

Jaeseok Kim · Hyunchul Shin *Editors*

---

# Algorithm & SoC Design for Automotive Vision Systems

For Smart Safe Driving System

 Springer

# Algorithm & SoC Design for Automotive Vision Systems

Jaeseok Kim · Hyunchul Shin  
Editors

# Algorithm & SoC Design for Automotive Vision Systems

For Smart Safe Driving System

 Springer

*Editors*

Jaeseok Kim  
Yonsei University  
Seoul  
Korea, Republic of (South Korea)

Hyunchul Shin  
Hanyang University  
Ansan  
Korea, Republic of (South Korea)

ISBN 978-94-017-9074-1      ISBN 978-94-017-9075-8 (eBook)

DOI 10.1007/978-94-017-9075-8

Springer Dordrecht Heidelberg New York London

Library of Congress Control Number: 2014942316

© Springer Science+Business Media Dordrecht 2014

This work is subject to copyright. All rights are reserved by the Publisher, whether the whole or part of the material is concerned, specifically the rights of translation, reprinting, reuse of illustrations, recitation, broadcasting, reproduction on microfilms or in any other physical way, and transmission or information storage and retrieval, electronic adaptation, computer software, or by similar or dissimilar methodology now known or hereafter developed. Exempted from this legal reservation are brief excerpts in connection with reviews or scholarly analysis or material supplied specifically for the purpose of being entered and executed on a computer system, for exclusive use by the purchaser of the work. Duplication of this publication or parts thereof is permitted only under the provisions of the Copyright Law of the Publisher's location, in its current version, and permission for use must always be obtained from Springer. Permissions for use may be obtained through RightsLink at the Copyright Clearance Center. Violations are liable to prosecution under the respective Copyright Law. The use of general descriptive names, registered names, trademarks, service marks, etc. in this publication does not imply, even in the absence of a specific statement, that such names are exempt from the relevant protective laws and regulations and therefore free for general use.

While the advice and information in this book are believed to be true and accurate at the date of publication, neither the authors nor the editors nor the publisher can accept any legal responsibility for any errors or omissions that may be made. The publisher makes no warranty, express or implied, with respect to the material contained herein.

Printed on acid-free paper

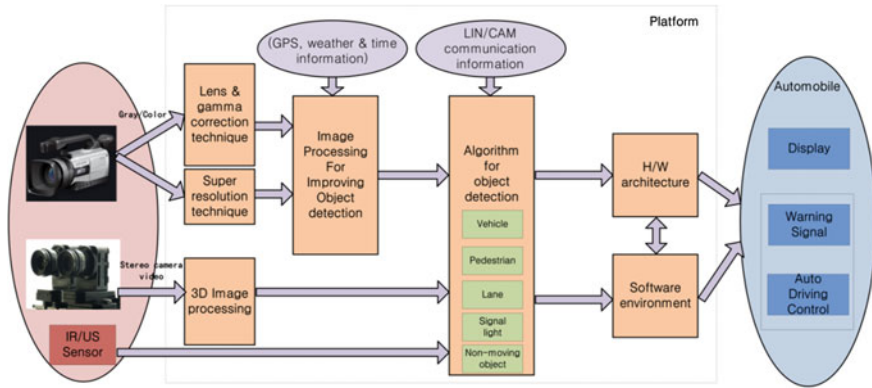
Springer is part of Springer Science+Business Media ([www.springer.com](http://www.springer.com))

# Preface

An emerging trend in the automobile industry is its convergence with information technology (IT). Indeed, it has been estimated that almost 90 % of new automobile technologies involve IT in some form. *Smart* driving technologies that improve safety as well as *green* fuel technologies are quite representative of the convergence between IT and automobiles. The former technologies, in particular, include three key elements: sensing of driving environments, detection of objects and potential hazards (pedestrians, lanes on the road, other vehicles, traffic signals, stationary objects, etc.), and the generation of driving control signals including warning signals. The first two elements are usually implemented using machine vision, radar, and sonar together with vehicle-to-vehicle and vehicle-to-infrastructure communication. These are the most commonly used technologies to realize novel systems such as a smart parking assistance, lane keeping assistance, smart cruise control, collision avoidance, and ultimately fully autonomous self-driving vehicles.

This book is organized into 10 chapters to cover system-on-a-chip (SoC) design—including both algorithms and hardware—related with image sensing and object detection by using the camera for smart driving systems, as shown in Fig. 1. First, lens correction techniques for distorted images captured by cameras installed on vehicles are presented. Second, novel super-resolution algorithm suited to a low-cost vehicle camera is presented. Next, image enhancement techniques to improve object detection from the captured images are discussed. Then, two chapters present algorithms and techniques for accurate detection of several different types of objects (pedestrians, road lanes other vehicles, traffic signals, stationary objects, etc.). This is followed by a discussion of the SoC architecture and hardware design necessary to implement these algorithms in real time. Finally, the software environment and reliability issues for automotive SoC platforms are discussed.

Chapter 1 introduces the needs and requirements of an image vision system for smart and safe driving in a novel IT-converged automobile. Next, we present and discuss the platform architecture for addressing the components of the automotive vision system. This platform is implemented as a System-on-a-Chip (SoC) platform, and its hardware architecture along with its software environment are introduced.



**Fig. 1** SoC platform architecture for automobile vision system

Chapter 2 presents a lens distortion correction algorithm based on a geometric invariant suitable to a vehicle camera. This method adopts cross-ratio invariability for a perspective projection and minimized distortion. In addition, we describe a new gamma correction approach to deal with rapid variation in luminous intensity. To reduce the computation complexity, we introduce an objective numerical descriptor for luminance and contrast as well as gamma correction based on a tone-mapping approach.

Chapter 3 presents a novel super-resolution algorithm suited to a low-cost vehicle camera. We introduce a smart and robust registration algorithm that takes into account rotation and shift estimation. To reduce the registration error, this algorithm determines the optimal reference image where even other super-resolution algorithms discard this registration error. The algorithm follows the warp-blur observation model because the blur parameter is considerably larger than the warp parameter for camera rotation and/or vibration.

Chapter 4 introduces image processing algorithms for improving object detection. These algorithms can be used to improve the object recognition rate for poor images that have been captured under inadequate conditions such as low illumination, rainfall, or snowfall. Furthermore, a high-dynamic-range tone-mapping technique that improves image quality is described.

Chapter 5 explains coarse-to-fine vehicle and pedestrian detection techniques. In the case of vehicle detection, we describe mono-camera-based vehicle detection systems in which low-level edges and high-level bag-of-features are incorporated. Specifically, once initial candidates are obtained using edge information, these candidates are further verified using a bag-of-features. On the other hand, in the case of pedestrian detection, we explain certain popular Histogram of Oriented Gradient (HOG) detectors and part-based detectors. The use of edge-based coarse detection in the base detectors greatly reduces the detection time.

Chapter 6 discusses various driver monitoring systems as well as some methods for predicting unsafe driving behaviors. We describe the design considerations that led us to develop the driver monitoring system architecture, discuss the software responsible for control and data acquisition, and present some of the data screening and feature extraction algorithms that predict unsafe driving behaviors.

Chapter 7 covers various aspects of SoC architectures for automobile vision system. After a brief introduction, it surveys existing SoC architectures as well as architecture design issues and methodologies, ranging from single- to multi-core SoC architectures. The chapter further discusses on-chip communications among cores, hardware blocks, and memories. It also covers the issue of mapping applications to SoC architectures.

Chapter 8 introduces a hardware accelerator that performs interest point detection and matching for image-based recognition applications in real time. Interest point detection and matching are basic and one of the most computation-intensive operations, in general, vision tasks such as object recognition/tracking, image matching/stitching, and simultaneous localization and mapping. This chapter describes a pattern matching-based image recognition processor that unifies architectures for features from accelerated segment tests and binary robust independent elementary features with low power dissipation and a high frame rate.

Chapter 9 introduces a software development environment for automotive SoC. AUTOSAR, a standardized automotive software architecture, is a partnership of automotive manufacturers and suppliers working in collaboration to establish an open industry standard for automotive E/E architectures. This chapter describes this software architecture in detail, covering the consideration of safety requirements, scalability of different vehicle and platform variants, and implementation and standardization of the basic functions based on the cooperation of standards. In addition, this chapter covers maintainability throughout the entire product life cycle, software updates, and upgrades over vehicle lifetime.

Chapter 10 covers reliability issues of automobile electronic system. Current vehicles are built with complex electronic systems embedded with more than a hundred microprocessors through complicated automotive networks. In the de facto ISO 26262 standard in the automotive industry, Automotive Safety Integrity Level (ASIL) is classified into four different levels. In this chapter, the ISO 26262 hardware ASIL is described in detail. Finally, we introduce fault diagnosis architectures that use various designs for testability techniques such as scan design, built-in self-test, and IEEE boundary scan design for increasing hardware reliability.

We would like to appreciate our colleagues who contributed to each chapter in this book. This work was supported by the Ministry of Trade, Industry and Energy (MOTIE) in Korea and the IDEC Platform center (IPC) for Smart cars.

We would like to thank the publishing editor Mark de Jongh and Mrs. Cindy Zitter at Springer for their encouragement and continuous support to prepare this book.

Finally, we wish to thank our graduate students who provided their efforts to prepare some materials in this book.

Jaeseok Kim  
Hyunchul Shin



# Contents

<b>1</b>	<b>Introduction</b> . . . . .	<b>1</b>
	Jaeseok Kim and Hyunchul Shin	
<b>2</b>	<b>Lens Correction and Gamma Correction</b> . . . . .	<b>11</b>
	Sang-Bock Cho	
<b>3</b>	<b>Super Resolution</b> . . . . .	<b>41</b>
	Hyo-Moon Cho	
<b>4</b>	<b>Image Enhancement for Improving Object Recognition</b> . . . . .	<b>73</b>
	Jaeseok Kim	
<b>5</b>	<b>Detection of Vehicles and Pedestrians</b> . . . . .	<b>107</b>
	Hyunchul Shin and Irfan Riaz	
<b>6</b>	<b>Monitoring Driver’s State and Predicting Unsafe Driving Behavior</b> . . . . .	<b>143</b>
	Hang-Bong Kang	
<b>7</b>	<b>SoC Architecture for Automobile Vision System</b> . . . . .	<b>163</b>
	Kyounghoon Kim and Kiyoung Choi	
<b>8</b>	<b>Hardware Accelerator for Feature Point Detection and Matching</b> . . . . .	<b>197</b>
	Jun-Seok Park and Lee-Sup Kim	
<b>9</b>	<b>Software Development Environment for Automotive SoC</b> . . . . .	<b>231</b>
	Jeonghun Cho	
<b>10</b>	<b>Reliability Issues for Automobile SoCs</b> . . . . .	<b>263</b>
	Sungju Park	

# Chapter 1

## Introduction

Jaeseok Kim and Hyunchul Shin

**Abstract** This chapter introduces concepts and trends related to the development of advanced driver assistance systems (ADAS), which aid human drivers in collision avoidance. Some industrial developments and major functional blocks of ADAS are also introduced. We have focused on the camera vision-based systems, which are very popular mainly because of their low development costs and very diverse potential applications. The system-on-chip (SoC) architecture and components of automobile vision systems are also presented.

### 1.1 Introduction to the Advanced Driver Assistance System

In 2009, 5.5 million automobile crashes, 2.2 million automobile-related injuries, and 33,963 motor vehicle fatalities were reported in the United States alone, while in Europe, more than 40,000 casualties and 1.4 million injuries are caused by vehicle-related accidents annually [1]. Based on these figures, researchers have developed crash survival techniques and systems, and emphasized the need for their installation in all vehicles. To some extent, these systems guarantee the survival of the driver. Top motor vehicle brands have stringently equipped their vehicles with these systems, i.e., airbags, active safety electronics, etc. Although these advances in passive safety have made passenger cars safer than ever before, the potential for further improvements in passive safety features is limited.

---

J. Kim (✉)  
Yonsei University, Seoul, Korea  
e-mail: jaekim@yonsei.ac.kr

H. Shin  
Hanyang University, Ansan, Korea  
e-mail: shin@hanyang.ac.kr

In the past, researchers have focused on developing systems that can help protect human lives in the event of motor vehicle accidents. Systems like anti-lock braking systems (ABS) [2] and electronic stability programs (ESP) [3] offer the possibility of improving traffic safety by assisting drivers when they are on the road. However, the emphasis is shifting from passive protection systems toward active protection systems, i.e., from crash survival to crash avoidance. This is where advanced driver assistance systems (ADAS) come in. ADAS are vehicle control systems that use environmental sensors, e.g., radar, laser, and vision, to improve driving comfort and traffic safety by assisting the driver in recognizing and reacting to potentially dangerous traffic situations. ADAS include facilities that enable vehicles to successfully deploy collision avoidance systems.

Over the last decade, several ADAS technologies, such as blind spot detection (BSD), the lane departure warning system (LDWS) or lane-keeping assistance system (LKAS), the smart parking assistance system (SPAS), the forward collision warning system (FCWS), pre-crash safety (PCS), and smart cruise control (SCC), have been introduced into the market and used as intelligent parts of new smart automobiles [4]. The BSD system operates sensors to avoid collisions on the rear-side blind spot of a vehicle, while LDWS employs front-side vision cameras to warn a driver when the vehicle begins to move out of its lane, thus minimizing potential accidents. Similarly, LKAS is a driving assistance device that maintains the vehicle's driving lane using the same vision cameras. SPAS uses infrared (IR) sensors and vision cameras to assist with parking, and can even control a vehicle's steering system. FCWS makes use of vision cameras to calculate the forward collision timing and to warn the driver of dangerous situations. By using radar sensors and vision cameras, PCS technology helps vehicles avoid collisions and minimizes damage when a collision cannot be avoided. Radar sensors and cameras allow the SCC system to start or stop a vehicle based on the vehicle's distance from the vehicle in front of it. It also provides warning signals when the vehicle is within another's blind spot [5].

ADAS were developed with the safety of the driver and the people outside the vehicle in mind. This is in contrast to the crash survival system, which only ensures the safety of the driver. ADAS provide a platform to save human life through technology, and have made it possible to make the transition from semi-autonomous to fully automatic driver systems. Researchers are currently working to develop a low-cost system that can be marketed openly. Since ADAS can even intervene autonomously, an ADAS-equipped vehicle is popularly referred to as an "intelligent vehicle" [5].

At the end of 2012, the global market for ADAS was estimated to be around US\$16.6 billion, and analysts at ABI Research forecasted that the market will increase to more than US\$261 billion by the end of 2020, representing a compound annual growth rate (CAGR) of 41 % [6]. A key catalyst inspiring the acceptance of ADAS in Europe over the next 5 years will be the Euro New Car Assessment Programme (NCAP) specifications. The specifications include three types of ADAS: the speed assistance system (SAS), autonomous emergency braking (AEB), and LDWS or LKAS. By 2017, it is likely that all three ADAS will

be required to be fitted as standard in all new models in order to qualify for the highest, five-star safety rating [6]. North America and the Asia-Pacific's original equipment manufacturers (OEMs) are expected to follow Europe's trend in one to three years.

According to several surveys, ADAS can prevent up to 40 % of traffic accidents, depending on the type of ADAS and the type of accident scenario [7]. The main challenges in this area have been customer acceptance and their understanding of the added value of ADAS, liability exposure, and regulatory issues. Drivers also expect ADAS to meet high standards in terms of (subjective) performance, reliability (low rate of false alarms), and safety (low rate of missed detections). Therefore, all ADAS must be tested in a huge variety of complex traffic situations so that the systems will be able to recognize and handle all types of situations. Unfortunately, the exhaustive testing of ADAS prototypes is usually impossible due to constraints in costs and time-to-market. Not only the design, but also the validation of ADAS requires growing effort in the development process. To address these issues, efficient methods for the design of ADAS controllers and validation of their safety and performance are required.

The final goal of ADAS is an autonomous vehicle that provides a 100 % guarantee of an accident-free and self-driving driving experience. The autonomous vehicle will be capable of sensing its driving environment and navigating on its own by controlling every function of a vehicle based on the information sensed. The driver of an autonomous vehicle may choose a destination but will not be required to perform any mechanical operation of the vehicle. This vehicle will sense the world with such techniques as radar, light-detection-and-ranging (LIDAR), global positioning system (GPS), and computer vision. The advanced control system will interpret the information to identify appropriate navigation paths, as well as obstacles and relevant traffic signs. The autonomous vehicle will typically update its maps based on sensory input, such that it will be able to navigate through uncharted environments.

The advantages of these cars would mainly be safety, freedom for the driver to do other tasks, and fuel efficiency. Automated highway systems using fully automated passenger cars are expected to significantly benefit traffic safety [8].

The capabilities of self-driving cars offer many advantages over those of human drivers. Although the emergence of a truly autonomous car is still several years away, currently available radars, LIDARs, and vision-based systems are coalescing into automotive-sensor platforms, which point the way toward tomorrow's robo- or self-driving cars.

Research conducted by iSuppli in 2011 showed that radar-based systems are primarily used by manufacturers to handle collisions in undesirable weather and driving conditions [9]. However, the camera (vision-based sensor) has gained importance in ADAS, while laser-based sensors are more popular in low-speed FCW systems and are capable of ranging.

## 1.2 Industrial Developments for ADAS

### 1.2.1 *The Google Self-Driving Car*

In 2010, Google revealed the results of its efforts to create an autonomous vehicle. Google's fleet of specially equipped cars (shown in Fig. 1.1) has logged more than 140,000 miles of daytime and nighttime driving in California and Nevada [10]. The license for the vehicles was issued by the Nevada Department of Motor Vehicles in May 2012. As of April 2012, Florida became the second state to allow the testing of driverless cars on public roads, and California became the third state to legalize the use of self-driven cars for testing purposes in September 2012.

Each Google vehicle uses LIDAR technology combined with vision cameras and algorithmic vision-processing systems to construct and react to a 3D view of the world through which a vehicle is driving [10]. The enabling sensor hardware in the vehicles enables the cars to see everything around them and make decisions about every aspect of driving. Although automakers are not yet close to a fully autonomous vehicle, the technology, including the sensor platform of radars, ultrasonic sensors, and cameras, is already available in today's intelligent vehicles. It remains for the car's hardware platform to be standardized and the software to be developed [10].

### 1.2.2 *Mobileye's Vision Processor*

Mobileye's collision avoidance technology, shown in Fig. 1.2, which uses a single smart camera combined with a sophisticated vision algorithm, is able to interpret a scene in real-time and provide drivers with an immediate evaluation based on its analysis [11]. Automakers such as Volvo, BMW, and General Motors are now adopting this technology, which is embedded with Mobileye's EyeQ<sup>®</sup> vision processor, into their rapidly expanding safety feature applications [11]. In 2010, Volvo introduced the S60 model with pedestrian detection, vehicle detection, and lane detection using this technology. The vehicle was equipped with an automatic emergency collision mitigation system and automatic emergency braking in the event of pedestrian detection. In 2011/12, lane departure warning, intelligent headlight control, traffic sign recognition, and forward collision warning technologies were added to the pedestrian detection system.

In 2012, BMW and Mobileye upgraded the BMW 6-series active cruise control (ACC) and the collision warning and braking features with a vision-radar fusion system, thus replacing the radar-only system. This was the first system to utilize Mobileye vision to enhance radar ACC to a high degree using a mature vision sensor that reports vehicle, lanes, and other information independently. The vision information improves target selection and target localization, and enables an earlier reaction to stationary targets than previous models allowed.



Fig. 1.1 Google car (From Wikipedia: Google car)

Fig. 1.2 Mobileye's pedestrian and bicycle collision avoidance



### 1.2.3 C2X Communication Technology and BMW

Car-to-X (C2X) technology demonstrates how, using a wireless networking of cars and road infrastructure, cooperative driving can make driving safer, more efficient, and more environmentally friendly. The C2X uses high-frequency WLAN IEEE802.11p/G5A-based networking technology [12]. A real-time and comprehensive C2X communication network leads to a significant increase in the predictive capabilities of vehicles. The driver can adjust the speed of the vehicle according to a green light signal using provided phase information. Sensor cooperation technology also detects a person who is in the driver's blind spot. This technology is supported by the C2X consortium, which consists of the major automotive manufacturers, including BMW and Audi.

BMW aims to adopt this technology for its ConnectedDrive ADAS, which is being developed to connect the automobile with its environment and daily traffic situations [13]. BMW ConnectedDrive provides connectivity to web servers, real-time traffic information in combination with a networked navigation system, and the integration of vehicle-specific apps with mobile devices [13]. The automobile is actually becoming a mobile internet exchange point. The ConnectedDrive

currently provides several ADAS features, such as adaptive headlights, a lane-change/departure warning system, speed limit info, a self-parking system, Stop & Go active cruise control, and collision warning.

## 1.3 System-on-Chip Platform Architecture for Automobile Vision Systems

### 1.3.1 Major Functional Blocks for ADAS

It is expected that ADAS will see sharp growth in the years to come. Customer demand is the one of the driving forces behind increased safety awareness and the desire for more driving comfort. However, Euro NCAP's tightened safety requirements will boost the installation of ADAS equipment in vehicles to almost a 100 % rate over the next few years [14].

Figure 1.3 shows the major functional blocks of smart ADAS, which consist of a sensing block, a decision-making block, and an active control with warning functions.

Figure 1.4 shows the major functions to be performed at each functional block in ADAS.

The first step in developing ADAS technology is the sensing of the driving environments. In this step, some objects, such as lanes, vehicles, pedestrians, animals, or fixed objects, should be detected and tracked. The position and condition of the vehicle must also be assessed continuously. In addition, information about traffic signals and weather conditions are monitored. The second step is to make a decision based on the information sensed and monitored from the first block. Several signal-processing algorithms are used to decide on the space required for safe driving or parking, and to make decisions to prevent a collision or accident. The third step is to generate a warning or announcing signal and to develop active control of the vehicle. The warning can be provided by sound or a visual cue. The traffic or driving information can be displayed onscreen, announced by a voice, or provided to another location via e-call. The active control of the vehicle facilitates collision avoidance and driving/parking guidance so that the vehicle will eventually be self-driving and self-parking.

Figure 1.5 shows the hardware and software components at each functional block in ADAS.

Several different types of sensors, such as radar, LIDAR, ultrasonic/IR sensors, cameras, V2X (C2X), and GPS, are presently being used in ADAS, with the selection of sensors being dependent on the ADAS tasks that need to be implemented. A component of the decision-making step entails several software algorithms, such as signal processing, image processing, and pattern recognition. However, these algorithms should be run in real-time on very powerful processors that contain fault-tolerant functions. The active control of the vehicle is applied to the steering, brakes, auto navigation, adaptive lights, seatbelts, sound, and e-call.

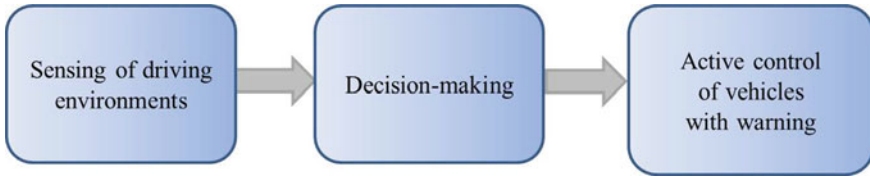


Fig. 1.3 Major functional blocks of the ADAS system

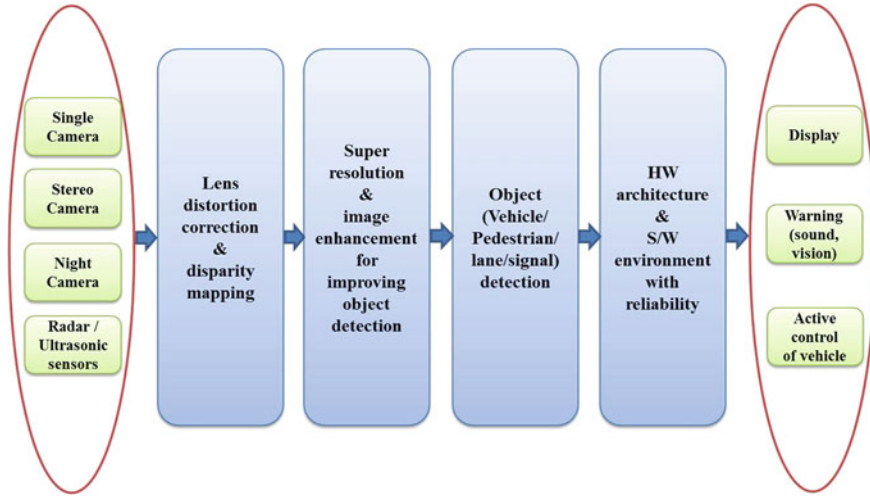
Block	Sensing	Decision-making	Active control
Functions to be performed	<ul style="list-style-type: none"> <li>Object detection &amp; tracking               <ul style="list-style-type: none"> <li>Lane</li> <li>Cars</li> <li>Pedestrian</li> <li>Animals</li> <li>Fixed objects</li> </ul> </li> <li>Vehicle position &amp; conditional tracking</li> <li>Sensing of traffic signal</li> <li>Sensing of weather</li> <li>Monitoring of driver</li> </ul>	<ul style="list-style-type: none"> <li>Continuous monitoring of detected information</li> <li>Space detection for safe driving &amp; parking</li> <li>Collision &amp; accident prevention</li> </ul>	<ul style="list-style-type: none"> <li>Warning/announcing               <ul style="list-style-type: none"> <li>Sound/vision warning</li> <li>Traffic/driving information display &amp; announcing</li> <li>e-call</li> </ul> </li> <li>Driving assistance               <ul style="list-style-type: none"> <li>Collision avoidance</li> <li>Adaptive head-light</li> <li>Driving guidance</li> <li>Self-driving</li> </ul> </li> <li>Parking assistance               <ul style="list-style-type: none"> <li>Parking guidance</li> <li>Self-parking</li> </ul> </li> </ul>

Fig. 1.4 Major functions to be performed at each block in ADAS

Sensing	Decision-making	Active control
<ul style="list-style-type: none"> <li>Radar (long, short)</li> <li>LIDAR</li> <li>Ultrasonic / IR sensors</li> <li>Camera / night-vision</li> <li>V2X(C2X)</li> <li>GPS</li> </ul>	<ul style="list-style-type: none"> <li>Signal processing algorithms</li> <li>Image processing / object recognition algorithms</li> <li>Real-time processing H/W processors with fault-tolerant</li> </ul>	<ul style="list-style-type: none"> <li>Steering</li> <li>Brake</li> <li>Auto navigation</li> <li>Light</li> <li>Seat belt</li> <li>Sound</li> <li>e-call</li> </ul>

Fig. 1.5 Hardware and software components at each block in ADAS





**Fig. 1.6** Platform architecture for an automobile vision system

The sensor components in the vehicles enable the cars to see everything around them and to detect any obstacles so that the potential for safe driving is maximized. Ultrasonic sensors are a low-cost solution that provide proximity detection at low speeds. Radar has short-range, medium-range, and long-range capabilities and works at any speed in any weather condition; however, it is expensive and takes up significant space. The presence of a camera boosts the sensor capabilities so that the car is able to have a 360° view and detect any objects or obstacles, all at a relatively low cost.

### 1.3.2 Vision Systems for ADAS

Camera-based vision systems are gaining more attention in the development of ADAS due to their ability to not only provide an extended outside view, but also detect and recognize objects such as pedestrians, vehicles, animals, or fixed objects. These systems also provide the additional function of looking inside the car to analyze the driver's state [14]. Nowadays, multiple cameras are used as standard equipment in automobile systems.

Figure 1.6 shows the platform architecture of an automobile vision system intended for advanced driver assistance. This book covers the issues and the new technologies pertaining to the activities shown in this figure in order to create a smart, safe driving automobile system using vision cameras.

The sensing of driving environments can be performed using a single camera, stereo cameras, and night (IR) cameras, combined with radars and ultrasonic sensors. The first functional block of the automobile vision system is the lens

distortion correction and gamma correction, which is covered in [Chap. 2](#). [Chapter 2](#) introduces vehicle-friendly lens correction algorithms and gamma correction algorithms with objective illumination estimation methods. In the lens correction section, we introduce a simple lens correction method for a low-cost camera and propose a method that simultaneously leads to a guarantee of the restrictions for the determination. In the gamma correction section, the objective illumination estimation methods and tone mapping-based gamma correction methods are introduced.

