} e^{-\frac{x^2+y^2}{2\sigma^2}} = \frac{r^2 - 2\sigma^2}{\sigma^4} e^{-\frac{r^2}{2\sigma^2}} \qquad \text{II (13)}$$

Thus, the typical 5 by 5 LoG convolution kernel is

$$\begin{bmatrix} 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 2 & 1 & 0 \\ 1 & 2 & -16 & 2 & 1 \\ 0 & 1 & 2 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 \end{bmatrix} \tag{II (14)}$$

2.6 Distance-to-Angle Signature Algorithm

The distance-to-angle signature is a 1D functional representation of a boundary. As Figure II-15 shows, the way to get the distance-to-angle signature of a figure is to plot the distance from its centroid to the boundary as a function of the angle [43].

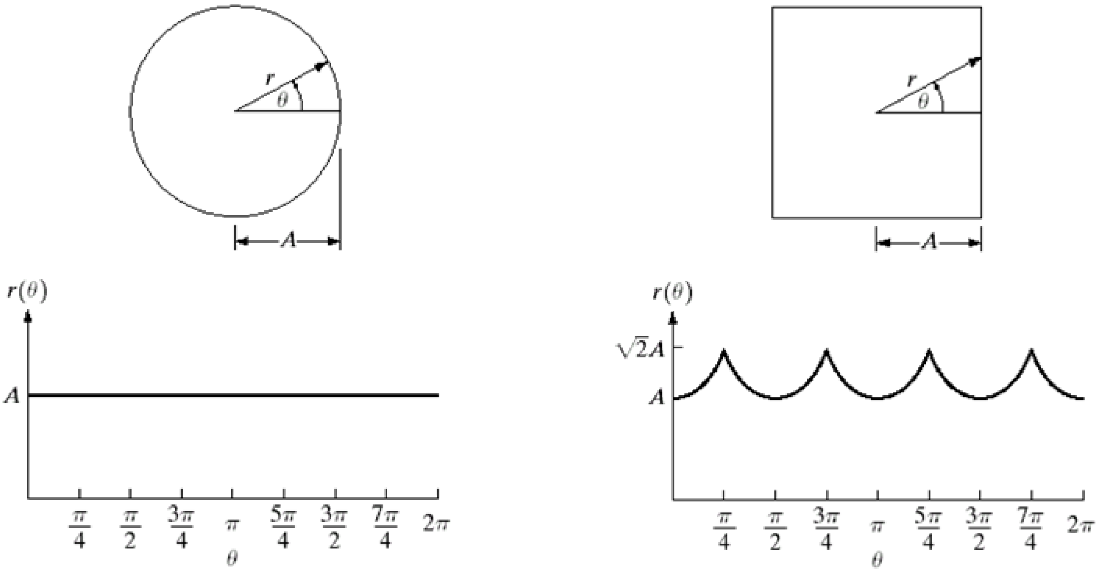


Figure II-15 Distance-to-Angle signature algorithm

The basic idea of the distance-to-angle signature is to transform the figure shape from the Cartesian coordinate system into a distance-angle coordinate system. Conventional DIP algorithms (e.g., block-based transform) are based on the Cartesian coordinate system, which is a good platform to analyze panning objects. However, if the objects are spinning or moving in/out, they become hard to analyze/recognize. One of the key features of the distance-to-angle signature is that no matter how the orientation or size of the object has changed, the only changes reflected in its distance-to-angle signature map would be the amplitude ratio or orientation point of the signature.

2.7 Artificial Neural Network System

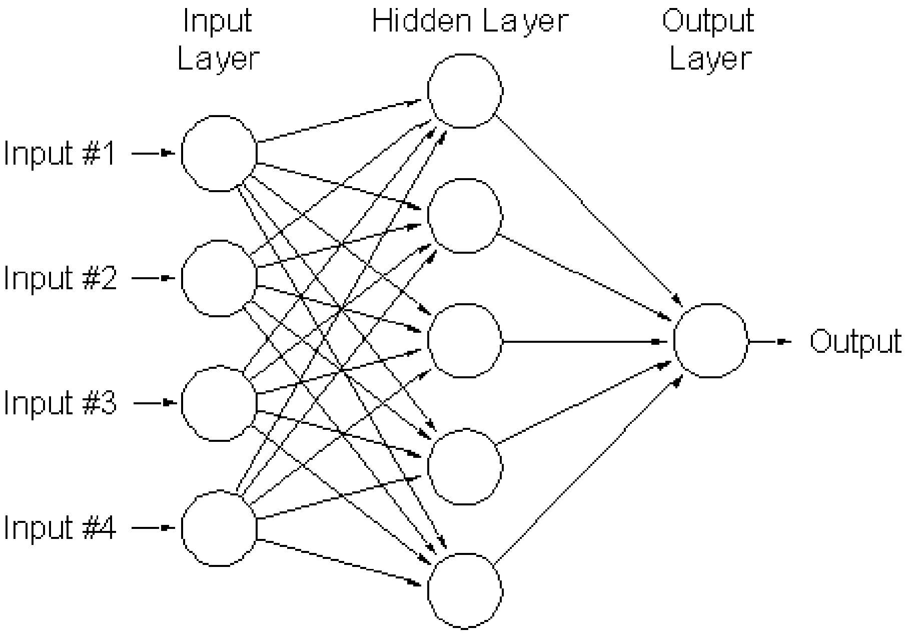


Figure II-16 Simple Artificial Neural Network (ANN) system

An Artificial Neural Network (ANN) system [44], [45], [46] is a computer algorithm model of the brain and nervous system. It is highly parallel and processes information much more like the brain than a serial computer. Basically, an ANN system

- has the ability to learn
- is based on very simple principles
- exhibits very complex behaviors

Figure II-16 shows a single-layer artificial neural network system. Data are presented to an input layer and passed onto a hidden layer by multiplying by a weight matrix. Finally, the data are passed onto the output layer. The information is distributed and processed in parallel. Figure II-17 shows the process for each of the neuron cells.

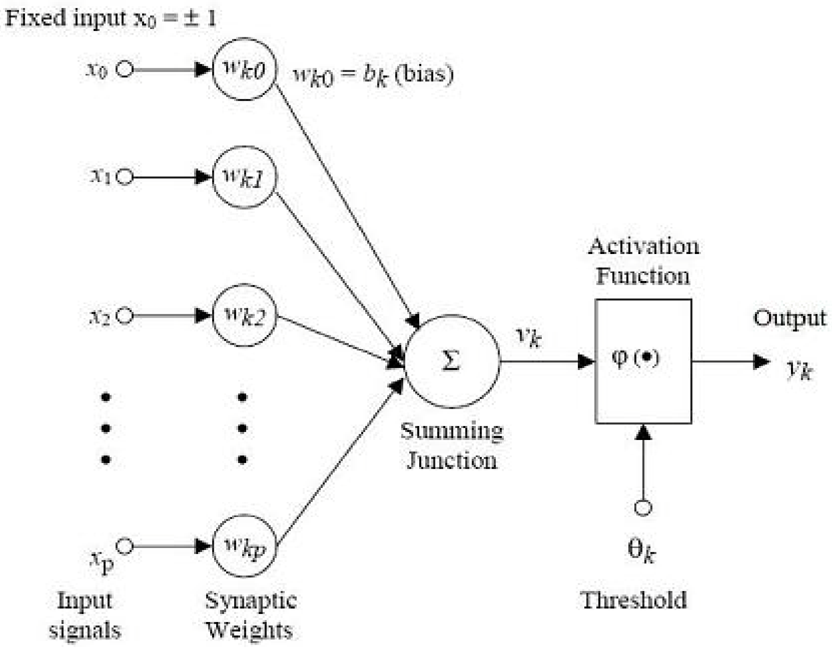


Figure II-17 Typical neural network process

Among the various types of ANNs, the most popular is Backpropagation (BP) [47], [48], [49]. BP is a common neural network learning algorithm. Its input signals propagate forward through the network, while error signals propagate backward. Weight adjustments are made to reduce error. A typical BP ANN should have the following features:

- it requires a training set (input/output pairs)
- it starts with small random weights
- the error is used to adjust the weights (supervised learning)

- the result is gradient descent on error landscape

In a real application, the input and training of a BP ANN could be very different. But the idea remains the same. We first need to find a series of features to represent the input. Then, we define the corresponding output values. We train the neural network with these inputs and outputs starting with some random weights. The training is counted in epochs (rounds). In each epoch, the random weights will be slightly corrected to become a better filter for the input-output combinations. At the end of the training, all of the inputs will be perfectly recognized and assigned to the outputs, and we can start to use the well-trained network to predict the results of unknown input data.

III SYSTEM HARDWARE PLATFORM DESIGN

3.1 Analog Surveillance Network System Architecture

At present, visual surveillance network systems are mainly designed using analog technology. Some surveillance systems have been partially digitized on the data storage end [50], [51]. Their storage format has been converted into digital files and stored on hard drives or optical discs. Because the signals that are transferred and processed remain analog, they are still analog systems. A practical structure for a visual surveillance network system is depicted in Figure III-1.

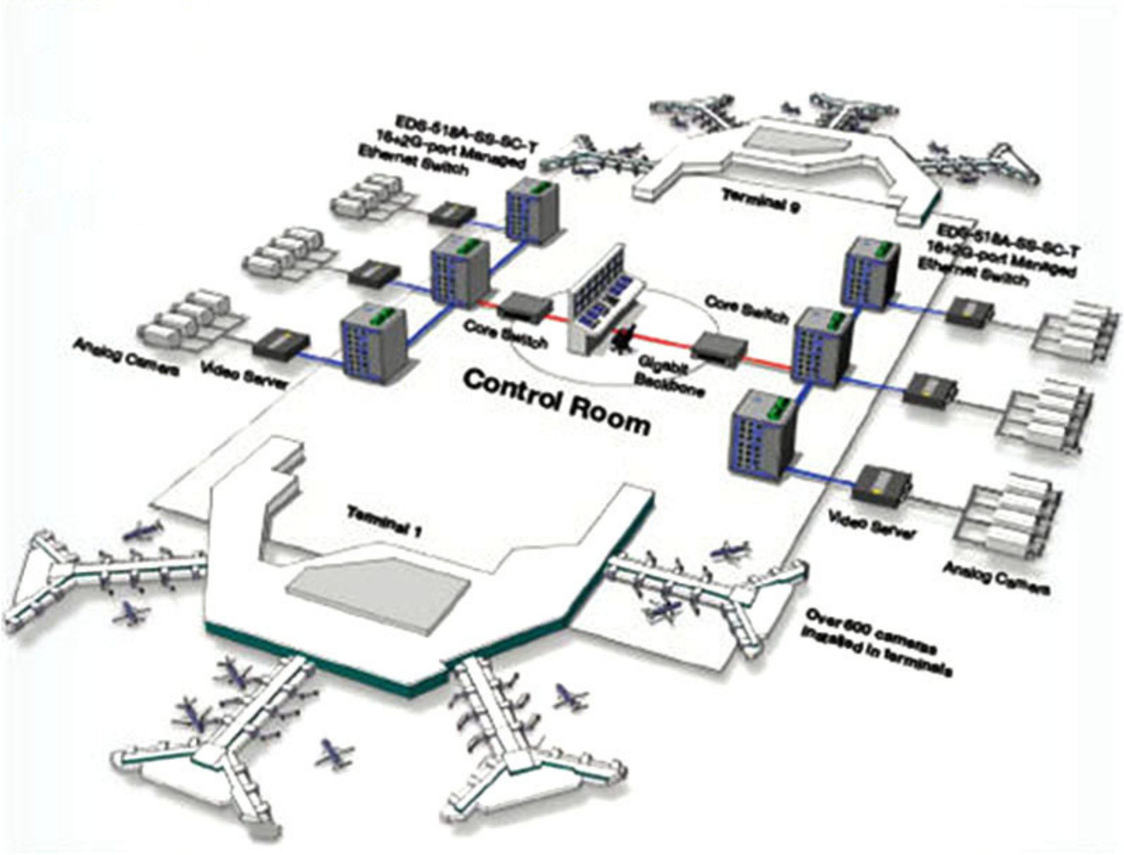


Figure III-1 Practical structure for visual surveillance network system

Essentially, in an analog surveillance system, multiple cameras with dedicated video cables are connected to a video link switch server. All of the video data or selected video data may then be transferred to a secure room (control room), which contains human guards to monitor the information. We can easily divide a surveillance system into four basic phases: the capture phase, transfer phase, analyze phase, and store/execute phase. Figure III-2 illustrates these phases in an analog surveillance system.

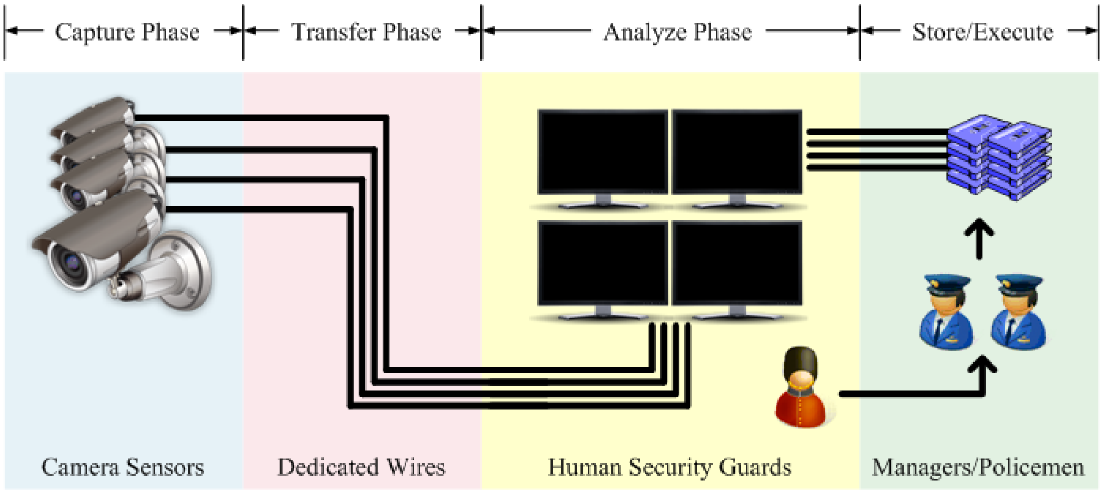


Figure III-2 Analog surveillance system phases

As discussed in Chapter I, because analog video signals are hard to process, an analog surveillance system needs human guards to "analyze" the incoming video information and make judgments about whether or not a potential threat exists.

3.2 Proposed Digital Surveillance System Phases

The core idea of this dissertation is to improve the surveillance network by changing it into a purely digital system. In order to achieve this target, our signal must be digitized from the very first phase of the system.

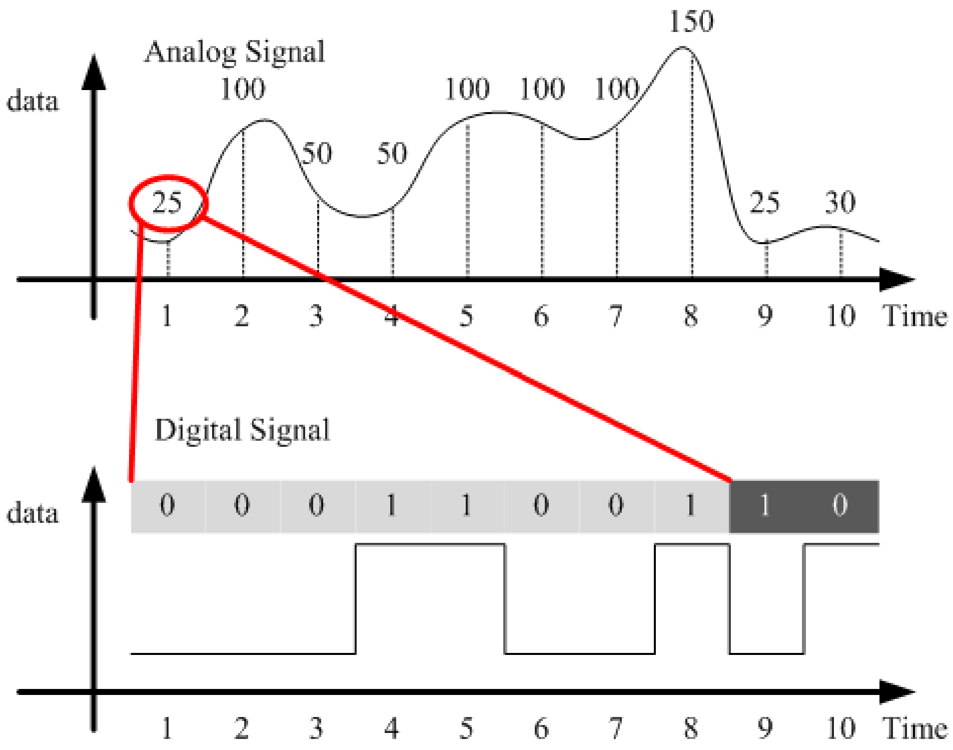


Figure III-3 Analog data transfer compared to digital data transfer

In a surveillance network system, over 99 percent of the data transferred is video color data. A typical unit of color data (R, G, or B) is a number from 0 to 255, which is an 8-bit binary number. As shown in Figure III-3, a digital system requires eight times the bandwidth of an analog system to transfer the same data file.

To reduce the bandwidth impact of a digital system, H.264 video coding is introduced. As discussed in Chapter 2.2, H.264 is an advanced video coding method that can reduce the video data dramatically (usually by 100 times or more). After using an H.264 encoding system, a typical 3.0 Gbps 1080p video signal is no more than 60 Mbps. However, it also requires huge computational resource support. Currently, no desktop machine is capable of encoding an HD H.264 movie in real time, which is why we use the H.264 codec SoC hardware platform to perform the encoding.

H.264 is a great solution for video data transfer and storage, but it also leads to some problems. After the data is transferred to the secure room, decoding all of the H.264 data into video data and analyzing it requires a large computational overhead. In the real world, even the fastest machine cannot decode a large number of H.264 videos simultaneously. If we have a hundred or even a thousand video cameras in our surveillance system, decoding-based analysis would be impossible. Apparently, we need to change the entire architecture of our digital surveillance network system.

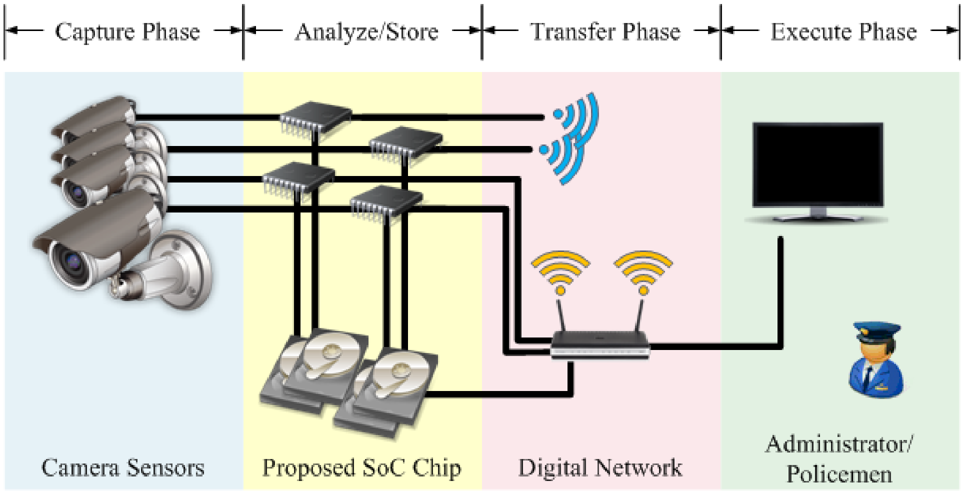


Figure III-4 Proposed digital surveillance system phases

Compared to Figure III-2, Figure III-4 illustrates a completely new architecture for a digital surveillance network system. By implementing a digital SoC chip in each video sensor, we can now encode data and analyze it simultaneously right after it is captured. The coded H.264 video data do not ever need to be transferred but can be stored locally at its camera. Through a private digital network, a security administrator can check either the real time streaming video data or the stored video data.

Each of the video cameras can capture, analyze, and store video data independently and is called a "surveillance node," as shown in Figure III-5.

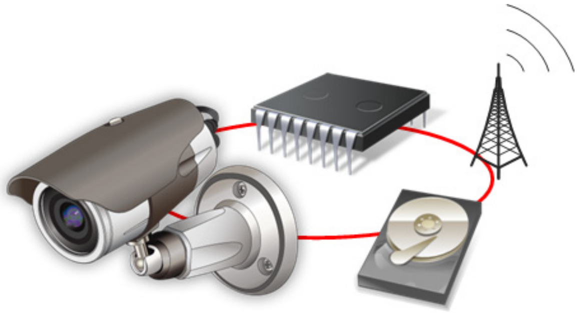
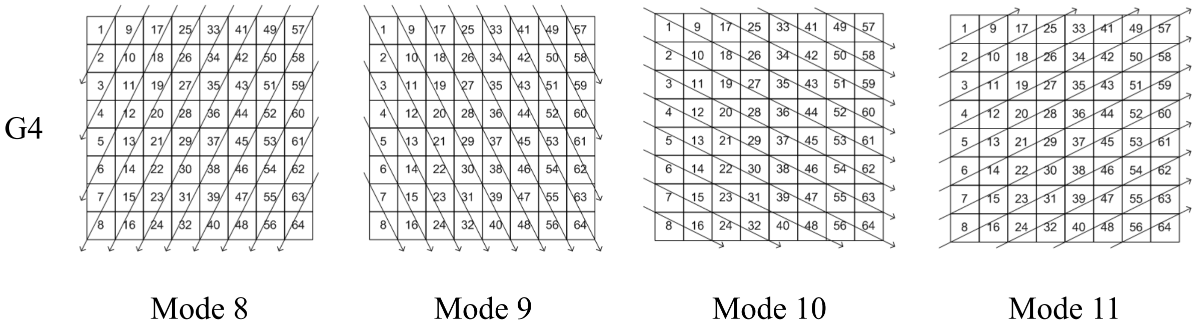


Figure III-5 Surveillance Node

3.3 Improved H.264 Core Architecture

3.3.1 Video optimized DDCT module implementation

As discussed in Chapter 2.3, conventional DDCT is optimized for image use, which is mostly based on 8 × 8 pixel block computation. However, the H.264 coding algorithm is based on a 16 × 16 pixel MB. The larger pixel block size brings more direction options. In order to maintain the efficiency of DDCT, we add four extra modes to the original DDCT.



Mode 8

1	9	17	25	33	41	49	57
2	10	18	26	34	42	50	58
3	11	19	27	35	43	51	59
4	12	20	28	36	44	52	60
5	13	21	29	37	45	53	61
6	14	22	30	38	46	54	62
7	15	23	31	39	47	55	63
8	16	24	32	40	48	56	64

Mode 9

1	9	17	25	33	41	49	57
2	10	18	26	34	42	50	58
3	11	19	27	35	43	51	59
4	12	20	28	36	44	52	60
5	13	21	29	37	45	53	61
6	14	22	30	38	46	54	62
7	15	23	31	39	47	55	63
8	16	24	32	40	48	56	64

Mode 10

1	9	17	25	33	41	49	57
2	10	18	26	34	42	50	58
3	11	19	27	35	43	51	59
4	12	20	28	36	44	52	60
5	13	21	29	37	45	53	61
6	14	22	30	38	46	54	62
7	15	23	31	39	47	55	63
8	16	24	32	40	48	56	64

Mode 11

1	9	17	25	33	41	49	57
2	10	18	26	34	42	50	58
3	11	19	27	35	43	51	59
4	12	20	28	36	44	52	60
5	13	21	29	37	45	53	61
6	14	22	30	38	46	54	62
7	15	23	31	39	47	55	63
8	16	24	32	40	48	56	64

Figure III-6 Four Extended Modes for Video Optimized DDCT

Figure III-6 shows the extra group of four modes created for the optimized 16 × 16 pixel video DDCT. Lab results show that these “semi-diagonal” modes would match another 20~25% of the area that other DDCT modes cannot match.

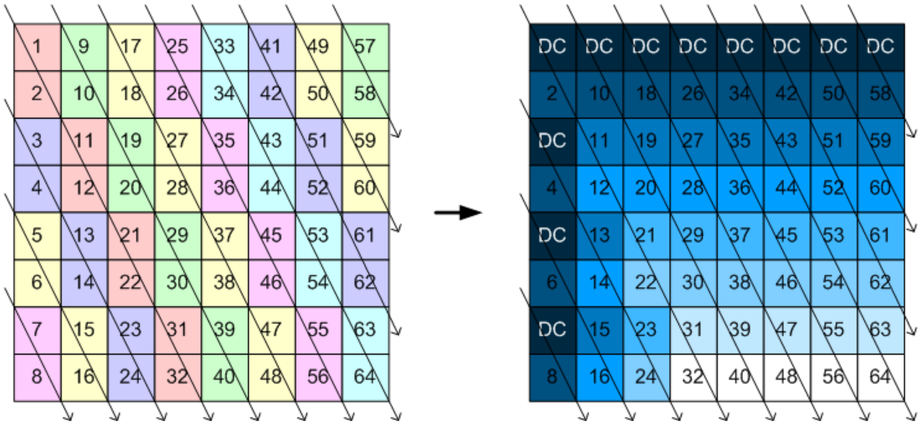


Figure III-7 First DCT of DDCT mode 9

Figure III-7 and Figure III-8 show two DCT processing examples of DDCT mode 9, which is one of the new modes we created. Thus, our new video optimized DDCT now contains a total of 12 modes.