The second functional block is to enhance the quality of the images captured on the camera. As the first part of this block, super-resolution methods to obtain high-resolution images from low-resolution images are discussed in [Chap. 3](#). As the second part, [Chap. 4](#) introduces some general image enhancement algorithms followed by special algorithms for improving object recognition in automotive environments.

The third functional block deals with object detection techniques, such as lanes, vehicles, pedestrians, and fixed objects. [Chapter 5](#) presents state-of-the-art techniques for vehicle and pedestrian detection. [Chapter 6](#) describes some techniques that can be employed to monitor a driver's state and predict unsafe driving behaviors.

The last functional block for the vision system is to design hardware architecture and a software environment with some consideration of reliability issues. To this end, [Chap. 7](#) discusses the system-on-chip (SoC) architecture issues for an automobile vision system. A pattern matching-based image recognition accelerator (PRA) is presented for embedded vision application. It is a hardware accelerator that performs interest-point detection and matching for image-based recognition applications in real-time—both in mobile and vehicle platforms—without any loss of performance or accuracy. [Chapter 8](#) presents the design of the image-recognition accelerator for feature-point detection and matching. [Chapter 9](#) describes the software development environment for the automotive SoC. AUTOSAR is a model-based software development method for automotive electronic systems that has been developed by the automotive industry. A case study of the AUTOSAR development for automotive SoC is also described. Finally, the reliability issues for automobile SoC are covered in [Chap. 10](#). In the final chapter, the ISO 26262 standard hardware, Automotive Safety Integrity Level (ASIL), is described in detail. We then introduce fault diagnosis architectures that use various designs for testability techniques, such as scan designs, built-in self-tests, Institute of Electrical and Electronics Engineers (IEEE) boundary scan designs, and error correcting codes, for increased hardware reliability.

## References

1. International Road Traffic Database (IRTAD), <http://www.irtad.com/>
2. J. Broughton, C. Baughan, The effectiveness of antilock braking systems in reducing accidents in Great Britain. *Accid. Anal. Prev.* **34**(3), 347–355 (2002)

3. A. Lie, C. Tingvall, M. Krafft, A. Kullgren, The effectiveness of ESP (electronic stability program) in reducing real life accidents. *Traffic Inj. Prev.* **5**(1), 37–41 (2004)
4. [www.hyundai.com](http://www.hyundai.com)
5. [http://en.wikipedia.org/wiki/Advanced\\_Driver\\_Assistance\\_Systems](http://en.wikipedia.org/wiki/Advanced_Driver_Assistance_Systems), (2012)
6. Forecasted US\$260 Billion Global market for ADAS systems by 2020, <http://www.abiresearch.com/press/>. Accessed 17 Apr 2013
7. Development of performance requirements for commercial vehicle safety application—final report, DOT HS 811 722, National Highway Traffic Safety Administration, Washington, DC, USA, May 2013
8. S.E. Shladover, Automated vehicles for highway operations (automated highway systems). *Proc. IMechE Part I J. Syst. Control Eng.* **11**, 53–75 (2005)
9. ADAS Technology trends, 3Q/2011, HIS-iSuppli topical report/automotive research, 2011
10. Automotive sensors may usher in self-driving cars, <http://www.edn.com/>. Accessed 26 May 2011
11. <http://www.mobileye.com/>
12. Drive C2X at ITS world Congress, <http://www.ertico.com/congress/vienna2012/>
13. <http://www.bmwgroup.com/>
14. B. Kusstatcher, P. Voss, *Camera Based Adas For Mass Deployments*, 21 June 2012

# Chapter 2

## Lens Correction and Gamma Correction

Sang-Bock Cho

**Abstract** Recently, the necessity of camera in vehicle industry is increasing now as increasing the smart car needs. Almost smart car concepts are implemented by using the camera system based on image processing technology. Actually, the rear view camera for parking assistance system, the forward view camera for lane departure warning and forward collision warning system, and multi-view camera for vehicle black box and blind spot warning system and so on, so many systems those adopted the camera system are already released. However, performance of these functions strongly depends on the image quality through the camera system. Especially, distortion of a thing pictured by the lens and suddenly illumination changing by environments are core factors affecting the image quality for smart car performance. Thus, in this chapter, we introduced a vehicle friendly lens correction algorithm and a gamma correction algorithm with objective illumination estimation method. In the lens correction part, we introduced a simple lens correction method in low-cost camera and we propose a method that leads to guarantee of the restrictions simultaneously for the determination. In the gamma correction part, we introduced the objective illumination estimation methods and the gamma correction methods based on tone mapping.

### 2.1 Lens Correction

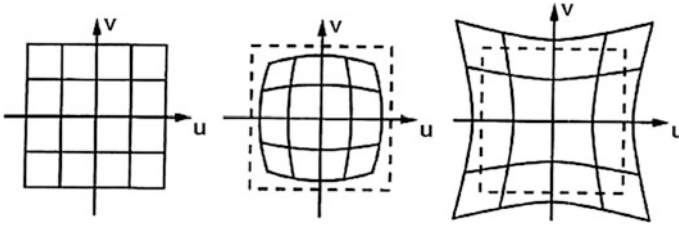
#### 2.1.1 Introduction to Lens Correction

Cameras being used in the field have been having lens distortion, and it has been considerably dealing in image process area. So far, many lens correction algorithms have been relying on the invariant properties of projective geometry in

---

S.-B. Cho (✉)

School of Electrical Engineering, University of Ulsan, Ulsan, Korea  
e-mail: sbcho@ulsan.ac.kr



**Fig. 2.1** Radial distortions—ideal image with no distortion, barrel distortion, and pincushion distortion (from left to right)

order to find distortion parameters. A common property is “the straight line in scene is straight line in image”. However, if the straight lines are also parallel together, the previous works have missed this restriction in determining lens distortion parameters.

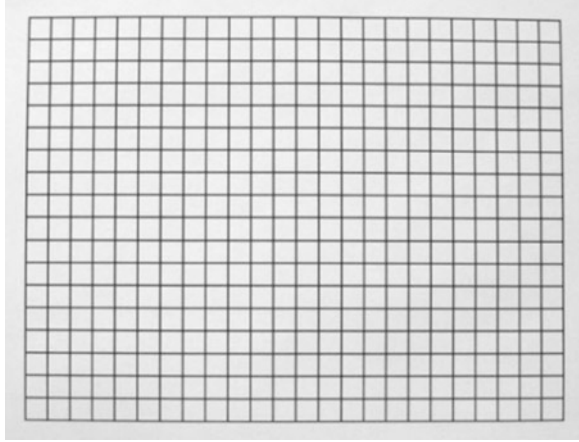
Model of lens distortion was proposed by Brown [1, 2] including radial distortion, decentering distortion and prism distortion. Radial distortion (barrel distortion or/and pincushion distortion) as shown in Fig. 2.1 is noticeable but other distortions are acceptably negligible.

Many algorithms adopted this model used 3D reference points to find the lens distortion parameters [3–9]. However, these algorithms can find the proper distortion parameter only when their experiments were precise and extracting control points were accurate.

In another class of lens distortion correction, there have been many proposed methods based on geometric invariants of some features to achieve distortion parameters [10–14]. The straight property of line—the straight line in scene is straight line in image—is used widely. However, if the straight lines are also parallel together, these methods have missed this restriction in determining lens distortion parameters. This can cause wrong distortion parameters of final solution in search algorithm. Finally, the other approaches were proposed to correct lens distortion that collect whole content of image to get distortion parameters [15–17]. These approaches take advantage of image content without requiring any pattern calibration. However, results are still limited to lead to convincing results.

Model of lens distortion was proposed by Brown [1, 2] including radial distortion, decentering distortion and prism distortion. Radial distortion (barrel distortion or/and pincushion distortion) is noticeable and other distortions are acceptably negligible, as shown in Fig. 2.1. The adopted methods to this model [3–8, 14] used 3D reference points to find lens distortion parameters. These methods encountered in setting up experiments precisely and extracting control points accurately under noise impact. In another class of lens distortion correction, there have been many proposed methods based on geometric invariants of some features to achieve distortion parameters [7, 10, 12, 18]. The straight property of line—the straight line in scene is straight line in image—is used widely. However, if the straight lines are also parallel together, these methods have missed this restriction in determining lens distortion parameters. Finally, the other approaches

**Fig. 2.2** Standard calibration template



were proposed to correct lens distortion that collect whole content of image to get distortion parameters [8, 15, 18]. These approaches take advantage of image content without requiring any pattern calibration. However, results are still limited to lead to convincing results.

In this chapter, our proposed method is based on the projective transformation “a straight line in a 3D scene domain is itself straight line in the 2D image domain” [2]. This means that curvature of lines in the image that should be straight is due only to lens distortion. Using this principal, iterative search algorithms are applied to seek adequate distortion parameters, which make cure lines in image become its original state—the straight lines. The proposed method in this chapter is different from above works. We use a template consisting of lines that are straight, parallel and perpendicular together, as shown in Fig. 2.2. After the correction, the restrictions of lines of template are guaranteed.

### 2.1.2 Lens Distortion Model

The standard model for the radial and decentering distortion is mapping from the distorted image coordinates  $q_d = (u_d, v_d)$  that are observable to the undistorted image coordinates  $q_p = (u_p, v_p)$  which are not physically measurable.

$$u_p = u_d + \bar{u}_d(k_1 r_d^2 + k_2 r_d^4 + k_3 r_d^6 + \dots) + \left[ p_1 (r_d^2 + 2\bar{u}_d^2) + 2p_2 \bar{u}_d \bar{v}_d \right] [1 + p_3 r_d^2 + \dots] \quad (2.1a)$$

$$v_p = v_d + \bar{v}_d(k_1 r_d^2 + k_2 r_d^4 + k_3 r_d^6 + \dots) + \left[ p_1 (r_d^2 + 2\bar{v}_d^2) + 2p_2 \bar{u}_d \bar{v}_d \right] [1 + p_3 r_d^2 + \dots] \quad (2.1b)$$

where

$$\overline{u}_d = u_d - c_u, \quad \overline{v}_d = v_d - c_v, \quad r_d^2 = \overline{u}_d^2 + \overline{v}_d^2 \quad (2.1c)$$

and  $k_1, k_2, k_3, \dots$  are coefficients of radial distortion and  $p_1, p_2, p_3, \dots$  are coefficients of decentering distortion,  $r_d$  is the radius of an image point from the distortion center defined as  $(c_u, c_v)$  above.

Typically, since we only consider two coefficients of radial distortion [5, 7, 12, 18] the model after discarding other terms:

$$u_p = u_d + \overline{u}_d(k_1 r_d^2 + k_2 r_d^4) \quad (2.2a)$$

$$v_p = v_d + \overline{v}_d(k_1 r_d^2 + k_2 r_d^4) \quad (2.2b)$$

$$r_p = r_d(1 + k_1 r_d^2 + k_2 r_d^4) \quad (2.2c)$$

Therefore, the lens distortion correction becomes recovering the practically significant distortion coefficients  $k_1, k_2$  and distortion center  $(c_u, c_v)$ .

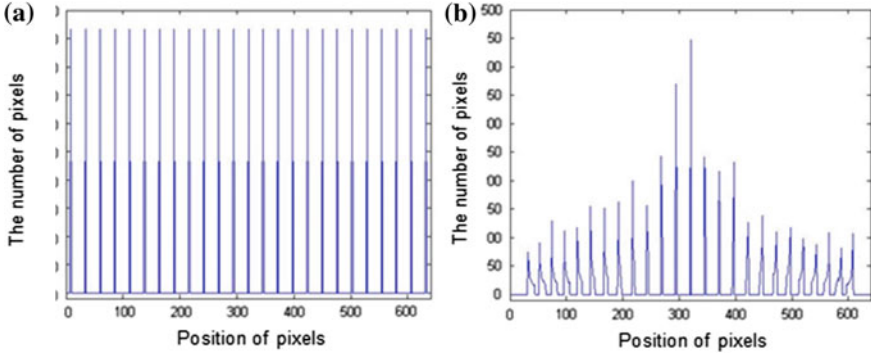
## 2.1.3 Proposed Method

### 2.1.3.1 Basic Idea

The standard calibration template as shown in Fig. 2.2, it consists of black straight lines in horizontal and vertical directions. These lines are parallel and perpendicular together on bright background. In horizontal direction, on a horizontal line that is the nearest to distortion center, we find out the intersection points of this horizontal line with vertical lines. Based on each intersection point, we count the number of pixels of vertical line which cross the intersection point. Since lens distortion exists, the number of pixels of the vertical line do not lie on vertical axis (origin of vertical axis is intersection point). We measure the farthest distance of pixels of each line to vertical axis respectively at each base point and seek lens distortion parameters of camera by minimizing this distance. The process is the same for vertical direction. Figure 2.3 shows the position of pixels in horizontal direction of a non-distorted image and a distorted image.

First of all, a binary image is made of a gray image. In the binary image, there are horizontal lines and vertical lines; the width of these lines is one pixel. Then the distortion center is initialized, e.g. image center. In horizontal direction, since a straight line crossing the distortion center is still straight line in distorted image, so a horizontal line that is through the distortion center (or the nearest the distortion center) is chosen. This line is named as driven line.

On the driven line, the points where vertical lines intersect with this line can be determined. These points are named base points. The number of the base points is



**Fig. 2.3** The number of pixels in horizontal direction, **a** in non-distortion image, **b** in distortion image

equal the number of vertical lines  $n$ . Based on each base point, in next step, we count the number of pixels of respective vertical line. Due to lens distortion, a number of pixels of vertical line do not lie on vertical axis (the origin of vertical axis is base point). In the close range of each base point, we determine one point on the left and another one on the right of base point where the number of pixels at these two points are greater than  $e$  pixels (in our implementation,  $e = 4$ ). They are called up-position points and down-position points. Therefore, each base point lies on the range between up-position point and down-position point respectively. We evaluate the distortion by measuring the distances from up-position points  $p_i^{uh}$  to down-position points  $p_i^{dh}$  respectively. This measurement is defined by a function  $E_1$  as

$$E_1 = \sum_{i=1}^n \sqrt{(p_i^{uh} - p_i^{dh})^2} / n \quad (2.3)$$

The process is the same as in vertical direction. We have:

$$E_2 = \sum_{i=1}^m \sqrt{(p_i^{uv} - p_i^{dv})^2} / m \quad (2.4)$$

Finally, we define a function  $E$  as evaluation of distortion in whole image:

$$E = (E_1 + E_2) / 2 \quad (2.5)$$

where  $n$  and  $m$  are the number of vertical lines and horizontal lines,  $p_i^{uh}$  and  $p_i^{uv}$  are the up-position points in horizontal and vertical direction, and  $p_i^{dh}$  and  $p_i^{dv}$  are the down-position points in horizontal and vertical direction.

The error  $E$  reflects the farthest distance of pixels of each line to axis respectively at each base point. Minimization of the error function  $E$  is our goal for



seeking distortion parameters. In this chapter, we optimize distortion parameters using the Nelder-Mead simplex method [9].

### 2.1.3.2 Linear Form Solution

Nonlinear optimization algorithm is an iterative progress and initial value plays an important role to lead to an exactly converged value. If initial value is so far from global minima, converged value can be a local minima and results in a false solution. In this chapter, we adopted the method in [5] for initial guess. The method is summarized as follows:

Suppose we have a line  $l$  in the undistorted image plane. Every point  $(u_p, v_p)$  on the line satisfies the equation

$$au_p + bv_p + c = 0 \quad (2.6)$$

where  $a$ ,  $b$  and  $c$  are constants for specific line  $l$ , with  $-a/b$  being the line slope. Each point on the line is related to a point  $(u_d, v_d)$  in the distorted image plane according to Eq. (2.1). This means that both coordinate of the line point are functions of  $(u_d, v_d)$ . Therefore, the last equation can be written as

$$f(u_d, v_d) = au_p(u_d, v_d) + bv_p(u_d, v_d) + c = 0 \quad (2.7)$$

where  $f(u_d, v_d)$  describes the equation of the corresponding curve in the distorted image plane. The elemental change in  $f$  at any distorted image point  $(u_d, v_d)$  can be expressed as

$$\delta f = a \left[ \frac{\partial u_p}{\partial u_d} \delta u_d + \frac{\partial u_p}{\partial v_d} \delta v_d \right] + b \left[ \frac{\partial v_p}{\partial u_d} \delta u_d + \frac{\partial v_p}{\partial v_d} \delta v_d \right] = 0 \quad (2.8)$$

where all the four partial derivatives can be directly computed from Eq. (2.1). Hence, one can see that the slope of the line  $S$  in the undistorted plane (which should equal  $-a/b$ ) is related to the slope of the tangent  $(\delta v_d / \delta u_d)$  to the curve at point  $(u_d, v_d)$  by

$$s(u_d, v_d) = \left( \frac{\partial v_p}{\partial u_d} + \frac{\partial v_p}{\partial v_d} \frac{\partial v_d}{\partial u_d} \right) / \left( \frac{\partial u_p}{\partial u_d} + \frac{\partial u_p}{\partial v_d} \frac{\partial v_d}{\partial u_d} \right) \quad (2.9)$$

In our case, we choose some lines in image template as reference lines in finding linear solution. We only estimate  $k_1$  and  $k_2$  equation of linear form solution  $Ax = b$  can be expressed as

$$A = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix}, \quad x = \begin{bmatrix} k_1 \\ k_2 \end{bmatrix} \quad \text{and} \quad b = \begin{bmatrix} \hat{s}_1 - s_{1,1} \\ \hat{s}_2 - s_{2,1} \end{bmatrix} \quad (2.10a)$$

with

$$a_{11} = 2\bar{u}_{1,1}\bar{v}_{1,1}(1 - \hat{s}_1 s_{1,1}) + (\bar{u}_{1,1}^2 + \bar{v}_{1,1}^2)(s_{1,1} - \hat{s}_1) + 2(\bar{v}_{1,1}^2 s_{1,1} - \bar{u}_{1,1}^2 \hat{s}_1) \quad (2.10b)$$

$$a_{12} = 4\bar{u}_{1,1}\bar{v}_{1,1}(\bar{u}_{1,1}^2 + \bar{v}_{1,1}^2)(1 - \hat{s}_1 s_{1,1}) + (\bar{u}_{1,1}^2 + \bar{v}_{1,1}^2)(s_{1,1} - \hat{s}_1) + (\bar{u}_{1,1}^2 + \bar{v}_{1,1}^2)(\bar{v}_{1,1}^2 s_{1,1} - \bar{u}_{1,1}^2 \hat{s}_1) \quad (2.10c)$$

$$a_{21} = 2\bar{u}_{2,1}\bar{v}_{2,1}(1 - \hat{s}_2 s_{2,1}) + (\bar{u}_{2,1}^2 + \bar{v}_{2,1}^2)(s_{2,1} - \hat{s}_2) + 2(\bar{v}_{2,1}^2 s_{2,1} - \bar{u}_{2,1}^2 \hat{s}_2) \quad (2.10d)$$

$$a_{22} = 4\bar{u}_{2,1}\bar{v}_{2,1}(\bar{u}_{2,1}^2 + \bar{v}_{2,1}^2)(1 - \hat{s}_2 s_{2,1}) + (\bar{u}_{2,1}^2 + \bar{v}_{2,1}^2)(s_{2,1} - \hat{s}_2) + (\bar{u}_{2,1}^2 + \bar{v}_{2,1}^2)(\bar{v}_{2,1}^2 s_{2,1} - \bar{u}_{2,1}^2 \hat{s}_2) \quad (2.10e)$$

where  $(u_{l,i}, v_{l,i})$  denotes the  $i$ -th distorted point on chain  $l$  with tangent slope  $s_{l,i}$ ,  $\hat{s}_l$  is the estimated slope of the line corresponding to chain  $l$ . The slope  $s_{l,i}$  is estimated by approximating the curve points within a window of size  $2W + 1$  (in our implementation,  $W = 5$ ) by a second order polynomial.

### 2.1.3.3 Straight Line Method

Given a set of distorted points  $(u_d, v_d)$  in the lines of distorted image, in search algorithms, a set of the distortion parameters is applied to the distorted points. Lines are fitted to the resulting points (corrected points)  $(u_p, v_p)$  and objective function is computed as the sum of the squared distances of the corrected points from their corresponding best-fit straight lines.

Let the best-fit straight line for a set of corrected points be:

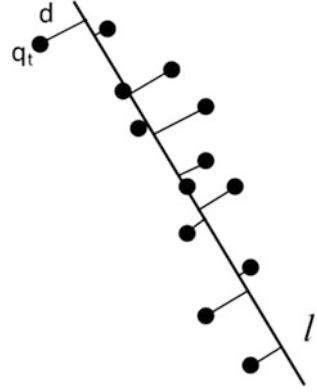
$$u_p \sin\theta - v_p \cos\theta + \rho = 0 \quad (2.11)$$

where  $\theta$  is the angle the line makes with the horizontal axis and  $\rho$  is the distance of the line from the distortion center. Therefore, the error due to a single point is given by:

$$e = (u_p \sin\theta - v_p \cos\theta + \rho)^2 \quad (2.12)$$

where the values for  $(u_p, v_p)$  are calculated from the correction equation Eq. (2.1). Let the number of the lines in the distorted image is  $L$ , and let the number of points on each line  $l$  be  $t_l$ . Then, the objective junction is given by:

**Fig. 2.4** Best-fit straight line of a set of points



$$\xi = \sum_{l=1}^L \sum_{p=1}^{t_l} \left( u_p^{t,l} \sin \theta_l - v_p^{t,l} \cos \theta_l + \rho_l \right)^2 \quad (2.13)$$

where  $\theta_l$  and  $\rho_l$  are the best-fit straight line parameters corresponding to the line  $l$  and  $(u_p^{t,l}, v_p^{t,l})$  is the corrected point to the line  $l$ . After getting the objective function, the distortion parameters are found out by minimizing this function using some non-linear search algorithms (Fig. 2.4).

### 2.1.4 Experimental Result

To evaluate performance of the proposed method, we carry out experiments on synthetic images and real images. In simulation, the distorted-synthetic images are created by warping the undistorted-synthetic template images with radial distortion model expressed in Eq. (2.11)

$$r_p = r_d (1 + k_1 r_d^2 + k_2 r_d^4) \quad (2.14)$$

While distortion coefficient  $k_2$  is fixed due to the weak impact compared with  $k_1$ , distortion coefficient  $k_1$  is changed for testing. In our experiment,  $k_2 = 3 \times 10^{-13}$ ,  $k_1 = -8 \times 10^{-7} : -4.8 \times 10^{-7}$ . Then, we recover the undistorted images from these distorted images using the linear solution, the straight line method and the proposed method. After completing the lens distortion correction, the performances of each method are evaluated based on the error in Eq. (2.5).

The errors of each method are shown in Fig. 2.5. In the first view, we see that the error of the linear method is the worst among three methods. The result also shows that distortion correction based on straight line method causes the error greater than the proposed method when it just guarantees straight restriction of

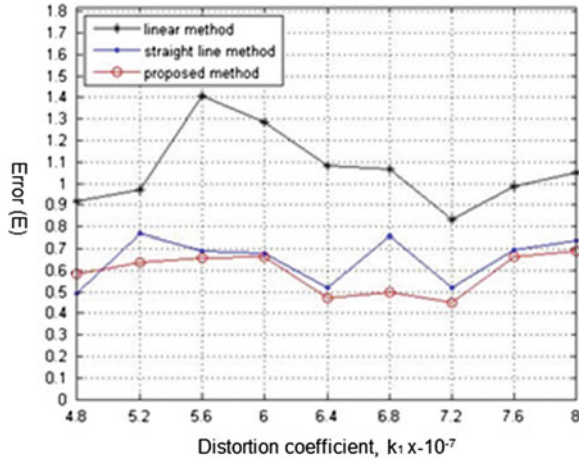


Fig. 2.5 Error of the methods after correcting distortion

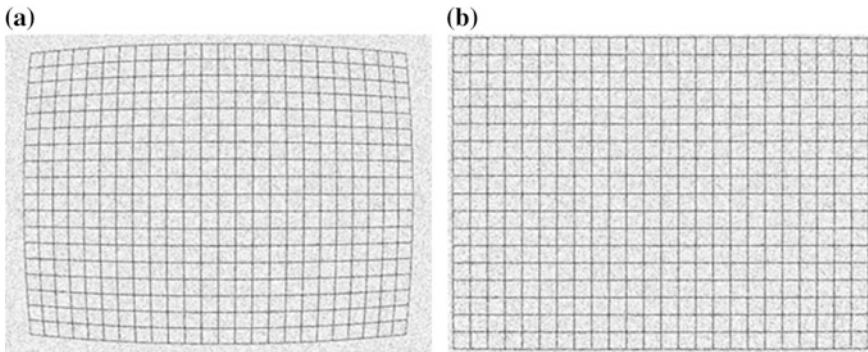
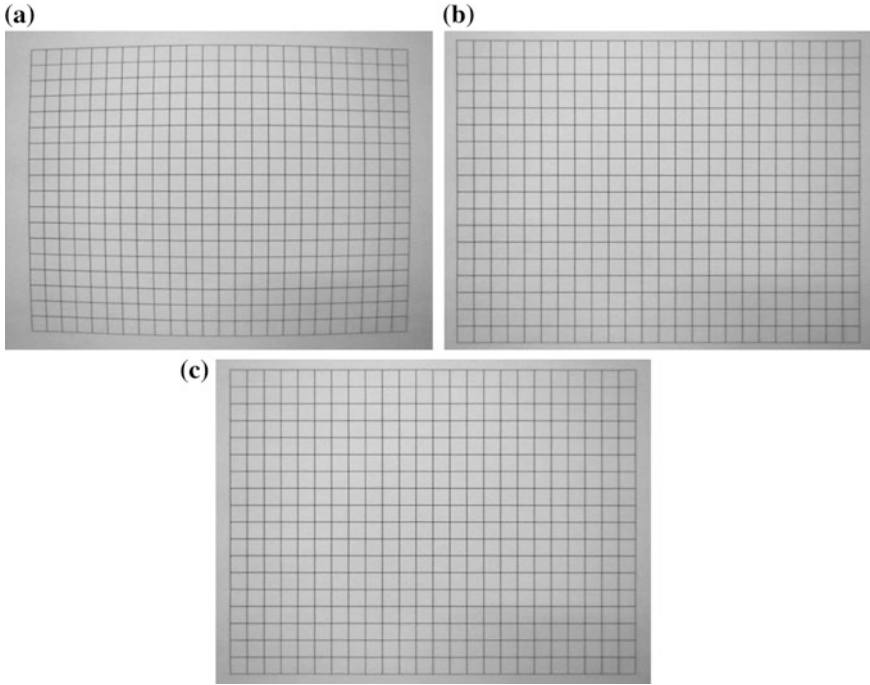


Fig. 2.6 The impact of noise in the proposed method (a input image, lens distorted image via camera, b lens distortion corrected image). Lens distorted image via camera (a), lens distortion corrected image (b)

each line without guaranteeing the parallel restriction among straight lines. By applying our method, however, these restrictions are guaranteed more stringent in distortion correction because they are considered simultaneously. It means that almost vertical lines (horizontal lines) coincide with vertical axes (horizontal axes) respectively at each base point.

In Fig. 2.6, we consider the noise impact in our method. The distorted-synthetic images are corrupted by zero-mean Gaussian noise of standard deviation 0.2 pixels. From the experiment, although the noise impact causes our image content to be very erroneous from original image content, corrected image is almost the same ideal image that is taken from pinhole camera model. Therefore, under the influence of noise in practical cases, the proposed method is better than other methods to correct lens distortion.

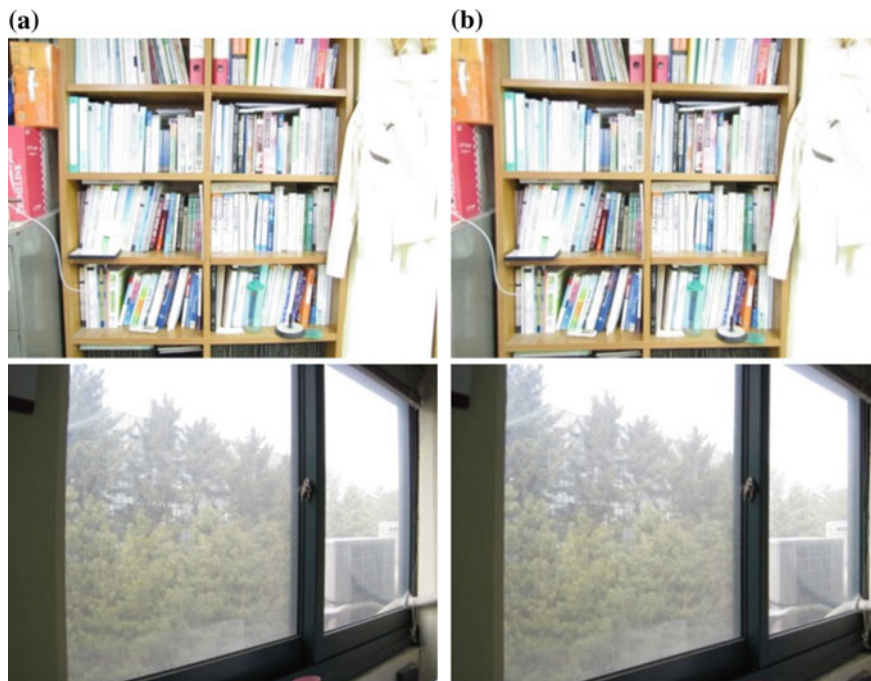


**Fig. 2.7** Radial distortion correction of three methods: **a** distorted image, **b** straight line method, **c** lens distortion corrected image by using the proposed method

### 2.1.5 Conclusion

To evaluate the effectiveness of the proposed method in practical case, Fig. 2.7 shows the radial distortion correction of camera Canon SX20IS using the straight line method and the proposed method. The error before correcting is 4.0758; and after correcting is 0.6821, 0.6421 of the straight line method, and our method respectively.

Figure 2.8 shows the result of radial distortion correction images by using the proposed method. In this chapter, a new method is presented for the lens distortion correction. Using a simple template including the parallel-perpendicular-straight lines, the method applies a common property of perspective geometry to seek the lens distortion parameters. This invariant property is expressed as “a straight line in a 3D scene domain is itself straight line in the 2D image domain”. The previous works have used this property to find the lens distortion parameters when they try to straighten each line in the image that should be straight if there is no lens distortion. However, the method of this chapter is different from previous works. It determines the lens distortion parameters by minimizing the farthest distances of pixels on vertical lines (horizontal lines) to vertical axes (horizontal axes) respectively.



**Fig. 2.8** Radial distortion correction results: **a** original distorted images, **b** corrected images

With the proposed method, it guarantees not only the straight property of each line but also the parallel property among lines for determining the distortion parameters. The experimental results prove the effectiveness and accuracy of the proposed method. Compared with the previous works, the error of the proposed method is almost less than that of the common methods when the distortion coefficient is changed from small value to large value. Moreover, the proposed method becomes more reliable and practical as it is verified under the impact of noise. In the experiment, the algorithm recovers the corrected image perfectly although the affect of noise on distorted image is serious.

In addition, this method is also provided the closed form solution in order to speed up the convergence and obviate the local minimum of objective function. Because the algorithm of the proposed method find out global minimum, the corrected image based on the distortion parameters will be close the ideal image taken from pinhole camera model.

In the future, the algorithm will be continued developing for more optimized process. First of all for improving purpose, we will get threshold automatically in converting from grayscale image to binary image. Next one is designing an adaptive coefficient(s) to reflect contribution of the distance from the distortion center in evaluating the distances between up positions and down positions respectively.

## 2.2 Gamma Correction

### 2.2.1 *Normalized Numerical Image Descriptor*

#### 2.2.1.1 Introduction

The acquired image during the vehicle driving gets degraded by the ambient luminance condition and this is one of the big problems which need proper solution. Therefore, recently, the studies for the image quality enhancement by using the gamma correction method, wide dynamic range, etc., are activated. Almost numerous these studies have been based on the image quality assessment.

The assessment of image quality has been previously considered by Oakley and Bu [19]. One of the purposes of them is to shed light on the relationship between local luminance and local contrast in order to estimate the image quality for different appearing images. However, their method is not clear whether there is an ideal dependence between dispersion and location for good visual quality images.

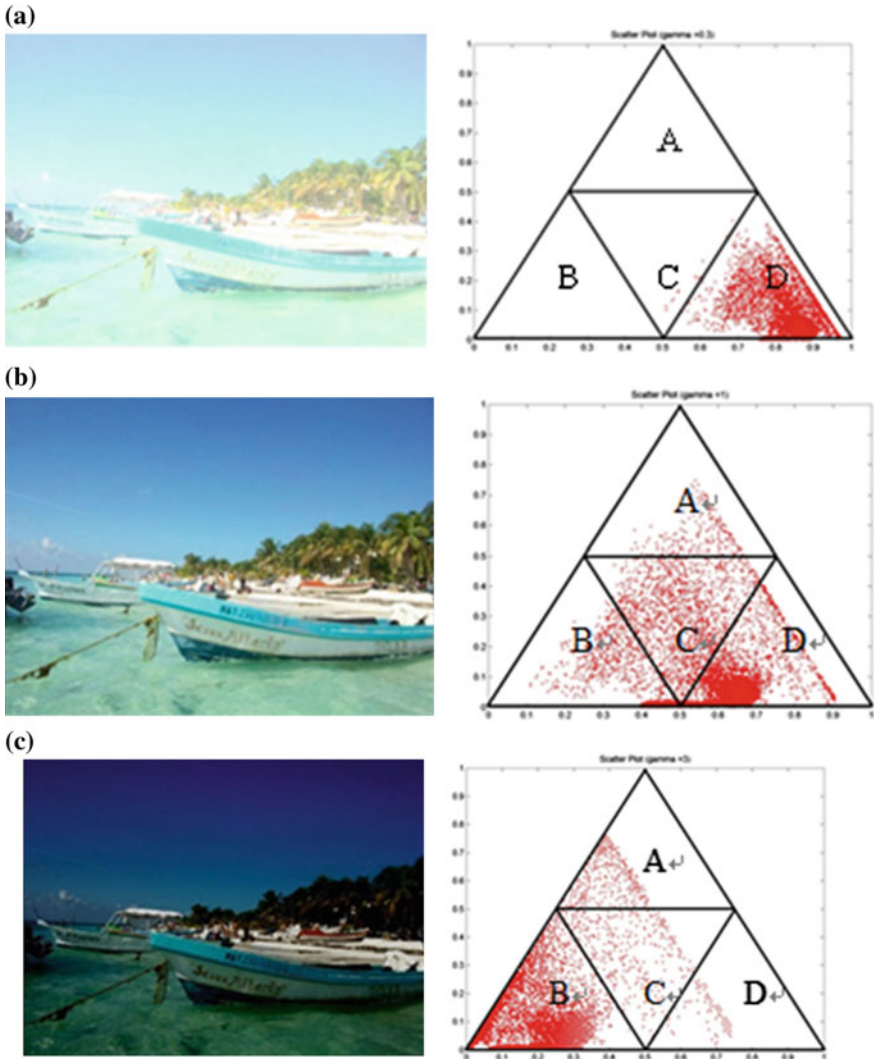
In their [1] method for example, takes the local standard deviation and the local average as estimators and assumes that the local standard deviation should be proportional to the pixel average (a situation analogous to that in Weber's law) for certain types of images. This in fact is not true in general since, as mentioned, many good quality images have a correlation coefficient of local dispersion and location very near to zero. With a slightly different definition of contrast, it is argued in [20] that there are independence between luminance and contrast in natural scenes.

Restrepo and Ramponi [21] considered that scatter plots of local dispersion versus location (d-l plots, for short) carry a fair amount of the information in the image. And then they proposed a word descriptor which classifies the points as belonging to one of four main regions, counting them, normalizing and ordering the numbers.

The four regions in their result are labeled as A, B, C and D and correspond to high-contrast/medium-luminance, low-contrast/low-luminance, medium-contrast/medium-luminance and low-contrast/high-luminance., respectively, and these are illustrated as in Fig. 2.9.

To obtain a word descriptor, however, this method requires very high computational complexity, because they have to compute the pixel intensity histogram and executing of dispersion and location distribution. It is very difficult or may be impossible to commercialize since these computing operations are mathematically complex, and those need the powerful and high performance system, as well.

In this chapter, we introduce a novel normalized numerical image descriptor and suggest the validity of the proposed algorithm by comparing with the word descriptor [21]. This can estimate the image quality more clearly and objectively



**Fig. 2.9** Three “boat” images with different gamma values and their scattering chart. **a**  $\gamma = 0.5$ , **b**  $\gamma = 1$ , **c**  $\gamma = 1.5$

than previous works. The image luminance assessment descriptor is based on the histogram calculation; we use the pixel data as weighted value. Also for the image contrast assessment descriptor, we use the same concept of luminance assessment. These two operations consist of just summation and multiplication. To verify our algorithm, we compare between the results of proposed and the results of the word descriptor by using several images with different gamma value as from 0.1 to 3.



Numerous gamma correction algorithms are based on the histogram. Its basic concept is that image quality can be enhanced using a kind of histogram equalization, it can be expressed as

$$s = T(r) \quad (2.15)$$

where  $s$  is the gamma corrected image by using transfer function  $T(\cdot)$ ,  $r$  is the data of original image. Thus, gamma correction is finding the suitable transfer function  $T(\cdot)$ . The histogram of a digital image with gray level in the range  $[0, L - 1]$  is a discrete function  $h(r_k) = n_k$ , where  $r_k$  is the  $k$ -th gray level and  $n_k$  is the number of pixels in the image having gray level  $r_k$ . The numerical luminance descriptor ( $LD$ ) can be expressed as

$$LD = \sum_{k=0}^{L-1} r_k n_k \quad (2.16)$$

And we apply normalization into Eq. (2.16), then

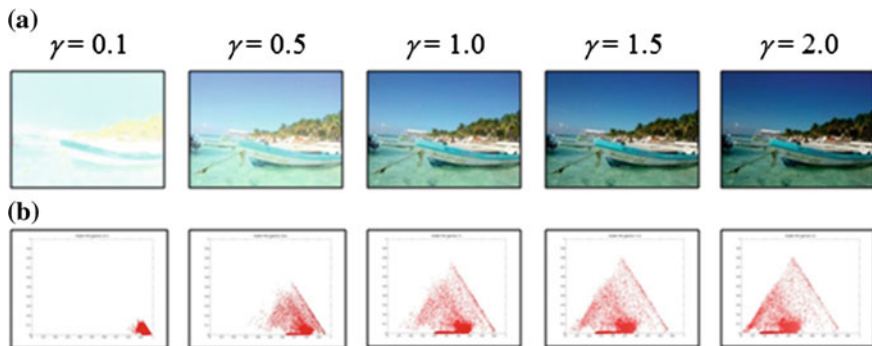
$$NLD = \frac{1}{LN} \sum_{k=0}^{L-1} r_k n_k, \quad 0 \leq NLD \leq 1 \quad (2.17)$$

where  $L$  is maximum quantized value and  $N$  is image size, they are taken from the original image. In Eq. (2.17), we understand that if an image has a normalized luminance descriptor ( $NLD$ ) value becomes 1 and this image is estimated as bright image. On the other hand, if it is near to 0 then this image expressed as dark. From these result, our algorithm suggests that the objective luminance assessment degree with numeric representation. We present example images of “boat” with different gamma value and each scattering chart are shown in Fig. 2.10. As shown in Table 2.1, our proposed algorithm has same result by simple method as compared with the word descriptor method.

However, this normalized luminance descriptor ( $NLD$ ) does not include the contrast information. Thus we have to consider the contrast estimation method. The contrast information can be estimated using the distribution of the difference between maximum and minimum pixel value of the image, thus it can be expressed as  $c = r_{\max} - r_{\min}$ .

As same with the luminance descriptor, the histogram of a contrast distribution with gray level in the range  $[0, L - 1]$  is a discrete function  $h(c_l) = b_l$ , where  $c_l$  is the  $l$ -th block contrast value and  $b_l$  is the number of blocks in the image having contrast value  $c_l$ . Here,  $c_l = b(r_{\max} - r_{\min})_l$  and  $b(r_{\max} - r_{\min})_l$  is the contrast value in  $l$ -th block. Thus, the numerical contrast descriptor ( $CD$ ) can be expressed as

$$CD = \sum_{l=0}^{B-1} c_l b_l \quad (2.18)$$



**Fig. 2.10** Example images of “boat” with different gamma value and each scattering chart. **a** Images with difference gamma value: image resolution =  $640 \times 480$ , **b** scattering charts for each gamma value

**Table 2.1** NLD for each gamma value and block size

Gamma	0.1	0.5	1	1.5	2
Block size					
global	0.94	0.74	0.56	0.42	0.33
$4 \times 4$	0.94	0.74	0.56	0.42	0.33
$8 \times 8$	0.94	0.74	0.56	0.43	0.33
$16 \times 16$	0.94	0.74	0.56	0.43	0.33

And we apply normalization into Eq. (2.18), then

$$NCD = \frac{1}{LB} \sum_{l=0}^{B-1} c_l b_l, \quad 0 \leq NCD \leq 1 \quad (2.19)$$

where  $B$  is the total number of segmented block in the image. In Eq. (2.19), we understand that if an image has a normalized contrast descriptor ( $NCD$ ) value near to 1 then this image is estimated as high contrast image, whereas if it is near to 0 then this image is low contrast. From these result, it also suggests the objective contrast. Therefore, the image quality assessment can be represented as Eq. (2.20) by combining with Eqs. (2.16) and (2.18).

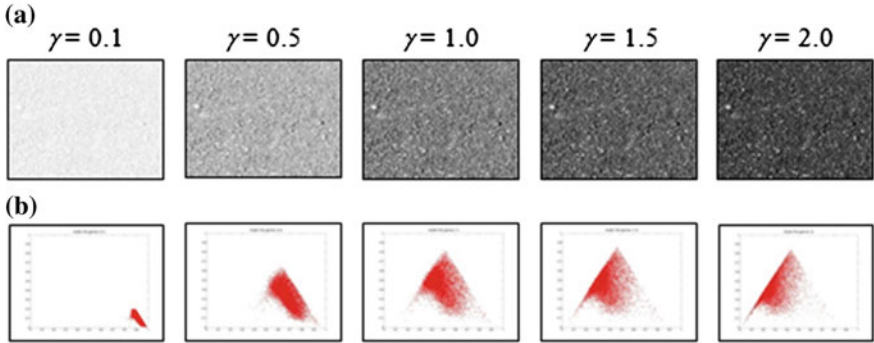
$$NIQ = (NLD, NCD) \quad (2.20)$$

And its results for Fig. 2.10 are shown in Table 2.2.

In another example image, “gravelly field,” as shown in Fig. 2.11 is one such representative image of the relatively high contrast. The  $NCD$  assessment in this chapter can clearly and objectively express the image quality which has same meaning compared with the scattering method or word descriptor.

**Table 2.2**  $NIQ = (NLD, NCD)$  for each gamma value and block size of “boat” image

Gamma	0.1	0.5	1	1.5	2
Block size					
$4 \times 4$	(0.94, 0.02)	(0.74, 0.06)	(0.56, 0.08)	(0.42, 0.09)	(0.33, 0.10)
$8 \times 8$	(0.94, 0.03)	(0.74, 0.10)	(0.56, 0.17)	(0.42, 0.15)	(0.33, 0.15)
$16 \times 16$	(0.94, 0.04)	(0.74, 0.15)	(0.56, 0.21)	(0.42, 0.22)	(0.33, 0.22)

**Fig. 2.11** Example images of “gravelly field” with different gamma value and each scattering chart. **a** Images with difference gamma value: image resolution =  $640 \times 480$ , **b** scattering charts for each gamma value**Table 2.3**  $NIQ = (NLD, NCD)$  for each gamma value and block size of “gravelly field” image

Gamma	0.1	0.5	1	1.5	2
Block size					
$4 \times 4$	(0.93, 0.09)	(0.70, 0.30)	(0.51, 0.39)	(0.37, 0.40)	(0.28, 0.37)
$8 \times 8$	(0.93, 0.14)	(0.70, 0.46)	(0.51, 0.58)	(0.37, 0.59)	(0.28, 0.55)
$16 \times 16$	(0.93, 0.16)	(0.70, 0.57)	(0.50, 0.71)	(0.37, 0.72)	(0.28, 0.69)

As shown in Tables 2.1, 2.2 and 2.3, the  $NLD$  maintains almost same values without concern of block size, but the  $NCD$  has different values according to the images. Generally, the segmented block size is decided by using Shannon’s ratio-distortion ( $RD$ ) cost. But the study for this area is mainly studied in mode decision site since it is as another research area. Therefore, we only use widely used the  $16 \times 16$  block segmentation in this chapter.

### 2.2.1.2 Experiments for Normalized Numerical Image Descriptor

We played the comparison experiment with word descriptor (or scattering chart) method to verify proposed algorithm. For the word descriptor, we followed [19], and for proposed algorithm process as below;

**Table 2.4**  $NIQ = (NLD, NCD)$  for each gamma value and block size of “airplane” image

Gamma	0.1	0.5	1	1.5	2	2.5	3
Block size							
$4 \times 4$	(0.96, 0.03)	(0.81, 0.10)	(0.67, 0.14)	(0.56, 0.16)	(0.47, 0.16)	(0.40, 0.16)	(0.34, 0.15)
$8 \times 8$	(0.96, 0.04)	(0.82, 0.16)	(0.68, 0.23)	(0.57, 0.26)	(0.48, 0.27)	(0.41, 0.26)	(0.35, 0.25)
$16 \times 16$	(0.96, 0.06)	(0.82, 0.22)	(0.68, 0.32)	(0.57, 0.36)	(0.48, 0.37)	(0.41, 0.36)	(0.35, 0.34)

Step 1: set a segmented block size from among these; global,  $4 \times 4$ ,  $8 \times 8$ , and  $16 \times 16$ .

Step 2: calculation for the normalized numerical luminance descriptor

- (1) Choose one pixel data from each block
- (2) Normalize the chosen pixel data
- (3) Obtain the summation of these normalized pixel data
- (4) Obtain the normalized numerical luminance descriptor by dividing the number of block.

Step 3: calculation for the normalized numerical luminance descriptor

- (1) Find the maximum and the minimum pixel data in each block
- (2) Normalize the difference in both of them which is obtained at 1)
- (3) Perform the summation of these normalized differences
- (4) Obtain the normalized numerical contrast descriptor by dividing the number of block.

In this chapter, we present total four experiments’ results wise “Airplane”, “Baboon”, “Butterfly”, and “Uniform noise” sample images. The resolution sizes of them are  $200 \times 200$ ,  $200 \times 200$ ,  $500 \times 500$ ,  $356 \times 356$ , respectively, and these are 24-bit RGB bitmap color images. To estimate image quality of them, we convert color image to gray level image by using as

$$Y = 0.3R + 0.6G + 0.1B \quad (2.21)$$

The experiments’ results are as below.

This chapter has described the normalized numeric image quality descriptor. It suggests the clear and objective image quality assessment degree. The word descriptor requires the high computational complexity, whereas this proposed method is simple and easily obtained using the summation and mean value. To perform the gamma correction algorithm to enhance the image quality, the assessment of image quality must be executed first since we can find image quality enhancement method depending upon what image quality is. In numerous gamma correction algorithms, most computations belong to this image quality estimation (Tables 2.4, 2.5 and 2.6).

Therefore, clear and objective image descriptor by this chapter can increase the efficiency and performance of numerous gamma correction or wide dynamic range algorithms.

**Table 2.5**  $NIQ = (NLD, NCD)$  for each gamma value and block size of “baboon” image

Gamma Block size	0.1	0.5	1	1.5	2	2.5	3
$4 \times 4$	(0.93, 0.04)	(0.70, 0.14)	(0.50, 0.19)	(0.36, 0.19)	(0.28, 0.18)	(0.20, 0.16)	(0.15, 0.14)
$8 \times 8$	(0.93, 0.06)	(0.70, 0.21)	(0.50, 0.28)	(0.36, 0.29)	(0.28, 0.27)	(0.19, 0.23)	(0.14, 0.21)
$16 \times 16$	(0.93, 0.08)	(0.70, 0.29)	(0.50, 0.38)	(0.36, 0.39)	(0.28, 0.37)	(0.20, 0.33)	(0.15, 0.29)

**Table 2.6**  $NIQ = (NLD, NCD)$  for each gamma value and block size of “butterfly” image

Gamma Block size	0.1	0.5	1	1.5	2	2.5	3
$4 \times 4$	(0.91, 0.04)	(0.62, 0.12)	(0.41, 0.14)	(0.28, 0.13)	(0.19, 0.12)	(0.15, 0.11)	(0.11, 0.08)
$8 \times 8$	(0.91, 0.07)	(0.62, 0.22)	(0.41, 0.26)	(0.28, 0.25)	(0.19, 0.23)	(0.15, 0.20)	(0.11, 0.18)
$16 \times 16$	(0.90, 0.10)	(0.62, 0.32)	(0.41, 0.40)	(0.28, 0.39)	(0.19, 0.36)	(0.15, 0.32)	(0.11, 0.29)

## 2.2.2 Gamma Correction Based on Tone Mapping Method

### 2.2.2.1 Introduction

In everyday life we experience real world scene displays the wide range of luminance values. The range of light we experience in the course of day is vast. The light of the noon sun can be as much as 10 million times more intense than moonlight. The human visual system is capable of perceiving scenes spanning five orders of magnitude, and adapting more gradually to over nine orders of magnitude. Although adaptation provides visual functioning over a wide range of ambient intensities, this does not mean that we see equally well at all intensity levels. For dim light our eyes are very sensitive to luminance and we can detect small difference in illumination, but acuity and colour sensitivity reduces. That is why we can't read in twilight. In day light we have sharp colour vision but absolute sensitivity is low and to detect luminance difference it should be great enough (Figs. 2.12, 2.13 and 2.14).

In practice media used to display these images is either a video display or a print on chapter and they cannot reproduce the computed luminance, or span more than a few orders of magnitude. The realistic image synthesis has shown that, it is possible to produce images that convey the appearance of the simulated scene by mapping to a set of luminance that can be produced by the display media.

It is widely agreed that image reproduction algorithms should aim to preserve intensity ratios. Using system dynamic range as a measure, we can determine in advance whether preserving intensity ratios is possible or not. If the original scene has a dynamic range that is less than or equal to the system dynamic range, we can preserve intensity ratios. Rendering the image in this scenario is a simple process. However if the original scene's dynamic range is greater than the system dynamic range, intensity ratios cannot be preserved and dynamic range compression algorithms are needed.

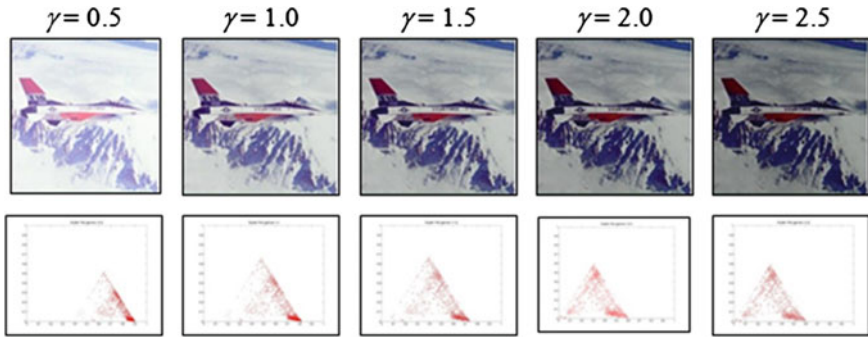


Fig. 2.12 Test images of “airplane” with different gamma value and each scattering chart

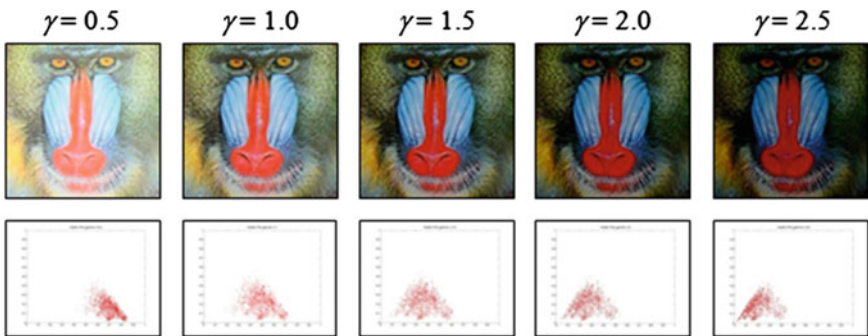


Fig. 2.13 Test images of “baboon” with different gamma value and each scattering chart

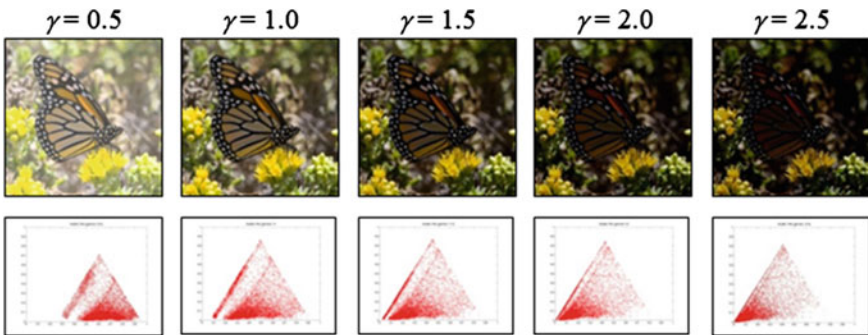
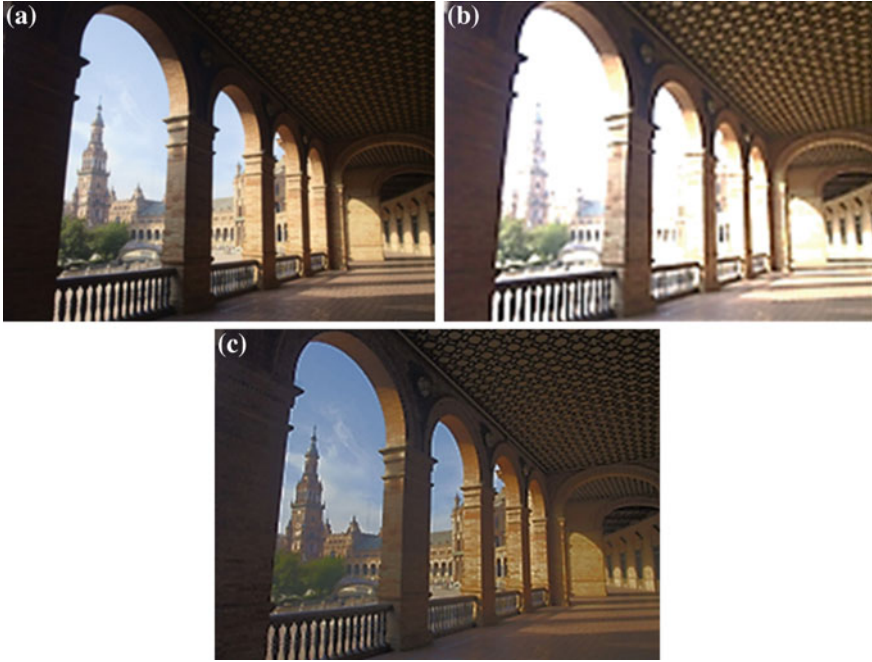


Fig. 2.14 Test images of “butterfly” with different gamma value and each scattering chart

In a typical scene that poses problem for tone reproduction both for photography and computer graphics image synthesis system—an outside sunlit landscape looked through the window. Human eyes can easily adapt the luminance level.