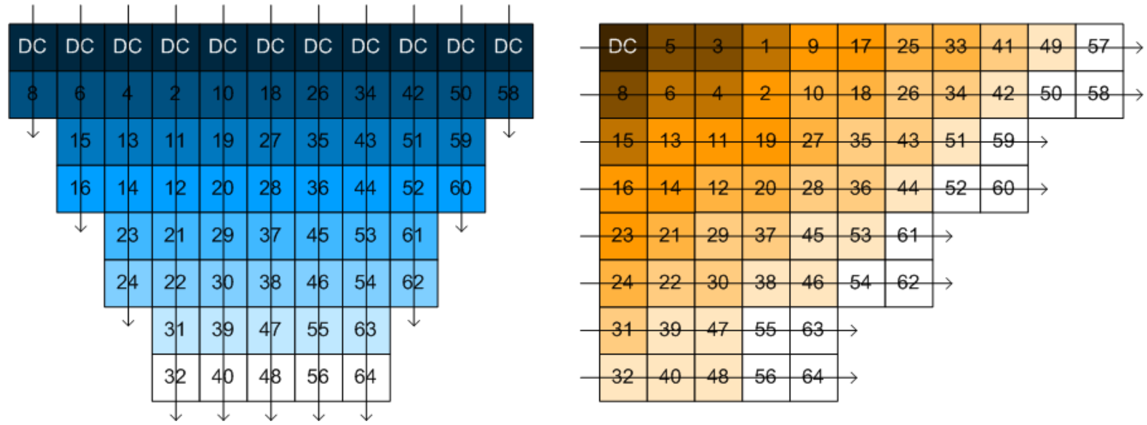


Figure III-8 Reformatted pixel block and second DCT of DDCT mode 9

3.3.2 Background elimination module implementation

For surveillance cameras, the background information is relatively static. Although it does not offer any information, it still occupies system bandwidth. Eliminating the background information from every single frame of the streaming video would dramatically save system bandwidth, and using the “object-only” frames would make video analysis (e.g., motion detection, object recognition, etc.) much easier.

On the other hand, the speed requirement for background elimination is high. If the input and output do not continuously match the frame speed of the system, the H.264 codec core cannot operate normally. In addition, the background information is sometimes needed when the administrator wants to replay the video. Therefore, we also need to add the background information back in during playback.

Generally, background elimination is based on an “object mask” generated by the differential image of the current frame and reference background frame. However, this subtraction cannot be performed directly because of various lighting issues. A general purpose surveillance camera can be placed anywhere, inside a government building or outside on a football field. Lighting issues involving rain, snow, or clouds can easily lead to the failure of the background elimination.

There are mainly two categories of lighting issues. Figure III-9 shows normalized luminance graphs of them. The luminance axis is normalized by the reference background so that all of the luminance values from the reference background maintain a value of “1.” Panel A shows a typical “exposure” situation. This situation occurs when the lighting condition for the background is changing (e.g., the sky gets cloudy, one of the lights goes off, etc.). Panel B shows a "ripple" situation. This situation occurs when the lighting condition for the background is very unstable spatially (e.g., rain or snow).

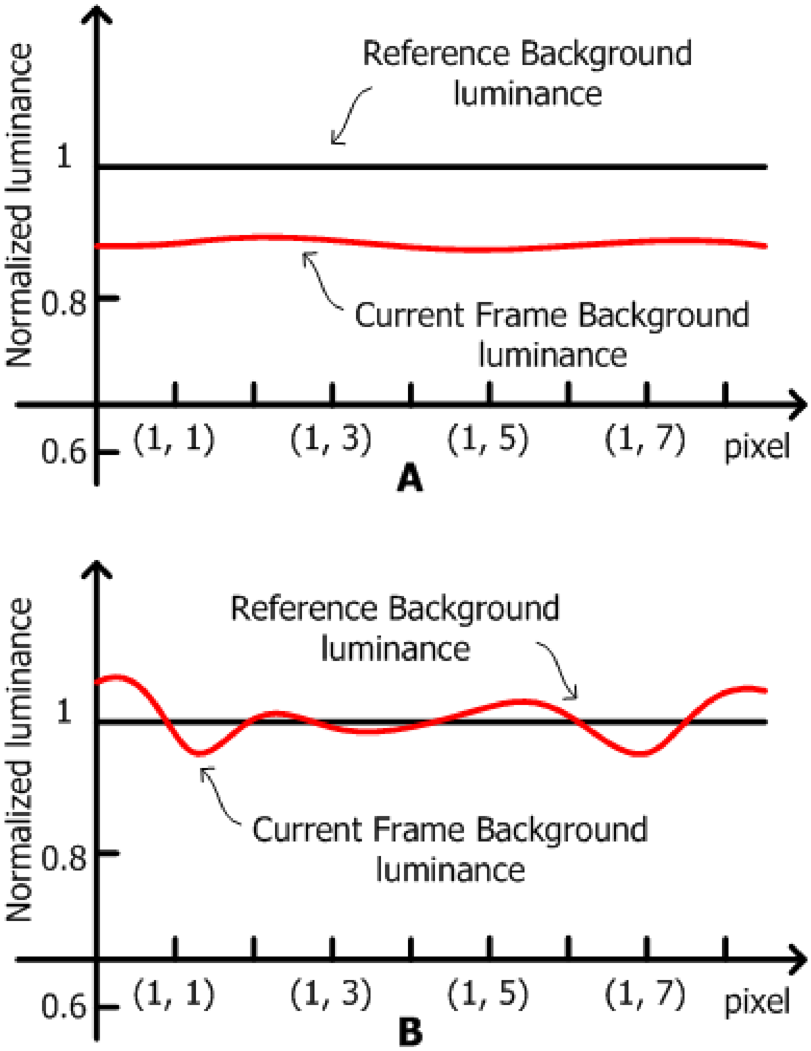


Figure III-9 Normalized luminance map of two categories of lighting issues

Figure III-10 shows an example of direct subtraction without solving the ripple lighting issue. Solving the lighting issue is a very important part of background elimination.

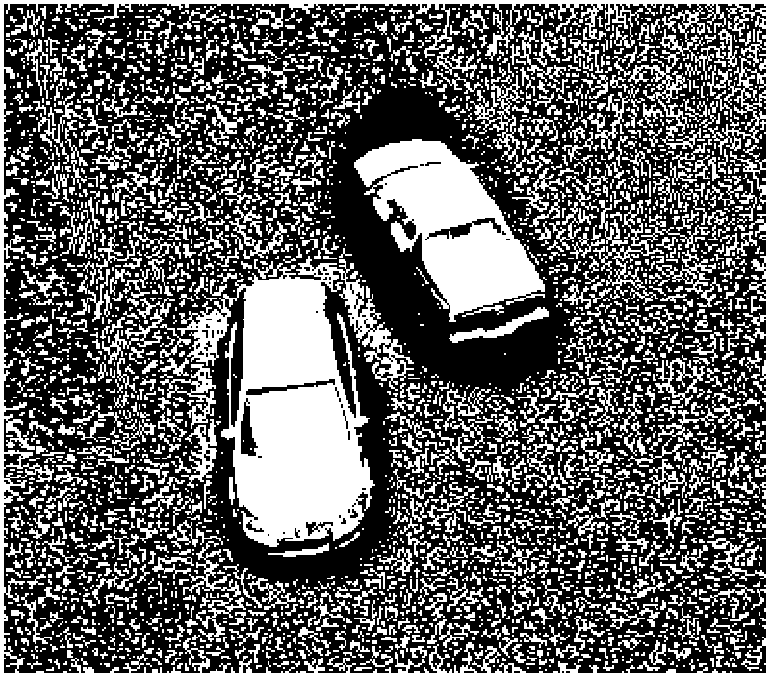


Figure III-10 Direct Subtraction example with ripple lighting issue

Figure III-11 demonstrates the background elimination algorithm. In order to solve the lighting issue, we have to sample and analyze a portion of the current frame's background (10 × 10 pixels). We will analyze the average value and the standard deviation value of the differential 10 × 10 pixel block to detect whether this portion of the current frame is background. The average difference value represents the total exposure luminance difference (exposure lighting issue), while the standard deviation value represents the instability of the differential block (ripple lighting issue).

If the average value and standard deviation value are both small, the current frame is perfect, with no lighting issue. On the other hand, if the average value and standard deviation value are both large, the 10 × 10 pixel block might have a lighting issue or be disturbed by some portion of the object. Another 10 × 10 pixel block needs to be processed to confirm this. If the average value is small but the standard deviation is large, there is a ripple lighting issue with the current frame. Similarly, if the only large number is the average value, this indicates an exposure lighting issue.

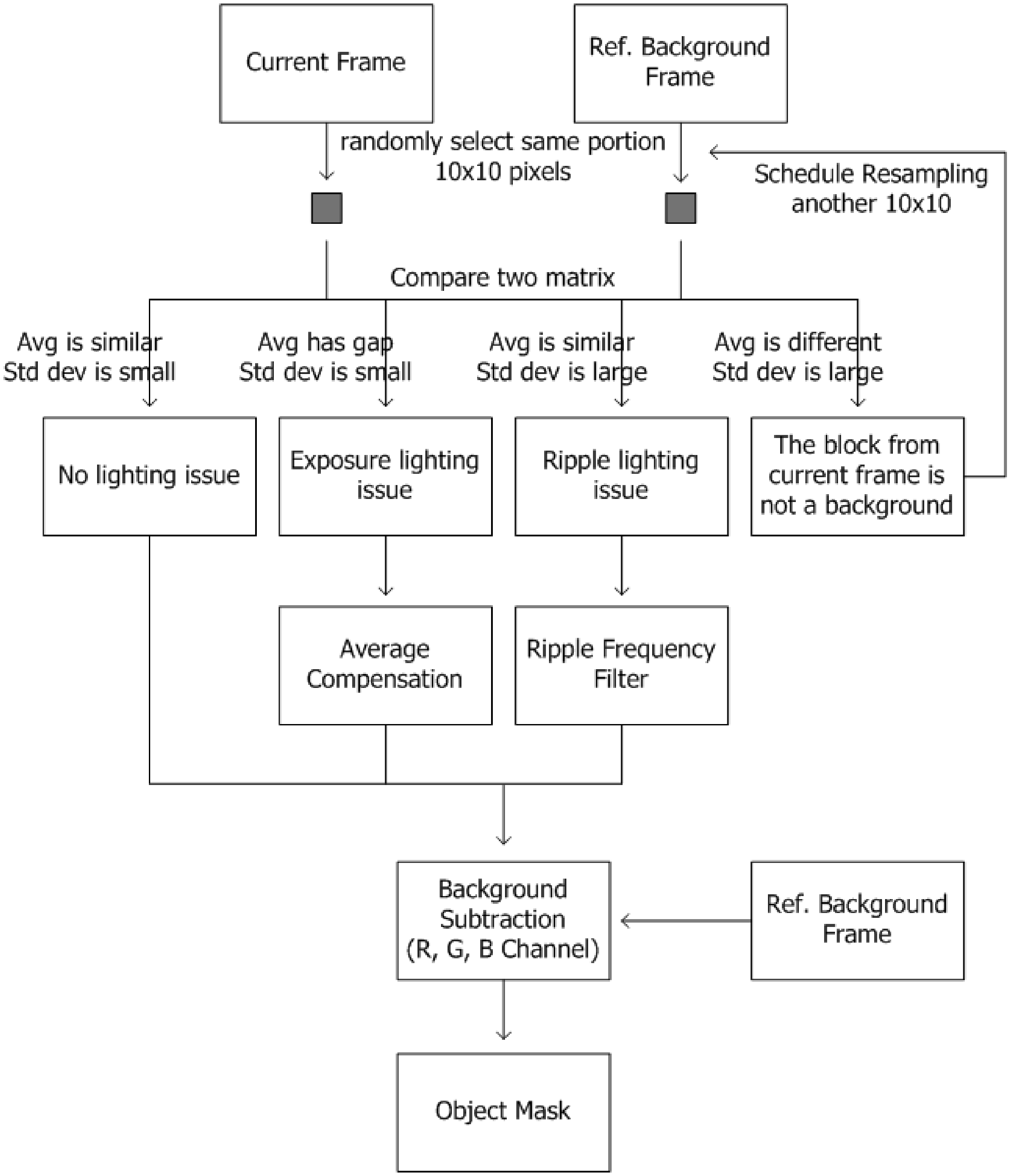

Current Frame
Ref. Background Frame
randomly select same portion 10x10 pixels
Schedule Resampling another 10x10
Compare two matrix
Avg is similar Std dev is small
Avg has gap Std dev is small
Avg is similar Std dev is large
Avg is different Std dev is large
No lighting issue
Exposure lighting issue
Ripple lighting issue
The block from current frame is not a background
Average Compensation
Ripple Frequency Filter
Background Subtraction (R, G, B Channel)
Ref. Background Frame
Object Mask

Figure III-11 Background Elimination Algorithm work flow

For the exposure lighting issue, an average luminance will be added to (subtracted from) the current frame. For the ripple lighting issue, a band-stop filter will be applied to eliminate the unwanted frequency ripples from the current frame.

3.3.3 Proposed H.264 Core Architecture

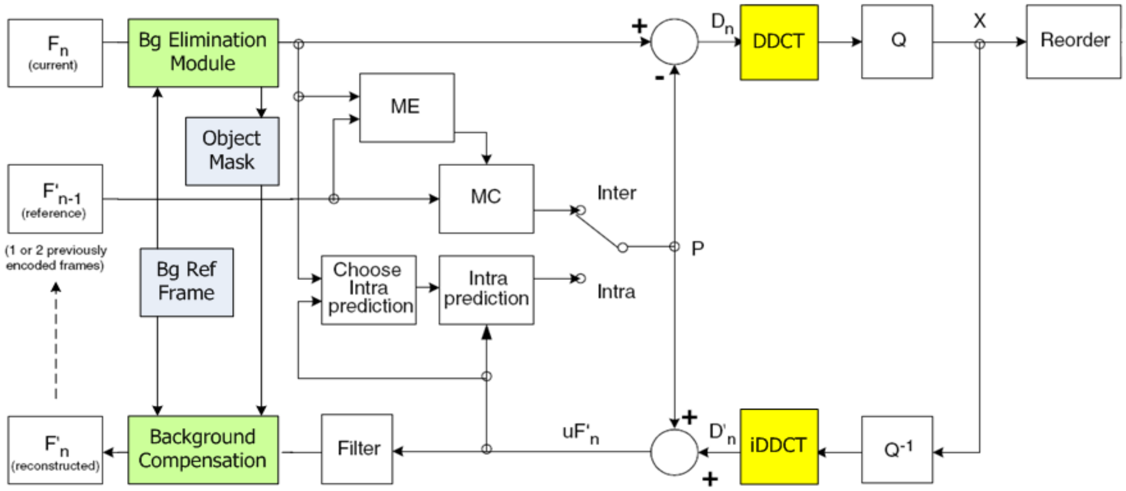


Figure III-12 Proposed H.264 codec core architecture

As Figure III-12 shows, two major modifications are applied to the conventional H.264 codec core (Figure II-2):

1) The DCT transform module is replaced by our newly designed video optimized DDCT and inversed DDCT (iDDCT) module in the proposed H.264 core.

2) Before H.264 coding takes place, the background elimination module will eliminate the background sampled from the background reference frame. Before the decoding output, the background will be added back to the decoded object-only frame.

3.4 H.264 and DSP-Based SoC Hardware Architecture

3.4.1 H.264-based SoC Top-down design

A typical H.264 codec-core-based SoC chip architecture was thoroughly discussed in Chapter 2.2. In order to fit it into the proposed digital surveillance system, in addition to the ability to encode/decode the video, the SoC should also be able to analyze the video. Video analysis has to be done in several different ways. Thus, some modifications are required for the H.264 SoC architecture.

SoC system design is different from hardware-only H.264 core design. It is more of an application-based hardware-software cooperation design. Generally, the hardware is a platform that runs software. It offers basic fundamental computation abilities. In addition, the software is going to use these abilities to expand the application possibilities. Table III-1 shows the basic advantages of hardware and software solutions.

Table III-1 Hardware solutions vs. Software solutions

	HARDWARE SOLUTIONS
Advantages	It is the fastest solution. Computations can be parallel.
Disadvantages	Hard to modify, expand, or reuse for other applications.
	SOFTWARE SOLUTIONS
Advantages	Very flexible. Can be re-programmed anytime.
Disadvantages	Relatively slow, depends on its hardware platform (CPU/DSP/MEM).

The choice to use a hardware or software solution is based on the specified applications. We implement DDCT in the hardware to increase the quality-bitrates ratio while maintaining the efficiency of the H.264 coding. However, video analysis is quite complicated and could vary for different applications. Thus, a software solution is also needed.

3.4.2 Vector Bank hardware module implementation in H.264 SoC

Figure III-13 shows a typical portion of the Motion Vector Map of a video encoding procedure. By comparing the current frame and reference frames, the Motion Estimation Block generates the MB-based Motion Vectors.

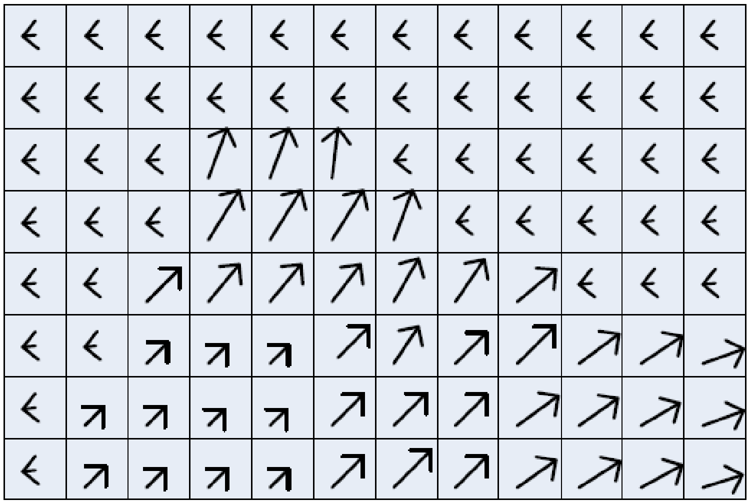


Figure III-13 Typical Motion Vector MB map in H.264 encoding

These vectors, along with residues, are going to be transformed, quantized, and compressed into video codes. After this, the Motion Estimation Block will dump the Motion Vectors to make room for the next MB. Actually, the motion vectors are calculated in the MB rather than in the frames. In previous studies [52] [53] [54], we found that it is very helpful for motion detection and object segregation to keep the vector information inside the video code. Thus, the memory-based hardware module used to keep these Motion Vectors is called a Vector Bank (VB).

The original format of a vector value described by the H.264 ME module is formatted as X by Y pixels. X and Y are the distances that the current MB is from the matched reference MB. The “X/Y” representation is very efficient in MB substitution but hard to categorize as motions. Therefore, we use the quantization to transform this vector containing X and Y into the angle value and length value of the vector to better represent its motion value:

$$Angle = tan^{-1}\left(\frac{Y}{X}\right) \qquad \text{III (1)}$$

$$Length = \sqrt{\left(\frac{Y}{16}\right)^{2} + \left(\frac{X}{16}\right)^{2}} \qquad \text{III (2)}$$

The angle-length vector description is better for storage and motion detection usage, but the “tan” and “square root” functions are unacceptably complex and consumptive in a hardware design. Thus, we use the two assumptions below to make it fit into the framework of hardware implementation:

- Sixteen approximate angle divisions. Sixteen angles are sufficient to detect the motion of objects. We approximate these angles from a rectangular plane as shown in Figure III-14 (b). The angles are not exactly the same, but are very easy to detect by comparing X and Y, as shown in Figure III-14 (d).

- For lengths, it would be easier to just use the relation of length r to X and Y within each angle rather than calculate them. Figure III-14 (c) shows how the four quadrants affect the length, and (d) shows the r approximation.

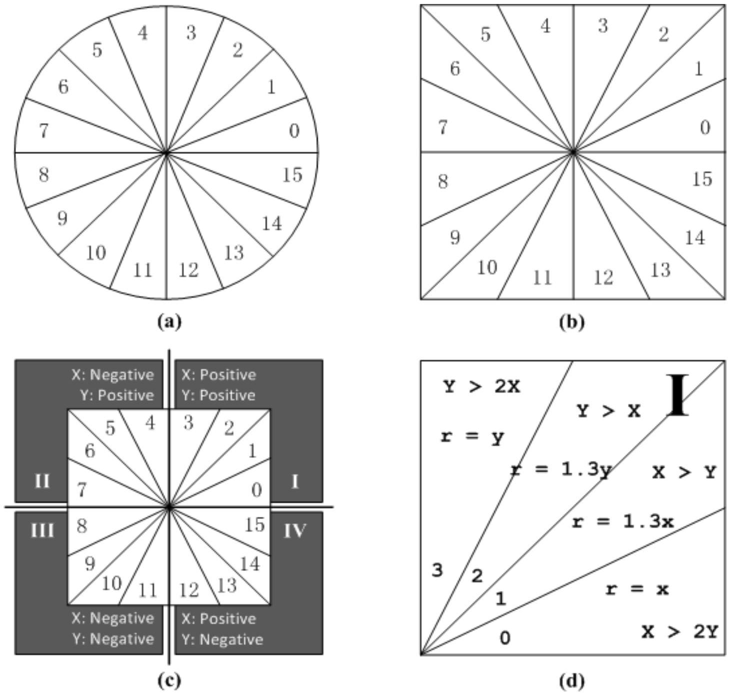


Figure III-14 Angle and length approximations for Motion Vectors

For example, if vector number (X, Y) equals (-2, 10), the vector is in the 2nd quadrant, and Y is more than twice the size of X. Therefore, the angle value is 4, and length value r = 10, which is equal to Y, as stated. We output the (angle, length), which is (4, 10), into the VB. It looks quite similar if we use (-2, 10) instead of (4, 10). Yet, actually there are essential differences. Both X and Y can be positive or negative numbers. Because these numbers are large, we probably need 16 bits to describe each of them. On the other hand, the angle number is 0~15, which only requires 4 digits. The length number is also a positive-only number, which probably only takes 8 bits. In hardware, that is a huge savings, from both the bandwidth and storage perspectives. More than that, the angle-distance model also offers a good starting point for the software to do its job. Hundreds or even thousands of instructions are saved in one second.

VB is not a typical module inside the H.264 codec core. With a dedicated channel built between the H.264 codec core and VB (called the "VB interface"), as shown in Figure III-15, the Motion Vector information could be reserved and formed frame-by-frame, making motion detection possible.

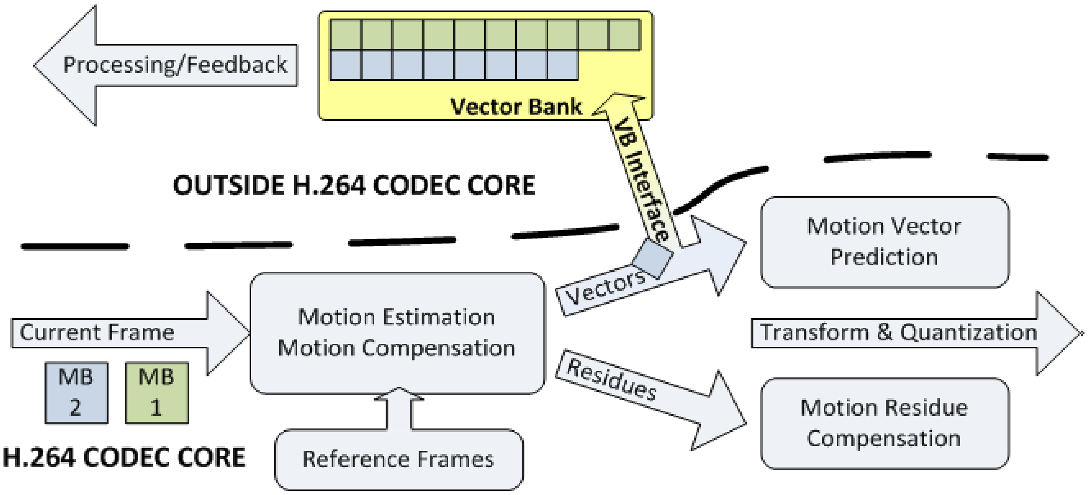


Figure III-15 Vector Bank gathering Motion Vectors

3.4.3 Digital Signal Processor implementation

The last part of the SoC hardware design is built as a video analysis software platform. This platform includes a processor (DSP processor), a dedicated memory bus, and, of course, a dedicated memory.

It would probably be easier to implement our platform based entirely on the original platform without a dedicated memory and buses. However, without expanding the bandwidth of the system bus, the system efficiency would be highly risky and more susceptible to crashing. By implementing a dedicated memory and bus, the original system balance is maintained.

On the other hand, in the software portion of the digital surveillance system architecture, the DSP processor should have the ability to

- detect and analyze the motion
- outline the object with LoG operators
- get the distance-to-angle signature of the result of an LoG operator
- identify or categorize the object using a known database

Obviously, these are tough jobs for any available software platform. Surely this is going to require a large amount of bandwidth and computation resources. A dedicated processor-to-memory system is the best solution for such a tough job. At the same time, it could relieve the burden of the original system bandwidth in order to keep the system healthy, preventing the occurrence of frame loss and other system hazards.

3.4.4 H.264 and DSP-based SoC chip in surveillance node

Figure III-16 illustrates the "surveillance node" defined at the end of Chapter 3.2 in more detail. Initially, the video signal will make its way from the camera sensor interface to the system memory module. Then, the proposed H.264 module will access the memory to obtain the data. The current frame data will be compared with background reference data and the background information will be eliminated. Then, through the video compression optimized DDCT module, the coded H.264 video data will move back to the main memory and finally out of the SoC chip, to be stored on the digital hard disk through the USB/1394 Controller. On the other branch of the H.264 codec core, the Motion Vector information is copied and transferred to another memory based module–VB. A DSP has direct access to VB, and an application-specified video analysis program is performed. Finally, the analysis results will be packed compactly and sent to a private network via a Wi-Fi/Ethernet controller. The real-time video can be viewed directly through the network. The stored video data can also be retrieved from the disk and sent to the network upon on the request of a surveillance administrator/officer.