**Fig. 2.15** Example images for high dynamic range: **a** underexposed photograph, **b** over exposed photograph, **c** high dynamic range image

But for photo, this is either overexposed or the dark region is under exposed. This shown in Fig. 2.15a, b along with corrected image Fig. 2.15c with proposed new method.

In this chapter, we will be using this computationally efficient and intuitive method and this approach addresses over or under exposure problem associated with the images capturing wide range of luminance. This tone reproducing operator or tone mapping operator is built from image own data and reliably map high dynamic luminance to display luminance.

### 2.2.2.2 Survey of Other Approaches and Research Trends

Jeffrey et al. [22] have grouped High Dynamic range image algorithms into two categories: tone reproduction curves (TRCs) and tone reproduction operators (TROs). TRCs operate point-wise on the image data and comparatively simpler. Author has reviewed several high dynamic range algorithms based on visual sensitivity. These are the visual concepts we have utilized while designing this method.

TRO use the spatial structure of the image data and attempt to preserve local image contrast by maintaining the local intensity ratios. TRO's detail preserving local gain control algorithms are developed by Pattnaik and Yee [23]. Ramponi [24] have proposed one such special transformation method based on Retinex theory. The comprehensive details can be found in [24–26]. These methods produce satisfactory results. But these are computationally heavy and memory consuming. Jian Duan and Bressan [27] have used Histogram adjustment based local tone mapping method. Segmentation of image and adaptive contrast adjustment has produced good results.

Pattanaik and Ferwada [28] have developed a computational model of visual adaptation for realistic image synthesis based on psychophysical experiments that accounts for changes in spatial acuity and color sensitivities as a function of light level. This may be good guideline for any method.

A tone mapping operator has been proposed by Tumblin and Rushmier [29] using the brightness function based on Steven and Steven [30]. Here preservation of the brightness is at the expense of visibility. TRC operate point-wise on the image data, making the algorithms simple and efficient. Popularly used TRC functions are Power or Logarithmic mapping. These functions compress the dynamic range. A visibility matching tone reproduction operator based on logarithmic function is proposed by Larson and Rushmeier [31]. They have accounted for glare, spatial acuity and colour sensitivities. It requires additional histogram adjustment methods to reduce its contrast exaggeration.

Our work is greatly inspired by these chapters. After studying various methods, this new method is designed to keep simple computation and utilises successful element of past methods and adds new element to make it efficient. Our chapter utilise TRC based tone operator which is computationally simple yet gives better results.

### 2.2.2.3 Main Features of Proposed Method

Visibility maintained along with brightness and contrast. Hence the fine details of the images can be seen clearly without histogram adjustments. For getting required amount of contrast in the image, compression factor is used with tan function. The method essentially increases the contrast in the image. But exaggerated contrast disturbs the correct visual impact and hence we need the adjustment which will be done by minimizing the gap between the luminance counts.

The main features of proposed method are

- (1) Simple and intuitive step solution which gives better contrast for brighter region of image and visibility for dark region.
- (2) Dark images are corrected as well.
- (3) Dynamic range compression using adjustable variable factor.
- (4) It uses sophisticated adoption factor based on trigonometric function Tan and root power to achieve the brightness and contrast level.



- (5) No Histogram adjustments needed.
- (6) Works in both for fully autoamized calculation as well as user control.

For this new method approach we will work on luminance of image and start with the Histogram. For high dynamic range luminance the above example of outdoor sunlight and the inside corridor of the structure or the following case of dark indoor on brightly lit outdoor landscape, the histogram shows that luminance level is occurred in clusters rather than uniform distribution over the range.

It is studied that any method fails that uses a single adaptation level that maps a large range of the sparsely populated real world luminance levels to large range of display level. As described above eyes are sensitive to the bright region with respect to dark region. To sense the difference in the bright light it should be big enough. But in dim light this small difference of illumination is easily detectable.

Step 1: We will start with undistorted image and its histogram.

The histogram of digital image with  $L$  the total possible intensity levels in the range is defined as discreet function

$$h(k) = n_k \quad (2.22)$$

where  $h(k)$  is  $k$ -th intensity level and  $n_k$  is number of pixels at intensity  $k$ . Cumulative Probability frequency distribution  $P(k)$  is given by

$$P(k) = \sum_{k_n < k} T(k_n)/T \quad (2.23)$$

where  $T$  is total number of samples (Fig. 2.16).

Step 2: Let Min and Max represent the level of minimum and maximum intensity present in the image.

Step 3: The result will look like as following probability distribution function in Fig. 2.17.

The new probability frequency distribution  $P_k(K)$  can be calculated and plotted with respect to Minimum and Maximum level of intensity present in the image and the cumulative distribution will look like Fig. 2.18.

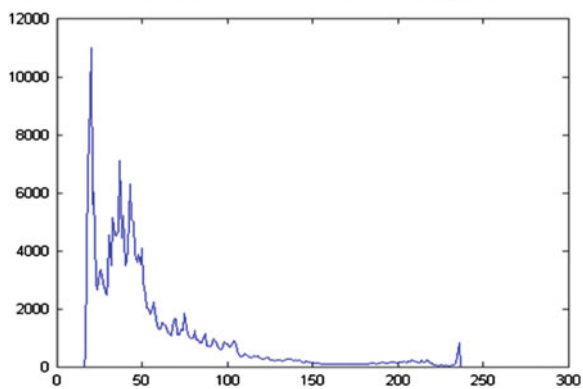
Step 4: We will be following tone reproduction operator for calculating display range as

$$\text{Exp}(DR) = C_f \left\{ \sqrt{[n]} \tan(p_k(k) \times 1.5) \right\} + \log(LD_{\min}) \quad (2.24)$$

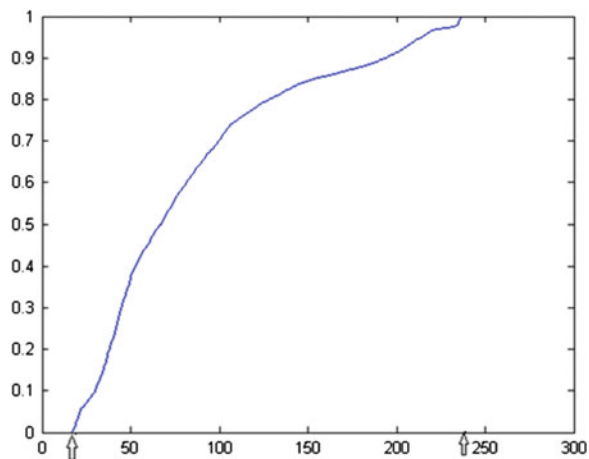
where DR is the display range luminance value,  $C_f$  notes a compression factor,  $L_{\min}$  is the display minimum luminance level in the image.

The display range is restricted less than the difference value  $k_I$  (i.e.  $k_I = \text{Max} - \text{Min}$ ). Various result shows that parameter compact will vary in the range of 0.1–5.54. As this will be maximum limits of natural log limits value 255. But the world luminance to display is as shown Fig. 2.19.

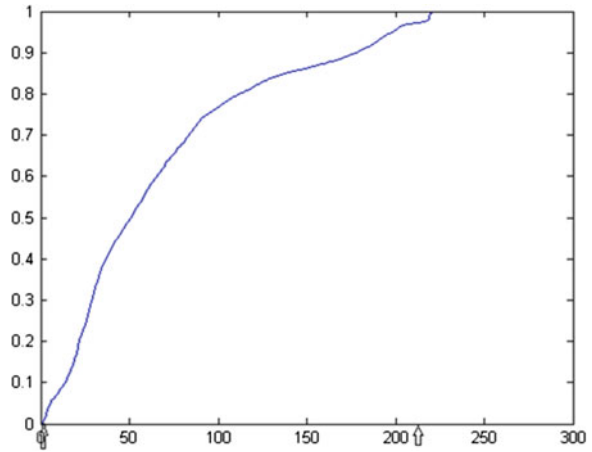
**Fig. 2.16** House on brightly lit landscape (size of image:  $600 \times 400$ )



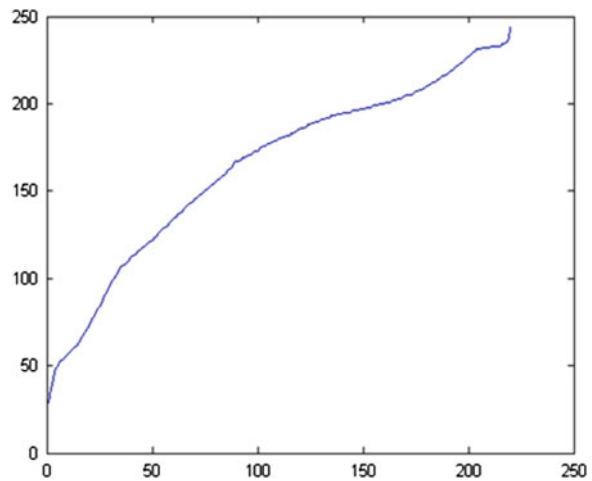
**Fig. 2.17** Curve showing probability distribution function



**Fig. 2.18** Probability distribution replotted with respect to minimum and maximum luminance



**Fig. 2.19** The world luminance to display luminance



This can be done by following loop code.

For c  $C_f$  0.1 to 5 in steps 0.1

For each  $k$  level

if  $k < LD_{max} - LD_{min}$

$$\text{Exp}(DR) = C_f \left\{ \sqrt[n]{\tan(p_k(k) \times 1.5)} \right\} + \log(LD_{min})$$

else

$$\text{Exp}(DR) = C_f \left\{ \sqrt[n]{\tan 2(p_k(k) \times 1.5)} \right\} + \log(LD_{min})$$

endif

Break if range exceeds the  $k1$

Then range as defined above

End for

End for

**Fig. 2.20** Effect comparison of with tan and only cube root function

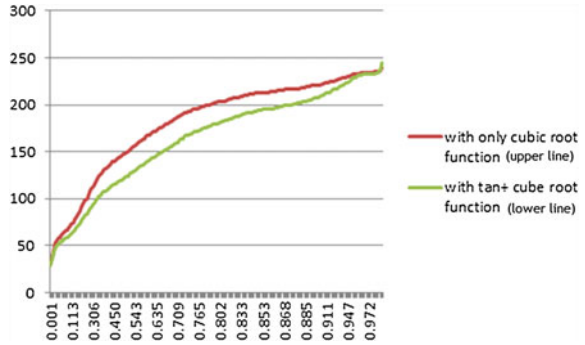


Figure 2.20 shows effect of trigonometric function tan with cube-root function and using only cube root function. Using tan it does add the required contrast in the image. The tan functionality curve is shown in figure and to utilize its full positive range the multiplication factor of 1.5 is used.

Contrast of the image is defined as the ratio of the difference in foreground and background to the background and therefore, adding value to both actually reduces the contrast. Merely using the root function gives benefit of better visibility but it is at the cost of losing fine detail especially in bright region. Hence we have added tan functionality. This one acts as complimentary to cube-root function. So we have taken the advantage of better visibility by using power root function and to overcome its disadvantage tan function is used along-with it.

Note that tan curve at higher luminance level is steep which gives better contrast. The image range get compressed aggressively at sparsely populated region compared to densely populated region and yields the desirable range using the adjustment of compression factor along with contrast adjusting tan function. Sometimes method results into enhanced amount of contrast in the image.

The exaggerated amount of contrast is reduced by closing the gap between two consecutive intensity bin and should be kept as minimum as possible maintaining the intensity ratio. For this the gap between the two levels is calculated and will be reduced linearly (Fig. 2.21).

Step 5: At last using this mapping function, display is reproduced by subtracting minimum luminance value of the image (min).

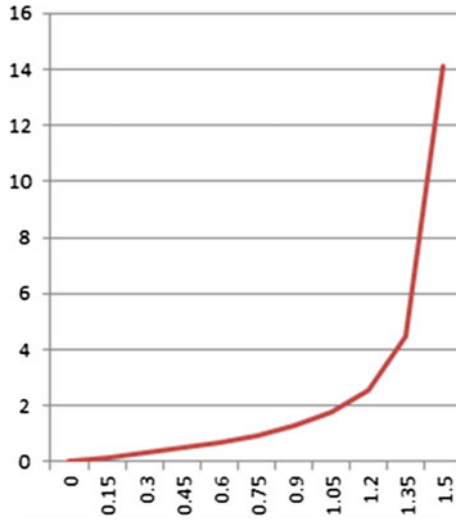


Fig. 2.21 Tan function



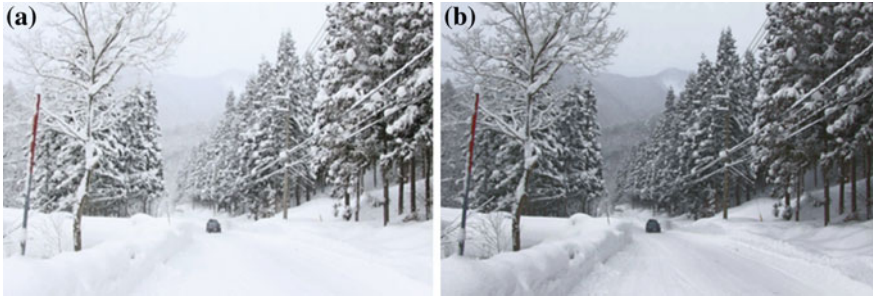
Fig. 2.22 The corrected image with proposed method (sample 1) **a** original image, **b** corrected image

### 2.2.2.4 Test Examples

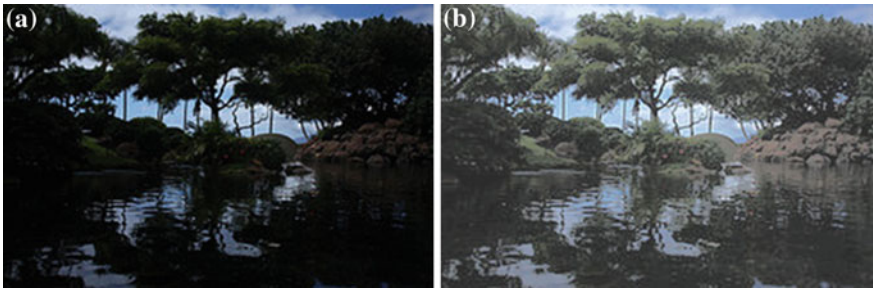
See Figs. 2.22, 2.23, 2.24, 2.25, 2.26, 2.27 and 2.28.

### 2.2.2.5 Comparison with Others

In this chapter we have presented a novel method based on the compression for high dynamic range images. After studying various method this new method is developed to keep fast, computationally simple and automatized. It can have user control for desired control on brightness or contrast. It enhances the contrast of the



**Fig. 2.23** The corrected image with proposed method (sample 2) **a** original image, **b** corrected image

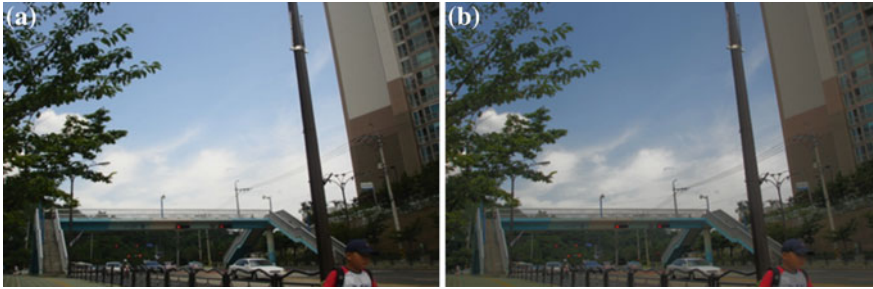


**Fig. 2.24** The corrected image with proposed method (sample 3) **a** original image, **b** corrected image

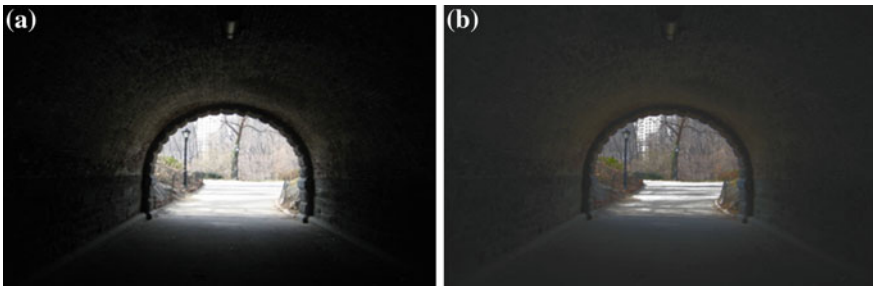


**Fig. 2.25** The corrected image with proposed method (sample 4) **a** original image, **b** corrected image

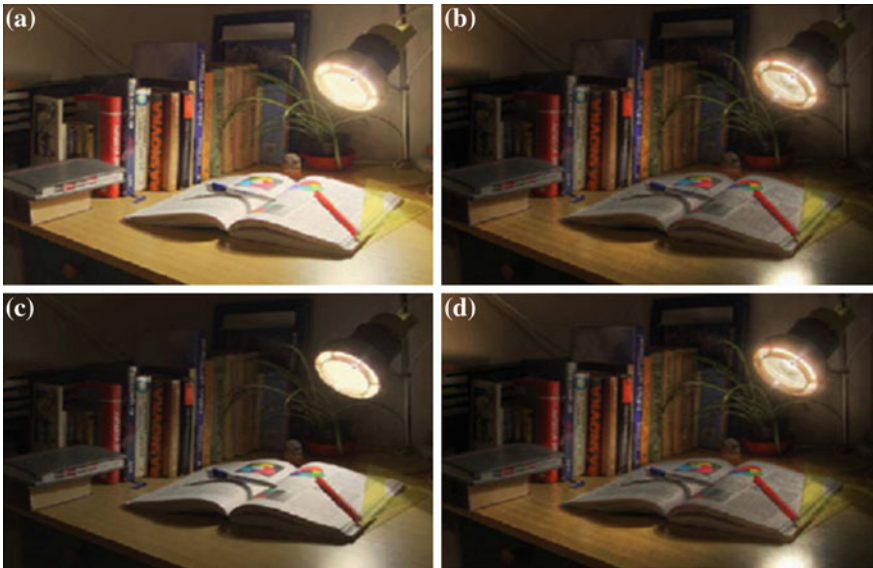
bright region of the image and at the same time increases the visibility of the dark region. For perfect contrast achievement the range variable compression factor is used along-with trigonometric tan function. Also, this is the automated calculation which yield the better result as shown in the examples.



**Fig. 2.26** The corrected image with proposed method (sample 5) **a** original image, **b** corrected image



**Fig. 2.27** The corrected image with proposed method (sample 6) **a** original image, **b** corrected image



**Fig. 2.28** Comparison with other algorithms **a** Pattanaik (02), **b** Durand (02), **c** Tumblin (99), **d** proposed

## References

1. D.C. Brown, Decentering distortion of lenses. *Photogram Eng. Remote Sens.* **24**, 555–566 (1966)
2. D.C. Brown, Close-range camera calibration. *Photogram. Eng. Remote Sens.* **42**, 855–866 (1971)
3. J. Heikkila, O. Silven, in *Proceedings of the 13th International Conference on “Calibration procedure for short focal length off-the-shelf CCD cameras,”* in *Pattern Recognition*, vol. 1 (1996), pp. 166–170
4. M. Yan, Z. Hanqi, What you see is what you get [self-calibrating camera lens distortion]. *IEEE Robot. Autom. Mag.* **11**, 123–127 (2004)
5. M. Ahmed, A. Farag, Nonmetric calibration of camera lens distortion: differential methods and robust estimation. *IEEE Trans. Image Process.* **14**, 1215–1230 (2005)
6. F. Devernay, O. Faugeras, Straight lines have to be straight. *Mach. Vis. Appl.* **13**, 14–24 (2001)
7. B. Prescott, G.F. McLean, Line-based correction of radial lens distortion. *Graph. Models Image Process.* **59**, 39–47 (1997)
8. H. Farid, A.C. Popescu, Blind removal of lens distortion. *J. Opt. Soc. Am. A* **18**, 2072–2078 (2001)
9. J.A. Nelder, R. Mead, A simplex method for function minimization. *Comput. J.* **7**(4), 308–313 (1965)
10. M.T. Ahmed, A.A. Farag, in *Proceedings of the 2001 IEEE Computer Society Conference on “Differential Methods for Nonmetric Calibration of Camera Lens Distortion,”* in *Computer Vision and Pattern Recognition CVPR 2001*, vol. 2 (2001), pp. II-477–II-482
11. R. Cucchiara et al., in *Proceedings of 12th International Conference on “A Hough Transform-Based Method for Radial Lens Distortion Correction,”* in *Image Analysis and Processing* (2003), pp. 182–187
12. S. Graf, T. Hanning, in *IEEE Computer Society Conference on “Analytically solving radial distortion parameters,”* in *Computer Vision and Pattern Recognition, CVPR 2005*, vol. 2 (2005), pp. 1104–1109
13. K. Sirisantisamrid et al., in *IEEE Region 10 Conference “A Simple Technique to Determine Calibration Parameters for Coplanar Camera Calibration,”* in *TENCON 2004*, vol. 1 (2004), pp. 677–680
14. R. Sagawa et al., in *IEEE/RSJ International Conference on “Calibration of Lens Distortion by Structured-Light Scanning,”* in *Intelligent Robots and Systems (IROS 2005)*, (2005), pp. 832–837
15. W. Yu, Image-based lens geometric distortion correction using minimization of average bicoherence index. *Pattern Recogn.* **37**, 1175–1187 (2004)
16. T. Nave, J.M. Francos, in *2nd International Conference on “Global Featureless Estimation of Radial Distortions,”* in *Signal Processing and Communication Systems, ICSPCS 2008* (2008), pp. 1–11
17. R. Tsai, A versatile camera calibration technique for high-accuracy 3D machine vision metrology using off-the-shelf TV cameras and lenses. *IEEE J. Robot. Autom.* **3**, 323–344 (1987)
18. Z. Wentao et al., A high-precision camera operation parameter measurement system and its application to image motion inferring. *IEEE Trans. Broadcast.* **47**, 46–55 (2001)
19. J.P. Oakley, H. Bu, Correction of simple contrast loss in color images. *IEEE Trans. Image Process.* **16**(2), 511–522 (2007)
20. V. Mante, R.A. Frazor, V. Bonin, W. Geisler, M. Carandini, Independence of luminance and contrast in natural scenes and in the early visual system. *Nat. Neurosci.* **8**(12), 1690–1697 (2005)



21. A. Restrepo (Palacios), G. Ramponi, Word descriptors of image quality based on local dispersion-versus-location distributions, in *16th EUSIPCO 2008*, Lausanne, Switzerland, 25–29 August 2008
22. G. Ramponi, Adaptive contrast improvement for still images and video frames, in *IEEE—EURASIP Workshop on Nonlinear Signal and Image Processing, NSIP-07*, Bucharest, Romania, 10–12 Sept 2007
23. S.N. Patnaik, H. Yee, Adaptive gain control for high dynamic range image display, in *Proceedings of Spring Conference in Computer Graphics (SCCG2002)*, Budmerice, Slovak Republic, 24–27 April 2002
24. G. Guarnieri, S. Marsi, G. Ramponi, High dynamic range image display with halo and clipping prevention. *IEEE Trans. Image Process.* **20**(5), 1351–1362 (2011)
25. K. Chiu, M. Herf, P. Shirley, S. Swamy, C. Wang, K. Zimmerman, Spatially nonuniform scaling functions for high contrast images, in *Proceedings of Graphics Interface '93*, Toronto, Canada, May 1993, pp. 245–253
26. R. Fallal, D. Lischinski, M. Werman, Gradient domain high dynamic range compression, in *Proceedings of SIGGRAH* (2002)
27. J. Duan, M. Bressan, C. Dance, G. Qiu, Tone-mapping high dynamic range images by novel histogram adjustment. *Pattern Recogn.* **43**(5), 1847–1862 (2010)
28. J.A. Ferwerda, S.N. Pattanaik, P. Shirley, D.P. Greenberg. A model of visual adaptation for realistic image synthesis. in *Proceedings of the SIGGRAPH'96* (1996), pp. 249–258
29. J. Tumblin, H. Rushmeier, Tone reproduction for realistic images. *IEEE Comput. Graphics Appl.* **13**, 42–48 (1993)
30. S.S. Stevens, J.C. Stevens, Brightness function: parametric effects of adaptation and contrast. *J. Optical Soc. Am.* **53**, 1139 (1960)
31. G.W. Larson, H. Rushmeier, C. Piatko, A visibility matching tone reproduction operator for high dynamic range scenes. *IEEE Trans. Visual. Comput. Graph.* **3**, 291–306 (1997)

# Chapter 3

## Super Resolution

Hyo-Moon Cho

**Abstract** We introduced, in this chapter, what the definition of a super resolution is and what the key approaching methods for major super resolution algorithms are. Numerous super resolution algorithms have based on the observation model and they have followed the warp-blur sequence. But, some cases which have large movements and warp factors such as video by taking in a vehicle are worse than normal interpolation methods. We introduce the smart and robust registration algorithm with rotation and shift estimation. To reduce the registration error, this algorithm decides the optimal reference image even other super resolution algorithms discard this registration error. This algorithm follows the warp-blur observation model because the blurring parameter is much bigger than warp parameter for camera rotation and/or vibration.

### 3.1 Introduction

In most imaging applications, a high resolution image is desired and widely required. Where, “high resolution image” means that an image has not only increasing the number of pixel but also increasing the resolving power. Therefore the resolution is related to the ability to distinguish details in the image.

The International Organization for Standardization (ISO) has described a precise method to measure the resolution of a digital camera [1]. The resolution can be measured as the highest frequency pattern of black and white lines where the individual black and white lines can still be visually distinguished in the image. It is expressed in line widths per picture height.

---

H.-M. Cho (✉)  
Engineering Faculty, Universiti Malaya, Kuala Lumpur, Malaysia  
e-mail: hmcho67@hanmail.net

The standard also describes a method to compute the spatial frequency response (SFR) of a digital camera. The spatial frequency response is the digital imaging equivalent of the modulation transfer function (MTF) used in analogue imaging systems. It describes the variation between the maximum and minimum values that is visible as a function of the spatial frequency that is the number of black and white lines per millimeter. It can be measured using an image of a slanted black and white edge, and is expressed in relative spatial frequencies which is relative to the sampling frequency, line widths per picture height, or cycles per millimeter on the image sensor. The resolution chart that is used in the International Organization for Standardization (ISO) standard is shown in Fig. 3.1.

Image resolution is one of the important factors in digital camera design, since the cameras have been widely used to capture images for numerous imaging applications. Digital cameras have long fast evolved towards a steadily increasing number of pixels. From about 0.3 Mega-pixels in 1993, the number of pixels on the charge-coupled device (CCD) or the complementary metal oxide semiconductor (CMOS) sensor in a digital camera has increased to about 5 Giga-pixels in some of the latest professional models. This pixel count has become the major selling argument for the camera manufacturers. Although the number of pixels of camera is increased, the current resolution grade of digital cameras and their price do not satisfy consumer demands and may not satisfy the future demand also. Thus, finding a way to increase the current resolution level is needed.

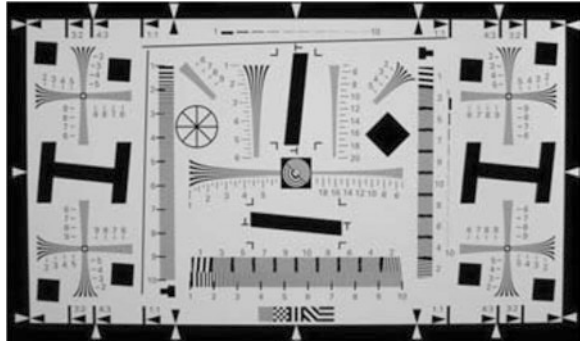
The most direct solution to increase spatial resolution of image is to reduce the pixel sizes of image sensor. It needs the high level sensor manufacturing technology, and the cost to do this is very high. The most critical problem of it is the shot noise that degrades the image quality severely, because the amount of light available is also decreased as decreasing the pixel size. Therefore the method reducing the pixel size has the limitation of the pixel size reduction and the current technical status of this is almost saturated.