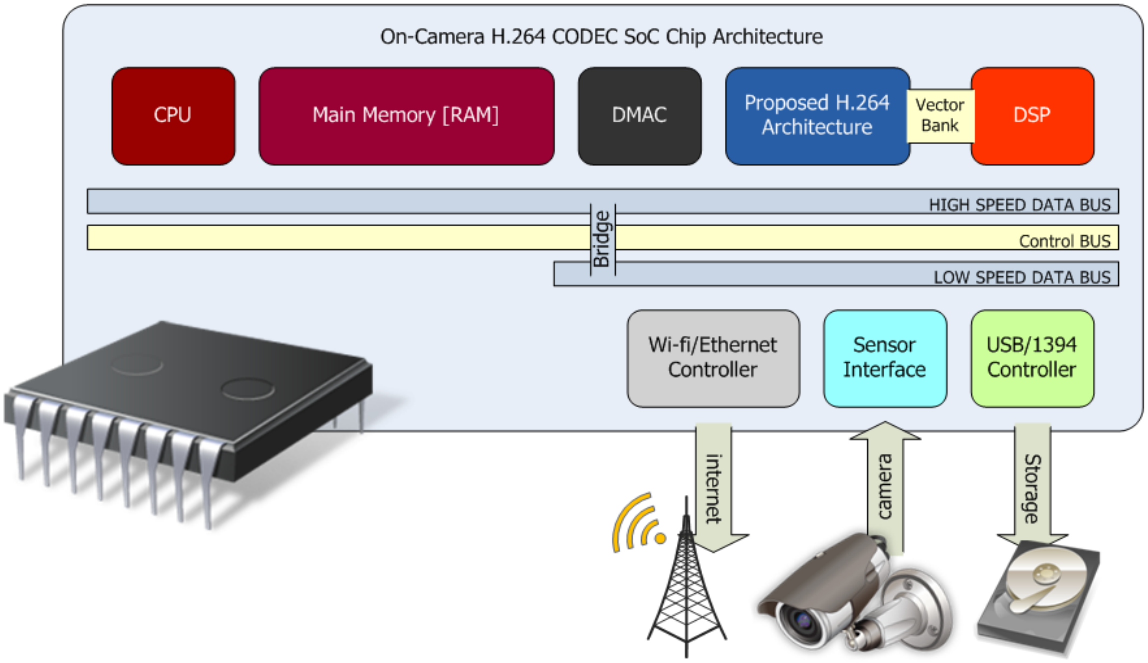


Figure III-16 Proposed H.264-Based SoC Architecture in surveillance node

3.5 Proposed Digital Surveillance Network System Architecture

The surveillance node is a cell of the entire surveillance network system. With the implemented hardware/software platform, it should have the ability to

- capture and compress the surveillance video
- send the H.264 coded video via a local private digital network
- store the captured video (or dump it) onto a digital data disk
- analyze the object motions using the motion vectors stored in VB
- analyze and categorize the object using an imaging DSP

Data is gathered, compressed, analyzed, and stored in every node. The various technologies previously discussed have contributed to the system, not individually, but in an organizational format. The whole architecture of this digital surveillance network is a distributed system. With the use of numerous distributed high-tech surveillance nodes, the intelligence of the system has been brought to another level.

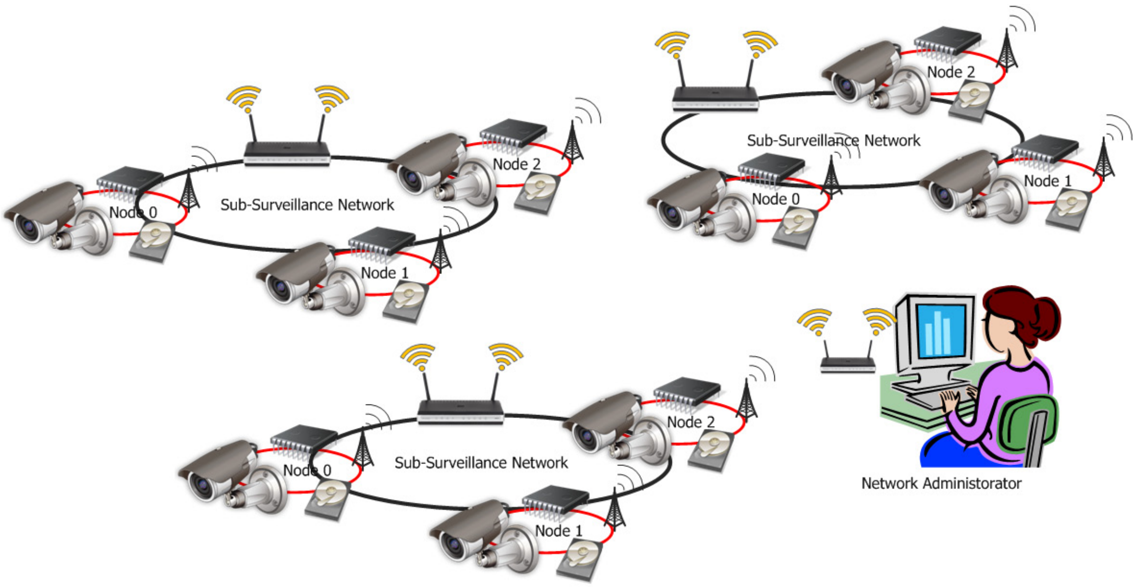


Figure III-17 Proposed distributed digital surveillance network architecture

Vector data goes from the H.264 core to VB and waits for analysis by DSP. DSP then analyzes these vector data, categorizing the results as “interesting data” or “regular data.” Interesting data means that by analyzing the vector data, DSP found data that carries some information that might be of interest to the surveillance system. This data would be sent to the group node and finally the control room, triggering an alarm. Regular data would not be transferred if the network load is high. Instead, it will be stored locally on the node in case a post check is ever needed.

A security administrator can also set the data saving strategy for each of the surveillance nodes. For example, the storage buffer time could be “30 minutes”. It means if there is no “interesting data” during this period of time, the video data file would be deleted automatically. For different applications, different strategies could be used, saving a large amount of bandwidth and data storage capacity.

IV SYSTEM SOFTWARE PLATFORM DESIGN

4.1 Proposed System Software Platform Architecture

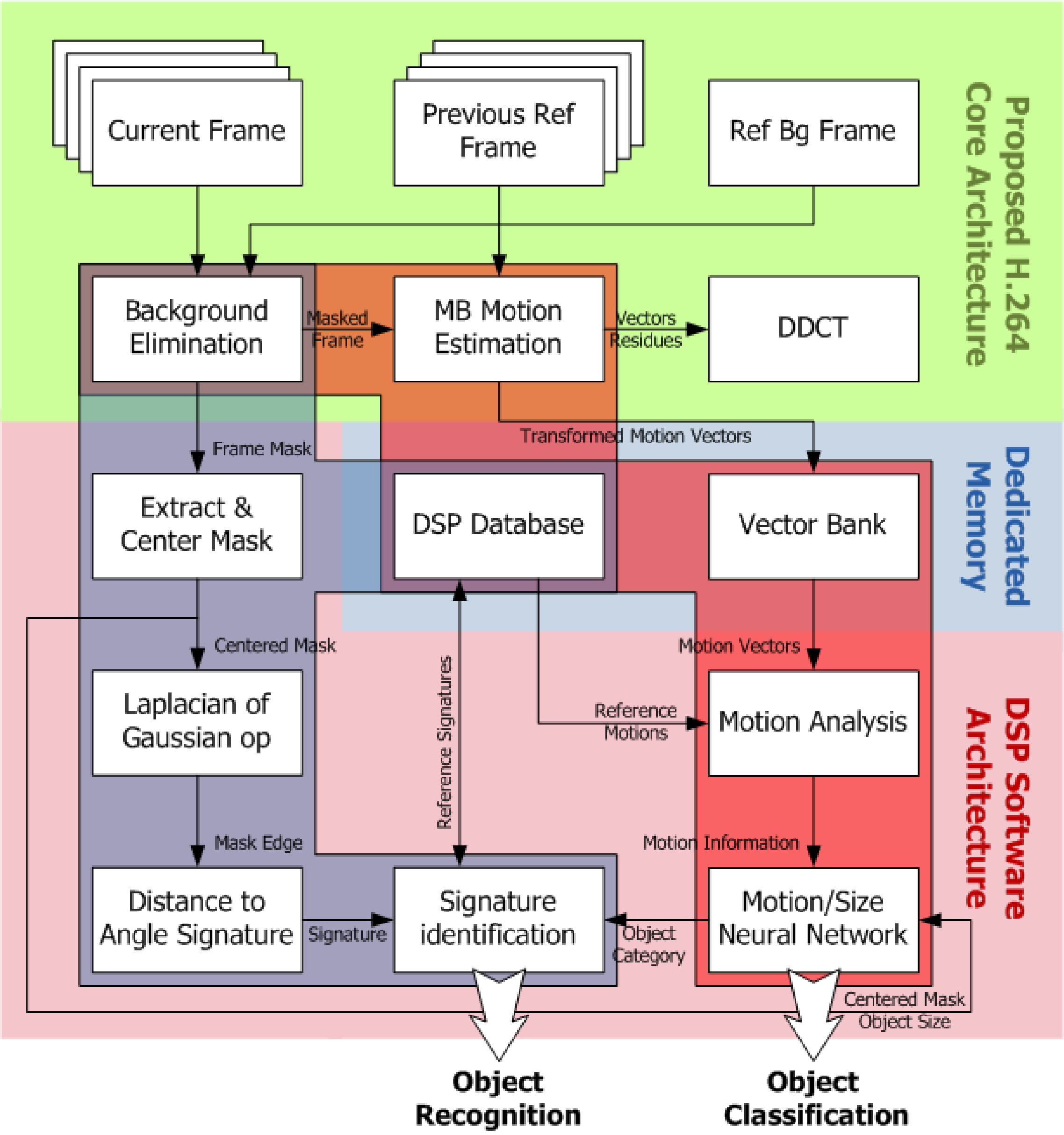


Figure IV-1 System software platform architecture

Figure IV-1 shows a block flowchart of the proposed digital surveillance network architecture. The green part is the H.264 codec hardware layer. Data passes through the green part and comes to the blue part, which is the dedicated memory for DSP. It contains the DSP database and Vector Bank. Finally, the red part is not hardware but the proposed software (program) architecture.

There are also two data paths shown in Figure IV-1, which represent the two major applications for the DSP software architecture. The blue one is the object recognition system, and the red one is the object categorization system.

4.2 Object Categorization System

4.2.1 Object Classification

Object classification is a very important step and is followed by motion detection. It can offer great assistance to a surveillance system. In contrast to the term "object recognition," object classification can be considered as a standard pattern recognition issue. At present, there are three main categories of approaches for classifying moving objects.

1) Shape-based classification. Different descriptions of shape information (e.g., points, boxes, or even more sophisticated shapes) are available for classifying moving objects.

2) Motion-based classification. Motion behavior (e.g., spinning, turning, etc.) can also be a good aid for classifying moving objects.

3) Region-based classification. The region in which the motion is taking place (e.g., road, yard, motor lane, etc.) can also affect the object classification result.

4.2.2 Object Categorization System

The proposed digital surveillance system eases the bottlenecks of bandwidth and human intervention. However, for large scale use, the database will become huge. Performing one-by-one object recognition is nearly impossible in this case. Thus, an object categorization system is needed.

For small private residence, object categorization system would also improve the quality of security alerts. For example, considering a case of a small surveillance system for a private backyard, the system is programmed to send alert whenever it detects an un-indentified object. What is really happening is that the surveillance system will keep sending alerts every time even there is only a cat or dog walks by. Without object categorization system, even a very small surveillance system could become very annoying.

With an object categorization system, things get better. As Figure IV-2 shows, our backyard has a lot of “visitors”. The digital surveillance system will capture these visitors and isolate them as “objects”. The motion information for these objects is also detected and analyzed. Because the camera is positioned at a fixed angle, the size and speed in pixels is very easily transformed into their real world equivalents, like inches or meters. By providing sufficient information, a well trained “backyard” system can easily tell the difference between these various categories. It can also cooperate with the object recognize system. Imagine that a series of candidate object models is ready to be compared right after the category of the object has been detected.

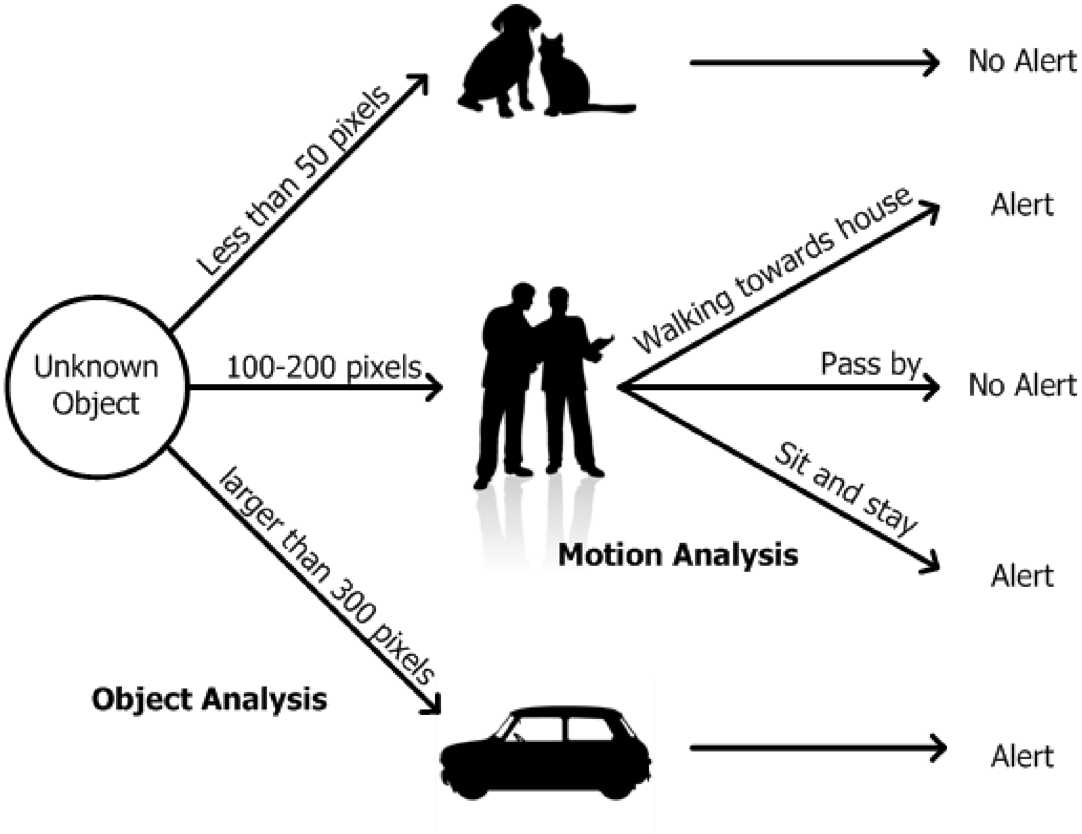


Figure IV-2 Backyard object categorization system

A very big surveillance system might contain thousands of recognition candidates in its database. Generally, a key feature of an object categorization system is the creation of a “tree map” to save a large amount of time in object recognition. As shown in Figure IV-3, when object recognition is needed, the system will first deliver the request to the object categorization system. Large systems (e.g., national security) may require multi-level tree systems, while small systems may need only a one-level category system. The system could vary based on specific applications. However, they all share some common features:

- greater accuracy is preferable, but it does not need to be 100% accurate
- the input could be anything – there is no “uniform” format for input
- have the ability to expand or narrow the category range it tries to detect

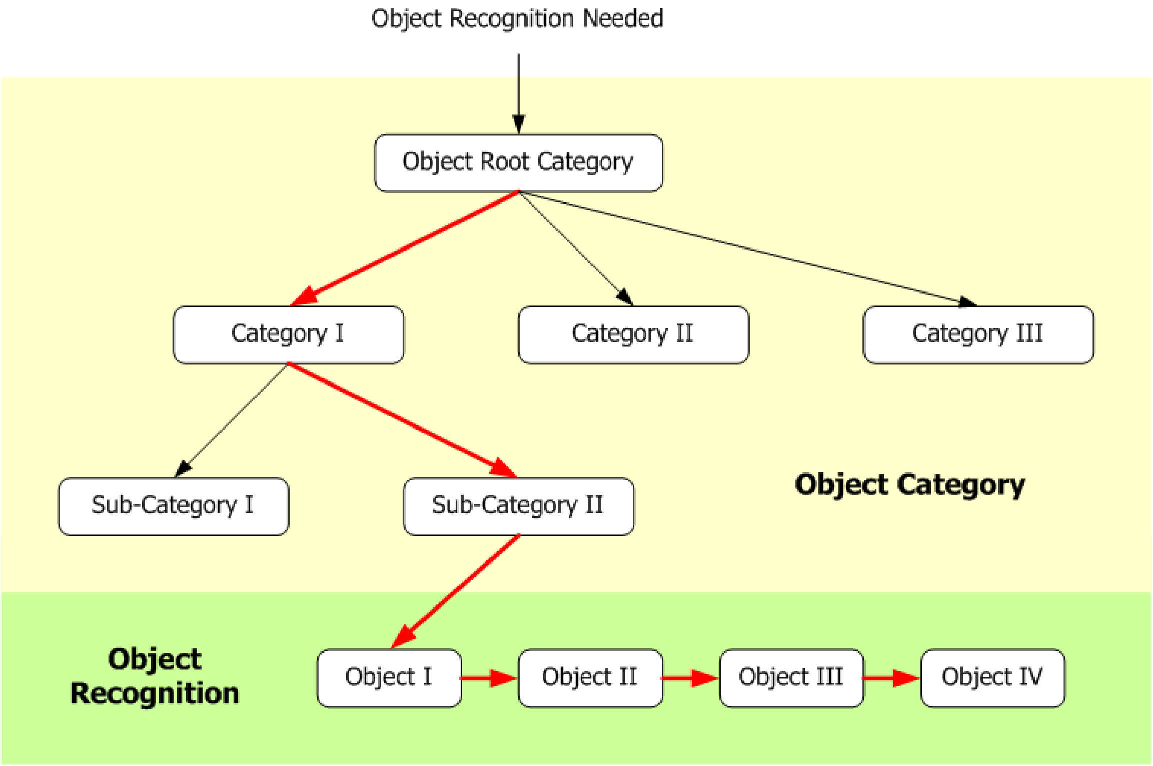


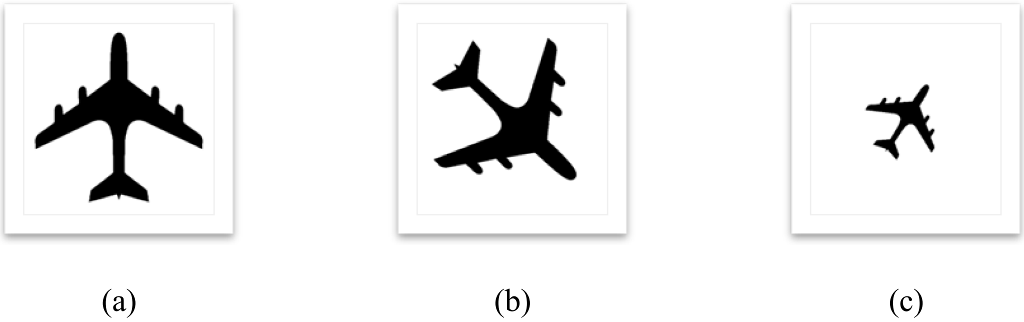
Figure IV-3 Object Categorization System and Object Recognition System

From the list above, we can clearly see that the object categorization system matches all of the requirements and features of an ANN system, specifically, a BP ANN system. The input of a BP system could be anything. We can take the object size (object mask) and object motion values (Motion Vectors) as the input and some real objects like “dog,” “human,” or “cars” as output.

4.3 Object Recognition System

An object recognition system is a special stage of the object categorization system. While the object categorization system focuses on the features, an object recognition system focuses more on the context. An object recognition algorithm should have the ability to recognize an object regardless of its “pixel size” or “orientation.” We decided to use the distance-to-angle signature algorithm as the fundamental structure of our object recognition system.

Figure IV-4 shows an example of a distance-to-angle signature. We have 3 symbol images. They are all airplanes. The differences between (a), (b), and (c) is that (b) is turned 230° compared to (a), and (c) is turned 320° and half the size of (a).



(a) (b) (c)

Figure IV-4 Distance-to-angle signature examples

After applying the distance-to-angle signature algorithm to these three images, we get the results shown in Figure IV-5. The top figure is the signature result of (a). We turned it 720° instead of 360° in order to give a better illustration of orientation shifting. As shown for the second signature (the result of symbol b), there is nothing different from (a) except the orientation point. It shows a 230° shift on this signature. The third signature is generated from (c). It has a 320° orientation shift and also a one-half amplitude shrinkage in the signature.

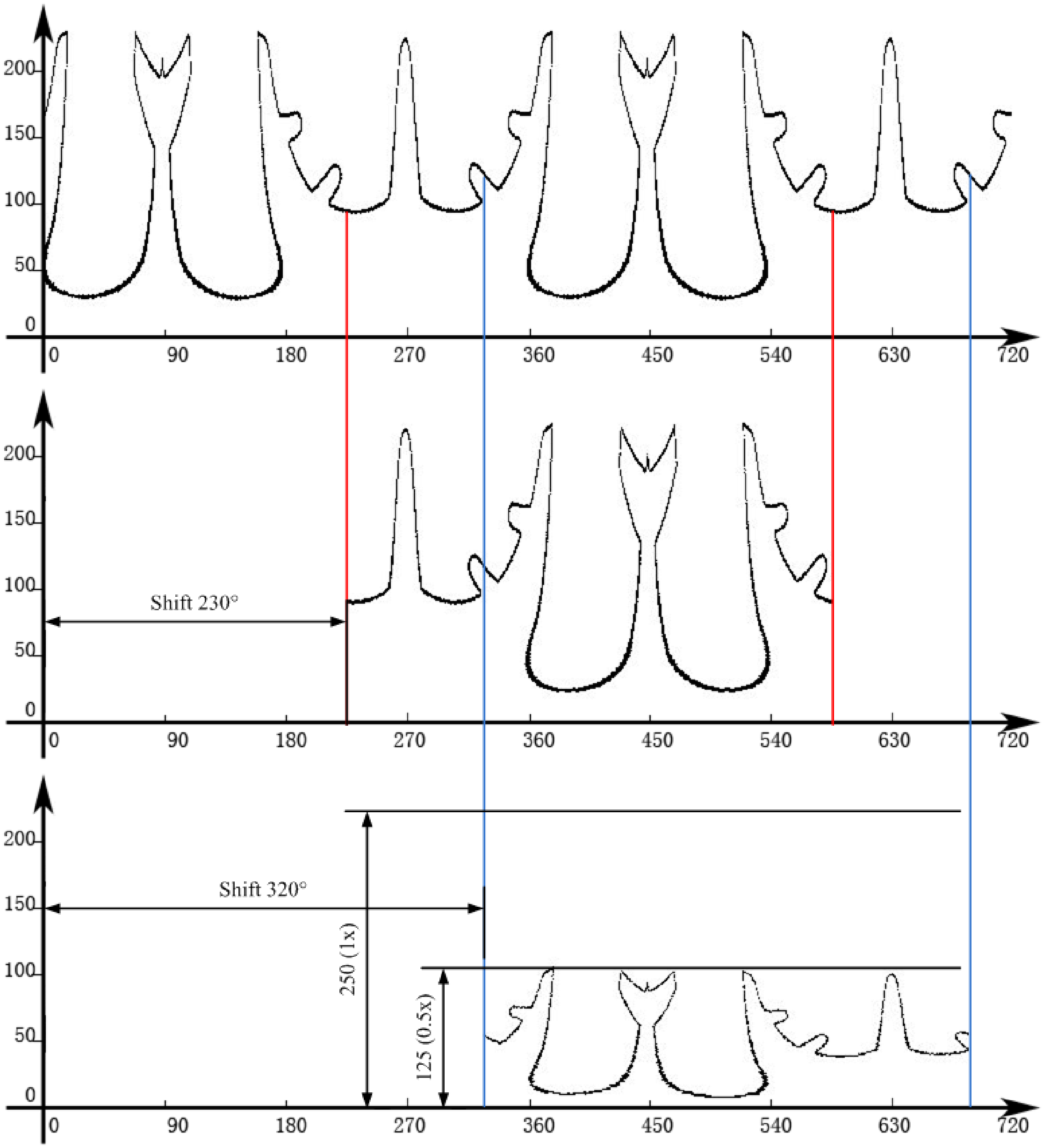


Figure IV-5 Distance-to-angle signature results

The example symbols from Figure IV-5 are identical. By using the distance-to-angle signature algorithm, we can detect the right symbol even with a slight difference.

We can merge 2 signatures into one plot, and calculate the distance differences. The smallest difference will have the best match. With proper programming, the distance-to-angle signature algorithm can be a very important component of the object analysis stage, as shown in Figure IV-6.

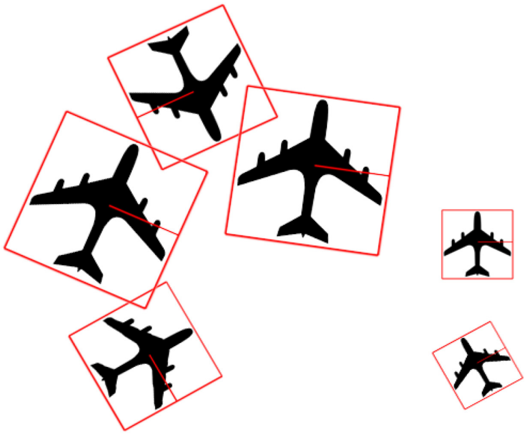


Figure IV-6 Multi-object detection by distance-to-angle signature algorithm.

V SYSTEM EXPERIMENTS AND RESULTS

5.1 H.264-Based DDCT System

5.1.1 DDCT Experiment Platform Design

As Figure V-1 shows, the pink area is the DDCT module. DDCT is a hardware parallel system. It is more like a Multi-DCT module.

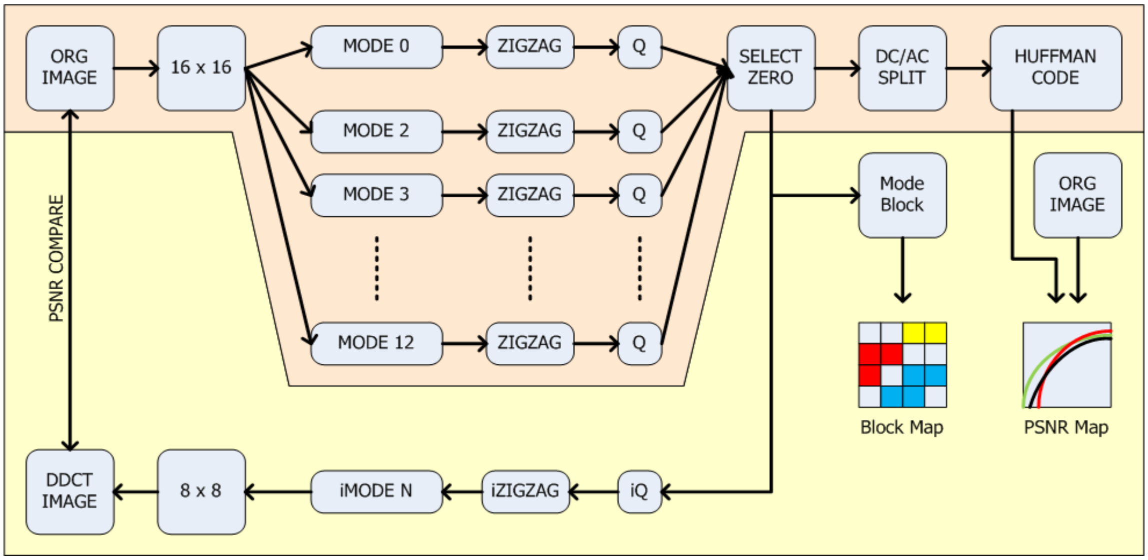


Figure V-1 DDCT experiment software platform

In the DDCT module, the original image (a 512 × 512 pixel image in this experiment) was divided into multiple 16 × 16 pixel MBs. Then, each of the MBs was sent to 11 different DDCT mode processors one-by-one. Every mode processor processed the 16 × 16 pixel MB just as was illustrated in Chapter 2.3. Then, the results were sent to a zigzag scan and quantization model, just as with 2D DCT. The results were then split between the DC part and Alternating Current (AC) part. The DDCT selected the most efficient compressed result (with the most zeros) as the best candidate and output the mode numbers with the DC/AC values. Then, the image was coded by the Huffman coding algorithm to eliminate the entropy redundancy.

The yellow area in the figure is the experiment test bench. After the best candidate was selected, the result was copied and processed in the opposite way in order to generate a recovered image compressed by DDCT. Then, the PSNR analyzer was applied to determine the difference between the original image and the recovered image. At the same time, the best candidate mode numbers were recorded and reformatted into another “block map.” Finally, the Huffman coding result and original image were compared, and a “PSNR-bitrates” map was generated to illustrate the efficiency of the DDCT in this case.

5.1.2 DDCT Test Cases Design

Selecting the right test candidate was very important for the system experiment. We used popular test images for image compression. For discussion purposes, we used multiple combination modes: one mode for conventional 2D DCT, 3 modes for only 2D DCT and diagonal DDCT, 7 modes for the original DDCT, and all 11 modes for the proposed video optimized DDCT to compare the results. Multiple quantization factors were also added to enhance the comparison.

5.1.3 Experimental Results

1) Replacement Efficiency

Table V-1 Replacement Efficiency experiment I

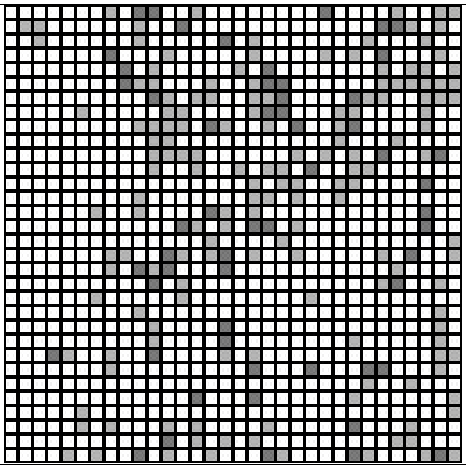
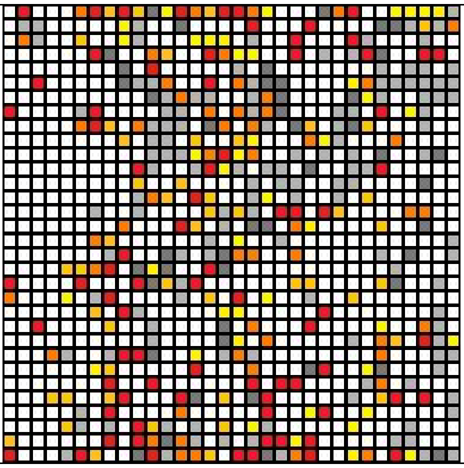
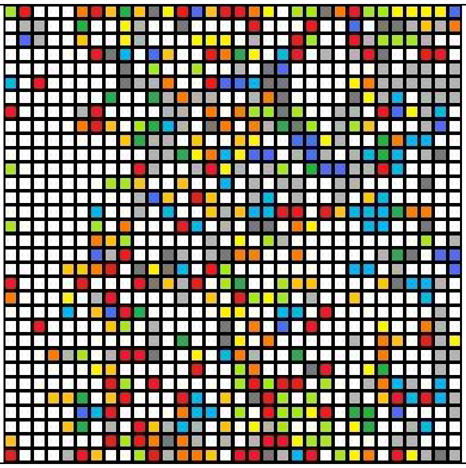
	
512 × 512 image “Lena”	0, 2, 3 Modes, Q = 1
	
0, 2, 3, 4, 5, 6, 7 Modes, Q = 1	0, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11 Mode, Q = 1

Table V-1 shows one set of the results coming from the replacement efficiency experiment. This is a single image but was processed in DDCT with different mode combinations. The resulting color block replacement map was created using small colored blocks. Each colored block represents a 16 × 16 pixel MB. A white block indicates that the 2D DCT or mode 0 is the best candidate. Gray blocks represent modes

2 and 3, which are the diagonal modes. Warm color blocks (red and yellow ones) indicate modes 4, 5, 6, and 7. Cold colors (green and blue ones) represent those blocks processed by modes 8, 9, 10, and 11. Q is the quantization factor. It shows a very simple result if we put more candidate modes into the pool: the output will show that more conventional DCT blocks are replaced.

Table V-2 Replacement Efficiency experiment II

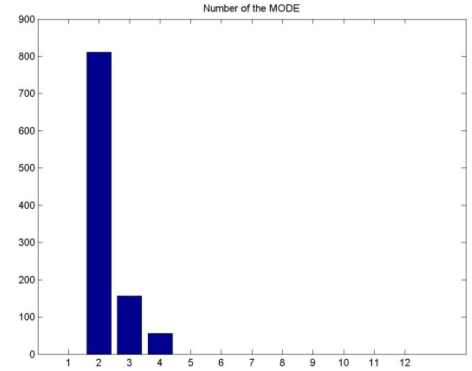
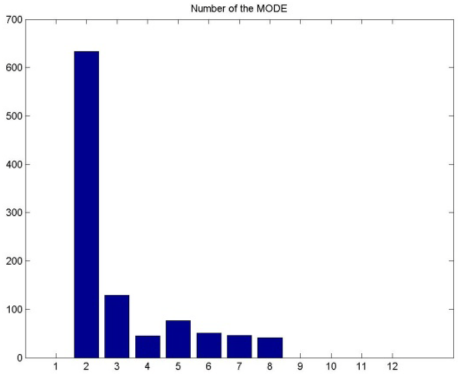
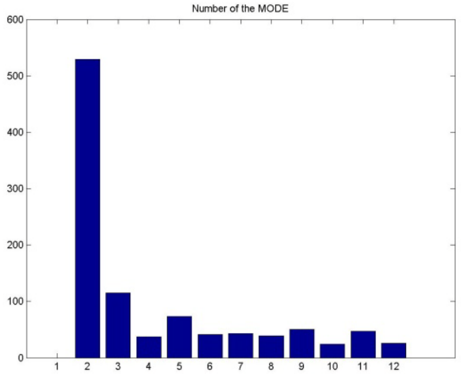
	
512 × 512 image "Lena"	0, 2, 3 Modes, Q = 1
	
0, 2, 3, 4, 5, 6, 7 Modes, Q = 1	0, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11Mode, Q = 1

Table V-2 shows the same case as Table V-1, but in these images, the horizontal axis number represents the corresponding mode number. The vertical axis represents the best candidate block amount for each of the modes. There are a total of 1024 MBs for a 512 × 512 image. Obviously, the conventional 2D DCT (DDCT mode 0) still takes the main portion of the "best candidates." Yet, it is also noticeable that with an increase in the number of modes joining the pool, the mode 0 number decreases.

Table V-3 Replacement Efficiency experiment III

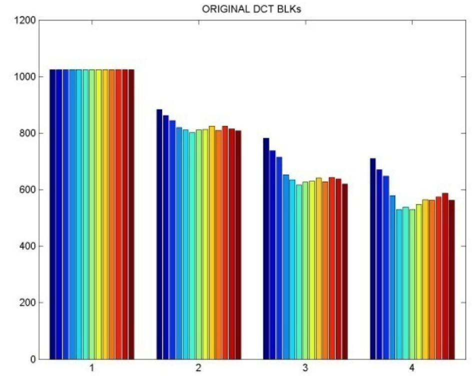
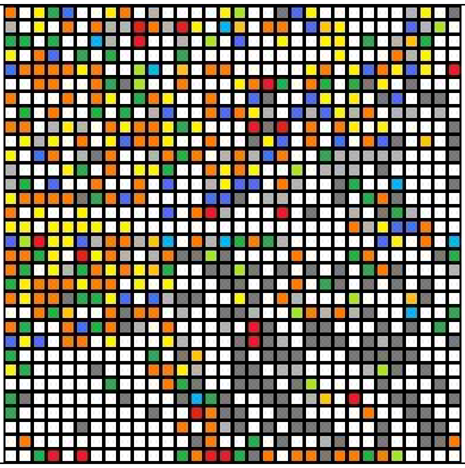
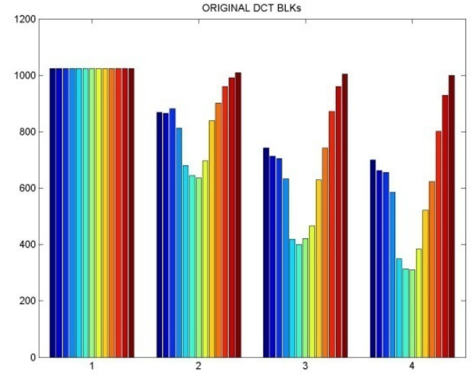
	
512 × 512 image "Lena"	Mode 0 amount in different modes and Qs
	
512 × 512 image "Cameraman"	Mode 0 amount in different modes and Qs

Table V-3 shows the mode 0 results for two different images. Each bar figure has four sections: "mode 0 only," "modes 0, 2, and 3," "modes 0 to 7," and "modes 0 to 11." In each mode, the original picture is compressed using different quantization numbers. In this example, we used Q numbers 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1.0, 2.0, 3.0, 4.0, and 5.0. A larger Q number represents the use of a greater compression ratio by DDCT and the retention of less image quality.

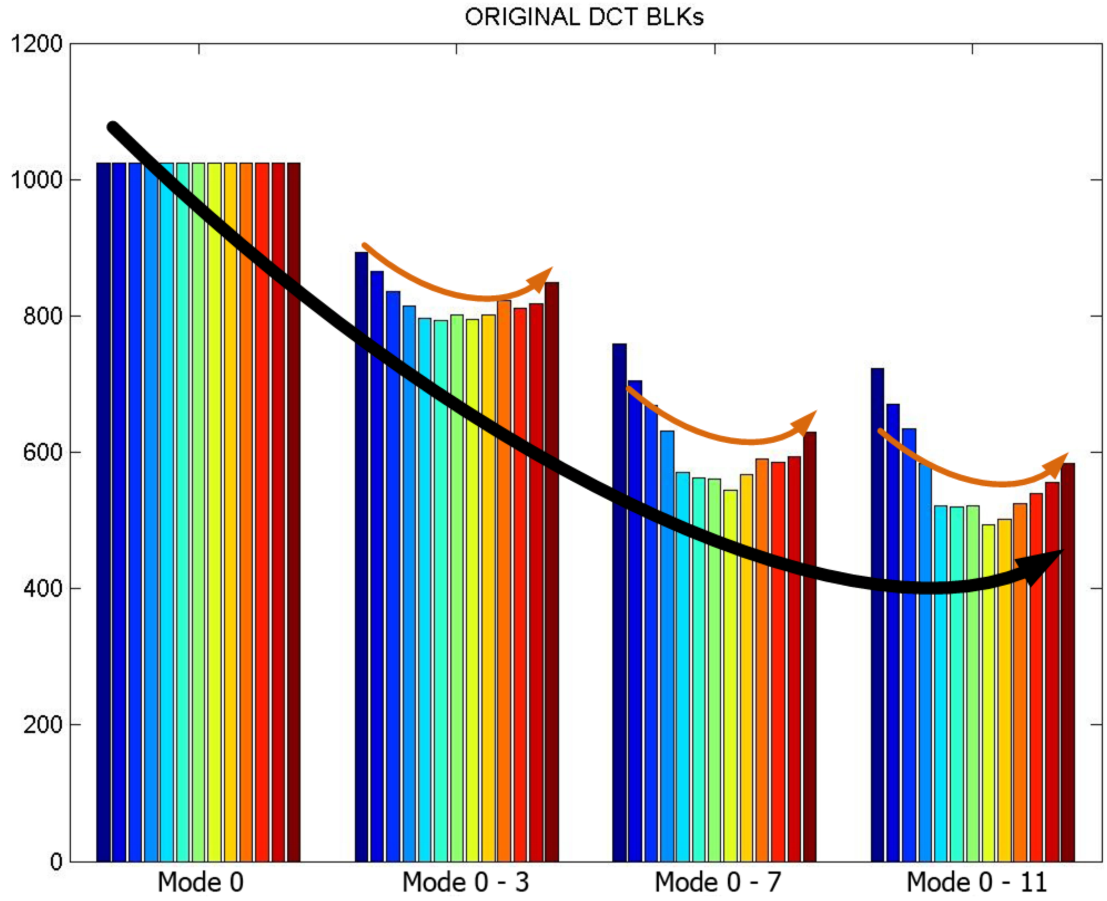


Figure V-2 Mode 0 trend for different mode and Q combinations

Figure V-2 shows that as the Q number becomes increasingly larger, more MBs will not always be represented by other modes. There is a “sweet spot” for the best replacement result, and it is different for different images. From the general trend, we can see that as more modes continue to be added, the replacement impact on mode 0 is going to decrease.

2) DC and AC compression ratio

Figure V-3 and Figure V-4 illustrate the DC/AC compression ratio change for two images caused by the change in the quantization factors in different modes.

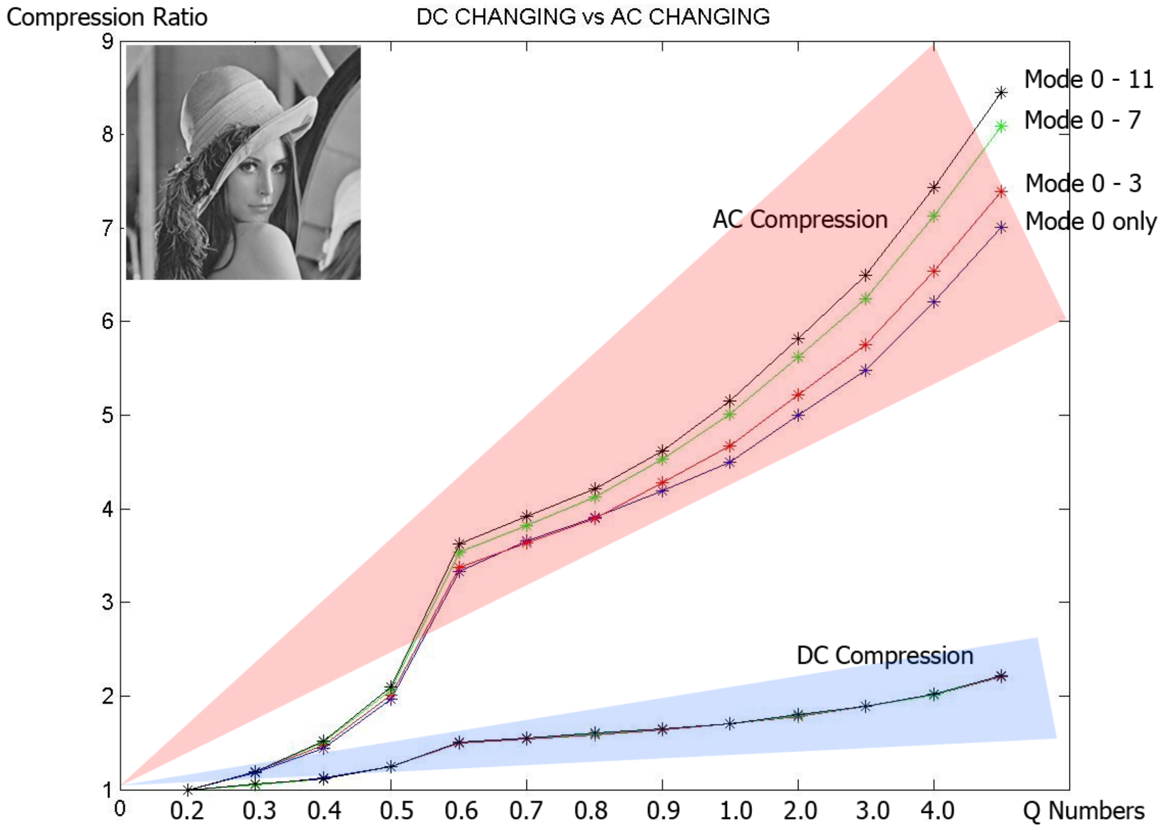


Figure V-3 DC/AC compression ratio with Q numbers

The compression ratio constantly increases as the quantization factors become larger, which means the image quality continues to decrease. For a single MB, it does not matter which mode of DDCT is applied; the DC value, which is the average value of the 16 × 16 (256) pixels, will always remain the same. Thus, the DC compression ratio will be affected by the Q numbers but will not change very much as the result of the different mode combinations. The blue sections of these two figures show the DC compression ratios caused by the different Q numbers and different mode combinations.

In contrast to the DC value, the AC value (the red area) could be compressed better with more DDCT modes. The directional DCT algorithm caused better compression of the directional AC value, and it obviously achieved its target.

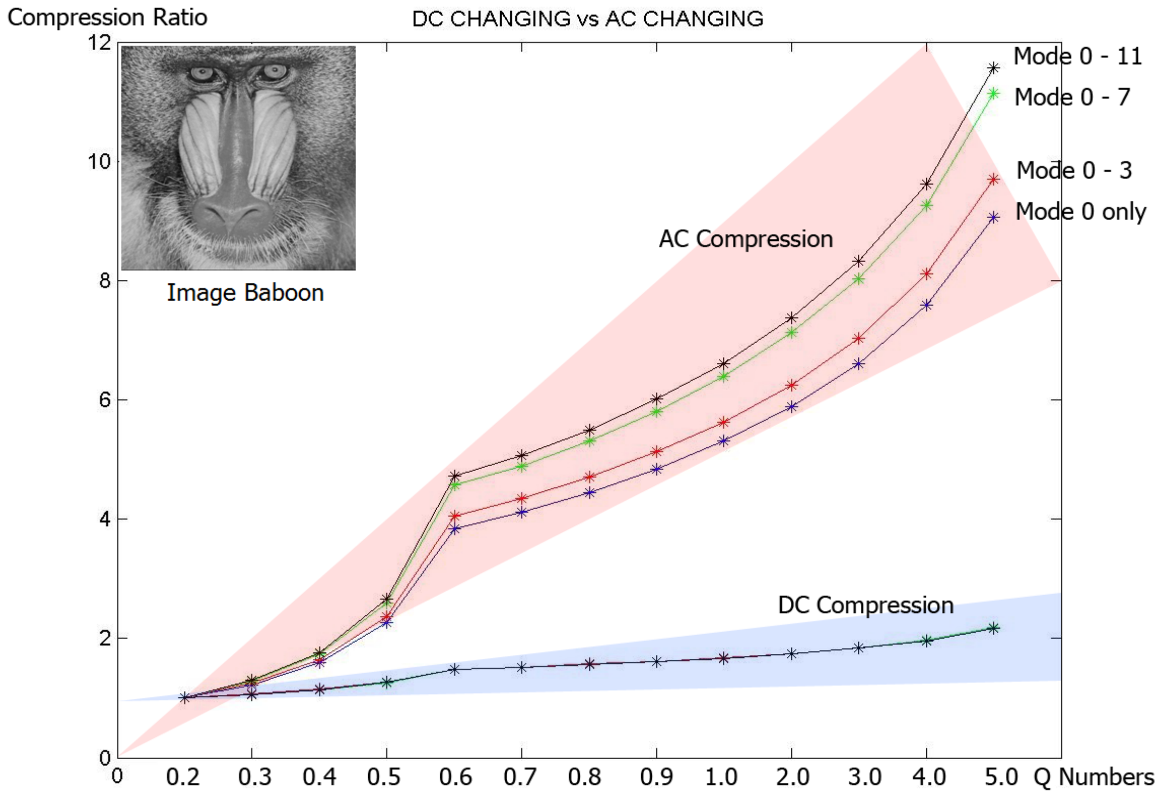


Figure V-4 DC/AC compression ratio with Q numbers II

3) PSNR-Bit/Pixel Ratio

The peak signal-to-noise ratio (PSNR) for the Bitrates map is one of the most important key feature demonstrations for a digital image compression algorithm. Figure V-5 and Figure V-6 show how different mode combinations affected the PSNR-Bitrates map.

In Figure V-5, the conventional 2D DCT offers nearly 30 dB in PSNR when using 1 bit per pixel. By using a smaller quantization factor, the bitrate becomes 2 bits per pixel, and the PSNR reaches 36 dB. In contrast, our fully loaded 11 mode video optimized DDCT offers approximately 34 dB in PSNR at a 1 bit per pixel transfer rate. If the rate becomes 2 bits per pixel, the PSNR could reach nearly 40 dB.

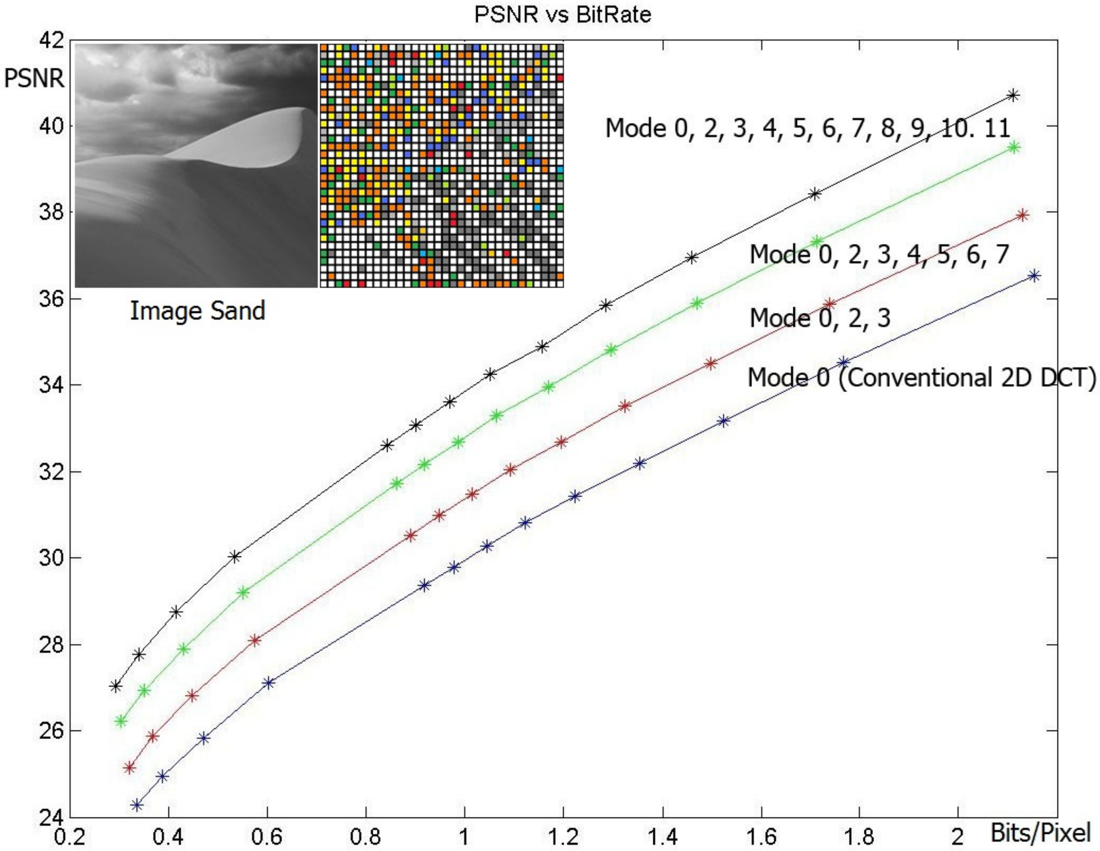


Figure V-5 DDCT efficiency of PSNR–Bits/Pixel map

Not every experiment had an output as good as that shown in Figure V-5. In Figure V-6, the optimized 11 mode DDCT does not show a big lead. As can be seen, the advantage of the multi-mode combination DDCT compared to the conventional 2D DCT is relatively small throughout the entire figure–a lead of approximately 0.5 dB in PSNR values when they share the same bitrates.

One of the big reasons for the variation in the efficiency of multi-mode DDCT is the replacement efficiency. In Figure V-5, the image “Sand” has a large number of MBs featured in a diagonal or semi-diagonal pattern, and most of them have been processed and replaced by modes 2 and 3. In contrast, the image “Cubic” in Figure V-6 has a large number of white/gray areas. These areas will have the best PSNR-Bitrate value even when processed by the conventional 2D DCT.