The other approach is to increase the chip size, it uses the reverse concept of above method and it leads to an increase in capacitance [2]. This second approach is difficult to speed up a charge transfer rate by large capacitance. Thus, this approach is also not a considered effective.

Therefore, a novel approach to overcome above limitations of sensors and optic manufacturing technologies is required. One promising approach is to use the digital image signal processing techniques to obtain a high resolution image or video sequence from observed multiple low resolution images. Recently, such a resolution enhancement approach has been one of the most active research areas, and it is called super resolution or simply resolution enhancement in the literature [2–62]. The major advantage of super resolution algorithms is that it may costless, and the existing low resolution images can be still utilized. In here, the meaning of using the existing low resolution images is that the obtained high resolution image by a super resolution algorithm consists of real data from several low resolution images not artificial data by computing data from just one image.

The basic condition of super resolution techniques is that the multiple low resolution images captured from the same scene and these are sub-sampled and

**Fig. 3.1** ISO resolution chart used to compute the SFR of a digital camera



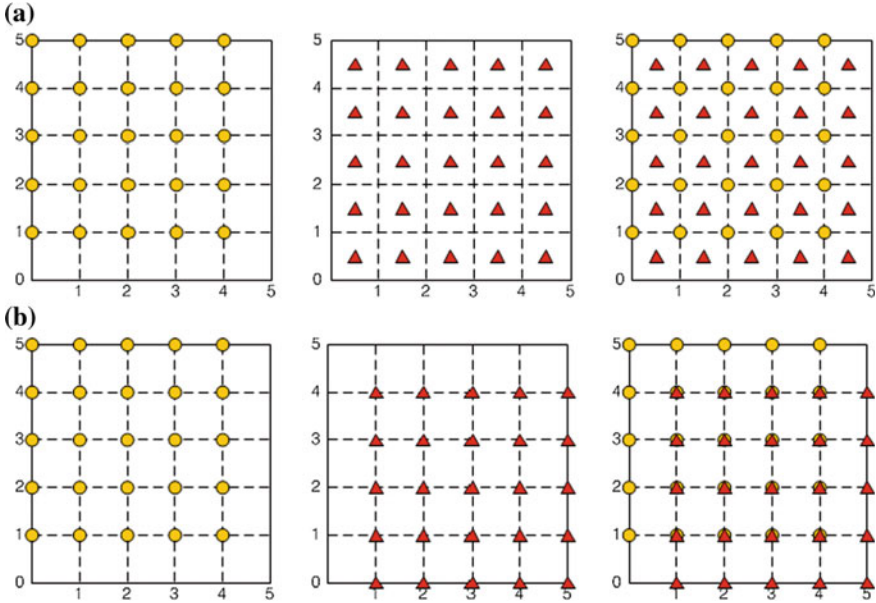
aliased, as well as shifted with sub-pixel displacement. If the low resolution images have different sub-pixel displacement from each other and if aliasing is present, then a high resolution image can be obtained, since the new information in each low resolution image can be exploited to generate a high resolution image as shown in Fig. 3.2a. Whereas, the low resolution images are shifted by integer pixel units, then it is difficult to generate a high resolution image because each image contains the same information for each other as shown in Fig. 3.2b. That is, there is no new information that can be used to reconstruct a high resolution image.

Generally, the super resolution algorithm covers image restoration techniques [63, 64] that produce high quality images from noisy, blurred images although the main concern of it is to reconstruct high resolution image from under-sampled low resolution images. Therefore, the goal of super resolution techniques is to restore a high resolution image from several degraded and aliased low resolution images as illustrated in Fig. 3.3.

Big difference between restoration and super resolution is that the restoration does not change the size of image. In fact, restoration and super resolution reconstruction are closely related theoretically, and super resolution reconstruction can be considered as a second-generation problem of image restoration.

Another problem related to super resolution reconstruction is image interpolation that has been used to increase the size of a single image. Although this field has been extensively studied [65–67], the quality of an image magnified from an aliased low resolution image is inherently limited even though the ideal “sinc” basis function is employed. That is, single image interpolation cannot recover the high-frequency components lost or degraded during the low resolution sampling process. For this reason, image interpolation methods are not considered as super resolution techniques.

To achieve further improvements in this field, the next step requires the utilization of multiple data sets in which additional data constraints from several observations of the same scene can be used. The fusion of information from various observations of the same scene allows us super resolution reconstruction of the scene.



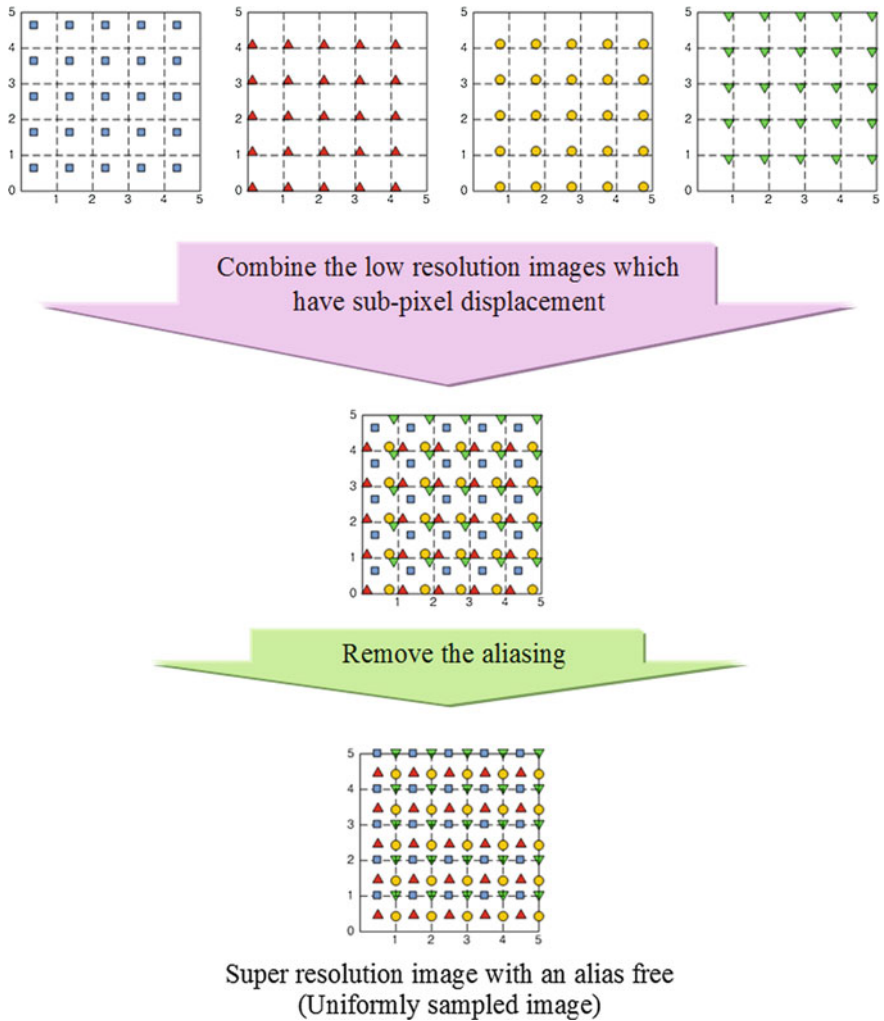
**Fig. 3.2** Basic condition for Super resolution. **a** Sub-pixel displacement. **b** Integer-pixel displacement

### 3.2 Observation Model

To comprehensively analyse the super resolution algorithm, the definition of the relation between a high resolution image and several low resolution images is necessary. One of famous and widely used the image formulation model is the observation model. The basic concept of observation model is if we can know how several low resolution images are generated from a high resolution image, then we can reconstruct a high resolution image from several low resolution images by using reverse process of observation model.

In this chapter, we employed the observation model for video sequence since the goal of this chapter is to obtain super resolution image on general video recording system. Let us denote by  $f(x, y, t)$  the continuous in time and space dynamic scene which is being captured. If the scene is sampled according to the Nyquist criterion in time and space, it is represented by the high resolution sequence  $f_l(m, n)$ , where  $l = 1, \dots, L$ ,  $m = 0, \dots, PM - 1$ , and  $n = 0, \dots, PN - 1$ , the discrete temporal and spatial coordinates, respectively.

For reasons that will become clear right away, the parameter  $P$  is referred to as the magnification factor. Note that although different magnification factors  $P_r$  and  $P_c$  can be used for rows and columns, respectively, for simplicity and without lack of generality, we used the same factor  $P$  for both directions. It is, however, important to low resolution images that depending on the available images we may



**Fig. 3.3** Process sequence of Super resolution technique

not be able to improve the spatial image resolution in both directions at the same degree.

Before we proceed, a matrix-vector representation of images and image sequences is introduced to use in addition with the point-wise representation. Using matrix-vector notation, each  $PM \times PN$  image can be transformed into a  $(PM \times PN) \times 1$  column vector, obtained by lexicographic image ordering.

The  $(PM \times PN) \times 1$  vector that represents the  $l$ -th image in the high resolution sequence is denoted by  $f_l$ , with  $l = 1, \dots, L$ . Additionally, if all frames  $f_l$ ,

$l = 1, \dots, L$ , are lexicographically ordered, the vector  $f$  of dimensions  $(L \times PM \times PN) \times 1$  is obtained.

The high resolution sequence  $f$  is input to the imaging system which generates the low resolution images denoted by  $g$  as illustrated in Fig. 3.4. The goal of super resolution is to obtain a high resolution frame,  $f_k$ , from the available low resolution images. All of the described techniques, however, may be applied to the super resolution of video by using, for example, a sliding window approach, as illustrated in Fig. 3.5. Alternatively, temporally recursive techniques can be developed in estimating a super resolution sequence of images. To obtain  $f_k$ , the imaging system and the temporal relationship between high resolution and low resolution sequences need to be modeled.

$\mathbf{f}_k$ : the lexicographically ordered image of the  $k$ -th high resolution frame, vector  $\mathbf{f}$   
 $\mathbf{g}_k$ : the lexicographically ordered image of the  $k$ -th low resolution frame, vector  $\mathbf{g}$

For the majority of the published work, sought after high resolution images  $f_1, \dots, f_L$ , are assumed to satisfy

$$f_l(m, n) = f_k \left( m + d_{l,k}^x(m, n), n + d_{l,k}^y(m, n) \right) \quad (3.1)$$

where  $d_{l,k}^x(m, n)$  and  $d_{l,k}^y(m, n)$  denote respectively the horizontal and vertical components of the displacement, that is,

$$d_{l,k}(m, n) = \left( d_{l,k}^x(m, n), d_{l,k}^y(m, n) \right) \quad (3.2)$$

The model of Eq. (3.1) is a reasonable one under the assumption of constant illumination conditions in the same scene. It leads to the estimation of the optical flow in the scene, not necessarily, to the estimation of the true motion. Note that the above model applies to both local and global motion. Also note that there may exist pixels in one image for which no motion vector exists (occlusion problem), and pixels for which the displacement vectors are not unique. Finally, note that we are not including noise in the above model, since we will incorporate it later when describing the process to obtain the low resolution observations.

Equation (3.1) can be rewritten using matrix-vector notation as

$$\mathbf{f}_l = \mathbf{C}(\mathbf{d}_{l,k}) \mathbf{f}_k \quad (3.3)$$

where  $\mathbf{C}(\mathbf{d}_{l,k})$  is the  $(PM \times PN) \times (PM \times PN)$  matrix that maps frame  $\mathbf{f}_l$  to frame  $\mathbf{f}_k$ , and  $\mathbf{d}_{l,k}$  is the  $(PM \times PN) \times 2$  matrix defined by lexicographically ordering the vertical and horizontal components of the displacements between the two frames. We will be using the scalar and matrix-vector notation interchangeably through this manuscript.

The motion estimation problem, as encountered in many video processing applications, consists of the estimation of  $\mathbf{d}_{l,k}$  or  $\mathbf{C}(\mathbf{d}_{l,k})$  given  $\mathbf{f}_l$  and  $\mathbf{f}_k$ . What makes the problem even more challenging in super resolution is the fact that

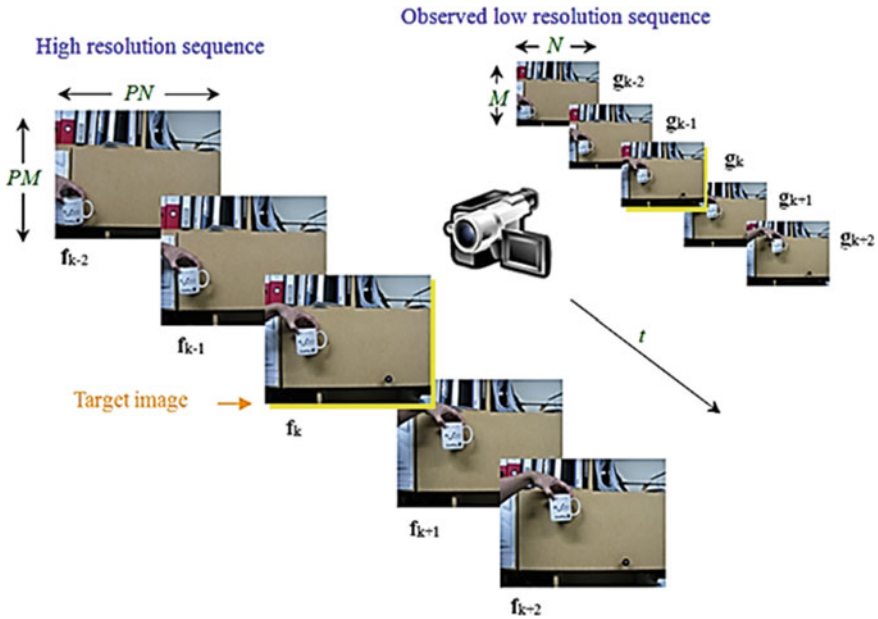


Fig. 3.4 Low resolution video acquisition model

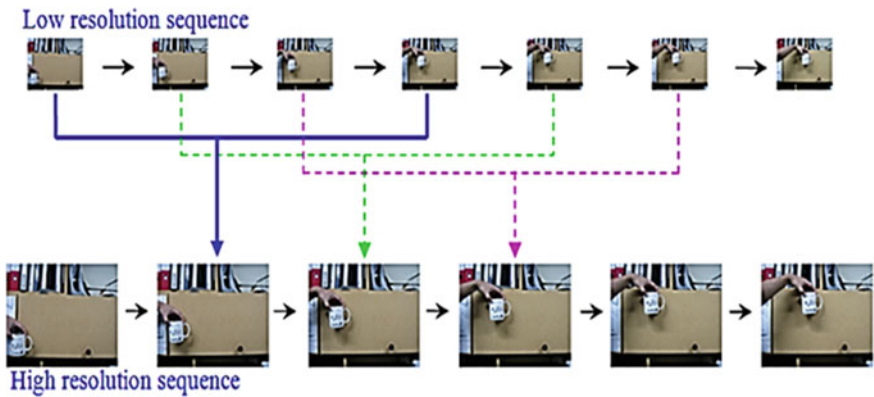


Fig. 3.5 Obtaining sequence of high resolution images from a set of low resolution images (The sliding window approach)

although the high resolution motion vector field is required, to get the high resolution images are not available, and therefore this field must be estimated utilizing the low resolution images. The accuracy of the  $\mathbf{d}_{l,k}$  is of the outmost important in determining the quality of the sought after high resolution images.



### 3.2.1 The Warp-Blur Model

As the name implies, with this model the warping of an image is applied before it is blurred. This case is shown as Fig. 3.6.

The low resolution discrete sequence is denoted by  $\mathbf{g}_l(i, j)$ , with  $i = 0, \dots, M - 1, j = 0, \dots, N - 1$ . Using matrix-vector notation, each low resolution image is denoted by the  $(M \times N) \times 1$  vector  $\mathbf{g}_l$ . The low resolution image  $\mathbf{g}_l$  is related to the high resolution image  $\mathbf{f}_l$  by

$$\mathbf{g}_l = \mathbf{A}_l \mathbf{H}_l \mathbf{f}_l + \eta_l \quad (3.4)$$

where the matrix  $\mathbf{H}_l$  of size  $(PM \times PN) \times (PM \times PN)$  describes the filtering of the high resolution image,  $\mathbf{A}_l$  is the down sampling matrix of size  $MN \times (PM \times PN)$ , and  $\eta_l$  denotes the observation noise. The matrices  $\mathbf{A}_l$  and  $\mathbf{H}_l$  are generally assumed to be known.

Equation (3.4) expresses the relationship between the low resolution and high resolution frames  $\mathbf{g}_l$  and  $\mathbf{f}_l$ , while Eq. (3.3) expresses the relationship between frames  $l$  and  $k$  in the high resolution sequence. Combining these two equations we obtain the following equation which describes the acquisition of a low resolution image  $\mathbf{g}_l$  from the unknown high resolution image  $\mathbf{f}_k$ ,

$$\mathbf{g}_l = \mathbf{A}_l \mathbf{H}_l \mathbf{C}(\mathbf{d}_{l,k}) \mathbf{f}_k + \eta_l + \mu_{l,k} = \mathbf{A}_l \mathbf{H}_l \mathbf{C}(\mathbf{d}_{l,k}) \mathbf{f}_k + \mathbf{e}_{l,k} \quad (3.5)$$

where  $\mu_{l,k}$  represents the registration noise and  $\mathbf{e}_{l,k}$  represents the combined acquisition and registration noise. It is clear from Eq. (3.5) that  $\mathbf{C}(\mathbf{d}_{l,k})$ —the warp—is applied first on  $\mathbf{f}_k$ , followed by the application of the blur  $\mathbf{H}_l$ . This process is pictorially illustrated in Fig. 3.7.

Note that the above equation shows the dependency of  $\mathbf{g}_l$  on both unknowns, the high resolution image  $\mathbf{f}_k$  and the motion vectors  $\mathbf{d}_{l,k}$ . This observation model was first formulated in [34], without matrix notation, and later written in matrix form by [34]. Wang and Qi [68] attributes this model to [31]. The acquisition model utilized in [11] for deriving frequency domain super resolution methods can also be written using this model. If we assume that the noise  $\mathbf{e}_{l,k}$  in Eq. (3.5) is Gaussian with zero mean and variance  $\sigma^2$ , denoted by  $N(0, \sigma^2 I)$ , the above equation produces the following conditional probability density functions to be used within the Bayesian framework,

$$\mathbf{P}_G(\mathbf{g}_l | \mathbf{f}_k, \mathbf{d}_{l,k}) \propto \exp \left[ -\frac{1}{2\sigma^2} \|\mathbf{g}_l - \mathbf{A}_l \mathbf{H}_l \mathbf{C}(\mathbf{d}_{l,k}) \mathbf{f}_k\|^2 \right] \quad (3.6)$$

such as a noise model has been used widely.

A uniform noise model is proposed by [25–28]. The noise model used by these authors is oriented toward the use of the projection onto convex sets (POCS) method in super resolution problems. The associated conditional probability density functions has the form

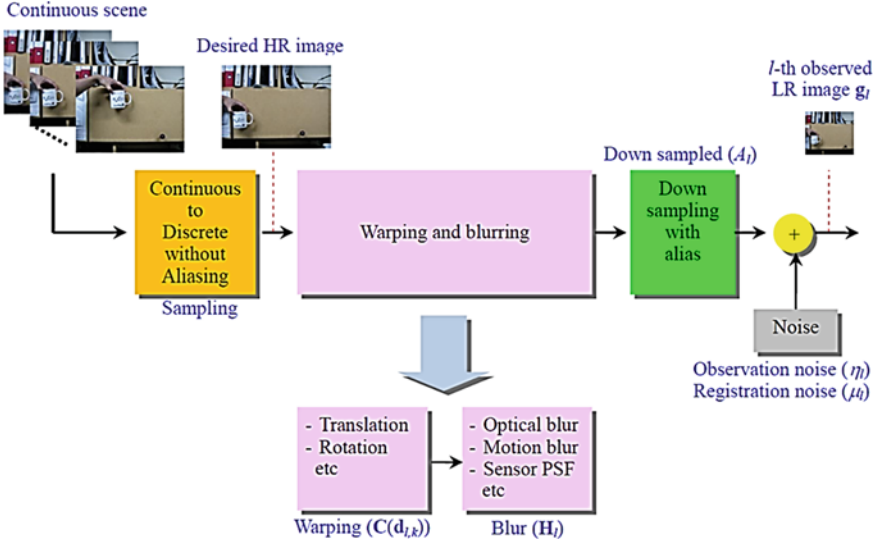


Fig. 3.6 Warp-blur model relating low resolution images to high resolution images

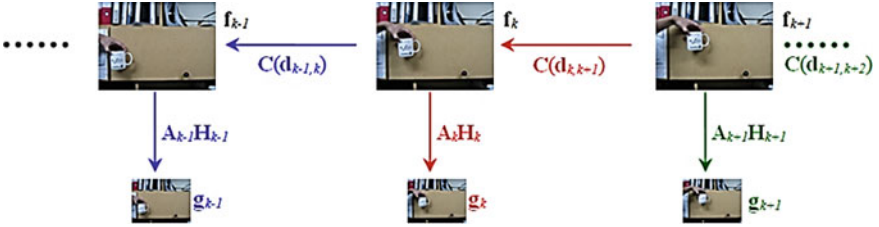


Fig. 3.7 Graphical depiction of the relationship between the observed low resolution images and the high resolution images

$$\mathbf{P}_G(\mathbf{g}_l | \mathbf{f}_k, \mathbf{d}_{l,k}) \propto \begin{cases} \text{const} & \text{if } |[\mathbf{g}_l - \mathbf{A}_l \mathbf{H}_l \mathbf{C}(\mathbf{d}_{l,k}) \mathbf{f}_k](i)| \leq c, \forall i \\ 0 & \text{elsewhere} \end{cases} \quad (3.7)$$

where the interpretation of the index  $i$  is that it represents the  $i$ -th element of the vector inside the brackets.

The zero value of  $c$  can be thought of as the limit of  $\mathbf{P}_G(\mathbf{g}_l | \mathbf{f}_k, \mathbf{d}_{l,k})$  in Eq. (3.6) when  $\sigma = 0$ . Farsiu et al. [69, 70] have recently proposed the use of a generalized Gaussian Markov random field (GGMRF) [71] to model the noise in the image formation process for super resolution problems. Thus, Eq. (3.7) can be written as

$$\mathbf{P}_{GG}(\mathbf{g}_l | \mathbf{f}_k, \mathbf{d}_{l,k}) \propto \exp \left[ -\frac{1}{2\sigma^p} \|\mathbf{g}_l - \mathbf{A}_l \mathbf{H}_l \mathbf{C}(\mathbf{d}_{l,k}) \mathbf{f}_k\|_p^p \right] \quad (3.8)$$

### 3.2.2 The Blur-Warp Model

Another acquisition model which has been used in the literature [29, 71, 72] first considers the blurring of the high resolution image, followed by warping and down-sampling, as shown in Fig. 3.8. In this case, the observation model becomes

$$\mathbf{g}_l = \mathbf{A}_l \mathbf{H}_l \mathbf{M}(\mathbf{m}_{l,k}) \mathbf{f}_k + \eta_l + \mu_{l,k} = \mathbf{A}_l \mathbf{M}(\mathbf{m}_{l,k}) \mathbf{B}_l \mathbf{f}_k + \mathbf{w}_{l,k} \quad (3.9)$$

where  $\mathbf{w}_{l,k}$  denotes the acquisition and registration noise,  $\mathbf{B}_l$  the blurring matrix for the  $l$ -th high resolution image,  $\mathbf{M}(\mathbf{m}_{l,k})$  the motion compensation operator for the blurred high resolution images through the use of motion vector  $\mathbf{m}_{l,k}$ , and  $\mathbf{A}_l$  again the down-sampling matrix.

Different notation has been used in Eqs. (3.5) and (3.9) for the blur and warping operators in order to distinguish between these two models for the rest of the text. The three-conditional probability density functions in Eqs. (3.6)–(3.8) can be rewritten now for the blur-warp model, by substituting  $\mathbf{A}_l \mathbf{H}_l \mathbf{C}(\mathbf{d}_{l,k})$  by  $\mathbf{A}_l \mathbf{M}(\mathbf{m}_{l,k}) \mathbf{B}_l$  (for brevity we do not reproduce them here). The question as to which of the two models (blur-warp or warp-blur) should be used is addressed in [68]. The authors claim that when the motion has to be estimated from the low resolution images, using the warp-blur model may cause systematic errors and, in this case, it is more appropriate to use the blur-warp model. They showed that when the imaging blur is spatiotemporally shift invariant and the motion has only a global translational component the two models coincide. Note that in this case, the blur and motion matrices correspond to convolution matrices and thus they commute.

Before concluding this section on image formation for uncompressed observations, we mention here that for both the warp-blur and the blur-warp models we have defined conditional probability density functions for each low resolution observation  $\mathbf{g}_l$  given  $\mathbf{f}_k$  and  $\mathbf{d}_{l,k}$ . Our goal, however, is to define the conditional probability density functions  $\mathbf{P}(\mathbf{g}|\mathbf{f}_k, \mathbf{d})$ , that is, the distribution when all the observations  $\mathbf{g}$  and all the motion vectors  $\mathbf{d}$  for compensating the corresponding high resolution frames to the  $k$ -th frame are taken into account. The approximation used in the literature for this joint-conditional probability density functions is

$$\mathbf{P}(\mathbf{g}|\mathbf{f}_k, \mathbf{d}) = \prod_{l=1}^L \mathbf{P}(\mathbf{g}_l|\mathbf{f}_k, \mathbf{d}_{l,k}) \quad (3.10)$$

which implies that the low resolution observations are independent given the unknown high resolution image  $\mathbf{f}_k$  and motion vectors  $\mathbf{d}$ .

## 3.3 Survey of the Super Resolution Algorithms

The idea of super resolution was first introduced in 1984 by Tsai and Huang [11] for multi-frame image restoration of band-limited signals. A good overview of existing algorithms is given by [3] and [73]. Most super resolution methods are

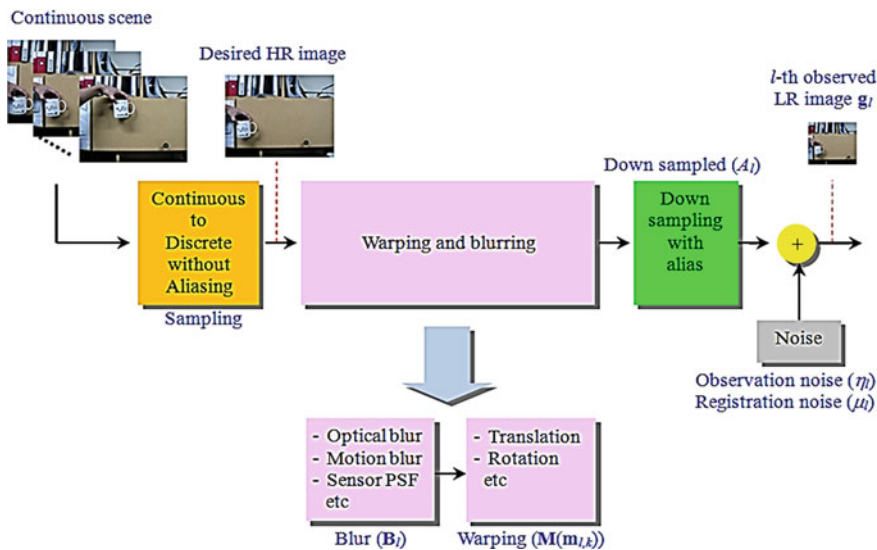


Fig. 3.8 Blur-warp model relating low resolution images to high resolution images

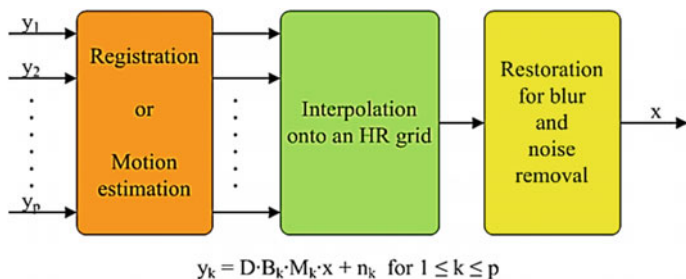


Fig. 3.9 Scheme for super resolution

composed of two main steps: first all the images are aligned in the same coordinate system in the registration step, and then a high-resolution image is reconstructed from the irregular set of samples. In this second step, the camera point spread function is often taken into account. The scheme of super resolution is illustrated in Fig. 3.9.