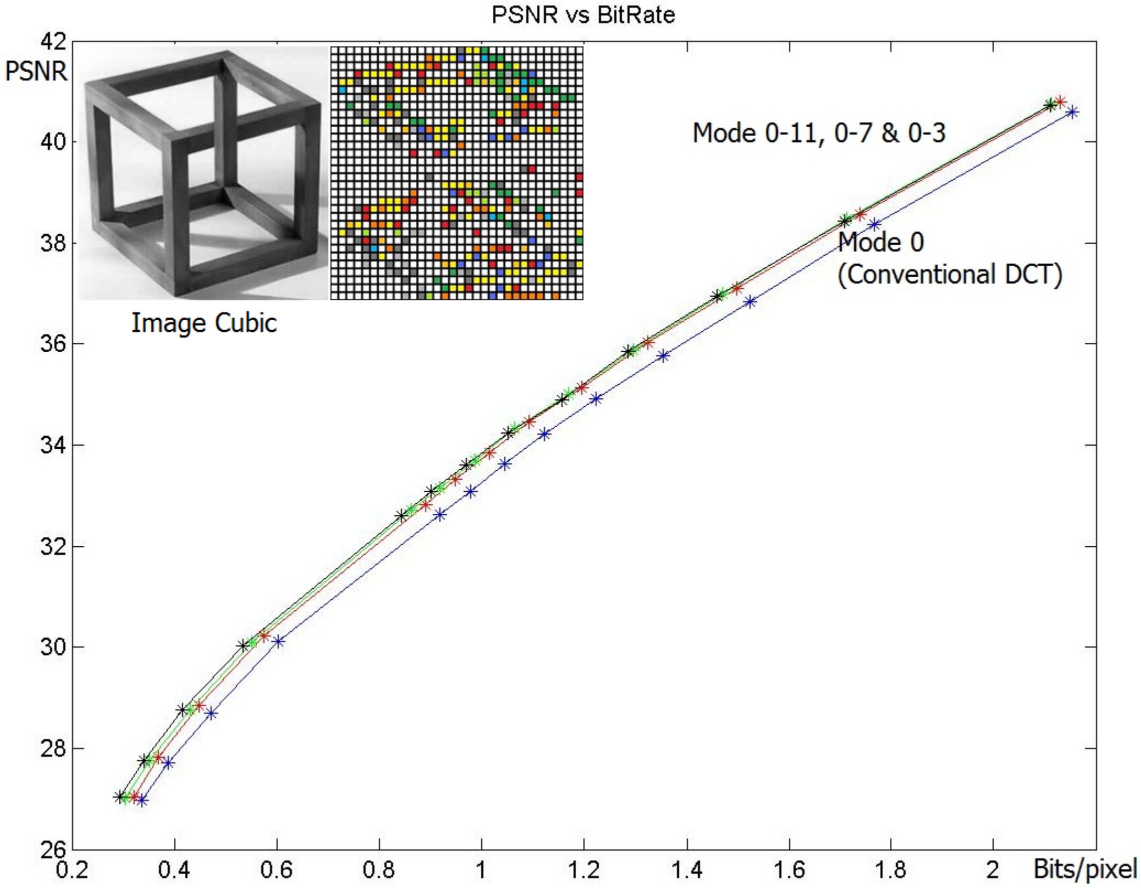


Figure V-6 DDCT efficiency of PSNR–Bits/Pixel map II

5.1.4 Summary

The PSNR-Bitrates map is the best conclusion of our video optimized DDCT. As expected, it shows its advantage when using 16 × 16 pixel MBs. By replacing DDCT with 2D DCT of the H.264 codec core, we can get higher video quality while maintaining or lowering the transfer rate.

5.2 Background Elimination System

All of the video analyses start from motion detection. The Motion Vectors that are extracted from the H.264 ME module by VB offer a great platform for the Motion Detection algorithms. As discussed in Chapter 2.4.1 and Figure III-10, the luminance/color matching can be easily fooled by various lighting conditions. Using the most similar MB to generate Motion Vectors and Residues is probably the best way to compress an image but is not perfect in motion detection. That is why background elimination is so important. It is implemented in the H.264 codec core to help the ME block obtain more accurate object/motion detection results.

The working situation for a surveillance camera is quite different from a general purpose camera. It usually remains in a static status, which means the background could be considered static over a relatively longer period of time (say every 10 minute). Background references can be captured by the administrator and reused for future comparison, analysis, and processing. Once we eliminate the background of the current frame, leaving just the object, the motion vectors generated by the H.264 ME module will become relatively accurate and usable.

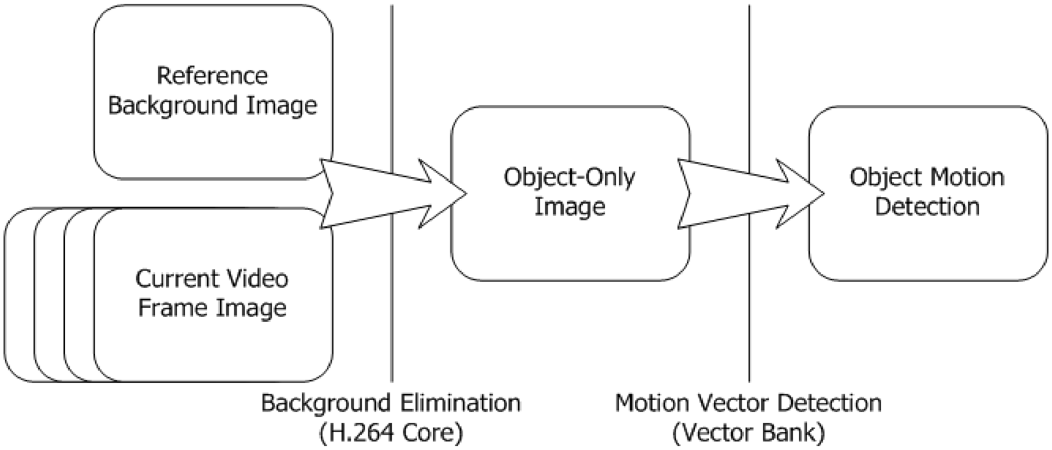

Reference Background Image
Current Video Frame Image
Object-Only Image
Object Motion Detection
Background Elimination (H.264 Core)
Motion Vector Detection (Vector Bank)

Figure V-7 Motion Detection with Background Elimination System

Figure V-7 shows a model of the motion detection system used with the background elimination system. Before the video stream is captured, a reference background image is captured. The background elimination algorithm eliminates the background of every input image. Thus, an object-only image will be prepared for the following ME module. With object-only images, ME will be sent to the H.264 core, and finally object motions will be detected by the ME module.

Another huge benefit of using hardware for Background Elimination is the system bandwidth. As mentioned above, surveillance cameras usually work in a fixed position. If we can eliminate the background and send the “object-only” frame as the “current frame” prior to H.264 encoding, and of course merge the reference background after the “object-only” frames are decoded, the frames should remain the same, while saving tremendous bandwidth. All we need to do is send our reference background frame before the decoding occurs. For example, we can send it as a single I frame at the beginning of every 10 minutes of H.264 stream capture.

5.2.1 Experiment Setup

1) Reference Background Image

Figure V-8 shows a reference background image captured by a surveillance administrator. All of the video frames in this experiment were shot via a typical consumer grade video camcorder. Within the frame, there are numerous horizontal lines on the door. These will easily be confused with human hair. There is also a skin-tone switch on the right side and several frustrating towels on the left.

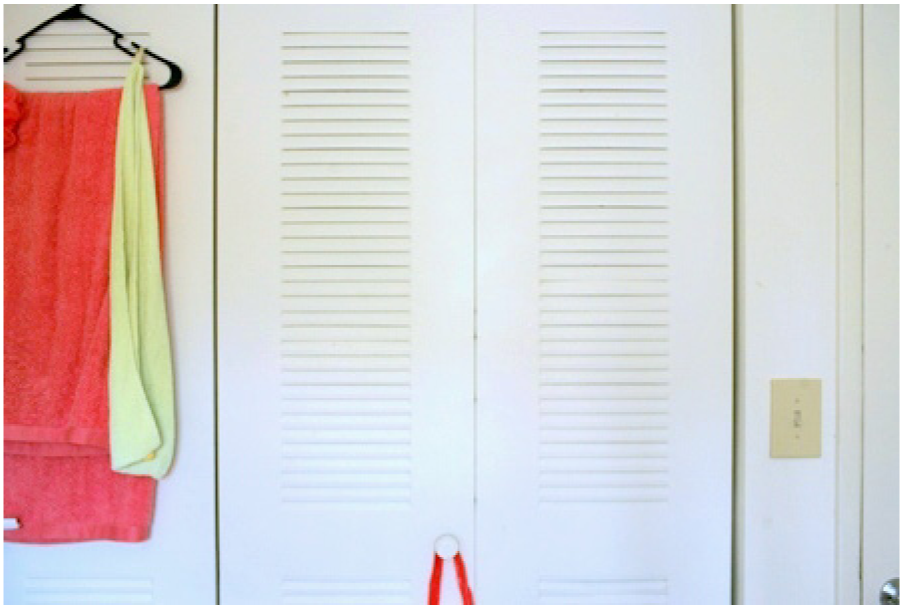


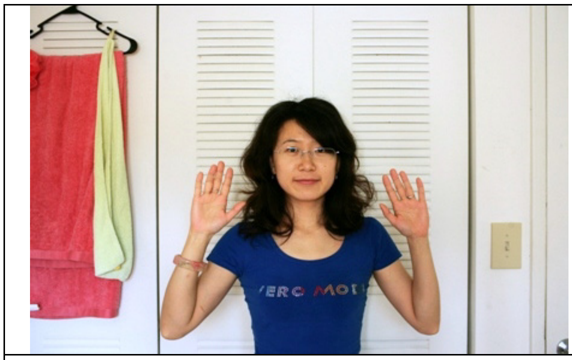
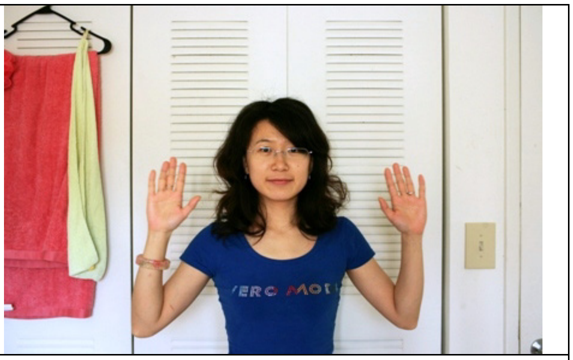
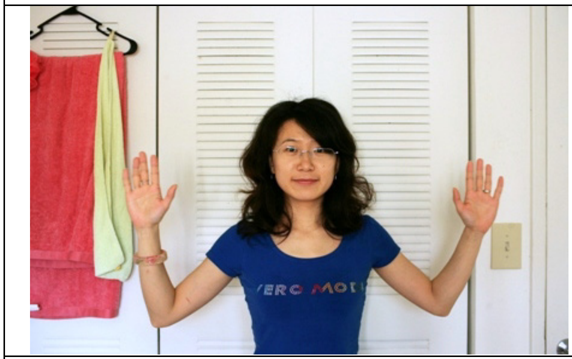
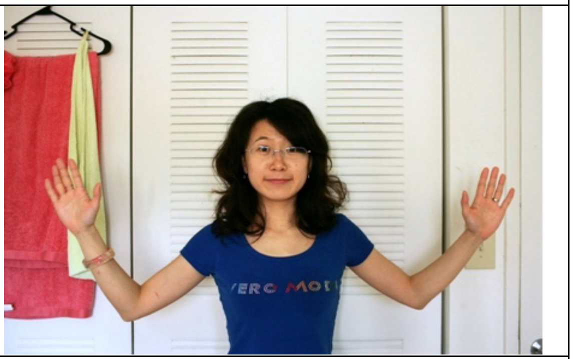
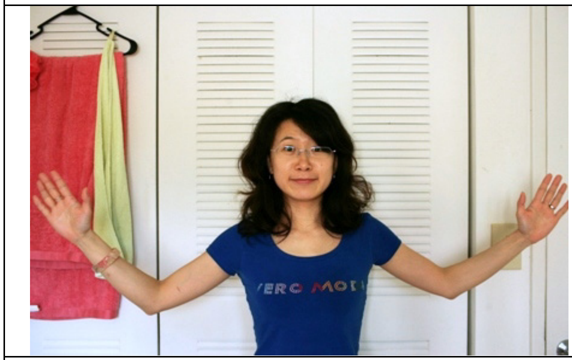
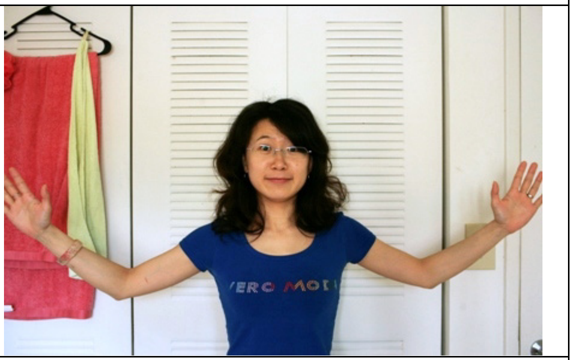
Figure V-8 Captured reference background image

The camera was fixed on a stable tripod throughout the entire experiment. The exposure mode was automatic, and it performed an exposure simulation automatically and compensated for the exposure by using different aperture index numbers and shutter speed combinations. The artificial lighting condition in this room was changed occasionally to simulate the real world environment.

2) Consecutive “current frame” images

Table V-4 shows 6 consecutive frames captured by the camera.

Table V-4 Consecutive Experiment Frames

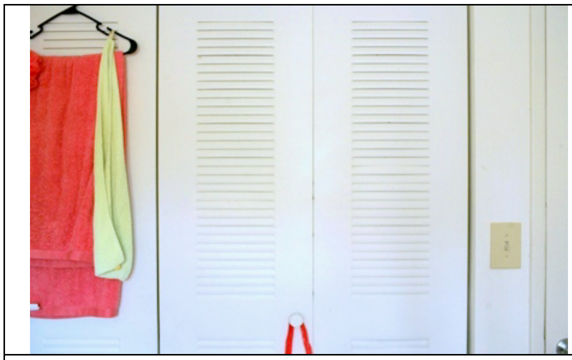
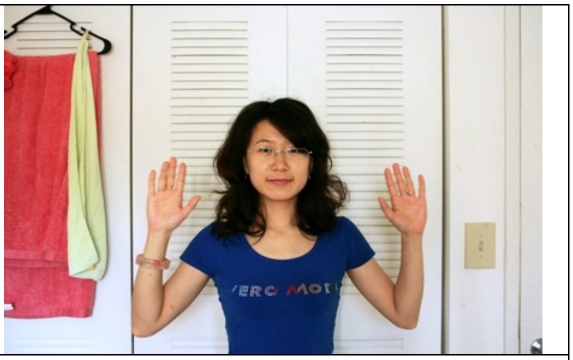
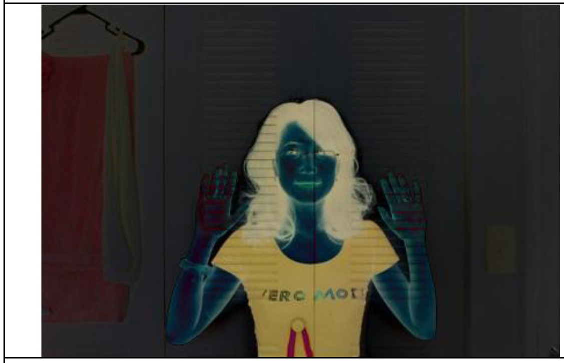
	
Frame Seq. #1	Frame Seq. #2
	
Frame Seq. #3	Frame Seq. #4
	
Frame Seq. #5	Frame Seq. #6

5.2.2 Experimental Results

1) Background elimination

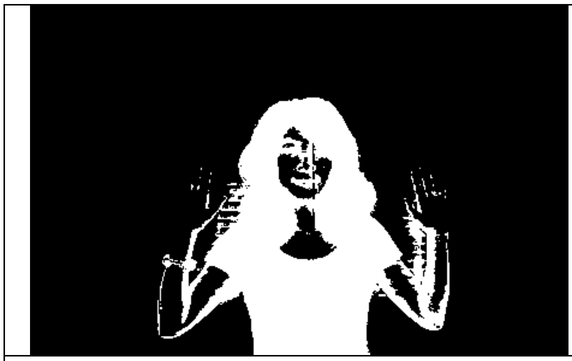
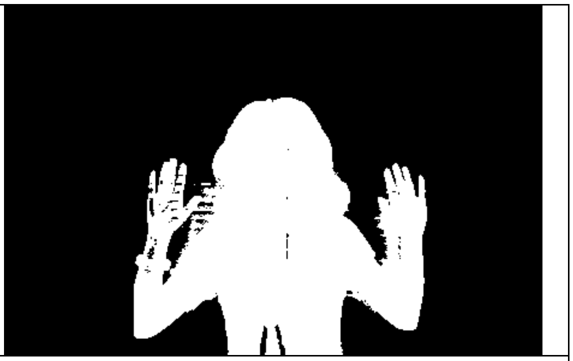
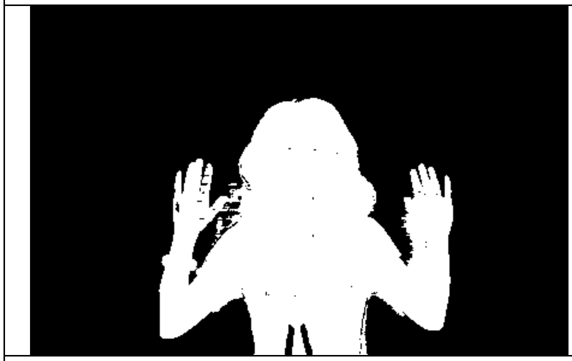
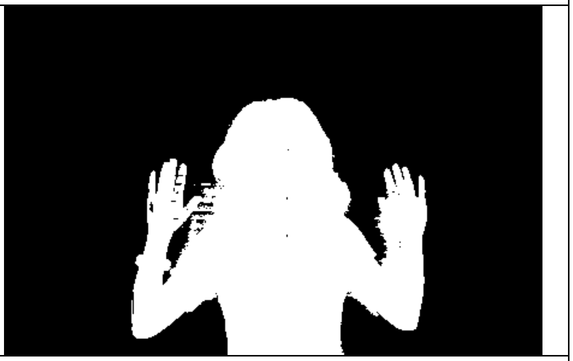
As Table V-5 shows, for the current input frame, the direct subtraction result is not ideal. A 10 × 10 pixel matrix portion of a direct subtraction result shows that the current frame has an average of 45 luminance differences other than the reference background frame. This means the current frame has an exposure lighting issue.

Table V-5 Consecutive Experiment Frames

<table>
<tr><td></td><td></td></tr>
<tr><td>Reference Background Frame</td><td>Current Frame</td></tr>
<tr><td></td><td>
<table>
<tr><td>46</td><td>45</td><td>45</td><td>48</td><td>46</td><td>48</td><td>49</td><td>47</td><td>47</td><td>47</td></tr>
<tr><td>47</td><td>49</td><td>46</td><td>48</td><td>45</td><td>49</td><td>48</td><td>49</td><td>47</td><td>47</td></tr>
<tr><td>45</td><td>48</td><td>49</td><td>46</td><td>46</td><td>48</td><td>46</td><td>48</td><td>47</td><td>46</td></tr>
<tr><td>48</td><td>50</td><td>45</td><td>48</td><td>45</td><td>47</td><td>46</td><td>49</td><td>46</td><td>50</td></tr>
<tr><td>46</td><td>50</td><td>46</td><td>49</td><td>49</td><td>46</td><td>46</td><td>48</td><td>49</td><td>47</td></tr>
<tr><td>49</td><td>48</td><td>49</td><td>46</td><td>46</td><td>48</td><td>47</td><td>47</td><td>46</td><td>49</td></tr>
<tr><td>45</td><td>50</td><td>47</td><td>47</td><td>45</td><td>46</td><td>50</td><td>50</td><td>45</td><td>47</td></tr>
<tr><td>50</td><td>46</td><td>46</td><td>47</td><td>47</td><td>46</td><td>47</td><td>49</td><td>48</td><td>45</td></tr>
<tr><td>49</td><td>46</td><td>47</td><td>49</td><td>50</td><td>49</td><td>48</td><td>48</td><td>47</td><td>48</td></tr>
<tr><td>47</td><td>46</td><td>49</td><td>49</td><td>49</td><td>47</td><td>47</td><td>46</td><td>47</td><td>48</td></tr>
</table>
</td></tr>
<tr><td>Direct Subtraction Results</td><td>10 × 10 background differential matrix</td></tr>
</table>

By compensating for the differential light, the exposure of the current frame was finally normalized with the reference background frame. Then, R, G, B channel subtractions were performed, and each channel generated an object mask. By merging them, we got the object mask. Using this object mask, the original sequential frame became an object-only frame (Figure V-9).

Table V-6 Consecutive Experiment Frames

	
Red channel mask	Green channel mask
	
Blue channel mask	Merged final object mask

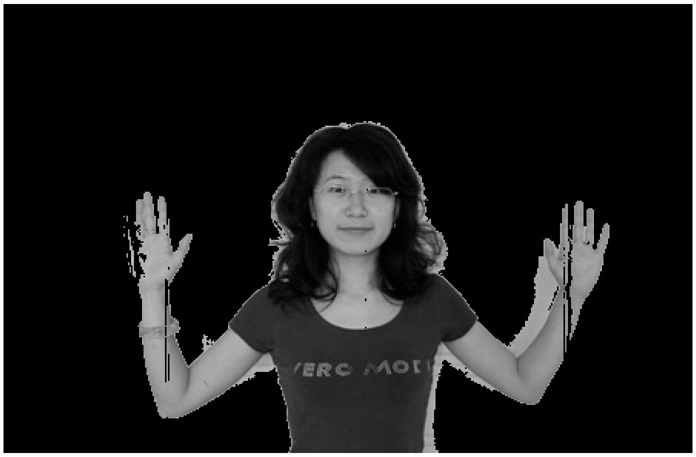


Figure V-9 Background eliminated object-only luminance frame.

2) Motion Detection after Background Elimination

Table V-7 Sequential Object Masks

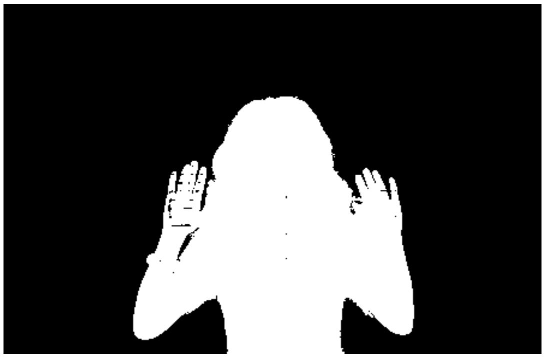
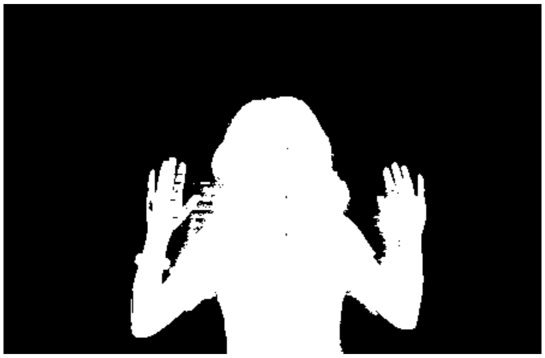
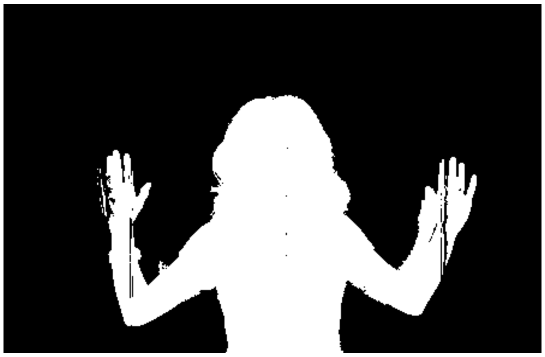
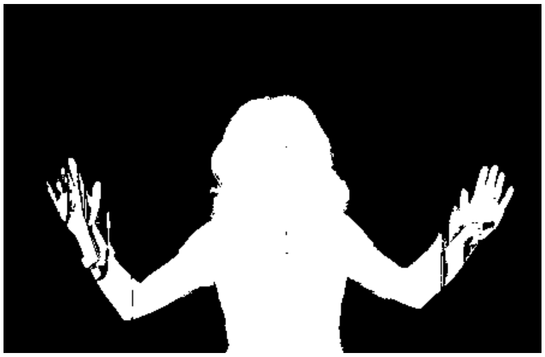
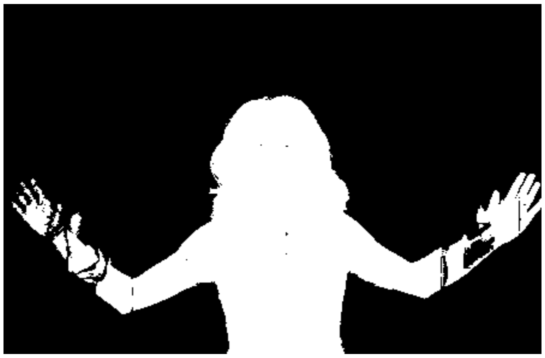
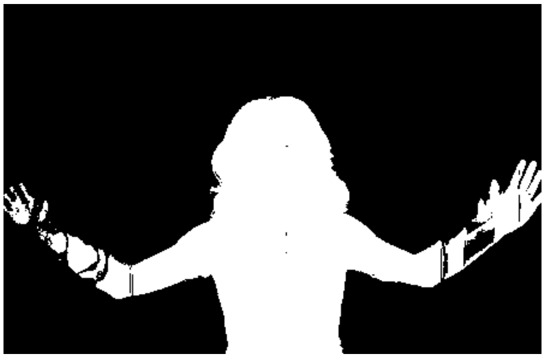
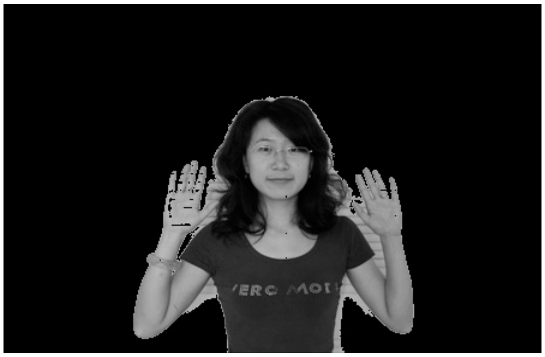
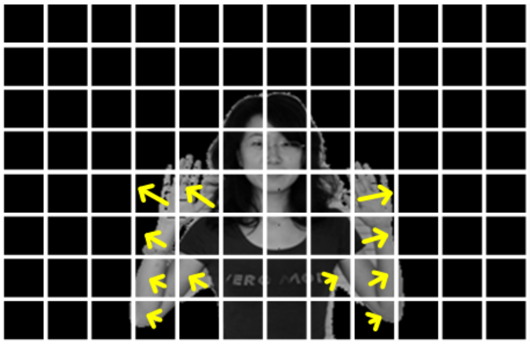
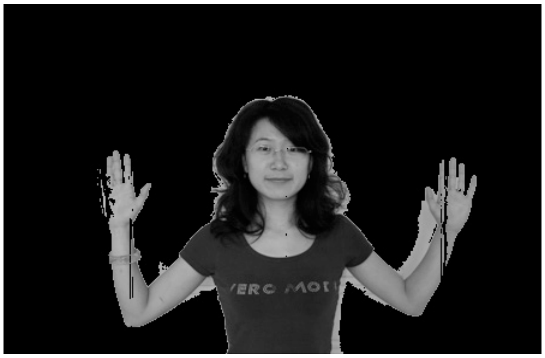
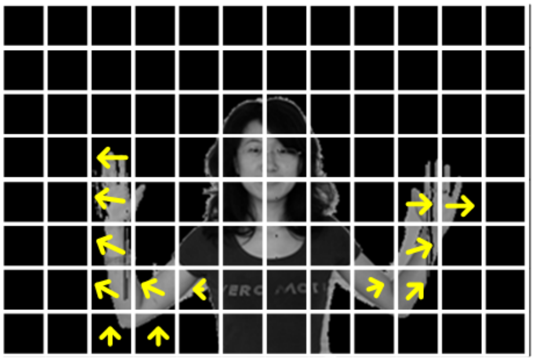
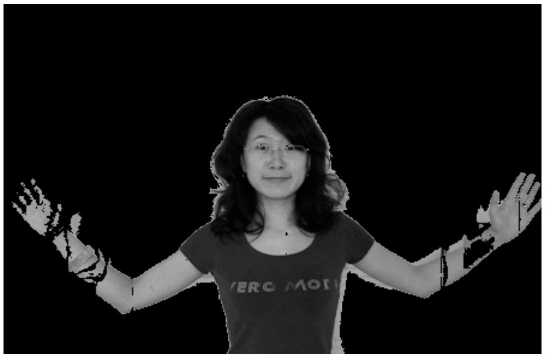
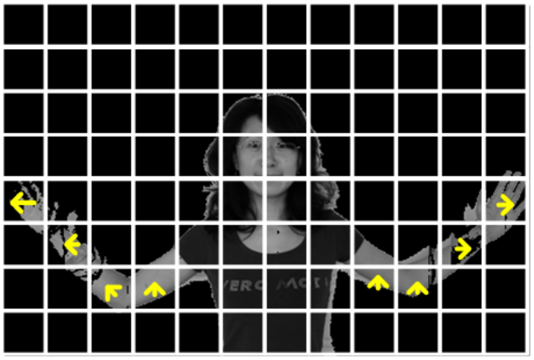
	
Frame Seq. #1 Object Mask	Frame Seq. #2 Object Mask
	
Frame Seq. #3 Object Mask	Frame Seq. #4 Object Mask
	
Frame Seq. #5 Object Mask	Frame Seq. #6 Object Mask

Table V-7 shows a list of object masks generated by the background elimination module. By applying the object mask to the original sequential frames, object-only frames will be generated. These frames will be sent to the ME module, and accurate motion vectors will be easily detected there. Table V-8 shows a list of sequential object-only frames and the motion detection results from the ME module.

Table V-8 Sequential Object-only frames and Motion Detection Results

	
Frame Seq. #1 Object Only Frame	Frame Seq. #1 Object Only Frame
	
Frame Seq. #3 Object Only Frame	Frame Seq. #3 Motion Detection
	
Frame Seq. #5 Object Only Frame	Frame Seq. #5 Motion Detection

3) H.264 compression improvement after Background Elimination

Figure IV-10 shows the experimental setup for a background elimination-based H.264 compression analysis. In this experiment, the original video sequence (Table V-4) will be compressed in H.264 as reference compression data. Then, as Figure IV-10 shows, the same video stream frames will be sent to the background elimination module. As the results of the background elimination algorithm, the "object-only" frames will be considered as standard input, and H.264 will be used as with the original video sequence. Finally, we compensate for the original background using the output of the compressed video, and get the real "decoded" video frame stream.

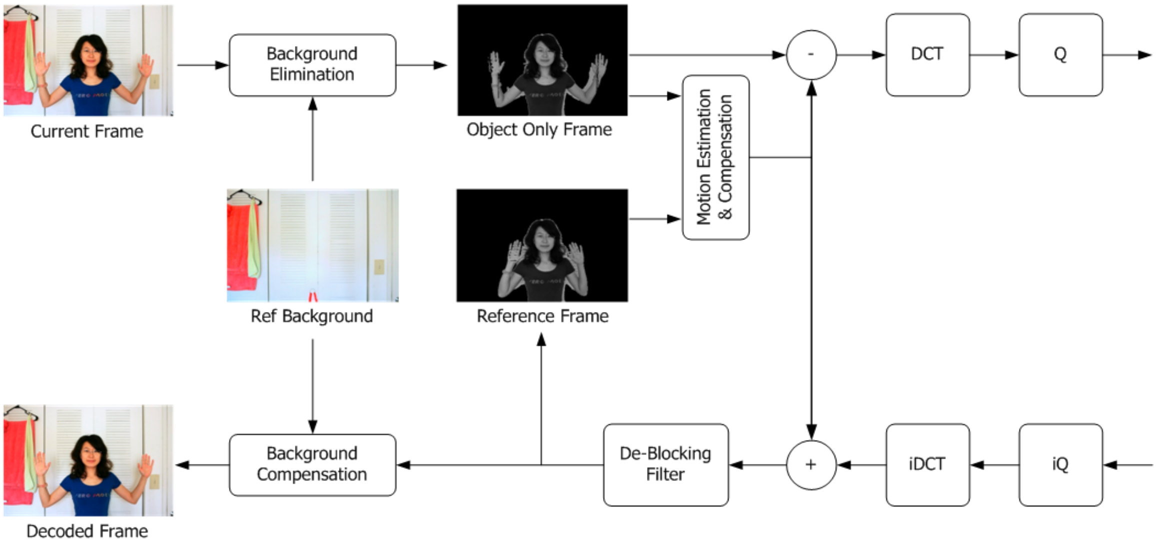


Figure V-10 Background Elimination experiment platform setup

As shown in Figure V-11 and Figure V-12, two video streams were analyzed for this experiment. The figures show the PSNR-Bitrate Maps for both the original video stream compressed in H.264 and the "background eliminated" version of the same video stream compressed using the same H.264.

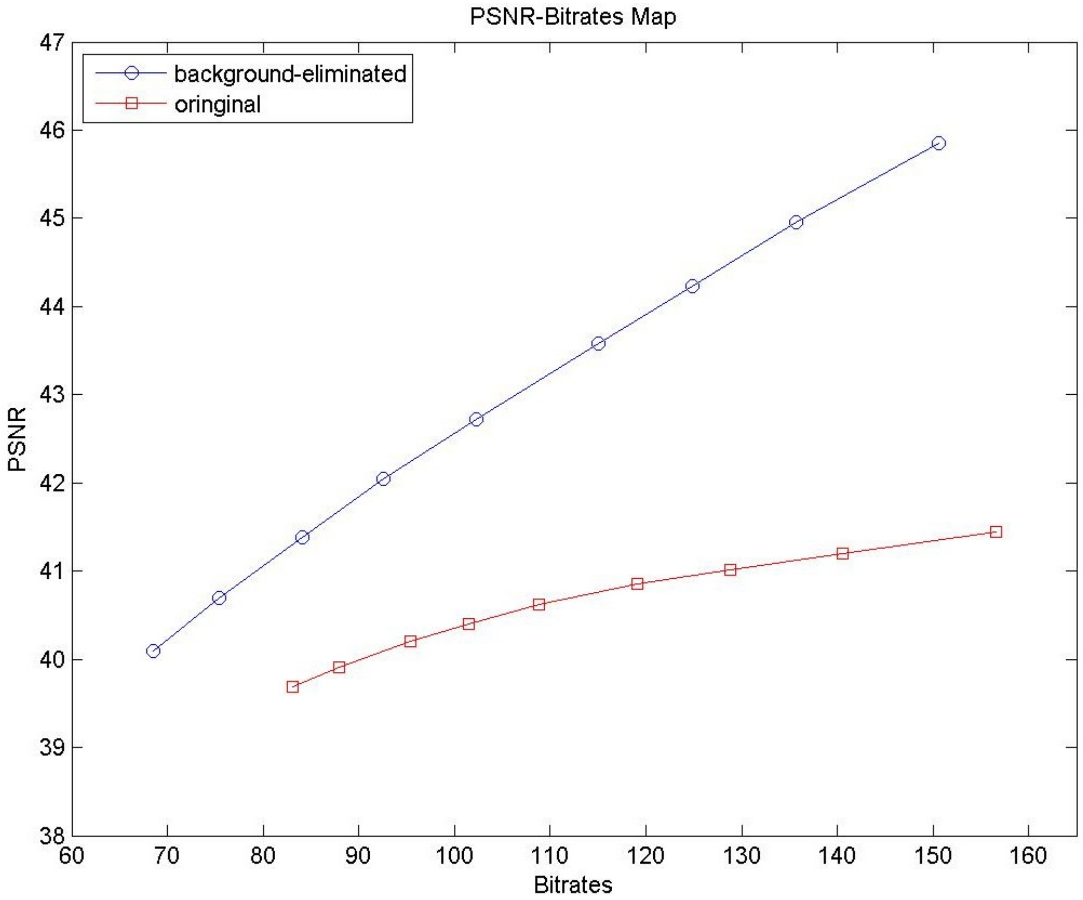


Figure V-11 H.264 compression result of original and Background Eliminated video PSNR-Bitrates comparison I

As can be seen in Figure V-11, the background eliminated video streams show better compression efficiencies than the original video streams. However, this conclusion is not always true. Figure V-12 (a larger object inside video) shows a crossover point where both the original and background eliminated video streams transfer a PSNR = 39.2 dB video at 115 Kb/s. There could be several reasons for this result. The most important one is the object size. If an object is large, the advantages of background elimination will be less. On the opposite side, if the object of the frame is relatively small, the bandwidth can be improved dramatically.

5.2.3 Summary of Experiment

The background elimination module plays a very important role in the whole digital surveillance system. It is the first part of the object recognition system, object categorization system, and motion detection system. It eliminates redundant and useless information from a frame, which will save the system bandwidth dramatically. It isolates the foreground object, allowing the H.264 ME module to obtain more accurate motion results for the following object categorization and recognition systems.

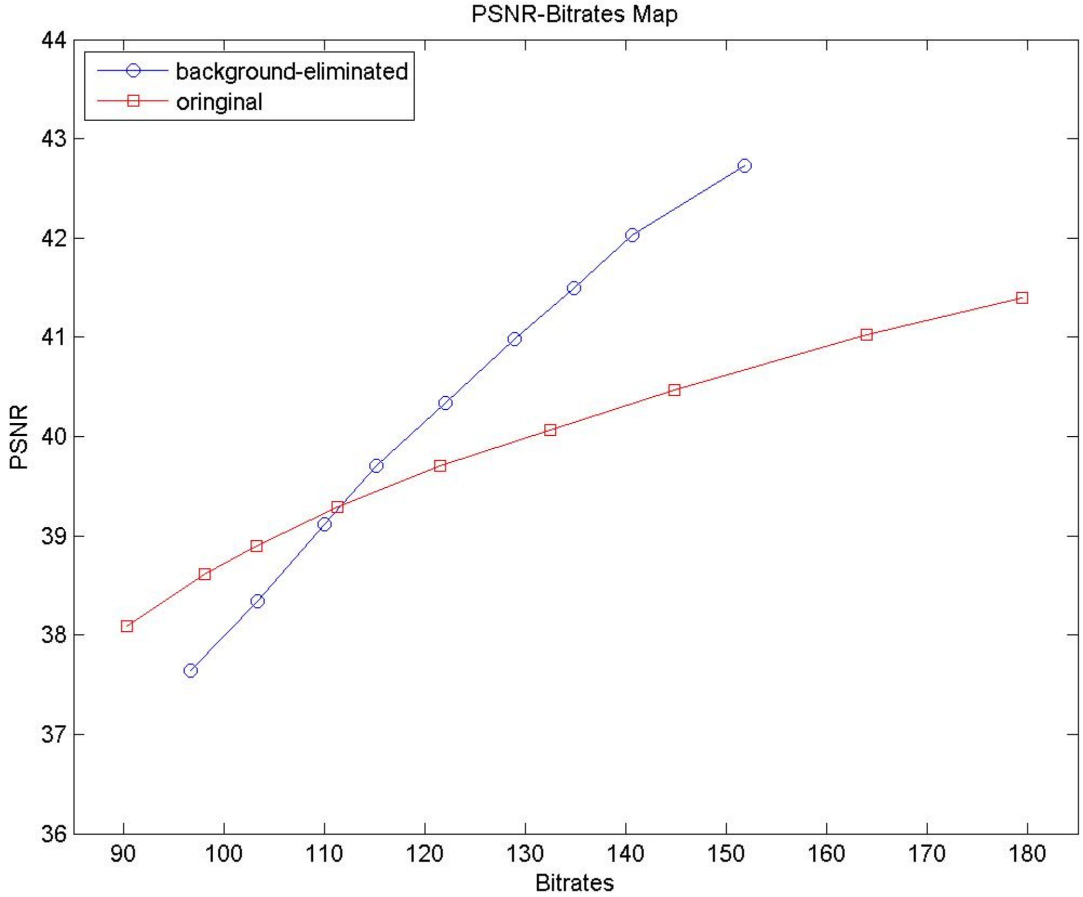


Figure V-12 H.264 compression result of original and Background Eliminated video PSNR-Bitrates comparison II

5.3 Object Recognition System

Different applications may require different object recognition types. As an application-based algorithm, an object recognition system has to have the ability to expand or change its recognition targets and methodologies. Thus, it is better to build the object recognition system in the software platform rather than in hardware modules.

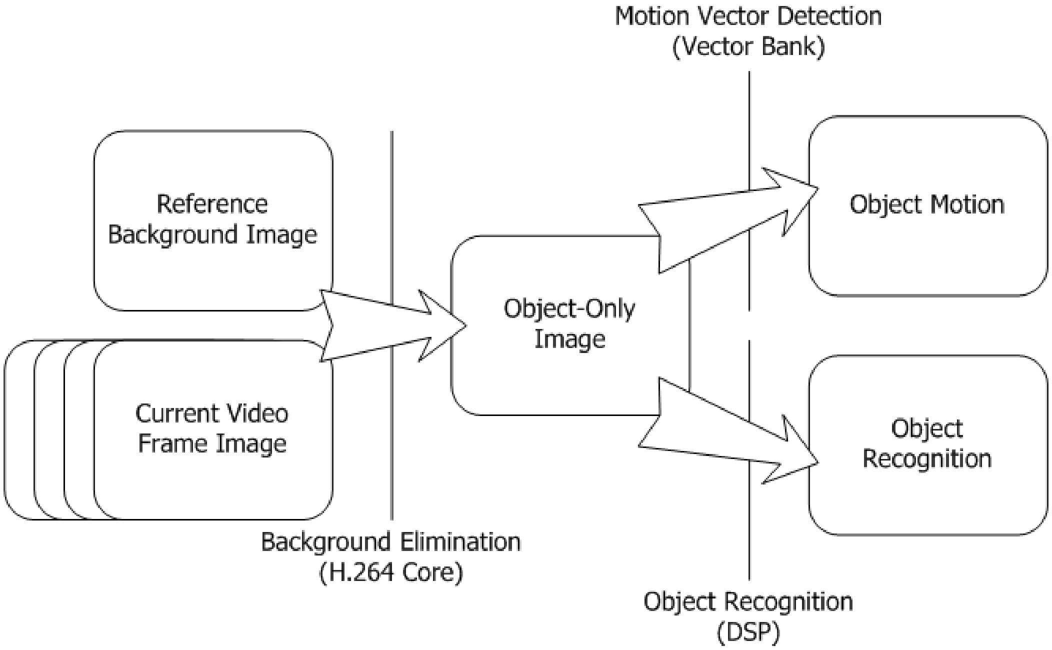


Figure V-13 Object Recognition in surveillance system

In an object recognition system, as shown in Figure V-13, the video data will be processed through the background elimination module, which is inside the proposed H.264 core module, and finally reach the object recognition system, which is within the DSP stage. The object recognition module is independently programmed. Future objects and models can be added by simply updating the database.

5.3.1 System Input

The system input for an object recognition experiment includes the reference background image, a series of frames, and a “to-be-recognized” object database. Again, let us start with the reference background image.

1) Reference Background Image

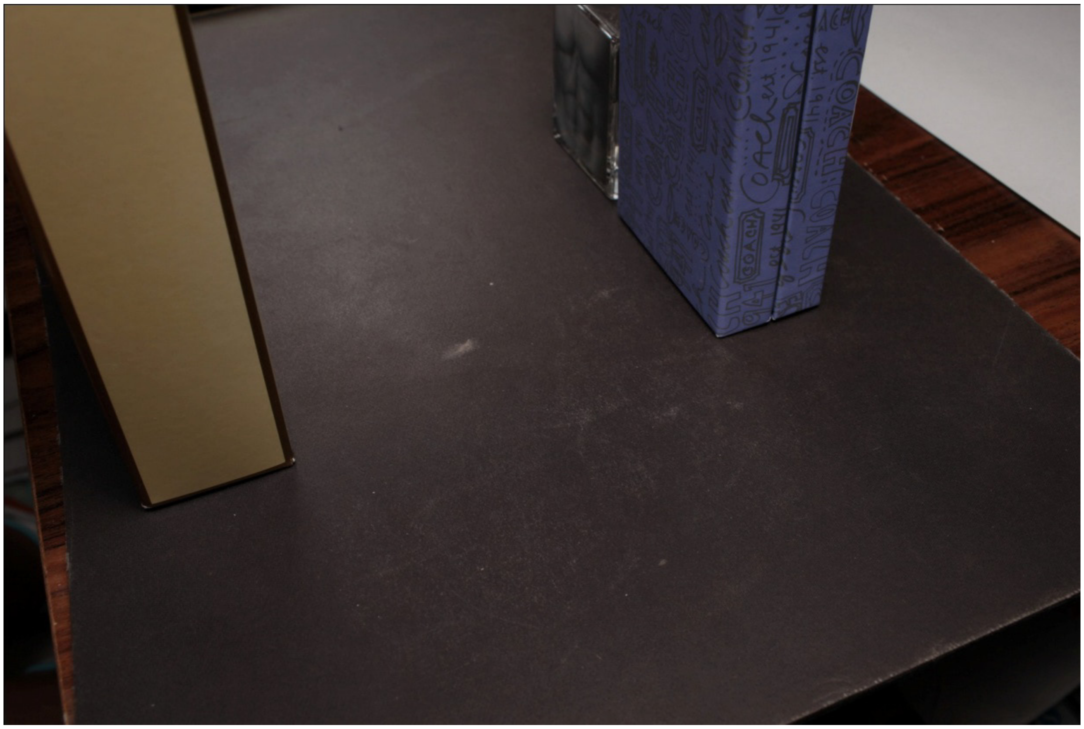


Figure V-14 Reference background frame picture

Figure V-14 shows the reference background frame image. This is a simulated road area. There are two paths crossing at the middle of the frame. The paths have a dark gray color. Some obstacles are placed beside the paths.

2) Object Database

Table V-9 Input database of three different cars

Cadillac	Volkswagen	Honda

Table V-9 shows the input database for this experiment. Three cars have been chosen. Because the cars are 3D models, each of them must have a set of appearances at different angles stored in the database. The real database actually contains a much larger amount of car shape information than is shown in Table V-9.

3) Sequential Frames

Table V-10 shows a series of frames taken when two cars meet at the intersection of the roads. This is a typical scene that occurs frequently.

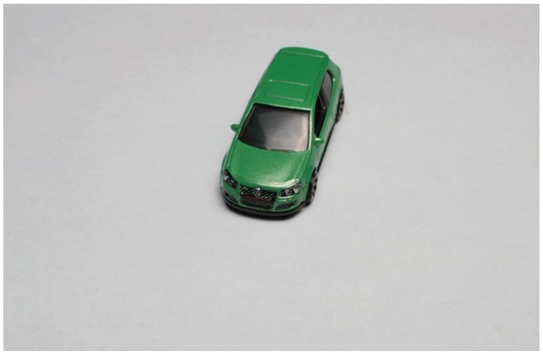
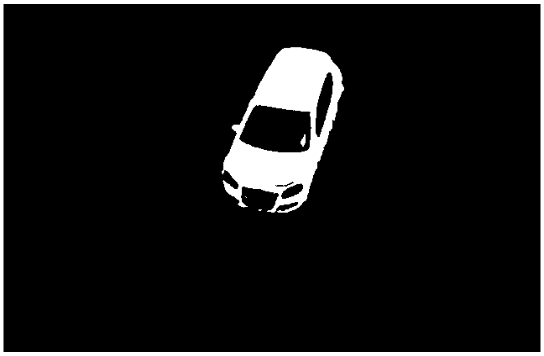
Table V-10 Sequential Frames

Frame Seq. #1	Frame Seq. #2	Frame Seq. #3
Frame Seq. #4	Frame Seq. #5	Frame Seq. #6
Frame Seq. #7	Frame Seq. #8	

5.3.2 Experiment Database Setup

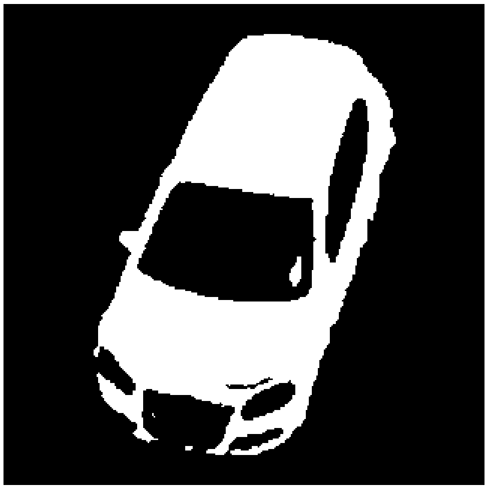
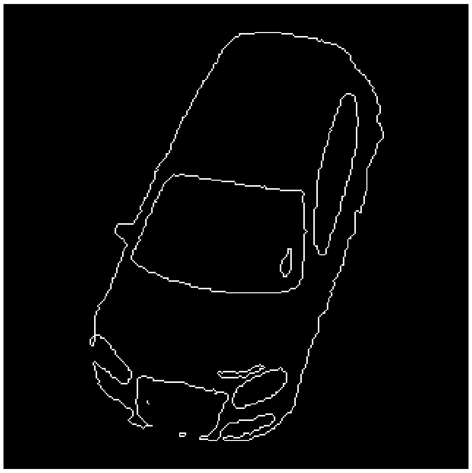
In this experiment, we first need to refine the image-based database into a data value-based database. As discussed in Chapter 2.5.1, the distance-to-angle signature algorithm is a very good algorithm for object recognition. In order to get uniform distance-to-angle signature values, we need to first refine our database pictures.

Table V-11 Object Mask generated by background elimination

The elements from the database are all pre-processed. They do not have to be real-time processed in this case. Let us take one figure from the database as an example. As Table V-11 shows, an object mask was generated by the original database image using the background elimination algorithm. To represent the shape of the object, an object mask figure is more useful than the object-only image.

Table V-12 Centered object image with its LoG edge detection result

After this, we need to center the object and crop the image to a fixed length. This is because the objects from the database might have different sizes. However, in order to recognize the object we need to unify all of the inputs and the database to a single size, as shown in the left figure of Table V-12.

The distance-to-angle algorithm is based on the outlines of the object. Thus, we must determine the outline of the object. The figure on the right side of V-12 is the result of using an LoG edge detection operator to process the left side image.

Finally, to build up the new database, we must determine the distance-to-angle signature values. By scanning 360° from the inner center to outer border of the object edge using different radii (Figure V-15), the angle-to-distance signature of the database object is finally generated, as seen in Figure V-16.

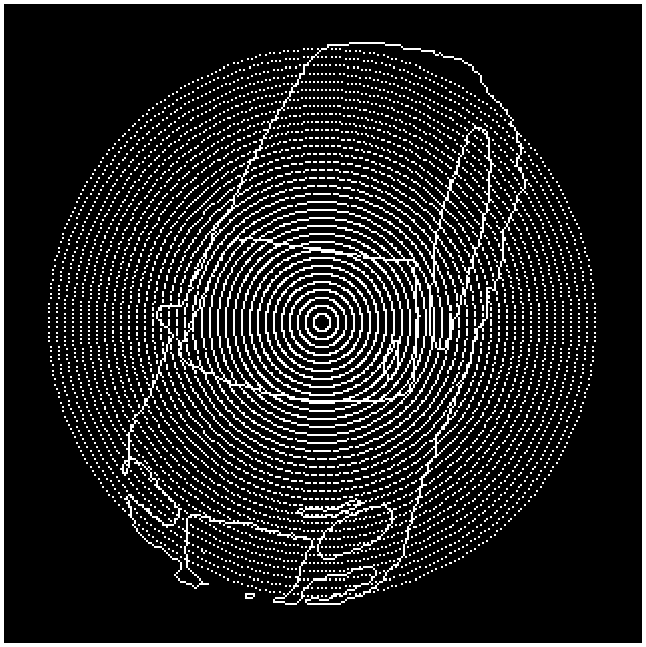


Figure V-15 Multi-Radius Scanning for angle-to-distance signature generation

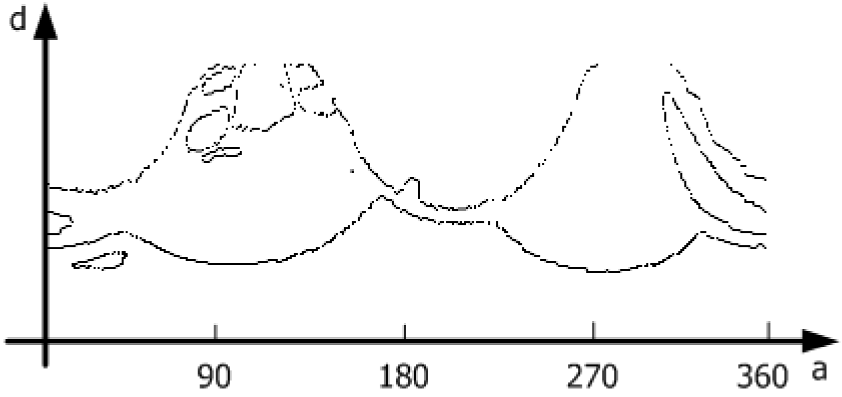


Figure V-16 Generated angle-to-distance signature

By following the same procedure, all of the images in our database can be transformed into a distance-to-angle value set.