Precise sub-pixel image registration is a basic requirement for a good reconstruction. If the images are inaccurately registered, the high-resolution image is reconstructed from incorrect data and is not a good representation of the original signal. Zitova and Flusser [74] presents an overview of image registration methods. Registration can be done either in spatial or in frequency domain. By the nature of the Fourier transform, frequency domain methods are limited to global motion models. In general, they also consider only planar shifts and possibly

planar rotation and scale, which can be easily expressed in Fourier domain. However, aliasing is much easier to describe and to handle in frequency domain than in spatial domain.

### 3.3.1 Registration

#### 3.3.1.1 Frequency Approach

Tsai and Huang [11] describes an algorithm to register multiple frames simultaneously using nonlinear minimization in frequency domain. Their method for registering multiple aliased images is based on the fact that the original, high resolution signal is band-limited. They derived a system equation that describes the relationship between low resolution images and a desired high resolution image by using the relative motion between low resolution images. The frequency domain approach is based on the following three principles: (i) the shifting property of the Fourier transform, (ii) the aliasing relationship between the continuous Fourier transform (CFT) of an original high resolution image and the discrete Fourier transform (DFT) of observed low resolution images, (iii) and the assumption that an original high resolution image is band-limited.

These properties make it possible to formulate the system equation relating the aliased discrete Fourier transform (DFT) coefficients of the observed low resolution images to a sample of the continuous Fourier transform (CFT) of an unknown image. For example, let us assume that there are two one-dimension low resolution signals sampled below the Nyquist sampling rate. From the above three principles, the aliased low resolution signals can be decomposed into the un-aliased high resolution signal as shown in Fig. 3.9.

Let  $f_l(m, n)$  denote a continuous high resolution image and  $\mathbf{F}_l(w_m, w_n)$  be its continuous Fourier transform (CFT). The global translations, which are the only motion considered in the frequency domain approach, yield the  $k$ -th shifted image of Eq. (3.1). By the shifting property of the continuous Fourier transform (CFT), the continuous Fourier transform of the shifted image,  $\mathbf{F}_k(w_m, w_n)$ , can be written as

$$\mathbf{F}_k(\mathbf{W}_m, \mathbf{W}_n) = \exp\left[j2\pi\left(d_{l,k}^x(m, n)\mathbf{W}_m, n + d_{l,k}^y(m, n)\mathbf{W}_n\right)\right] \mathbf{F}_l(\mathbf{W}_m, \mathbf{W}_n) \quad (3.11)$$

The shifted image  $f_k(m, n)$  is sampled with the sampling period  $T_m$  and  $T_n$  to generate the observed low resolution image  $g_k(m, n)$ . From the aliasing relationship and the assumption of band-limitedness of  $\mathbf{F}_l(w_m, w_n)$

$$|\mathbf{F}_k(\mathbf{W}_m, \mathbf{W}_n)| = 0 \text{ for } |\mathbf{W}_m| \geq (L_m\pi/T_m), |\mathbf{W}_n| \geq (L_n\pi/T_n) \quad (3.12)$$

The relationship between the continuous Fourier transform (CFT) of the high resolution image and the discrete Fourier transform (DFT) of the  $k$ -th observed low resolution image can be written as [75]

$$\gamma_k[\Omega_m, \Omega_n] = \frac{1}{T_m T_n} \sum_{m=0}^{L_m-1} \sum_{n=0}^{L_n-1} \left\{ \mathbf{F}_k \times \left( \frac{2\pi}{T_m} \left( \frac{\Omega_m}{M} + m \right), \frac{2\pi}{T_n} \left( \frac{\Omega_n}{M} + n \right) \right) \right\} \quad (3.13)$$

By using lexicographic ordering for the indices  $m, n$  on the right-hand side and  $k$  on the left-hand side, a matrix vector form is obtained as:

$$\mathbf{Y} = \Phi \mathbf{X} \quad (3.14)$$

where  $\mathbf{Y}$  is a  $p \times 1$  column vector with the  $k$ -th element of the discrete Fourier transform (DFT) coefficients of  $y_k[m, n]$ ,  $\mathbf{F}$  is a  $L_m L_n \times 1$  column vector with the samples of the unknown continuous Fourier transform of  $f_i(m, n)$ , and  $\Phi$  is a  $p \times L_m L_n$  matrix which relates the discrete Fourier transform of the observed low resolution images to samples of the continuous high resolution image.

Therefore, the reconstruction of a desired high resolution image requires us to determine  $\Phi$  and solve this inverse problem. It is not clear, however, if such a solution is unique and if such an algorithm will not converge to a local minimum. Most of the frequency domain registration methods are based on the fact that two shifted images differ in frequency domain by a phase shift only, which can be found from their correlation. Using a log-polar transform of the magnitude of the frequency spectra, image rotation and scale can be converted into horizontal and vertical shifts. These can therefore also be estimated using a phase correlation method.

### 3.3.1.2 Phase Shift and Correlation

Reddy and Chatterji [76 and 77] describe such planar motion estimation algorithms. Authors apply a high-pass emphasis filter to strengthen high frequencies in the estimation. Kim and Su [78, 79 and 80] also apply a phase correlation technique to estimate planar shifts. To minimize errors due to aliasing, their methods rely on a part of the frequency spectrum that is almost free of aliasing. Typically this is the low-frequency part of the images. [81] showed that the signal power in the phase correlation corresponds to a poly phase transform of a filtered unit impulse. [82] developed a rotation estimation algorithm based on the property that the magnitude of the Fourier transform of an image and the mirrored version of the magnitude of the Fourier transform of a rotated image has a pair of orthogonal zero-crossing lines. The angle that these lines make with the axes is equal to half the rotation angle between the two images. The horizontal and vertical shifts are estimated afterwards using a standard phase correlation method.

### 3.3.1.3 Regularization

An extension of this approach for a blurred and noisy image was provided by [12], resulting in a weighted least squares formulation. In their approach, it is assumed

that all low resolution images have the same blur and the same noise characteristics. This method was further refined by [13] to consider different blurs for each low resolution image. Here, the Tikhonov regularization method is adopted to overcome the ill-posed problem resulting from blur operator. Bose et al. [14] proposed the recursive total least squares method for super resolution reconstruction to reduce effects of registration errors (errors in  $\Phi$ ). A discrete cosine transform (DCT) based method was proposed by [15]. They reduce memory requirements and computational costs by using discrete cosine transform (DCT) instead of discrete Fourier transform (DFT). They also apply multichannel adaptive regularization parameters to overcome ill-posed such as underdetermined cases or insufficient motion information cases.

Theoretical simplicity is a major advantage of the frequency domain approach. That is, the relationship between low resolution images and the high resolution image is clearly demonstrated in the frequency domain. The frequency method is also convenient for parallel implementation capable of reducing hardware complexity. However, the observation model is restricted to only global translational motion and LSI blur. Due to the lack of data correlation in the frequency domain, it is also difficult to apply the spatial domain a priori knowledge for regularization.

Generally, the super resolution image reconstruction approach is an ill-posed problem because of an insufficient number of low resolution images and ill-conditioned blur operators. Procedures adopted to stabilize the inversion of ill-posed problem are called regularization. In this section, we present deterministic and stochastic regularization approaches for super resolution image reconstruction. Typically, constrained least squares (CLS) and maximum a posteriori (MAP) super resolution image reconstruction methods are introduced.

### 3.3.1.4 Spatial Approach

Spatial domain methods generally allow for more general motion models, such as homographies. They can be based on the whole image or on a set of selected corresponding feature vectors, as discussed by [83] and by RANSAC algorithm [84]. Keren et al. [85] developed an iterative planar motion estimation algorithm based on Taylor expansions. A pyramidal scheme is used to increase the precision for large motion parameters. A hierarchical framework to estimate motion in a multi resolution data structure is described in [86]. Different motion models, such as affine flow or rigid body motion, can be used in combination with this approach. Irani et al. [87] presented a method to compute multiple, possibly transparent or occluding motions in an image sequence. Motion is estimated using an iterative multi resolution approach based on planar motion. Different objects are tracked using segmentation and temporal integration. Gluckman [88] described a method that first computes planar rotation from the gradient field distribution of the images to be registered. Planar shifts are then estimated after cancellation of the rotation using a phase correlation method.

### 3.3.2 Reconstruction

#### 3.3.2.1 Interpolation-Based and Frequency Domain

In the subsequent image reconstruction phase, a high resolution image is reconstructed from the irregular set of samples that is obtained from the different low-resolution images. This can be achieved using an interpolation-based method as the one used by [85]. Tsai and Huang [11] describes a frequency domain method, writing the Fourier coefficients of the high-resolution image as a function of the Fourier coefficients of the registered low-resolution images. The solution is then computed from a set of linear equations. This algorithm uses the same principle as the formulation in time domain given by [89].

#### 3.3.2.2 POCS

A high-resolution image can also be reconstructed using a projection onto convex sets (POCS) algorithm [27], where the estimated reconstruction is successively projected on different convex sets. Each set represents constraints to the reconstructed image that are based on the given measurements and assumptions about the signal. Capel and Zisserman [83] and [90] use a maximum a posteriori (MAP) statistical method to build the high-resolution image.

Other methods iteratively create a set of low-resolution images from the estimated image using the imaging model. The estimate is then updated according to the difference between the real and the simulated low-resolution images [32, 85]. This method is known as iterative back-projection. Zomet et al. [91] improved the results obtained with typical iterative back-projection algorithms by taking the median of the errors in the different back-projected images. This proved to be more robust in the presence of outliers. Farsiu et al. [70] proposed a new and robust super resolution algorithm.

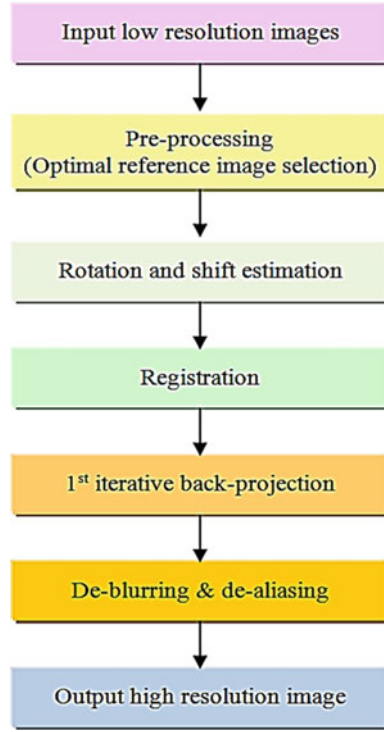
Instead of the more common L2 minimization, they use the L1 norm, which produces sharper high-resolution images. They also showed that this approach performs very well in combination with the algorithm by [91]. Elad and Feuer [31] present a super resolution framework that combines a maximum-likelihood/MAP approach with a projection onto convex sets (POCS) approach to define a new convex optimization problem. Next, they show the connections between their method and different classes of other existing methods.

## 3.4 Novel Super Resolution Registration Algorithm Based on Frequency

In this chapter, we show that the flowchart of the proposed algorithm and each implementation sources. And we describe the detail methodologies and show their experiment results such as the obtained high resolution images, their image quality



**Fig. 3.10** Main flowchart of the proposed algorithm



and the computational complexity comparing with the results of other super resolution algorithms. First, we show our main flow chart as in Fig. 3.10.

Secondly, we obtained the low resolution video sequence by applying the down-sampling factor of two into the original video sequence, as shown in Fig. 3.11, and their resolution size is  $320 \times 240$ .

### 3.4.1 Pre-processing

In the second step, we designed the automatic low resolution input image selection algorithm to reduce the registration error. In the whole video sequence, there are unsuitable images according to the reference image. Therefore, it is very important to choose this.

The video sequence has some linearity since it is made with 30 frames per second (fps) or 25 frames per second (fps). However, the accuracy of this is not high. According to the numerous literatures for the motion estimation and the motion compensation, the probability of inner 1/4-pixel distance motion vector is over 90 % for the practical video sequences, and the motion compensation error has maximum value at 1/2-pixel distance [92–100] as shown in Fig. 3.12.

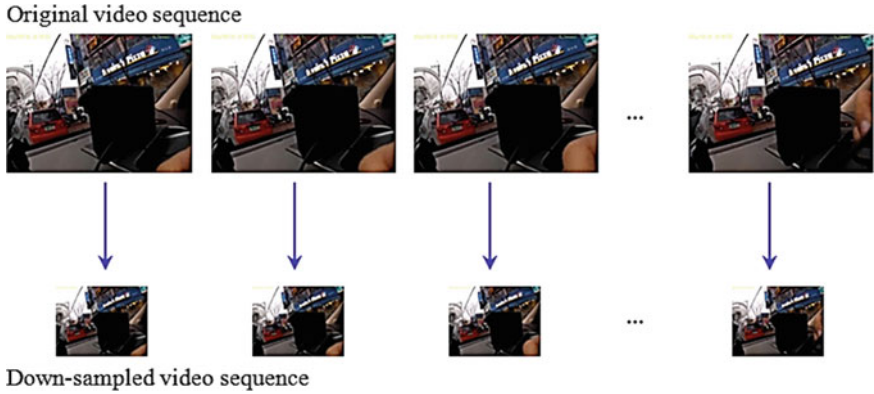
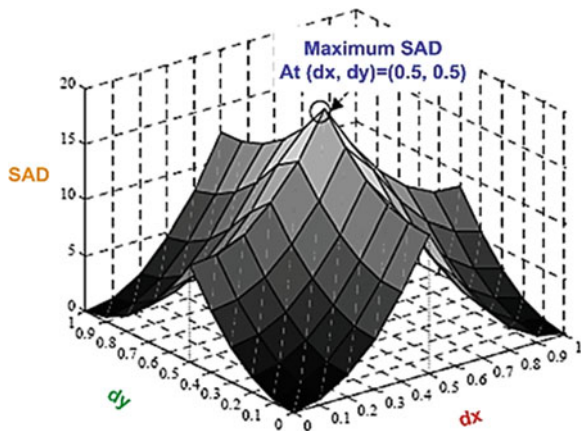


Fig. 3.11 Input low resolution video sequence generating scheme

Fig. 3.12 Distribution of the registration error depending on the sub-pixel shift



We designate the center image to the reference input image in the specified video sequence window, and analysis the registration error for each reference image and its comparing input low resolution images, at this time, we restrict the maximum number of input low resolution image is limited as five frames. The reason of this, it has very high computational complexity than others if we used many input low resolution images. The registration error is computed by the sum of difference (SAD) computing method since it can easily and simply calculate the motion compensation. Where, we assume that the block size computing the sum of difference (SAD) is as  $8 \times 8$  to low computational complexity. Thus, the sum of difference (SAD) calculation allows us to take the motion compensation error (MCE). If the sum of difference (SAD) of one input low resolution image (ILRI) has  $0 \leq SAD \leq$  maximum motion compensation error (MMCE, it is same with maximum SAD), then we can select it as an input low resolution image candidate (ILRIC). It is illustrated in Fig. 3.13.

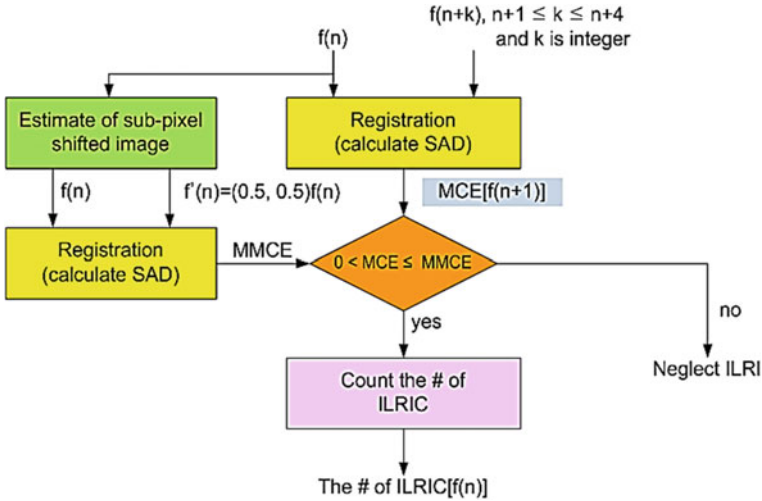


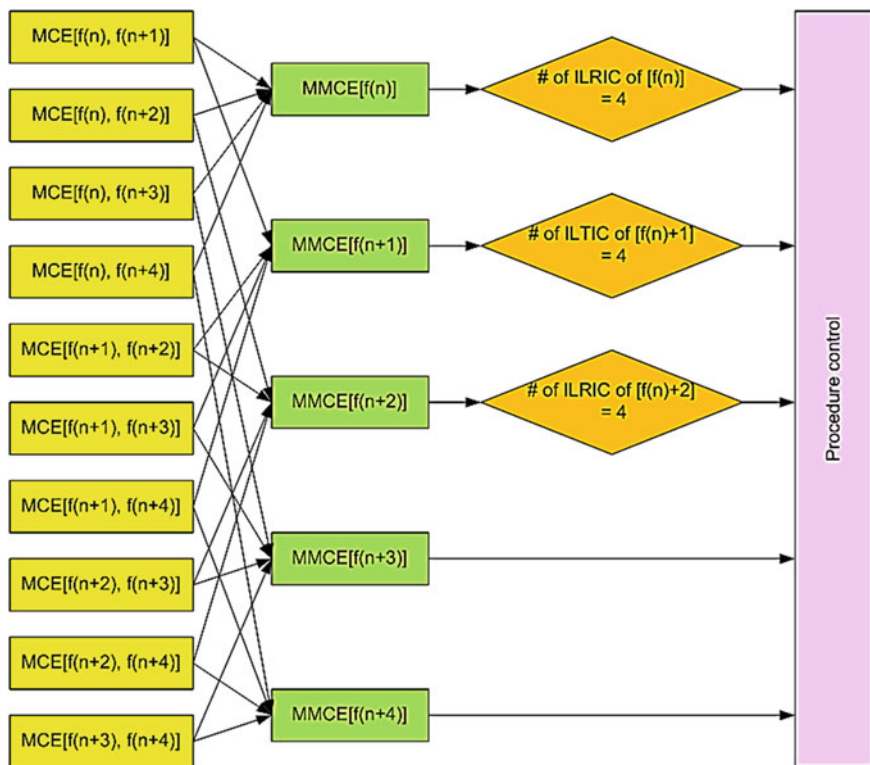
Fig. 3.13 The flowchart of input low resolution image candidate selection

In the next step, it compares the number of the input low resolution image candidates of each reference image. One reference image which has the largest the input low resolution image candidates is chosen as the optimal reference image. And also we propose an advanced architecture to choose the reference image to reduce computational complexity as shown in Fig. 3.14. This method can remove the duplication of the sum of absolute difference (SAD) calculation based on the partial distortion elimination (PDE) method at each frame. The basic concept of it is that if the difference between current and candidate block has small value then this candidate has higher probability to the optimal reference. Therefore, it is more efficient whenever as an input image which has larger initial accumulated sum of absolute difference (SAD) value is selected.

### 3.4.2 Planar Motion Estimation

Fourier based image registration methods only allow global motion in a plane parallel to the image plane. In such a case, the motion between two images can be described as a function of three parameters that are all continuous variables: horizontal and vertical shifts  $x_{1,h}$  and  $x_{1,v}$  and a planar rotation angle  $\theta_1$ .

A frequency domain approach allows us to estimate the horizontal and vertical shift and the (planar) rotation separately. Assume we have a continuous two-dimensional reference signal  $f_0(x)$  and its shifted and rotated version  $f_1(x)$ :



**Fig. 3.14** The flowchart of an advanced choosing the reference image based on the partial distortion elimination (PDE)

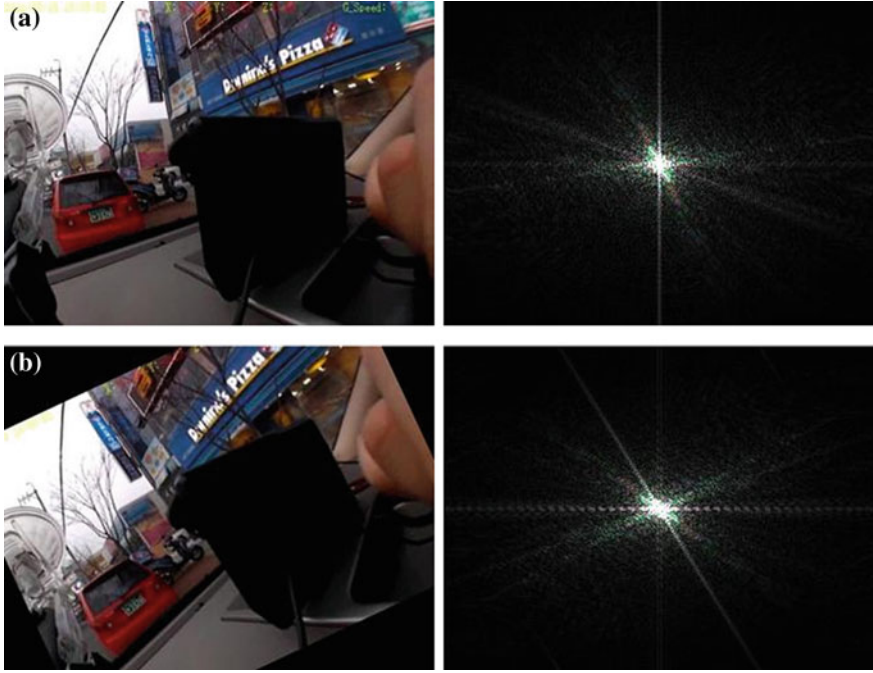
$$f_1(x) = f_0(R(x + x_1)) \quad (3.15)$$

$$\text{with } x = \begin{pmatrix} x_h \\ x_v \end{pmatrix}, x_1 = \begin{pmatrix} x_{1,h} \\ x_{1,v} \end{pmatrix}, R = \begin{pmatrix} \cos \theta_1 & -\sin \theta_1 \\ \sin \theta_1 & \cos \theta_1 \end{pmatrix}$$

This can be expressed in Fourier domain as

$$\begin{aligned} F_1(u) &= \iint_x f_1(x) e^{-j2\pi u^T x} dx \\ &= \iint_x f_0(R(x + x_1)) e^{-j2\pi u^T x} dx \\ &= e^{-j2\pi u^T x_1} \iint_x f_0(Rx') e^{-j2\pi u^T x'} dx' \end{aligned} \quad (3.16)$$

With  $F_1(u)$  the two-dimensional Fourier transform of  $f_1(x)$  and the coordinate transformation  $x' = x + x_1$ . After another transformation  $x'' = \mathbf{R} x'$ , the relation between the amplitudes of the Fourier transforms can be computed as



**Fig. 3.15** The rotation estimation ( $\theta_1 = 25^\circ$ ) and their Fourier transform. **a** Original image and its Fourier transformed amplitude. **b** Rotated image and its Fourier transformed amplitude

$$\begin{aligned}
 |F_1(u)| &= \left| e^{-j2\pi u^T x_1} \iint_x f_0(Rx') e^{-j2\pi u^T x'} dx' \right| \\
 &= \left| \iint_{x'} f_0(Rx') e^{-j2\pi u^T x'} dx' \right| \\
 &= \left| \iint_{x''} f_0(x'') e^{-j2\pi u^T (R^T x')} dx'' \right| \\
 &= \left| \iint_{x''} f_0(x'') e^{-j2\pi u^T (Ru)^T x'} dx'' \right| \\
 &= |F_0(Ru)|
 \end{aligned} \tag{3.17}$$

We can see that  $|F_1(u)|$  is a rotated version of  $|F_0(u)|$  over the same angle  $\theta_1$  as the spatial domain rotation in Fig. 3.15.  $|F_0(u)|$  and  $|F_1(u)|$  do not depend on the shift values  $x_1$ , because the spatial domain shifts only affect the phase values of the Fourier transforms. Therefore we can first estimate the rotation angle  $\theta_1$  from the amplitudes of the Fourier transforms  $|F_0(u)|$  and  $|F_1(u)|$ . After compensation for the rotation, the shift  $x_1$  can be computed from the phase difference between  $|F_0(u)|$  and  $|F_1(u)|$ .

### 3.4.3 Rotation Estimation

The rotation angle between  $|F_0(u)|$  and  $|F_1(u)|$  can be computed as the angle  $\theta_1$  for which the Fourier transform of the reference image  $|F_0(u)|$  and the rotated Fourier transform of the image to be registered  $|F_1(Ru)|$  have maximum correlation. This implies the computation of a rotation of  $|F_1(u)|$  for every evaluation of the correlation, which is computationally heavy and thus practically difficult.

If  $|F_0(u)|$  and  $|F_1(u)|$  are transformed in polar coordinates, the rotation over the angle  $\theta_1$  is reduced to a (circular) shift over  $\theta_1$ . We can compute the Fourier transform of the polar spectra  $|F_0(u)|$  and  $|F_1(u)|$ , and compute  $\theta_1$  as the phase shift between the two [76, 77]. This requires a transformation of the spectrum to polar coordinates. The data from the uniform  $u_h, u_v$ -grid need to be interpolated to obtain a uniform  $u_h, u_v$ -grid. Mainly for the low frequencies, which generally contain most of the energy, the interpolations are based on very few function values and thus introduce large approximation errors. An implementation of this method is also computationally intensive.

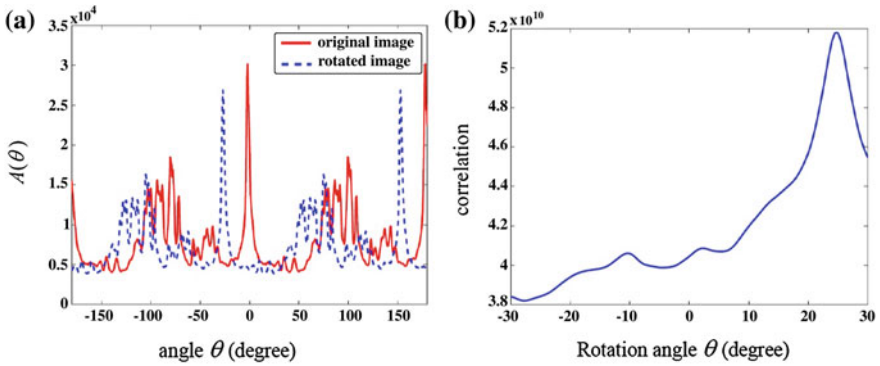
Our approach is computationally much more efficient than the two methods described above. First of all, we compute the frequency content  $\mathbf{A}$  as a function of the angle  $\theta$  by integrating over radial lines:

$$\mathbf{A}(\theta) = \int_{\theta-\Delta\theta/2}^{\theta+\Delta\theta/2} \int_0^{\infty} |F(u_r, u_\theta)| du_r du_\theta \quad (3.18)$$

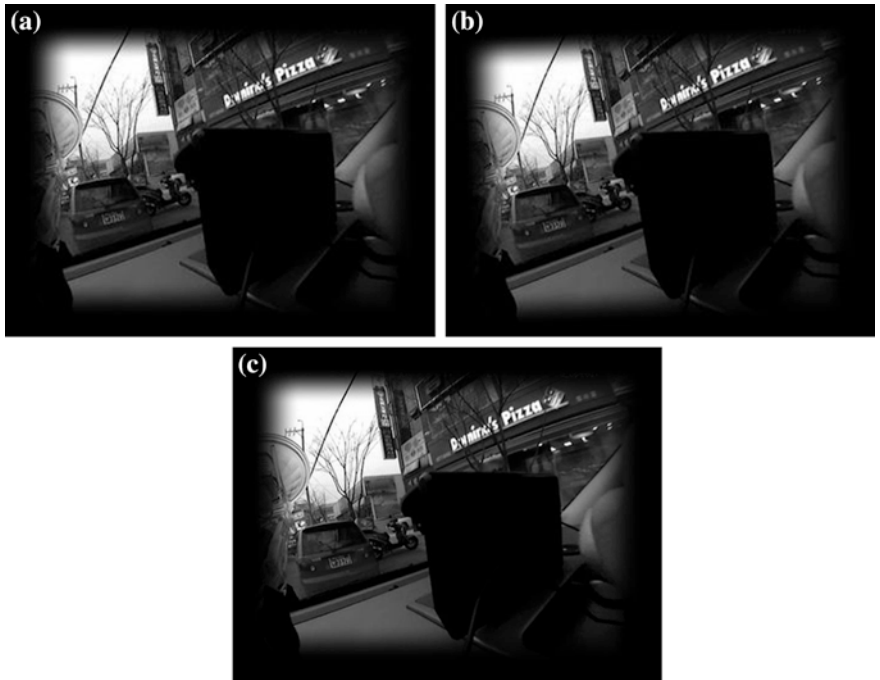
In practice,  $|F_0(u_r, u_\theta)|$  is a discrete signal. Different methods exist to relate discrete directions to continuous directions, like for example digital lines [101]. Here, we compute the discrete function  $\mathbf{A}(\theta)$  as the average of the values on the rectangular grid that have an angle  $\theta - \Delta\theta/2 < u_\theta < \theta + \Delta\theta/2$ . As we want to compute the rotation angle with a precision of 0.1 degrees,  $\mathbf{A}(\theta)$  is computed every 0.1 degrees. To get a similar number of signal values  $|F_0(u_r, u_\theta)|$  at every angle, the average is only evaluated on a circular disc of values for which  $u_r < \rho$  (where  $\rho$  is the image radius, or half the image size). Finally, as the values for low frequencies are very large compared to the other values and are very coarsely sampled as a function of the angle, we discard the values for which  $u_r < \varepsilon\rho$ , with  $\varepsilon = 0.1$ . Thus,  $\mathbf{A}(\theta)$  is computed as the average of the frequency values on a discrete grid with  $\theta - \Delta\theta/2 < u_\theta < \theta + \Delta\theta/2$  and  $\varepsilon\rho < u_r < \rho$ .

This results in a function  $\mathbf{A}(\theta)$  for both  $|F_0(u)|$  and  $|F_1(u)|$  as shown in Fig. 3.16. The exact rotation angle can then be computed as the value for which their correlation reaches a maximum. Note that only a one-dimensional correlation has to be computed, as opposed to the two-dimensional correlation approaches in [76] and [77].

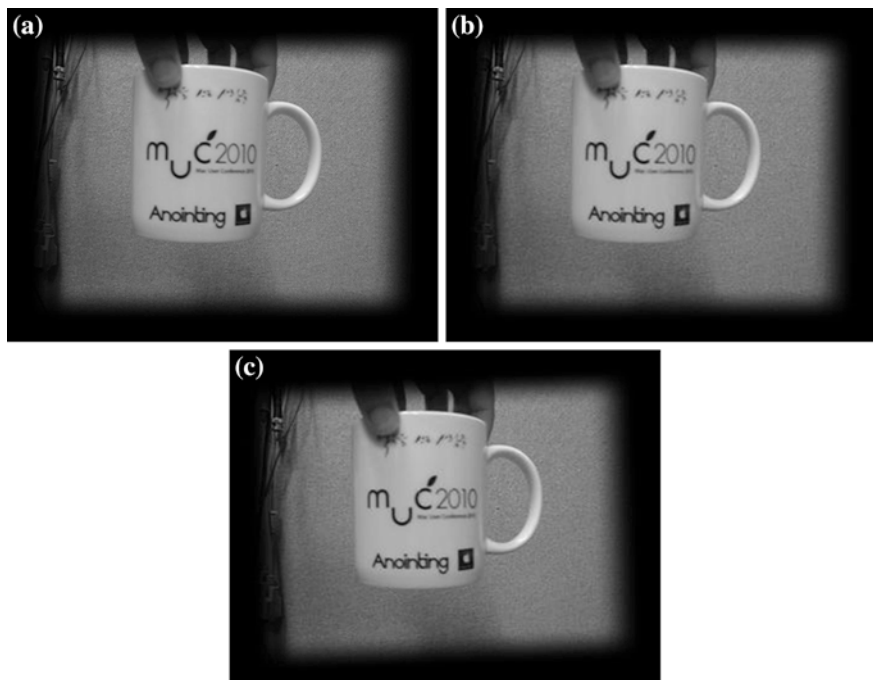
Of course, the use of such a radial projection also reduces the available information, and might introduce ambiguities in the estimation. The simulation result of our rotation estimation algorithms is shown in Fig. 3.17.



**Fig. 3.16** Rotation estimation. **a** Average Fourier domain amplitude as a function of the angle  $A(\theta)$  for the two image from Fig. 3.11. **b** Correlation between  $A_o(\theta)$  and  $A_l(\theta)$ , with a maximum at the rotation angle  $\theta_l = 25^\circ$



**Fig. 3.17** The simulation result of the rotation estimation. **a** Reference image. **b** Object image. **c** Inverse rotation estimated image of (b)



**Fig. 3.18** The simulation result of the shift estimation. **a** Reference image. **b** Object image. **c** Inverse shift estimated image with after the rotation estimation of **(b)**

### 3.4.4 Shift Estimation

A shift of the image parallel to the image plane can be expressed in Fourier domain as a linear phase shift:

$$\begin{aligned}
 F_1(u) &= \iint_x f_1(x) e^{-j2\pi u^T x} dx = \iint_x f_0(x + x_1) e^{-j2\pi u^T x} dx \\
 &= e^{j2\pi u^T x_1} \iint_{x'} f_0(x') e^{-j2\pi u^T x'} dx' = e^{j2\pi u^T x_1} F_0(u)
 \end{aligned} \tag{3.19}$$

It is well known that the shift parameters  $x_1$  can thus be computed as the slope of the phase difference  $\angle(F_1(u)/F_0(u))$  [76–79, 81, 82, 102]. To make the solution less sensitive to noise, a least squares method is widely used.

Here, the shift parameters  $x_1$  can be computed as the slope of the phase difference  $\angle(F_1(u)/F_0(u))$ . To make the solution less sensitive to noise, a least squares method is widely used. When we apply the inverse shift estimation into the object image after the rotation estimation for the reference image, the result image is exactly same with the reference image (see Fig. 3.18); therefore this shift estimation process is used in initial registration operation.



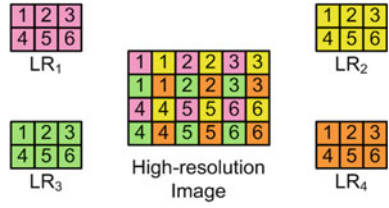


Fig. 3.19 The initial registration process



Fig. 3.20 Four sample images to generate a high resolution image

We decide three candidates for a reference image. To do this, we use the Hilbert space method. That is, we execute the initial registration process as Fig. 3.19.

In Fig. 3.19, LR<sub>1</sub> denotes as a reference image and from LR<sub>2</sub> to LR<sub>4</sub> are chosen candidates low resolution images. These candidate images are located at each high resolution grid by using the inverse shift estimation. For example, four sample images of 320 × 240 resolution to generate a high resolution image are shown in Fig. 3.20.

**Fig. 3.21** Graphical diagram and results image for four sample images

1	1	?	2	3	3
1	?	+	2	3	3
4	4	5	5	6	6
4	4	5	5	6	6

**Fig. 3.22** Secondly obtained high resolution image by using the mean value filtering



### 3.4.5 Reconstruction

And then, we can obtain a high resolution image, but its resolving power is not good. Because, obtained high resolution image has multichannel sampling frequencies and has unknown the offsets. To reduce the offsets and the number of multichannel sampling frequency, we apply the mean value filtering. That is, all of each pixel value is regenerated by using neighbor 5 pixels with cross-shape. Its graphical diagram and results image for four sample images are shown in Fig. 3.21.

Secondly obtained high resolution image, it looks not clear. Therefore, we apply the de-blurring operation to more reduce multichannel sampling frequencies, and then we apply sharpening process as shown in Fig. 3.23.

We apply mean value filtering, bi-cubic interpolation, de-blurring and sharpening process again like a kind iterative back-projection (IBP) method. And then we can obtain as Fig. 3.24.

These processes can be expressed as below equations. The initial registered image by the rotation and shift estimation has non-uniformed sampling frequency with unknown offsets. It can be expressed as

$$Y_m = \sum_{i_1, i_2} e^{j2\pi(i_1 N_1 t_{m,1} + i_2 N_2 t_{m,2})} D'_{t_m} \alpha_{i_1, i_2} \quad (3.20)$$

**Fig. 3.23** Result image after applying the de-blurring and sharpening to Fig. 3.22



**Fig. 3.24** Result image after applying IBP to Fig. 3.22





















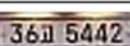
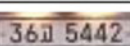
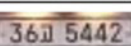
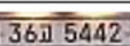
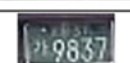

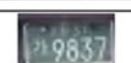
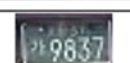





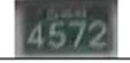








### 3.5 Conclusion

We have described super resolution methods, and especially we have focused to super resolution imaging with multichannel sampling with unknown offsets. In such algorithms, an accurate registration can decide the algorithm performance. We propose an advanced registration algorithms with smart rotation and shift estimation. The sequence for these two processes in our algorithm followed the warp-blur observation model. Generally, the cases that the blurring parameters are depend on the camera rotations and vibrations are much more than vice versa.

Firstly, our algorithm decides the optimal reference image to reduce the registration error, on the other hand another numerous super resolution algorithms discard considering this registration error or assume as uniform value. In this frame work, the registration error is calculated by using the sum of absolute

**Table 3.1** Comparison of the different methods presented in this chapter

	Proposed	Rank-based	Projection-based	Gröbner basis
Sample 1				
Sample 2				
Sample 3				
Sample 4				
Sample 5				
Sample 6				
Sample 7				
Sample 8				
Sample 9				
Sample 10				
Average PSNR	37.89	29.01	31.64	35.23

difference (SAD) based on the partial distortion elimination (PDE). This process has been obtained the noticeable result comparing conventional algorithms.

Secondly, the proposed algorithm estimates the rotation and the shift in order successively, because our algorithm is based on the warp-blur observation model. The blurring effects are by the point spread function (PSF) of camera, and it is subject to changes according to the rotation parameter of the image.

Finally, we have reconstructed a high resolution image by using the planar motion estimation. This results in a set of nonlinear equations in the unknown signal coefficients and the offsets. Using this formulation, we have shown that the solution is generally unique if the total number of sample values is larger than or equal to the total number of unknowns (signal parameters and offsets).

We present the one reference image and their three candidate images for 10 sample images to reconstruct a high resolution image by using our proposed registration algorithm. These candidate images are obtained from first step. And also we represent all images for each sample and show a bi-cubic interpolated image and a super resolution image by proposed algorithm. The image quality of

proposed algorithm is much higher than the bi-cubic interpolation method. We take the average PSNR of proposed algorithm is about 38 dB and another's are lower than our method. It is as shown in Table 3.1.

## References

1. International Organization for Standardization, ISO 12233:2000 - Photography- Electronic still picture cameras - Resolution measurements (2000)
2. T. Komatsu, K. Aizawa, T. Igarashi, T. Saito, Signal-processing based method for acquiring very high resolution image with multiple cameras and its theoretical analysis, in *Proceedings of the Institute of Electrical Engineering*, vol. 140, no. 1, pt. I (1993), pp. 19–25
3. S. Borman, R.L. Stevenson, Spatial resolution enhancement of low-resolution image sequences-a comprehensive review with directions for future research. Technical Report, Laboratory for Image and Signal Analysis (LISA). University of Notre Dame, Notre Dame, Ind (1998). Available at <http://www.nd.edu/~sborman/publications/>
4. S. Borman, R.L. Stevenson, Super-resolution from image sequences-a review, in *Proceedings of 1998 Midwest Symposium Circuits and Systems* (1999), pp. 374–378
5. S. Chaudhuri (ed.), *Super-Resolution Imaging* (Kluwer, Norwell, 2001)
6. H. Ur, D. Gross, Improved resolution from sub-pixel shifted pictures. *CVGIP: Graph. Models Image Process.* **54**, 181–186 (1992)
7. T. Komatsu, T. Igarashi, K. Aizawa, T. Saito, Very high resolution imaging scheme with multiple different-aperture cameras. *Signal Process. Image Commun.* **5**, 511–526 (1993)
8. M.S. Alam, J.G. Bognar, R.C. Hardie, B.J. Yasuda, Infrared image registration and high-resolution reconstruction using multiple translationally shifted aliased video frames. *IEEE Trans. Instrum. Meas.* **49**, 915–923 (2000)
9. N.R. Shah, A. Zakhor, Resolution enhancement of color video sequences. *IEEE Trans. Image Process.* **8**, 879–885 (1999)
10. N. Nguyen, P. Milanfar, An efficient wavelet-based algorithm for image superresolution. *Proc. Int. Conf. Image Process.* **2**, 351–354 (2000)
11. R.Y. Tsai, T.S. Huang, Multipleframe image restoration and registration, in *Advances in Computer Vision and Image Processing* (JAI Press Inc., Greenwich, 1984), pp. 317–339
12. S.P. Kim, N.K. Bose, H.M. Valenzuela, Recursive reconstruction of high resolution image from noisy undersampled multiframes. *IEEE Trans. Acoust. Speech Sig. Process.* **38**, 1013–1027 (1990)
13. S.P. Kim, W.Y. Su, Recursive high-resolution reconstruction of blurred multiframe images. *IEEE Trans. Image Process.* **2**, 534–539 (1993)
14. N.K. Bose, H.C. Kim, H.M. Valenzuela, Recursive implementation of total least squares algorithm for image reconstruction from noisy, undersampled multiframes, in *Proceedings of the IEEE Conference on Acoustics, Speech and Signal Processing*, vol. 5 (Minneapolis, 1993), pp. 269–272
15. S.H. Rhee, M.G. Kang, Discrete cosine transform based regularized high-resolution image reconstruction algorithm. *Opt. Eng.* **38**(8), 1348–1356 (1999)
16. M.C. Hong, M.G. Kang, A.K. Katsaggelos, A regularized multichannel restoration approach for globally optimal high resolution video sequence. in *SPIE VCIP*, vol. 3024 (San Jose, 1997), pp. 1306–1317
17. M.C. Hong, M.G. Kang, A.K. Katsaggelos, An iterative weighted regularized algorithm for improving the resolution of video sequences, in *Proceedings of the International Conference on Image Processing*, vol. 2 (1997), pp. 474–477

18. M.G. Kang, Generalized multichannel image deconvolution approach and its applications. *Opt. Eng.* **37**(11), 2953–2964 (1998)
19. R.C. Hardie, K.J. Barnard, J.G. Bognar, E.E. Armstrong, E.A. Watson, High-resolution image reconstruction from a sequence of rotated and translated frames and its application to an infrared imaging system. *Opt. Eng.* **37**(1), 247–260 (1998)
20. N.K. Bose, S. Lertrattanapanich, J. Koo, Advances in superresolution using L-curve, in *Proceedings of the International Symposium on Circuits and Systems*, vol. 2 (2001), pp. 433–436
21. B.C. Tom, A.K. Katsaggelos, Reconstruction of a high-resolution image by simultaneous registration, restoration, and interpolation of low-resolution images, in *Proceedings of 1995 IEEE International Conference on Image Processing*, vol. 2, Washington (1995), pp. 539–542
22. R.R. Schulz, R.L. Stevenson, Extraction of high-resolution frames from video sequences. *IEEE Trans. Image Process.* **5**, 996–1011 (1996)
23. R.C. Hardie, K.J. Barnard, E.E. Armstrong, Joint MAP registration and high-resolution image estimation using a sequence of undersampled images. *IEEE Trans. Image Process.* **6**, 1621–1633 (1997)
24. P. Cheeseman, B. Kanefsky, R. Kraft, J. Stutz, R. Hanson, Super-resolved surface reconstruction from multiple images. NASA Ames Research Center, Moffett Field, Technical Report FIA-94-12 (1994)
25. H. Stark, P. Oskoui, High resolution image recovery from image-plane arrays, using convex projections. *J. Opt. Soc. Am. A* **6**, 1715–1726 (1989)
26. A.M. Tekalp, M.K. Ozkan, M.I. Sezan, High-resolution image reconstruction from lower-resolution image sequences and space varying image restoration, in *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, vol. 3 (San Francisco, 1992), pp. 169–172
27. A.J. Patti, M.I. Sezan, A.M. Tekalp, Superresolution video reconstruction with arbitrary sampling lattices and nonzero aperture time. *IEEE Trans. Image Process.* **6**(8), 1064–1076 (1997)
28. P.E. Eren, M.I. Sezan, A.M. Tekalp, Robust, object-based high-resolution image reconstruction from low-resolution video. *IEEE Trans. Image Process.* **6**(10), 1446–1451 (1997)
29. A.J. Patti, Y. Altunbasak, Artifact reduction for set theoretic super resolution image reconstruction with edge adaptive constraints and higher-order interpolants. *IEEE Trans. Image Process.* **10**(1), 179–186 (2001)
30. B.C. Tom, A.K. Katsaggelos, An iterative algorithm for improving the resolution of video sequences, in *Proceedings of the 1996 SPIE Conference on Visual Communications and Image Processing* (Orlando, 1996), pp. 1430–1438
31. M. Elad, A. Feuer, Restoration of a single super-resolution image from several blurred, noisy, and undersampled measured images. *IEEE Trans. Image Process.* **6**(12), 1646–1658 (1997)
32. M. Irani, S. Peleg, Improving resolution by image registration. *CVGIP: Graph. Models Image Process.* **53**, 231–239 (1991)
33. S. Mann, R.W. Picard, Virtual bellows: constructing high quality stills from video, in *Proceedings of the IEEE International Conference on Image Processing* (Austin, 1994), pp. 13–16
34. M. Irani, S. Peleg, Motion analysis for image enhancement resolution, occlusion, and transparency. *J. Visual Commun. Image Represent.* **4**, 324–335 (1993)
35. M. Elad, A. Feuer, Superresolution restoration of an image sequence: adaptive filtering approach. *IEEE Trans. Image Process.* **8**, 387–395 (1999)
36. M. Elad, A. Feuer, Super-resolution reconstruction of image sequences. *IEEE Trans. Pattern Anal. Mach. Intelli.* **21**(9), 817–834 (1999)
37. M.C. Chiang, T.E. Boulton, Efficient super-resolution via image warping. *Image Vis. Comput.* **18**, 761–771 (2000)

38. D. Rajan, S. Chaudhuri, Generation of super-resolution images from blurred observations using an MRF model. *J. Math. Imaging Vision* **16**, 5–15 (2002)
39. D. Rajan, S. Chaudhuri, Simultaneous estimation of super-resolved intensity and depth maps from low resolution defocused observations of a scene, in *Proceedings of the IEEE International Conference on Computer Vision* (Vancouver, 2001), pp. 113–118
40. D. Rajan, S. Chaudhuri, Generalized interpolation and its applications in super-resolution imaging. *Image Vis. Comput.* **19**, 957–969 (2001)
41. M.V. Joshi, S. Chaudhuri, Super-resolution imaging: use of zoom as a cue, in *Proceedings of the Indian Conference on Vision, Graphics and Image Processing* (Ahmedabad, 2002), pp. 439–444
42. N.K. Bose, H.C. Kim, B. Zhou, Performance analysis of the TLS algorithm for image reconstruction from a sequence of undersampled noisy and blurred frames, in *Proceedings of the ICIP-94, IEEE International Conference on Image Processing*, vol. 3 (1994), pp. 571–575
43. M. Ng, J. Koo, N. Bose, Constrained total least squares computations for high resolution image reconstruction with multisensors. *Int. J. Imaging Syst. Technol.* **12**, 35–42 (2002)
44. M.K. Ng, N.K. Bose, Analysis of displacement errors in high-resolution image reconstruction with multisensors. *IEEE Trans. Circuits Syst. I* **49**, 806–813 (2002)
45. M. Park, E. Lee, J. Park, M.G. Kang, J. Kim, DCT-based high-resolution image reconstruction considering the inaccurate sub-pixel motion information. *SPIE Opt. Eng.* **41**(2), 370–380 (2002)
46. W. Lim, M. Park, M.G. Kang, Spatially adaptive regularized iterative high resolution image reconstruction algorithm, in *Proceedings of the VCIP2001, Photonicswest* (San Jose, 2001), pp. 20–26
47. E.S. Lee, M.G. Kang, Regularized adaptive high-resolution image reconstruction considering inaccurate subpixel registration. *IEEE Trans. Image Process.* **12**, 826–837 (2003)
48. Wirawan, P. Duhamel, H. Maitre, Multi-channel high resolution blind image restoration, in *Proceedings of the IEEE ICASSP* (AZ, 1989), pp. 3229–3232
49. N. Nguyen, P. Milanfar, G. Golub, Efficient generalized cross-validation with applications to parametric image restoration and resolution enhancement. *IEEE Trans. Image Process.* **10**, 1299–1308 (2001)
50. N. Nguyen, P. Milanfar, G. Golub, A computationally efficient superresolution image reconstruction algorithm. *IEEE Trans. Image Process.* **10**, 573–583 (2001)
51. M. Elad, Y. Hel-Or, A fast super-resolution reconstruction algorithm for pure translational motion and common space-invariant blur. *IEEE Trans. Image Process.* **10**(8), 1187–1193 (2001)
52. M. Ng, R. Chan, T. Chan, A. Yip, Cosine transform preconditioners for high resolution image reconstruction. *Linear Algebra Appl.* **316**, 89–104 (2000)
53. D.S. Messing, M.I. Sezan, Improved multi-image resolution enhancement for colour images captured by single-CCD cameras, in *Proceedings of the International Conference on Image Processing*, vol. 3 (2000), pp. 484–487
54. M.K. Ng, An efficient parallel algorithm for high resolution color image reconstruction, in *Proceedings of the 7th International Conference on Parallel and Distributed Systems: Workshops* (2000), pp. 547–552
55. B.C. Tom, A.K. Katsaggelos, Resolution enhancement of monochrome and color video using motion compensation. *IEEE Trans. Image Process.* **10**(2), 278–287 (2001)
56. M. Ng, W. Kwan, High-resolution color image reconstruction with Neumann boundary conditions. *Ann. Oper. Res.* **103**, 99–113 (2001)
57. D. Chen, R.R. Schultz, Extraction of high-resolution video stills from MPEG image sequences, in *Proceedings of the 1998 IEEE International Conference on Image Processing*, vol. 2 (1998), pp. 465–469

58. Y. Altunbasak, A.J. Patti, A maximum a posteriori estimator for high resolution video reconstruction from MPEG video, in *Proceedings of the 2000 IEEE International Conference on Image Processing*, vol. 2 (2000), pp. 649–652
59. B. Martins, S. Forchhammer, A unified approach to restoration, deinterlacing and resolution enhancement in decoding MPEG-2 video. *IEEE Trans. Circuits Syst. Video Technol.* **12**(9), 803–811 (2002)
60. C.A. Segall, R. Molina, A.K. Katsaggelos, J. Mateos, Reconstruction of high-resolution image frames from a sequence of low-resolution and compressed observations, in *Proceedings of the 2002 IEEE International Conference on Acoustics, Speech, and Signal Processing*, vol. 2 (2002), pp. 1701–1704
61. S.C. Park, M.G. Kang, C.A. Segall, A.K. Katsaggelos, Spatially adaptive high-resolution image reconstruction of low resolution DCT-based compressed images, in *Proceedings of the 2002 IEEE International Conference Image Processing*, vol. 2 (2002), pp. 861–864
62. B.K. Gunturk, Y. Altunbasak, R.M. Mersereau, Multiframe resolution-enhancement methods for compressed video. *IEEE Signal Process. Lett.* **9**, 170–174 (2002)
63. H.C. Andrews, B.R. Hunt, *Digital Image Restoration* (Prentice-Hall, Englewood Cliffs, 1977)
64. A.K. Katsaggelos (ed.), *Digital Image Restoration*, vol. 23 (Springer, Heidelberg, 1991)
65. I.J. Schoenberg, Cardinal interpolation and spline functions. *J. Approx. Theory.* **2**, 167–206 (1969)
66. R.E. Crochiere, L.R. Rabiner, Interpolation and decimation of digital signals—a tutorial review. *Proc. IEEE* **69**(3), 300–331 (1981)
67. M. Unser, A. Aldroubi, M. Eden, Enlargement or reduction of digital images with minimum loss of information. *IEEE Trans. Image Process.* **4**(3), 247–258 (1995)
68. Z. Wang, F. Qi, On ambiguities in super-resolution modeling. *IEEE Signal Process. Lett.* **11**, 678–681 (2004)
69. S. Farsiu, D. Robinson, M. Elad, P. Milanfar, Robust shift and add approach to super-resolution, in *Proceedings of SPIE Applications of Digital Image Processing XXVI*, vol. 5203 (San Diego, 2003), pp. 121–130
70. S. Farsiu, D. Robinson, M. Elad, P. Milanfar, Fast and robust multiframe super-resolution. *IEEE Trans. Image Process.* **13**, 1327–1344 (2004)
71. A. López, R. Molina, A. K. Katsaggelos, A. Rodríguez, J. M. López, J. M. Llamas, Parameter estimation in Bayesian reconstruction of SPECT images: an aide in nuclear medicine diagnosis. *Int. J. Imaging Syst. Technol.* **14**, 21–27 (2004)
72. S. Lertrattanapanich, N.K. Bose, High resolution image formation from low resolution frames using Delaunay triangulation. *IEEE Trans. Image Process.* **11**, 1427–1441 (2002)
73. S.C. Park, M.K. Park, M.G. Kang, Super-resolution image reconstruction: a technical overview. *IEEE Sig. Process. Mag.* **20**(3), 21–36 (2003)
74. B. Zitová, J. Flusser, Image registration methods: a survey. *Image Vis. Comput.* **21**(11), 977–1000 (2003)
75. A.M. Tekalp, *Digital Video Processing* (Prentice Hall, Englewood Cliffs, 1995)
76. B.S. Reddy, B.N. Chatterji, An FFT-based technique for translation, rotation, and scale-invariant image registration. *IEEE Trans. Image Process.* **5**(8), 1266–1271 (1996)
77. B. Marcel, M. Briot, R. Murrieta, Calcul de translation et rotation par la transformation de Fourier. *Traitement du Signal* **14**(2), 135–149 (1997)
78. S.P. Kim, W.-Y. Su, Subpixel accuracy image registration by spectrum cancellation, in *Proceedings of IEEE International Conference Acoustics, Speech, Signal Processing (ICASSP '93)*, vol. 5 (Minneapolis, 1993), pp. 153–156
79. H.S. Stone, M.T. Orchard, E.-C. Chang, S.A. Martucci, A fast direct Fourier-based algorithm for subpixel registration of images. *IEEE Trans. Geosci. Remote Sens.* **39**(10), 2235–2243 (2001)
80. P. Vandewalle, S.E. Šusstrunk, M. Vetterli, Super-resolution images reconstructed from aliased images, in *Proceedings of SPIE/IS&T Visual Communications and Image*



- Processing Conference*, ed. by T. Ebrahimi, T. Sikora, vol. 5150 (Lugano, 2003), pp. 1398–1405
81. H. Foroosh, J.B. Zerubia, M. Berthod, Extension of phase correlation to subpixel registration. *IEEE Trans. Image Process.* **11**(3), 188–200 (2002)
  82. L. Lucchese, G.M. Cortelazzo, A noise-robust frequency domain technique for estimating planar roto-translations. *IEEE Trans. Sig. Process.* **48**(6), 1769–1786 (2000)
  83. D. Capel, A. Zisserman, Computer vision applied to super-resolution. *IEEE Sig. Process. Mag.* **20**(3), 75–86 (2003)
  84. M.A. Fischler, R.C. Bolles, Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Comm. of the ACM* **24**(6), 381–395
  85. D. Keren, S. Peleg, R. Brada, Image sequence enhancement using sub-pixel displacements, in *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 88)*, (Ann Arbor, 1988), pp. 742–746
  86. J.R. Bergen, P. Anandan, K.J. Hanna, R. Hingorani, Hierarchical model-based motion estimation, in *Proceedings of 2nd European Conference on Computer Vision (ECCV'92)*, Lecture Notes in Computer Science (Santa Margherita Ligure, 1992), pp. 237–252
  87. M. Irani, B. Rousso, S. Peleg, Computing occluding and transparent motions. *Int. J. Comput. Vis.* **12**(1), 5–16 (1994)
  88. J. Gluckman, Gradient field distributions for the registration of images, in *Proceedings of IEEE International Conference on Image Processing (ICIP '03)*, vol. 3 (Barcelona, 2003), pp. 691–694
  89. A. Papoulis, Generalized sampling expansion. *IEEE Trans. Circuits Syst.* **24**(11), 652–654 (1977)
  90. R.R. Schultz, L. Meng, R.L. Stevenson, Subpixel motion estimation for super-resolution image sequence enhancement. *J. Vis. Commun. Image Represent.* **9**(1), 38–50 (1998)
  91. A. Zomet, A. Rav-Acha, S. Peleg, Robust super-resolution, in *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR '01)*, vol. 1, (Kauai, 2001), pp. 645–650
  92. W. Li, E. Salari, Successive elimination algorithm for motion estimation. *IEEE Trans. Image Process.* **4**(1), 105–107 (1995)
  93. X.Q. Gao, C.J. Duanmu, C.R. Zou, A multilevel successive elimination algorithm for block matching motion estimation. *IEEE Trans. Image Process.* **9**(3), 501–505 (2000)
  94. M. Brúning, W. Niehsen, Fast full search Block matching. *IEEE Trans. Circuit Syst. Video Technol.* **11**(2), 241–247 (2001)
  95. H.G. Musmann, P. Pirsch, H.J. Grallert, Advanced in picture coding. *Proc. IEEE* **73**(4), 523–548 (1995)
  96. S. Zhu, K. Ma, A new diamond search algorithm for fast block matching motion estimation. *IEEE Trans. Image Process.* **9**, 287–290 (2000)
  97. ITU-T, Video coding for low bitrate communication, *Draft Recommendation H.263* (1995)
  98. Y.H. Jeong, J.H. Kim, The FASCO block matching algorithm based on motion vector prediction using spatio-temporal correlations. *Korean Inst. Commun. Sci.* **26**(11A), 1925–1937 (2002)
  99. J.N. Kim, S.C. Byun, Y.H. Kim, B.H. Ahn, Fast full search motion estimation algorithm using early detection of impossible candidate vectors. *IEEE Trans. Sig. Process.* **50**(9), 2355–2365 (2002)
  100. V. Ayala Ramirez, M. Devy, C. Parra, Active tracking based on Hausdorff matching, in *Proceedings of Pattern Recognition 15th International Conference*, vol. 4 (2000), pp. 706–709
  101. V. Velisavljevic, “Directionlets,” Ph.D. dissertation, Ecole Polytechnique Fédérale de Lausanne (EPFL), Lausanne, Switzerland, 2005, ph.D. Thesis EPFL 3358 (2005), School of Computer and Communication Sciences
  102. W.S. Hoge, A subspace identification extension to the phase correlation method. *IEEE Trans. Med. Imaging* **22**(2), 277–280 (2003)

# Chapter 4

## Image Enhancement for Improving Object Recognition

Jaeseok Kim

**Abstract** Image enhancement techniques are increasingly needed for improving object recognition in automobile driving. In driving conditions, there are many variables that degrade the quality of the image captured from the camera, such as fog, rain, a sudden change of illumination, or lack of illumination. If the quality of the obtained image is degraded, object recognition (cars, pedestrians, fixed objects, and traffic signals) can be unsatisfactory. To improve the recognition rate of objects, several image enhancement algorithms are proposed and evaluated. In this chapter, general image enhancement techniques are introduced, followed by a discussion of advanced techniques for the driving environment.