5.3.3 Experiment procedures

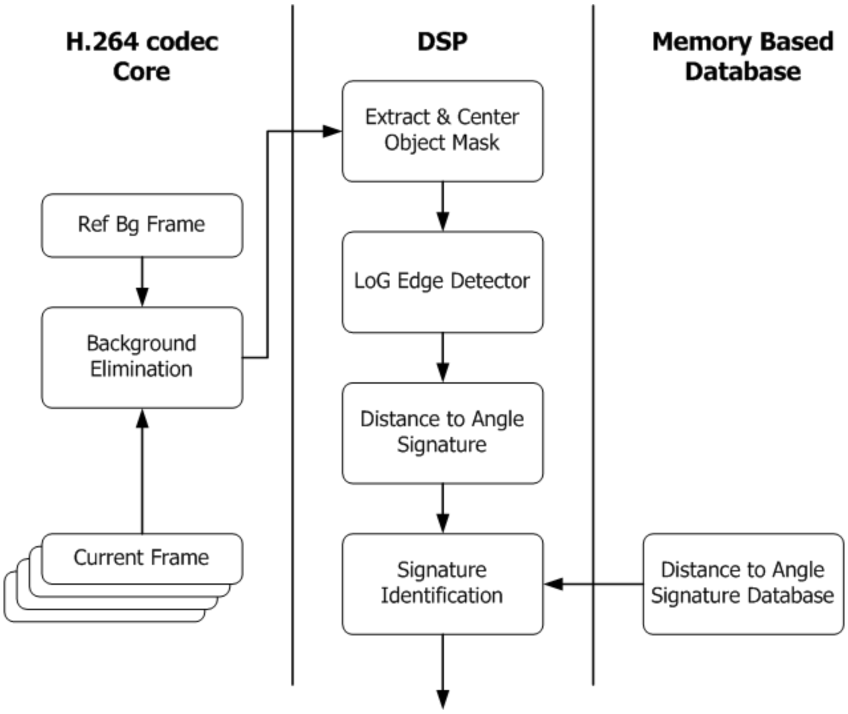


Figure V-17 Distance-to-Angle-based Signature Object Recognition Architecture

Figure V-17 shows a flow chart map of the object recognition experiment. As shown in the figure, object recognition is accomplished by three hardware devices. The first device is the proposed H.264 codec core. Using the same part as the motion detection, the background elimination module generates object mask frames from the current frame and reference background frame. Then, the frame mask is transferred into the DSP module. A series of DSP programs is then applied to the object mask:

1) Centering-Resizing-Extraction algorithm. This procedure analyzes the size and center point of the object mask. Then, it extracts the object portion out of the entire frame as a square image. Finally, it resizes the image to enlarge or shrink it to the original size of the database signature.
2) LoG edge detection operator. This program applies the LoG matrix convolution to the resized square object mask. The outlines will be kept but the context will be removed.
3) Distance-to-angle signature generator. This program resets the origin point of the object image to the center of the object and transposes it into the angle-radius system. Thus, we get the object distance-to-angle signature.
4) Finally, the signature has to go through the entire library to search for matching signatures. The absolute distance of the object signature and database candidate signature will determine which candidate will be the final matched one.

The software process may take longer than the frame rate (e.g. 1/30 seconds) to complete the object recognition. This depends on the ability of DSP, the efficiency of the program, the scale of the database, and numerous other conditions. However, no frame rate object recognition is required. Once the system has a clear object mask, the DSP can begin processing in an attempt to identify the object totally asynchronously with the coding process back in the H.264 codec core. On the other hand, in the future, if more proposed modules are implemented in the hardware, it might also be possible to keep the object recognition as a frame rate design.

5.3.4 Experiment Results

1) Background Elimination

Table V-13 Background Elimination Result

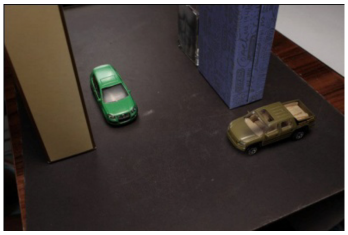
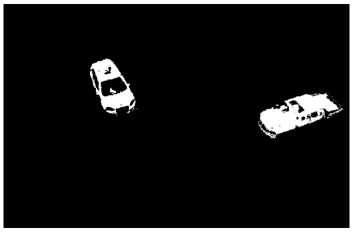
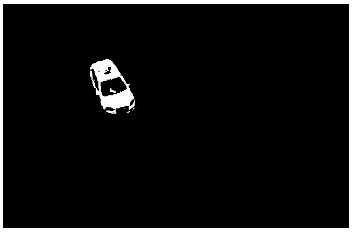
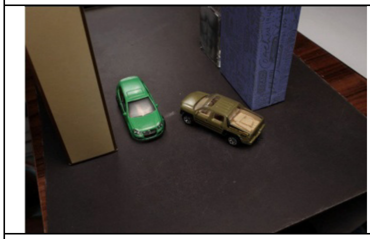
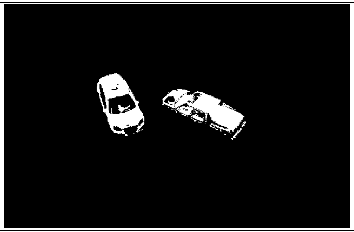
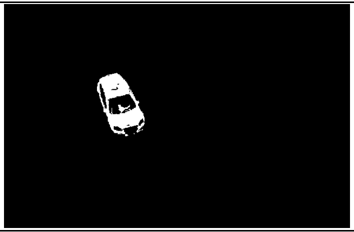
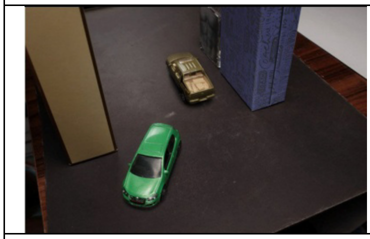
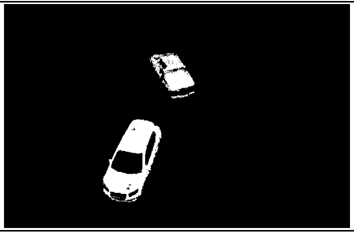
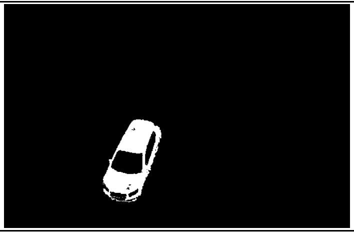
		
Frame Seq. #1	BG elimination	Object isolation for VW
		
Frame Seq. #4	BG elimination	Object isolation for VW
		
Frame Seq. #7	BG elimination	Object isolation for VW

Table V-10 shows a series of captured sequential frames as our test case. In this case, we have two objects, a Volkswagen GTI and a Honda truck. Obviously, we have to recognize one object at a time. Therefore, an object isolation program will be used and will finally leave only one object (the Volkswagen this time) in the object mask frame (Table V-13).

2) Centering-Resizing-Extraction algorithm results

Table V-14 Centering-Resizing-Extraction algorithm results

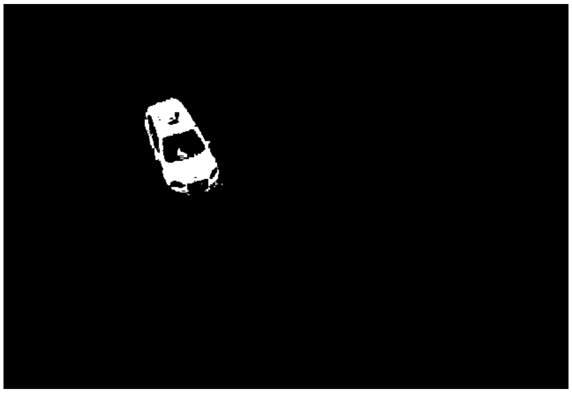
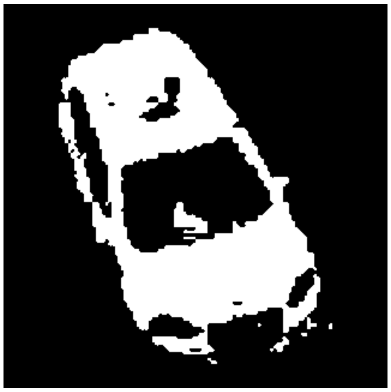
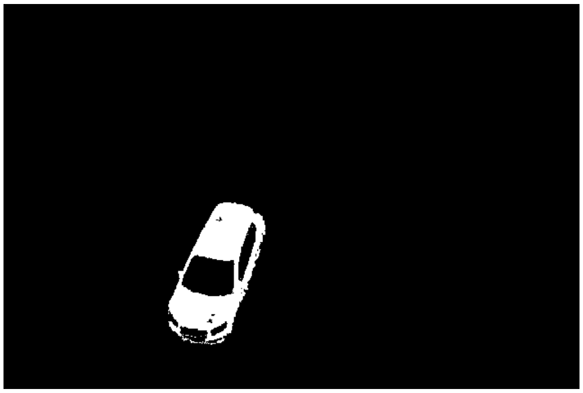
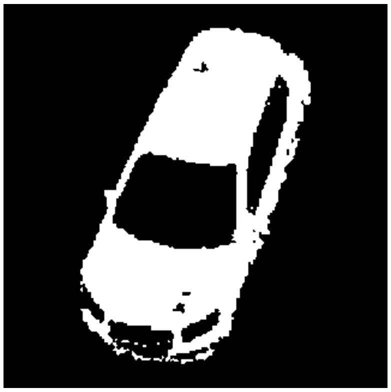
	
Frame #1 Object mask from BG Elimination	Centering-Extraction Result
	
Frame #7 Object mask from BG Elimination	Centering-Extraction Result

Table V-14 shows the experiment results of the Centering-Resizing-Extraction algorithm. The object has been extracted, centered, and enlarged into a 320 × 320 image block. The image quality is relatively low because the real object in the frame (480p) is not large enough for the experiment. HD cameras will offer better results in this case.

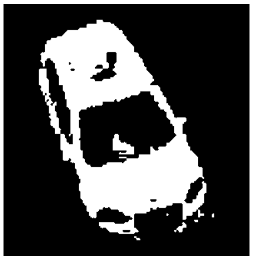
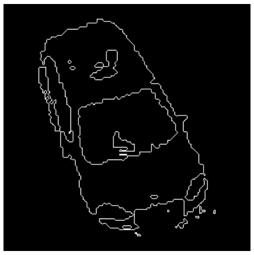
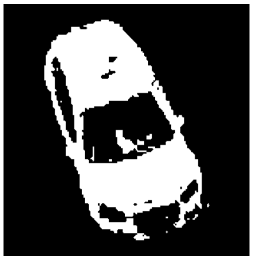
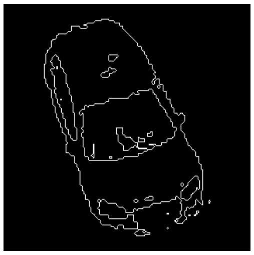
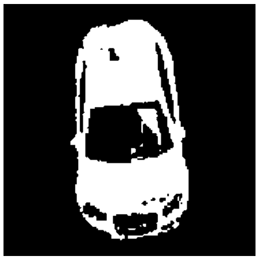
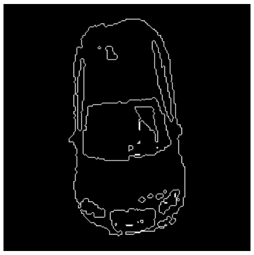
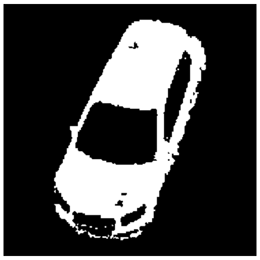
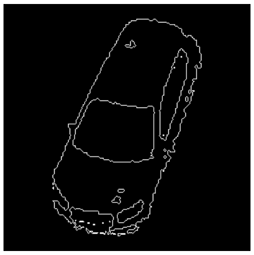
3) Laplacian of Gaussian Edge Detector Results

0	-1	0
-1	4	-1
0	-1	0

Figure V-18 LoG 3 × 3 operator

The next step is the Laplacian-of-Gaussian edge detection. The LoG operator is a mathematical 2D matrix. By applying a convolution using the LoG 3 × 3 operator (Figure V-16) through the centered object mask, we can easily get a clean edge of the object. Table V-18 shows the results of the LoG convolution. So far, we have successfully extracted the moving object out of the streaming frame, centered it, resized it, and finally, acquired its outline.

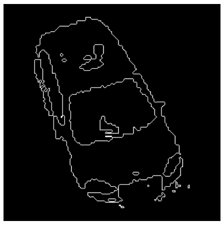
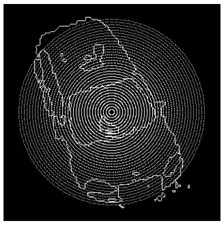
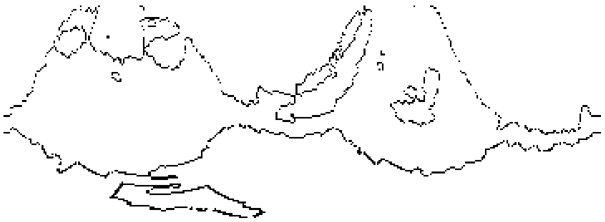
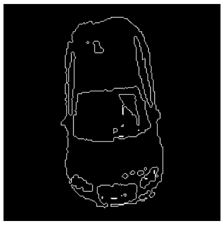
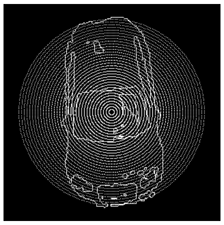
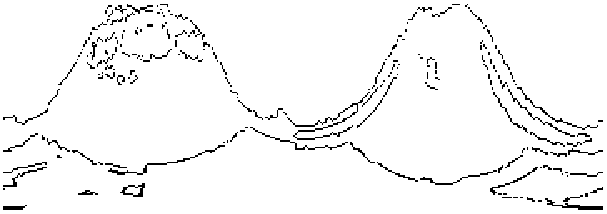
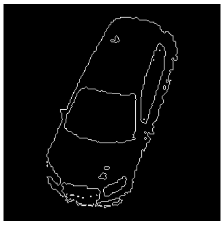
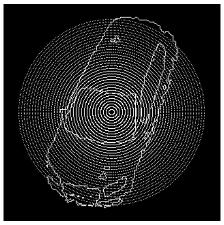
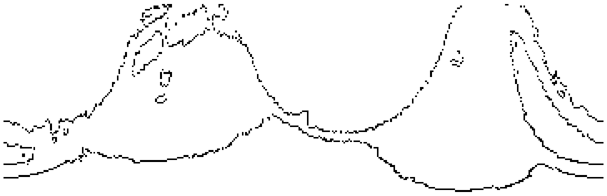
Table V-15 Laplacian of Gaussian edge detection algorithm results

			
#01 Object Mask	#01 LoG Result	#03 Object Mask	#03 LoG Result
			
#06 Object Mask	#06 LoG Result	#07 Object Mask	#07 LoG Result

4) Distance-to-angle signature generator

The Laplacian of Gaussian edge detection result has multiple lines. In order to describe every outline of the object edge on the distance-to-angle signature plane, we need to run circular scanning from the very center of the object to the very edge. Table V-16 shows the whole procedure of generating the distance-to-angle signature results from scanning an edged object mask.

Table V-16 Distance-to-Angle Signature Generator Result

		
Frame #01 Edge	Radius Scanning	Distance-to-Angle Signature Result
		
Frame #06 Edge	Radius Scanning	Distance-to-Angle Signature Result
		
Frame #07 Edge	Radius Scanning	Distance-to-Angle Signature Result

5) Signature Matching Algorithm

The object recognition of the system is based on a pre-loaded database. The distance-to-angle signature is an algorithm that could transform the object into an orientation and size independent pattern. As described in Chapter 2.5.1, the key for the distance-to-angle signature is the pattern differences. Basically, if two patterns have very close outline features, they are considered to be the same object.

The analysis is based on the total differences (or distances) between the outlines. Basically, for two patterns, fewer differences mean that they are more identical. We randomly picked the Volkswagen in the 7th frame as our "to be recognized" object. As shown in Table V-17, the green GTI is detected, and we have its signature.

Table V-17 Object to be recognized

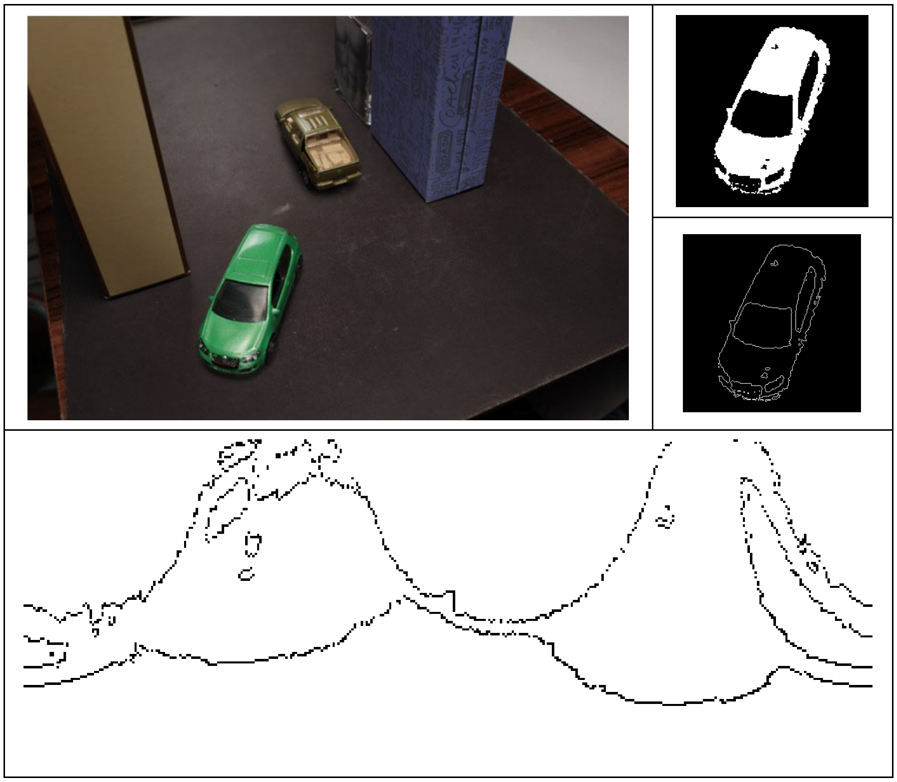
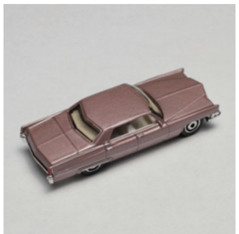
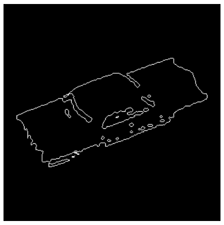
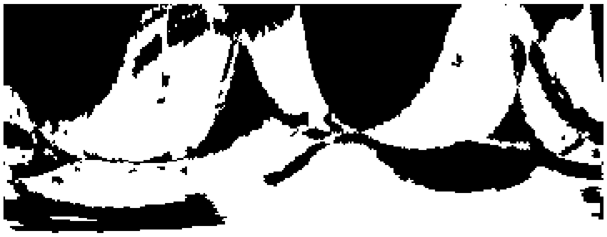
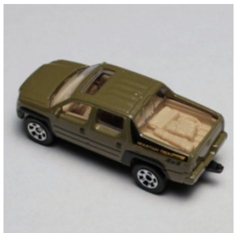
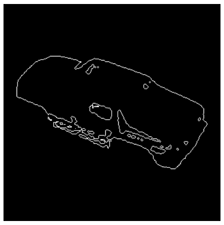
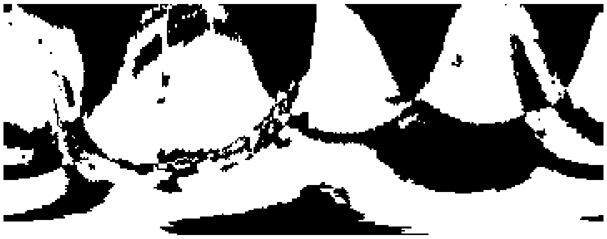
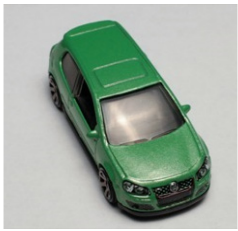
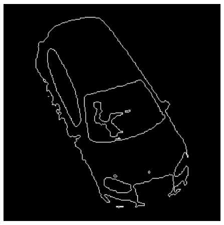
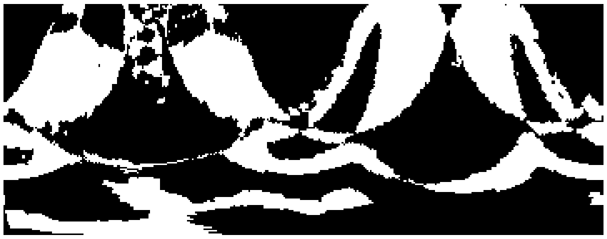
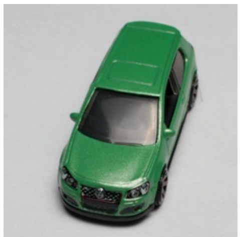
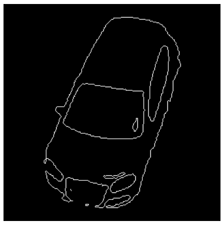
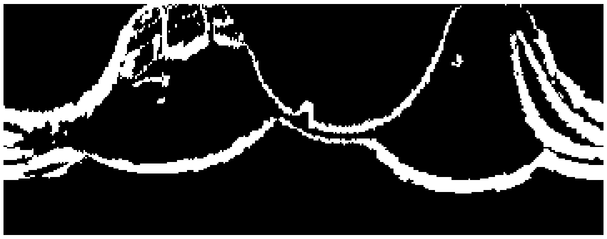


Table V-18 shows a number of candidates in the database, along with their signature differences (highlighted areas) from the GTI. It is very obvious that the last candidate from the database has the best match with the real object in the scenes (less than 10% difference).

Table V-18 Distance-to-Angle Signature Difference (Distance)

		
Database 04-7	Object Edge	Signature Distances (70% difference)
		
Database 08-2	Object Edge	Signature Distances (70% difference)
		
Database 02-2	Object Edge	Signature Distances (50% difference)
		
Database 02-13	Object Edge	Signature Distances (less than 5% difference)

5.3.5 Summary of this Experiment

In this experiment, we successfully recognized the object in real scenes from the database we input earlier. By analyzing the distance-to-angle signature of the object, we can easily avoid the orientation and size differences, while focusing on the pattern differences. On the other hand, there are other ways to recognize the signature other than by just calculating the differences. The second-derivative gradient of the outline curve could also be used to recognize patterns.

5.4 Object Categorization System

Sometimes the database used is extremely large. It would be a waste of time to run the pattern recognition for each of the objects. In this situation, an object categorization system might be needed to obtain the category of the object prior to performing the object recognition.

As discussed in Chapter 4.2, the best way to implement an object categorization system is the use of the BP ANN system. However, for the BP system, we need to train our input and output data first.

5.4.1 System input and output training

Although it is true that an object recognition system is a special object categorization system, their algorithms are quite different. In an object recognition system, we prepare a system algorithm and recognition candidate database. If another recognition candidate needs to be considered in the system, the only thing we need to do is update the database. Yet, things are quite different in an object categorization system. By using the BP ANN algorithm, the category "candidates" build a relationship with the category algorithm itself. Therefore, if we want to expand the neural network, we must not only expand the database, but also re-train our network.

Table V-19 Input matrix of object categorization neural network system

Item	Input Matrix
1	Object size
2	Average luminance of object
3	Average color of object
4	Length-width ratio of object
5	Pixel speed (Motion) of object
6	Motion direction of object

The BP ANN system input can be very flexible. Moreover, there is not necessarily a "function" between the input and output–the system network itself is a relation function. As a simple example here, we pick several features of an object as the input matrix. Their values are shown in Table V-19.

Before we run the BP ANN, we need to first "train" the network. A group of sample images will be analyzed as system input. However, we cannot input the whole picture because it is too large. We thus need to extract six parameters from each of the sample pictures, as described in Table V-20. Here, the parameters are just for example use. They could be totally different in practical use. Numerous surveillance cameras even have the ability of infrared detection or spectrum analysis, which will give more options.

These parameters are assigned to be the BP ANN training input, and the corresponding category index numbers (binary number) are assigned as the BP ANN training output.

Table V-20 Input matrix of object categorization neural network system

System Input	System Output	
[35, 151, 129, 3.5, 2, 46]	Small cat/dog	[-1, -1, 1]
[28, 25, 112, 2, 12, 24]	Small running cat/dog	[-1, 1, -1]
[74, 40, 23, 4, 6, 11]	Human	[-1, 1, 1]
[108, 37, 66, 7, 24, 56]	Running Human	[1, -1, -1]
[98, 129, 74, 6, 11, 191]	Human walking to the house	[1, -1, 1]
[228, 201, 17, 3, 53, 17]	Car/Truck	[1, 1, -1]
[311, 109, 47, 2.3, 123, 47]	Fast Car/Truck	[1, 1, 1]

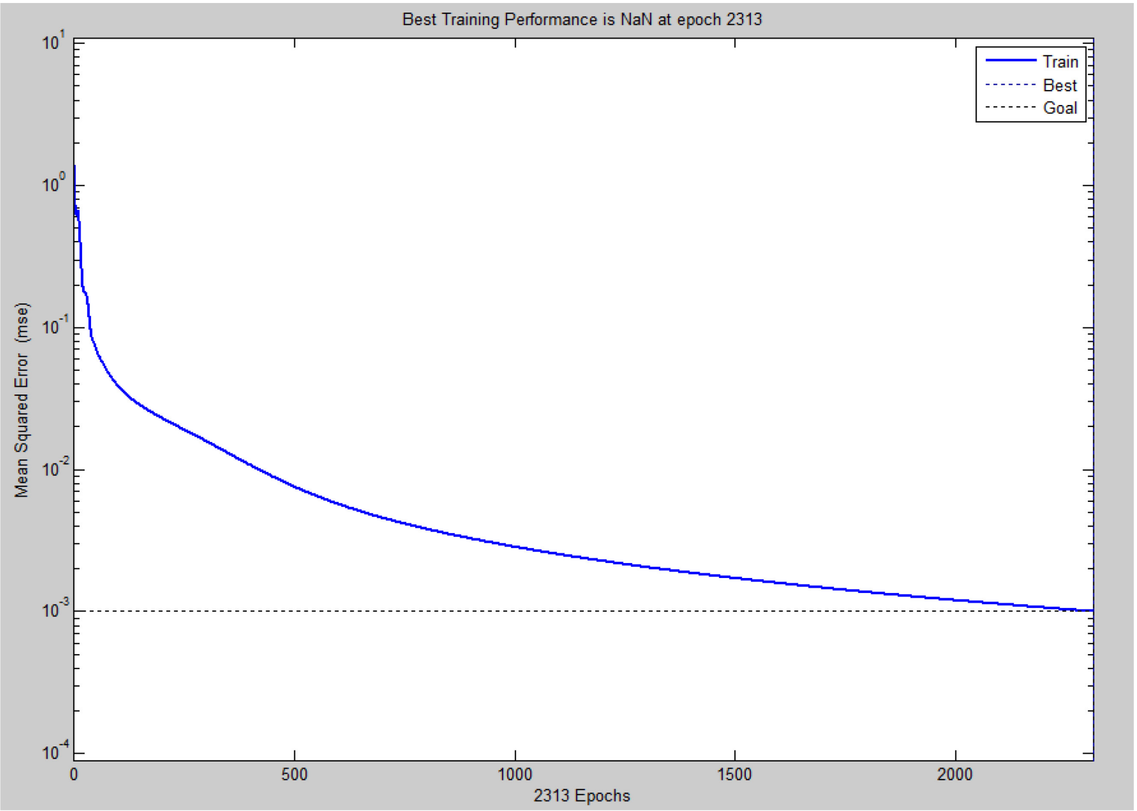


Figure V-19 BP ANN system training

In network training, the BP ANN generates random weights in the hidden layer and corrects them gradually by repeatedly analyzing the system input and output (for multiple epochs). For better result, we set the training Mean Square Error (MSE) to 1E-3 before the training starts. Figure V-19 shows the training process. This typical BP ANN system trains for 2313 epochs.

5.4.2 Experiment results

The results of the experiment were good. Because we did not have a real backyard in which to conduct our experiment, we used several random generators to regulate the system inputs. The random generation functions are shown below in Table V-21.

Table V-21 Input matrix of object categorization neural network system

Categories	System Input Random Constraints
Small cat/dog	[15-35, rand, rand, 0.2–5, 0–10, 0–200]
Small running cat/dog	[15-35, rand, rand, 0.2–5, 10–20, 0–200]
Human	[50-150, rand, rand, 0.1–10, 0–12, 0–190]
Running Human	[50-150, rand, rand, 0.1–10, 12+, 0–190]
Human walking to the house	[50-150, rand, rand, 0.1–10, rand, 190–200]
Car/Truck	[150-500, rand, rand, 0.1–10, 0–100, 0–200]
Fast Car/Truck	[150-500, rand, rand, 0.1–10, 100+, 0–200]

Then we used this generator to generate 10000 input sources and input them into our network to get the results. The results showed that if the data were closer to some input value, there was a better hit-rate (more than 99%). However, if the data were closer to the border of two categories, the hit-rate dropped quickly (less than 70%).

Because the BP ANN has the ability to learn, it will not be a big problem if the categorization system does not operate very well in the beginning. We can always input error data pairs at the training stage to improve the network.

5.4.3 Summary of this experiment

The experiments showed the basic steps to build and run an object categorization system based on the BP ANN algorithm. BP ANN is easy to implement and use. It is a great solution for an object categorization system. The category results are good and can be improved in the future using additional pairs of inputs and outputs to train it.

VI CONCLUSION & FUTURE WORK

6.1 Concluding Remarks

This dissertation described a digital surveillance network architecture based on the H.264 codec core. As mentioned before, more and more analog devices are being digitized today, including some surveillance networks. Because most of these have only digitized the storage stage, they are not full digital systems. A complete digital system employs digital signal from the sampling module at the beginning to digital signal analysis and transfer within the process and digital signal storage at the end. Finally, the fundamental change from analog signals to a digital data process will have a very big impact on the surveillance network industry. With its numerous unmatched advantages, digital surveillance network architecture will take the lead and becomes the main trend in the industry in the next 20 years.

The proposed digital surveillance network architecture includes two major parts: the proposed digital surveillance network and the proposed digital surveillance node as the foundation of the digital surveillance network.

6.1.1 Advantages of Digital Surveillance Network Architecture

Conventional surveillance network architecture has a centralized control center called a secure room. All of the surveillance equipment has dedicated wires to send data back to this secure room. The proposed digital surveillance network architecture has overthrown this traditional approach. It is a "distributed" architecture composed of numerous independent surveillance nodes. Each surveillance node has the ability to compress, analysis, store, and transmit video data. These surveillance nodes can be organized as clusters to determine what data should be sent. All of the captured analog signals become digital from the first second they enter the surveillance node. They then pass through the H.264 based SoC, which contains a series of digital signal processing algorithms within its hardware and software platform. Video signals will be analyzed and processed inside the surveillance node, and it will never be necessary for "security guards" to constantly watch a huge number of monitors. In summary, the advantages of digital surveillance network architecture are as follows.

1) Home-based surveillance network becomes possible.

Compared to the conventional analog surveillance network, a digital surveillance network will use a large number of high-tech image processing algorithms. This removes the dependency on security guards from a real-time surveillance system and makes a home-based surveillance network possible.

2) Extraordinarily large surveillance network becomes possible.

A traditional surveillance network requires dedicated cables to transfer the captured video stream. In contrast, a "distributed" surveillance network has the ability to analysis video in every single surveillance node. A group of surveillance nodes can be organized as clusters to determine what data should be sent. Only "interesting" data gets transferred, and the amount of data transfer can be very small. Thus, an extraordinarily large surveillance network (e.g., a nationwide or even global surveillance network) becomes possible.

3) The cost of building a huge surveillance network becomes very low.

Instead of dedicated video cables, a digital surveillance network will use well-established Local Area Network (LAN) technology to set up surveillance clusters. Thus, the building, re-building, and upgrading costs for the network, as well as the cost of routing and data transfer, become very low.

4) System security becomes higher.

A traditional surveillance network is "secure room" centered architecture. A large amount of money must be spent to ensure the security of this room. The new digital surveillance network architecture will not have such a room. Every single surveillance node has its own entropy-coded security system. Even if an individual surveillance node is compromised, there is no threat to the entire system because every cell has a different code. The only way to control the system is to actually have the ability to control or compromise every single surveillance node, which is likely to be impossible.

5) Data transfer security becomes higher.

As mentioned above, the "video data" sent by a digital surveillance network will be encrypted H.264-coded digital data. An advanced encryption algorithm ensures that the data content cannot be interpreted without the encryption key. Compared to analog data, digital data transfer is safer.

6) Data storage security becomes higher.

Just as with system security, centered data storage can bring a very high cost to secure the data. Once the system is compromised, all of the data is compromised. At the same time, a distributed data storage system is more efficient for data backup and data transfer. Even if some portion of the system storage nodes is compromised, this will not have a very big impact on the security of the entire storage system.

6.1.2 Digital Surveillance node Design

Compared to the conventional surveillance network architecture, the proposed digital surveillance network architecture has numerous advantages. These advantages all come from the distributed surveillance node design, which is basically based on two platforms: the improved H.264 SoC hardware platform and the DSP-based software platform.

1) H.264-based SoC Hardware Platform for Surveillance node

The H.264 SoC platform in the surveillance node design was developed from the original H.264 codec platform, which is used as a hardware encoder/decoder. With two major changes, we implement it into a platform for a digital surveillance network.

The first major change is that we implement a DSP/Memory platform in the main bus of the SoC platform. This becomes the hardware support system for our software architecture. The advanced “object recognition” and “object categorization” algorithms run on this platform.

The other major change is the H.264 codec core. We implement two new modules in the H.264 codec core: the background elimination module and DDCT module.

- By analyzing a reference background frame and the lighting condition for the current background, the background elimination module can eliminate the background from the image. The "object-only" frame is then compressed by the H.264 module. Because the background information is eliminated, the compression efficiency is 20–30% better than the old way. At the same time, as object feature information, the "object-mask" frame will be sent to DSP for further analysis.
- DDCT is based on 2-D DCT. By adding more modes (directions), it performs better compression in different directions. By being specially designed for the H.264 algorithm, the improved DDCT can compress 16 × 16 pixel blocks using 12 modes. Using the improved DDCT, the compression efficiency is 5~10% better.

2) DSP-Based Software Platform for Surveillance node

One of the most important features of the digital surveillance network architecture is its imaging analysis ability. This will eliminate the need for a human presence in a RT surveillance system. The true hero is our new DSP-based software platform. This platform is based on two parts: an object recognition system and object categorization system.

- By analyzing the object shape mask coming from the background elimination module, the object recognition system generates an object distance-to-angle signature. This signature stores the essential shape information for the object despite of its size and orientation. The object recognition system then compares this signature to those found in the system database, and finally finds a matching object.
- The object categorization system is built using a BP ANN model. We first define a set of feature data (input) and categories (output). Then, we train the BP ANN model before using it. By setting an MSE value and training it over multiple repetitions, the randomized weight number starts to become stable and the network starts to "understand" the category system. Then, we use the network to identify real time objects, compare the results, and upgrade the database to refine the network we built. Finally, we will get a trustable network capable of categorizing objects correctly.

Finally, as a conclusion for this dissertation, Figure VI-1 shows the top-down look for the entire proposed digital surveillance network.2

6.2 Future Work

This dissertation has discussed a complete digital surveillance network. This has numerous advantages compared to the most popular analog surveillance network today. However, there are also many ways to improve it.

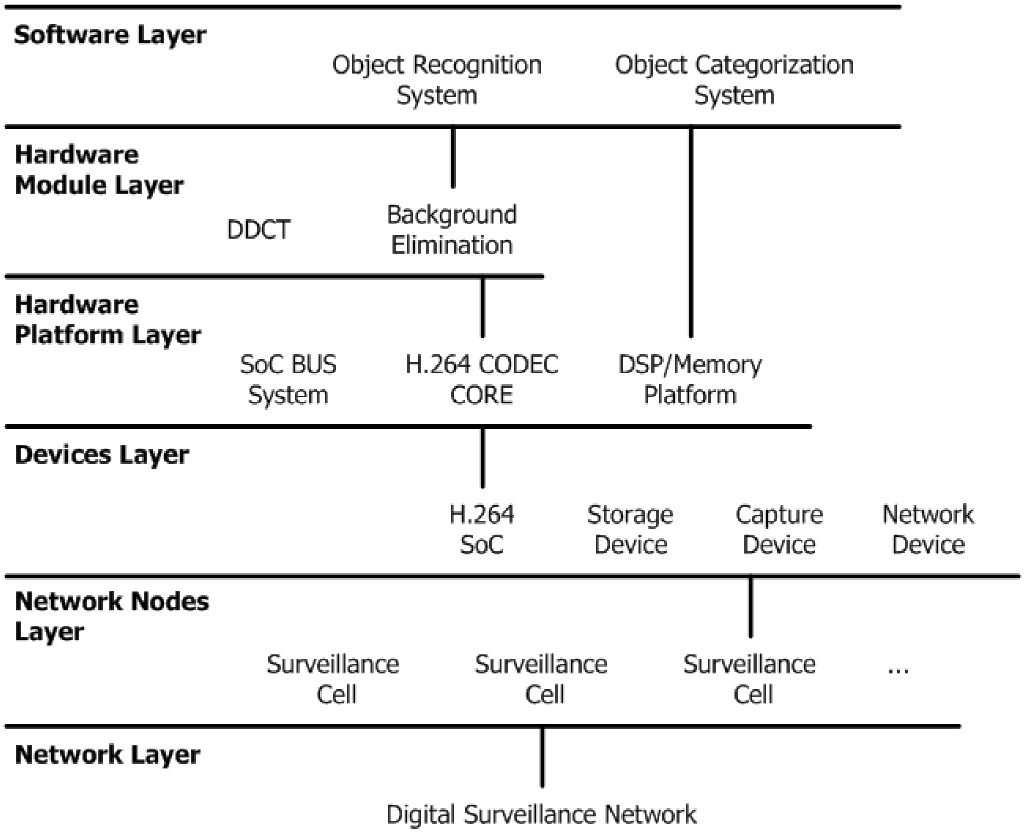


Figure VI-1 Top-Down look for proposed digital surveillance network architecture

- 3-D vision. At present, we use the pixel speed and pixel size to simulate the real speed and real size. Yet, these are not true values but only converted approximations. Things would be improved by implementing a bi-camera surveillance system. Just like 3-D movies, if we can implement 2 camera sensors in a single chip, they would work together to provide a 3-D picture of the real-time object. The height, width, and speed would be real world values rather than just pixel values.
- Taking a step further, we could make the intelligent surveillance node even more intelligent by allowing them to cooperate with each other through a LAN. There would then be no need to implement a bi-camera system because each of

our surveillance nodes could send its data to the nearby nodes. Because the distances between them would be larger than that in a bi-camera system, the results would be more accurate.

REFERENCES

[1] I. Haritaoglu, D. Harwood, and L. S. Davis, “W: Real-time surveillance of people and their activities,” IEEE Trans. Pattern Anal. Machine Intell., vol. 22, pp. 809–830, Aug. 2000.

[2] Iain E. G. Richardson. “H.264 and MPEG-4 Video Compression: Video Coding for Next-generation Multimedia,” John Wiley & Sons, Ltd. 2003.

[3] Anderson, M. “VCR quality video at 1.5 Mbits/s,” National Communication Forum, Chicago, Oct. 1990.

[4] ITU-T and ISO/IEC JTC 1, “Generic coding of moving pictures and associated audio information – Part 2: Video,” ITU-T Recommendation H.262 – ISO/IEC 13818-2 (MPEG-2), Nov. 1994.

[5] Coding of moving pictures and associated audio. Committee Draft of Standard ISO11172: ISO/MPEG 90/176, Dec. 1990.

[6] Benson, K. Blair, and Donald G. Fink. “HDTV--Advanced Television for the 1990s,” New York: Intertext Publications: McGraw-Hill Pub. Co., 1991.

[7] Joint Video Team of ITU-T and ISO/IEC JTC 1, “Draft ITU-T Recommendation and Final Draft International Standard of Joint Video Specification (ITU-T Rec. H.264 ISO/IEC 14496-10 AVC),” Joint Video Team (JVT) of ISO/IEC MPEG and ITU-T VCEG, JVT-G050, March 2003.

[8] ITU-T, Video Codec for Audiovisual Services at px64 kbit/s, ITU-T Recommendation H.261, Version 1: Nov. 1990; Version 2: Mar. 1993.

[9] ITU-T, “Video coding for low bit rate communication,” ITUT Recommendation H.263; version 1, Nov. 1995; version 2, Jan. 1998; version 3, Nov. 2000.

[10] T. Wedi, "Motion Compensation in H.264/AVC," in IEEE Transactions on Circuits and Systems for Video Technology, this issue.

[11] T. Wiegand, X. Zhang, and B. Girod, “Long-Term Memory Motion-Compensated Prediction,” IEEE Transactions on Circuits and Systems for Video Technology, vol. 9, no. 1, pp. 70-84, Feb. 1999.

[12] T. Wiegand and B. Girod, “Multi-frame Motion- Compensated Prediction for Video Transmission,” Kluwer Academic Publishers, Sept. 2001.

[13] M. Flierl, T. Wiegand, and B. Girod, "A Locally Optimal Design Algorithm for Block-Based Multi-Hypothesis Motion-Compensated Prediction", in Data Compression Conference, Snowbird, USA, Mar. 1998, pp. 239-248.

[14] D. Marpe, H. Schwarz, and T. Wiegand: “Context-Adaptive Binary Arithmetic Coding in the H.264/AVC Video Compression Standard,” in IEEE Transactions on Circuits and Systems for Video Technology, this issue.

[15] Chen, T.C. Lian, C.J. Liang, G. (2006) Hardware architecture design of an h.264/AVC video codec.

[16] Tourapis, H.-Y.C. Tourapis, A.M. (2003) Fast motion estimation within the h.264 codec, Conf. Multimedia and Expo, ICME ’03, vol. 3, pp. III – 517–20.

[17] He, Zhihai, S. M. A (2001) unified rate-distortion analysis framework for transform coding, Circuits and Systems for Video Technology, vol. 11.

[18] Chen C.Y. Chien S.Y. Huang Y.W. Chen T.C. Wang T.C. Chen L.G. (2005) "Analysis and Architecture Design of Variable Block Size Motion Estimation for H.264/AVC," IEEE Transactions on Circuits and Systems I, Volume PP, Issue 99.

[19] Seong-Min Kim, Ju-Hyun Park, etc. “Hardware-Software Implementation of MPEG-4 Video Codec,” ETRI J., vol.25, no.6, Dec. 2003, pp.489-502.

[20] Ahmed A. Jerraya and Wayne Wolf (editors), Multiprocessor Systems-on-Chip, Elsevier Morgan Kaufmann, San Francisco, California, 2005

[21] R. Chen, W. Zhao, Q. Liu, Jeffrey Fan, "Efficient H.264 architecture using modular bandwidth estimation", IEEE 5th International Conference on Embedded Software and Systems (ICESS'08), pp. 277-282, Chengdu, China, July 29-31, 2008.

[22] Vaidyanathan, P. P. (1993) Multirate Systems and Filter Banks. Englewood Cliffs, NJ: Prentice-Hall.

[23] Vetterli, M. and Kovacevic, J., (1995) Wavelets and Subband Coding. Englewood Cliffs, NJ: Prentice-Hall.

[24] Clark, R. J. (1985) Transform Coding of Images. London, U.K.: Academic.

[25] Gersho, A. and Gray, R. M. (1991) Vector Quantization and Signal Compression. Boston, MA: Kluwer, 1991.

[26] Jayant, N. S. and Noll, P. (1984) Digital Coding of Waveforms. Englewood Cliffs, NJ: Prentice-Hall.

[27] Rabiner, L. R. and Schafer, R. W. (1978) Digital Processing of Speech Signals. Englewood Cliffs, NJ: Prentice-Hall.

[28] Zeng, B. and Fu, J. (2008) Directional Discrete Cosine Transforms – A New Framework of Image coding, IEEE Transactions on Circuit and Systems for Video Technology, Vol. 18, No. 3.

[29] Ahmed, N. Natarajan, T. and Rao, K. R. (1974) Discrete cosine transform, IEEE Trans. Computer, vol. 23, no. 1, pp. 90–93, Jan. 1974.

[30] Rao, K. R. and Yip, P. (1990) Discrete Cosine Transform—Algorithms, Advantages, Applications. London, U.K.

[31] Kauff, P. and Schuur, K., (1998) Shape-adaptive DCT with block-based DC separation and ΔDC correction, IEEE Trans. Circuits Syst. Video Technol., vol. 8, no. 3, pp. 237–242.

[32] Y. Ivanov, A. Bobick, and J. Liu, "Fast Lighting Independent Background Subtraction," Technical Report no. 437, MIT Media Laboratory, 1997.

[33] C. Ridder, O. Munkelt, and H. Kirchner, "Adaptive Background Estimation and Foreground Detection Using Kalman-Filtering," Proc. Int'l Conf. Recent Advances in Mechatronics, ICRAM '95, pp. 193-199, 1995.

[34] S.S. Intille, J.W. Davis and A. F. Bobick, "Real-time Closed-World Tracking", Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR'97), 1997, pp. 697-703.

[35] F. De la Torre, E. Martinez, M. E. Santamaria and J.A.Moran, "Moving Object Detection and Tracking System: a Real-time Implementation", Proceedings of the Symposium on Signal and Image Processing GRETSI 97, Grenoble, 1997.

[36] Jung, Y.K. Lee, K.W. and Ho, Y.S. (2001) Content-based event retrieval using semantic scene interpretation for automated traffic surveillance, IEEE Transactions on Intelligent Transportation Systems, vol. 2, pp. 151-163.

[37] Montoliu, R. and Pla, F. (2001) Multiple parametric motion model estimation and segmentation, ICIP 2001, vol. 2, pp. 933-936.

[38] Li, D. (2000) Moving objects detection by block comparison, Electronics, Circuits and Systems, vol. 1, pp. 341-344.

[39] Cucchiara, R. Grana, C. Piccardi, M. and Prati, A. (2000) Statistic and knowledge-based moving object detection in traffic scenes, IEEE Proceedings. Intelligent Transportation Systems, pp. 27-32.

[40] Novak, C.L. and Shafer, S.A. (1987) Color Edge Detection, Proceedings DARPA Image Understanding Workshop, Vol. I, pp. 35-37, Los Angeles, CA, USA.

[41] Cumani, A (1991). Edge Detection in Multispectral Images, CVGIP: Graphical Models and Image Processing. Vol. 53, pp. 40-51.

[42] Marr, D. and Hildreth, E. (1980) Theory of Edge Detection, Proceedings of the Royal Society of London, B207, pp. 187-217.

[43] Gonzalez, R. C. and Woods, R. E. (2001) Digital image processing,vol. 10, no. 2, pp. 585–611.

[44] J. Hertz, A. Krogh, and R.G. Palmer, "Introduction to the theory of Neural Computation," Addison-Wesley, Reading, Mass., 1991.

[45] J. Eeldman, M.A. Fanty, and N.H. Goddard, "Computing with Structured Neural Networks," Computer, Vol. 21, No. 3, Mar. 1988, pp. 91-103.

[46] Lippmann, R. P., "An introduction to computing with neural nets," IEEE Acoust. Speech Signal Process, 4(2) (1987) 4-22.

[47] S. J. Hanson and L. Y. Pratt, "Comparing biases for minimal network construction with back-propagation," in Advances in Neural Information Processing I, D. S. Touretzky, Ed. Morgan Kaufmann, 1989, pp 177-185.

[48] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning internal representations by error propagation," in Parallel Distributed Processing, vol. 1, D. E. Rumelhart and J. L. McCleland, Eds. Cambridge, MA: M.I.T. Press, 1986, pp. 318-362.

[49] F. C. Chen, "Back-propagation neural network for nonlinear self-tuning adaptive control," Proc. IEEE Intelligent Machine, pp. 274-279, 1989.

[50] R. Goshorn, J. Goshorn, D. Goshorn, H. Aghajan, "Architecture for cluster-based automated surveillance network for detecting and tracking multiple persons," ICDSC07 (219-226), 2007.

[51] K Trivedi, K. Huang and I. Mikic, "Intelligent environments and active camera networks," Proc. IEEE Int'l. Conf. on Systems, Man, and Cybernetics, pp. 804-809, Oct. 2000.

[52] W. Zhao, Z. Luo, J. Fan, S. Tan, “Vector edge detection in H.264 Implementation,” IEEE 5th International Conference on Embedded Software and Systems Symposia (ISHSO'08), pp. 208-212, Chengdu, China, July 29-31, 2008.

[53] W. Zhao, J. Fan, A. Davari, “Vector bank based target tracking via vision sensors in aviation systems,” IEEE 41st Southeastern Symposium on System Theory (SSST'09), pp. 73-76, Tullahoma, TN, March 15-17, 2009.

[54] R. Chen, W. Zhao, J. Fan, A. Davari, “Vector bank based multimedia codec system-on-a-chip (SoC) design,” IEEE 10th International Symposium on Pervasive Systems, Algorithms and Networks (I-SPAN09), pp. 515-520, Kaohsiung, Taiwan, December 14-16, 2009.

VITA

WEI ZHAO

May 9, 1983	Born, Beijing, China P. R.
2001-2005	B.S., Electrical Engineering Zhejiang University Hangzhou, China P. R.
2005-2007	M.S., Electrical Engineering Florida International University Miami, Florida
2007-2010	Doctorate Candidate in Electrical Engineering Florida International University Miami, Florida

PUBLICATIONS AND PRESENTATIONS

Wei Zhao, R. Batista, J. Fan, J. Tan, (2010). *H.264 based architecture of digital surveillance network in application to computer visualization*. I-manager's Journal on Software Engineering (IJSE), Vol. 4, No. 4, pp. 18-26.

Wei Zhao, J. Fan, A. Davari (2009). *H.264 based wireless surveillance sensors in application to target identification and tracking*. I-manager's Journal on Software Engineering (IJSE), Vol. 4, No 2, pp 47-56.

Wei Zhao, X. Yuan, R. Batista, J. Fan, (2010). *Controller free gaming and gesture recognition via H.264 SoC*. IEEE 5th International Conference on Communications and Networking in China (ChinaCom'10), Beijing, China, session MCS02.2.

R. Chen, Wei Zhao, J. Fan, A. Davari, (2009). *Vector bank based multimedia codec system-on-a-chip (SoC) design*. IEEE 10th International Symposium on Pervasive Systems, Algorithms and Networks (I-SPAN09), pp. 515-520.

Wei Zhao, J. Fan, A. Davari, (2009). *Vector bank based target tracking via vision sensors in aviation systems*. IEEE 41st Southeastern Symposium on System Theory (SSST'09), pp. 73-76, Tullahoma, TN.

Wei Zhao, C. Castello, J. Fan, (2008). *Design considerations of SOPC-based H.264/AVC systems*. First International Workshop on Video Coding and Video Processing (VCVP'08), Session S7-3, Shenzhen, China.

Wei Zhao, Z. Luo, J. Fan, S. Tan, (2008). *Vector edge detection in H.264 Implementation*. IEEE 5th International Conference on Embedded Software and Systems Symposia (ISHSO'08), pp. 208-212, Chengdu, China.

R. Chen, Wei Zhao, Q. Liu, J. Fan, (2008). *Efficient H.264 architecture using modular bandwidth estimation*. IEEE 5th International Conference on Embedded Software and Systems (ICESS'08), pp. 277-282, Chengdu, China.

T. Chou, S. Fan, Wei Zhao, J. Fan, A. Davari, (2008). *Intrusion aware system-on-a-chip design with uncertainty classification*. IEEE 5th International Conference on Embedded Software and Systems (ICESS'08), pp. 527-531, Chengdu, China.