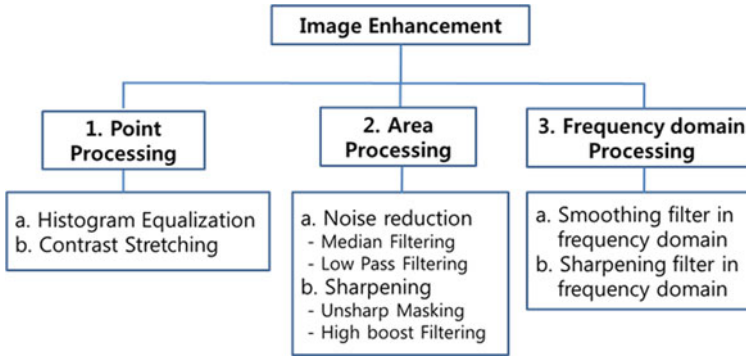
### 4.1 General Image Enhancement Techniques

Image enhancement is the process of manipulating an image so that the result is better than the original for a specific application [1]. It covers a broad scope of techniques that are present in numerous applications. These techniques involve enhancement or restoration of an image that has been degraded during image acquisition, highlighting certain features of an image, creation of a new image from other images, and so on [2]. Among the numerous applications of image enhancement techniques, we focus on techniques for improving object recognition in the automobile driving environment.

Image enhancement techniques applicable to this application can be divided into four major categories: point processing, area processing, frequency domain processing, and tone-mapping. Point processing modifies a pixel's value based on that pixel's original value and yields a new pixel with the same coordinates.

---

J. Kim (✉)  
Yonsei University, Seoul, Korea  
e-mail: jaekim@yonsei.ac.kr



**Fig. 4.1** Classification of image enhancement techniques

Area processing operates in the spatial domain, using a square mask that computes for the object pixel and neighboring pixels. Frequency domain processing is the technique that uses the advantage of the frequency domain to enhance an image. In order to convert data from the spatial domain to the frequency domain, the Fourier transform is the most widely used algorithm. The tone-mapping technique is the process of converting a high dynamic range (HDR) image to a low dynamic range (LDR) image that is suitable for display, by compressing the dynamic range of the HDR image while preserving the details of the image.

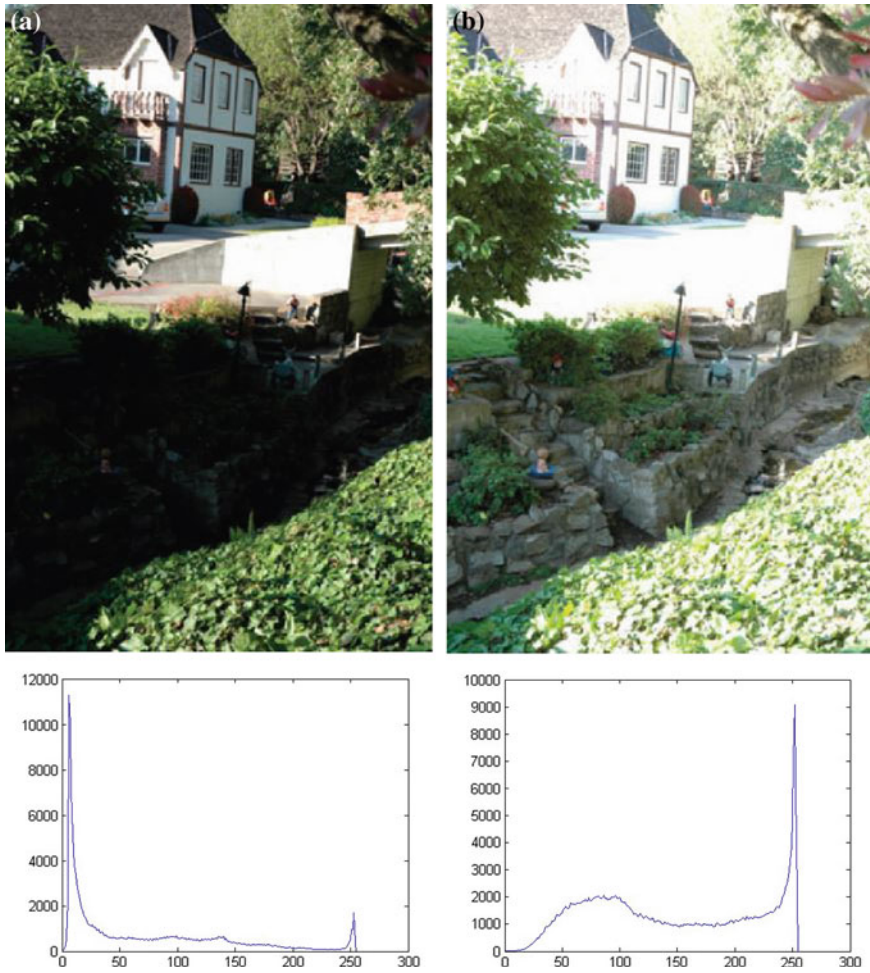
Figure 4.1 shows the classification of image enhancement techniques.

### 4.1.1 Point Processing Technique

Point processing techniques work with an image histogram, which is a valuable tool used to view the intensity profile of an image [2], providing information about the contrast and overall intensity distribution. The histogram of a digital image with intensity levels in the range  $[0, L-1]$  is a discrete function  $h(r_k) = n_k$ , where  $r_k$  is the  $k$ th intensity value, and  $n_k$  is the number of pixels in the image with intensity  $r_k$  [1]. It is common practice to normalize the histogram by dividing each of its components by the total number of pixels in the image. Thus, a normalized histogram is given by Gonzales and Woods [1]:

$$p(r_k) = n_k/n \quad (4.1)$$

where  $n$  is the total number of pixels in the image, and  $k = 0, 1 \dots L-1$ .  $p(r_k)$  gives an estimate of the probability of occurrence of intensity level  $r_k$ . Figure 4.2 presents some sample images and their histograms.

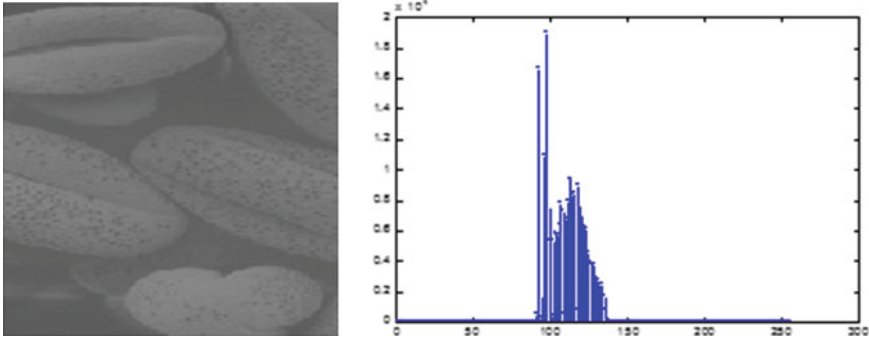


**Fig. 4.2** Sample images and their histograms [3] **a** Dark image **b** Light image

#### 4.1.1.1 Histogram Equalization

The histogram pattern gives information about the contrast of an image. For example, if an image has a narrow histogram, such as is shown in Fig. 4.3, the image is considered to be a low contrast image. Generally, it is not easy to recognize details in a low contrast image.

To increase the contrast of an image, the histogram equalization technique can be used. Histogram equalization redistributes intensity distributions, and the resulting image will have a more uniformly distributed histogram. The process of equalizing starts with computing a normalized histogram of the original image. The transform equation  $T(r_k)$  is given by [1]:



**Fig. 4.3** Histogram of a low contrast image [1]

$$T(r_k) = (L - 1) \sum_{j=0}^k p_r(r_j) = \frac{L - 1}{n} \sum_{j=0}^k n_j, \quad k = 0, 1, 2, \dots, L - 1 \quad (4.2)$$

where  $r_k$  is the intensity value of the input image pixel,  $n_j$  is the number of pixels in the image with intensity  $r_j$ , and  $n$  is the total number of pixels in the image. The resulting image is obtained by mapping each pixel in the input image using Eq. (4.2). Figure 4.4 shows the result of histogram equalization, whereby the contrast of the image is enhanced and the histogram becomes broader.

Histogram equalization works well when the distribution of the histogram is narrow. To adjust histogram equalization for more general cases, the histogram division algorithm [4] is suggested. The result of histogram division and equalization is an adaptive contrast stretched image. In this method, the histogram can be divided into two or more regions. For example, in the case of the bi-histogram, the original histogram will be divided into two sections. In this case, upper region and lower region can be decided by the threshold function. If the PDF of the original image is  $p(X_k) = \frac{n^k}{n}$ , each region is represented as [4]:

$$P_L(X_k) = \frac{n^k}{n}, P_U(X_k) = \frac{n^k}{n} \quad (4.3)$$

So the final transfer functions are represented as follows:

$$f_L(x) = X_0 + (X_m - X_0)c_L(x) \quad (4.4)$$

$$f_U(x) = X_{m+1} + (X_{L+1} - X_{m+1})c_U(x) \quad (4.5)$$

This algorithm shows an improved performance in preserving brightness.

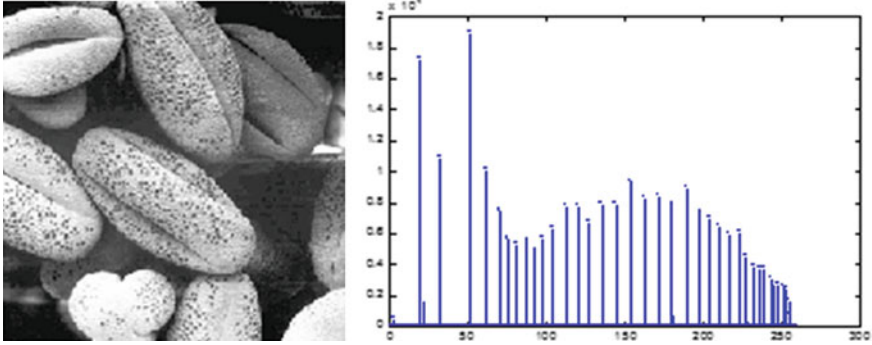


Fig. 4.4 Histogram equalized image and its histogram

#### 4.1.1.2 Contrast Stretching

The contrast of an image is its distribution of light and dark pixels. In the histogram of a low contrast image, the histogram is concentrated on the right, the left, or in the middle. Contrast stretching is a technique that enlarges the image's contrast, stretching the histogram to fill the full dynamic range of the image [2]. This makes the image clearer, as the human viewer can more readily discern an image with greater contrast. With this technique, the histogram of an image would be expanded to cover all ranges of pixels. The basic transformation equation is [2]:

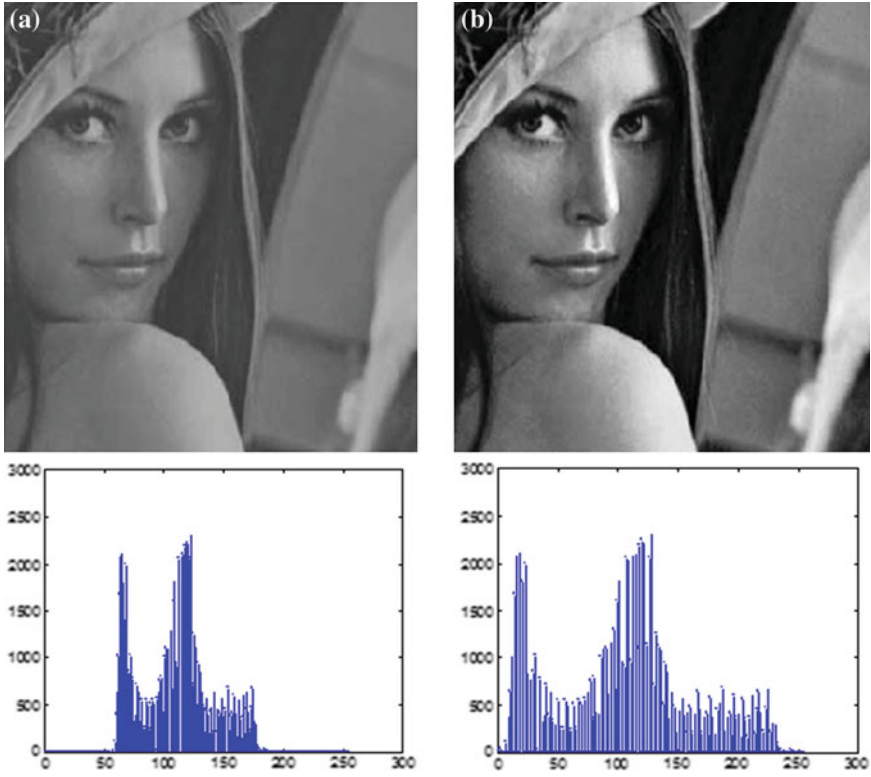
$$\text{new pixel} = \frac{\text{old pixel} - \text{low}}{\text{high} - \text{low}} \times 255 \quad (4.6)$$

Figure 4.5 shows the result of contrast stretching. The contrast stretched image has a broader histogram than the original image's histogram.

Contrast stretching and histogram equalization both improve the contrast of the image, with some differences. The contrast stretching technique extends the intensity range of an image with a fixed ratio (highest pixel value/lowest pixel value), whereas the histogram equalization technique redistributes the intensity range in accordance with the cumulative distribution function by occurrence.

### 4.1.2 Area Processing Technique

Area processing techniques operate on the input pixel and the neighboring pixels to generate a new output pixel. The most important concept in the area processing technique is filtering, which performs some operations to accept or reject certain components in the pixels of the image. For the filtering operation, a specific region is selected and the components in that region are filtered.



**Fig. 4.5** Original image and contrast stretched image **a** Low contrast image **b** Contrast stretched image

There are two kinds of filters: the non-linear filter and the linear filter. Non-linear filters do not have any fixed form, but linear filters are represented as a convolution, which is a weighted sum of pixels in the neighborhood of the source pixel. The range of the neighborhood and weights is determined by a convolution mask, a small matrix with predetermined weight.

The discrete form of the equation for spatial domain convolution is given by:

$$R(x, y) = \sum_{i=-d}^d \sum_{j=-d}^d w(i, j) f(x - i, y - j) \tag{4.7}$$

where  $(2d + 1) \times (2d + 1)$  is the mask size,  $w(i, j)$ 's are the weights of the mask,  $f(x, y)$  is the input pixel at coordinates  $(x, y)$ , and  $R(x, y)$  is the output value at  $(x, y)$ .

If the center of the mask is at location  $(x, y)$  in the image, the gray level of the pixel located at  $(x, y)$  is replaced by  $R$ , the mask is then moved to the next location in the image, and the process is repeated.

Figure 4.6 shows how convolution is performed.

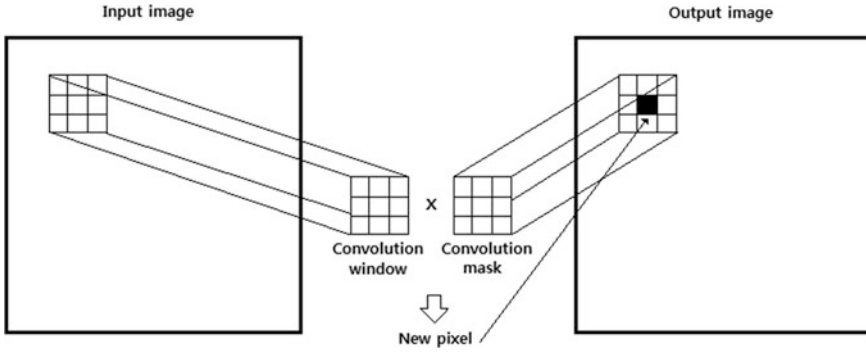


Fig. 4.6 Concept of discrete convolution

The next section describes two main types of filtering operations: noise reduction and sharpening.

#### 4.1.2.1 Noise Reduction

In image processing, noise reduction is always an issue. The most frequently occurring noises are impulse noise and Gaussian noise. Impulse noise is also called salt and pepper noise because of its appearance as white and black dots superimposed on an image. Gaussian noise is inherent in our environment, which has a Gaussian profile. Low pass filtering is useful for the reduction of Gaussian noise, and median filtering is suitable for the reduction of impulse noise.

##### (a) Low pass filtering

The low pass spatial filter (sometimes called an averaging filter) is used for smoothing edges and other fine details in the image. It is useful for removing Gaussian noise. Simultaneously, low pass filtering also blurs the edges of objects in the target image. Blurring can sometimes be used as a pre-processing task for certain image processing techniques, such as the removal of small details from an image or the smoothing of false contours.

For low pass filtering, the key requirement is that all coefficients of the mask are positive. The Gaussian mask and the averaging mask are well-known low pass filters. The coefficients of the Gaussian mask are determined by the bell-shaped Gaussian function. Equation (4.18) is the equation of Gaussian function.

$$G[x, y] = \frac{e^{-\frac{(x^2+y^2)}{2\sigma^2}}}{2\pi\sigma^2} \quad (4.8)$$

where  $\sigma$  controls the effective spread of coefficient. If the  $\sigma$  value is small, the weight of coefficients is dense. On the other hand, if the value of  $\sigma$  is large, the



**Fig. 4.7** Example of  $3 \times 3$  Gaussian blur mask

$1/16$	$1/8$	$1/16$
$1/8$	$1/4$	$1/8$
$1/16$	$1/8$	$1/16$

weight is broad. Figure 4.7 shows an example of a  $3 \times 3$  Gaussian blur mask where  $\sigma = 1$ .

Figure 4.8 shows the result of low pass filtering, notably some smoothing effects that are effective means of reducing Gaussian noise. However, this low pass filtering has the undesirable side effect that it blurs edges.

#### (b) Median filtering

Median filtering is a non-linear area processing technique that replaces the value of a pixel with the median of the intensity values in the neighborhood of that pixel. Median filtering is performed by selecting the median value from the values in the median mask region for the output pixel. For each input pixel  $f(x, y)$ , the values of the pixels are sorted and the median value is assigned to the output pixel  $g(x, y)$ . Figure 4.9 shows how the median filter works.

Median filtering is a widely used technique for noise reduction, especially impulse noise. Moreover, the process can preserve the sharpness of the edge and details in the target image [1, 5]. Edge preservation is one great advantage that the low pass filter cannot provide.

Figure 4.10 shows an example of the result of median filtering.

### 4.1.2.2 Sharpening

Sharpening has the opposite effect from smoothing. Sharpening emphasizes the details in an image and highlights the transitions in intensity. It is based on the high pass filtering operation, which is used to highlight a fine detail in an image or to enhance a detail that has become blurred. High pass filtering attenuates the low frequency components without disturbing the high frequency information in the frequency domain. In the spatial domain, sharpening can be accomplished by spatial differentiation. Unsharp masking and high boost filtering are commonly used techniques for high pass filtering.

The unsharp masking process can be performed by subtracting the unsharpened version of an image from the original image [1]. This process consists of the following steps:

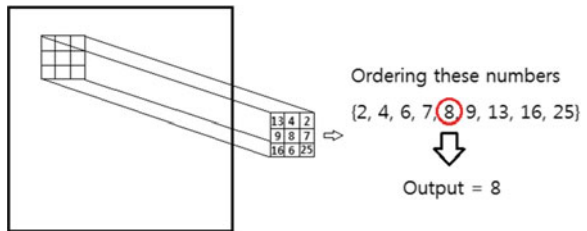
Step 1: Blur the original image.

Step 2: Subtract the blurred image from the original image (the resulting difference is called the mask).



**Fig. 4.8** Result of low pass filtering **a** Original image. **b** Low pass filtered image

**Fig. 4.9** How the median filter works ( $3 \times 3$  mask)



Step 3: Add the mask to the original.

Letting  $\bar{f}(x,y)$  denote the blurred image, the mask is given by [1]:

$$g_{mask}(x, y) = f(x, y) - \bar{f}(x, y) \tag{4.9}$$

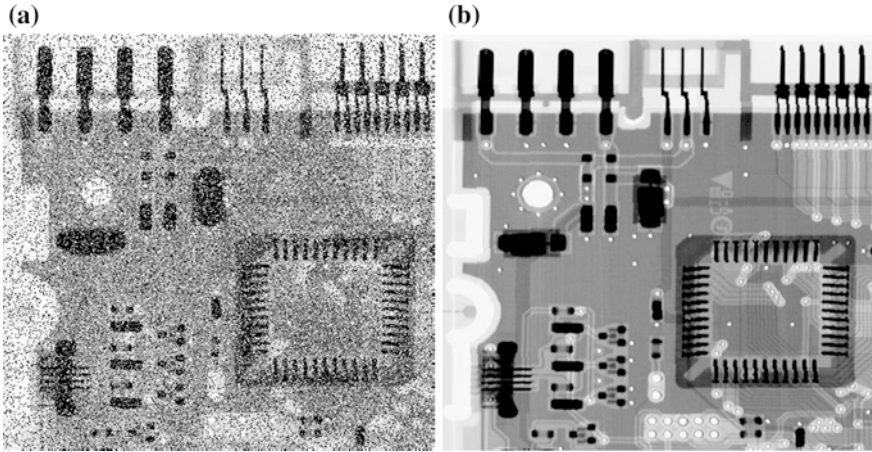
A weighted portion of the mask is then added to the original image, as follows:

$$g(x, y) = f(x, y) + k * g_{mask}(x, y) \tag{4.10}$$

where  $f(x, y)$  and  $g(x, y)$  are the input and sharpened images, respectively.

When the weight coefficient  $k = 1$ , the process is called unsharp masking, and when  $k > 1$ , the process is called high boost filtering.

Unsharp masking and high boost filtering have the effect of making the image look sharper than the original image, as shown in Figs. 4.11 and 4.12.



**Fig. 4.10** Example of the result of median filtering [1] **a** Image with impulse noise. **b** Median filtered image



**Fig. 4.11** Example of unsharp masking

### ***4.1.3 Frequency Domain Processing Technique***

Image processing with a domain transformed image can reduce computational cost and increase the accuracy of operation for some image processing techniques. In particular, the computation of convolution in the spatial domain is costly, but it can be replaced by simple multiplication in the frequency domain, for which the time required is much less than convolution in the spatial domain, even including transform and inverse transform time. In addition, the result of the operation in the frequency domain is more precise than that performed in the spatial domain, since

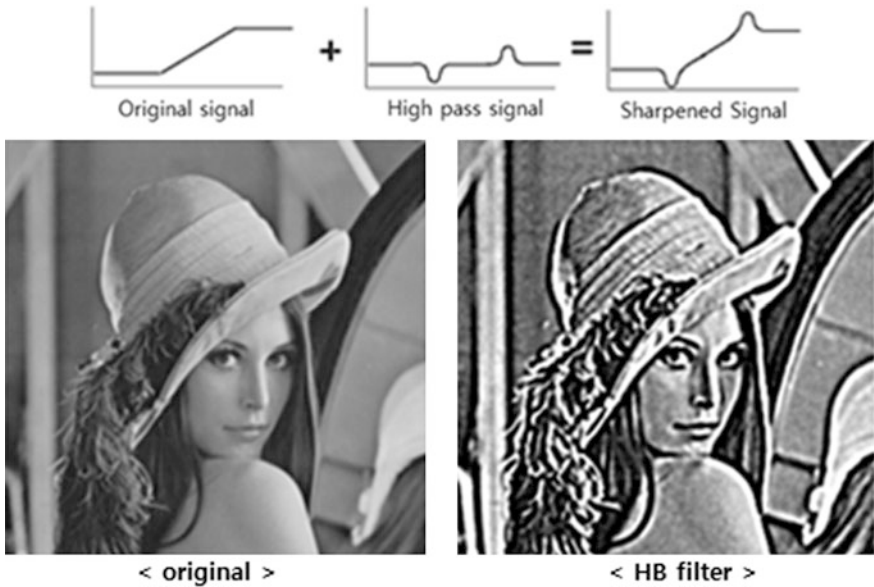


Fig. 4.12 Example of high boost filtering

masking is always an approximation of the original model in the spatial domain process, whereas, in the frequency domain, the ideal low pass or high pass filter can be used.

Another point to note in the frequency domain process is that there is a relationship between frequency components and intensity variations in an image. The slow varying frequency component is proportional to the average intensity of an image. Low frequencies correspond to the slowly varying intensity components of an image, whereas higher frequencies correspond to fast intensity changes. These are the edges of objects and other components of an image, characterized by abrupt changes in intensity.

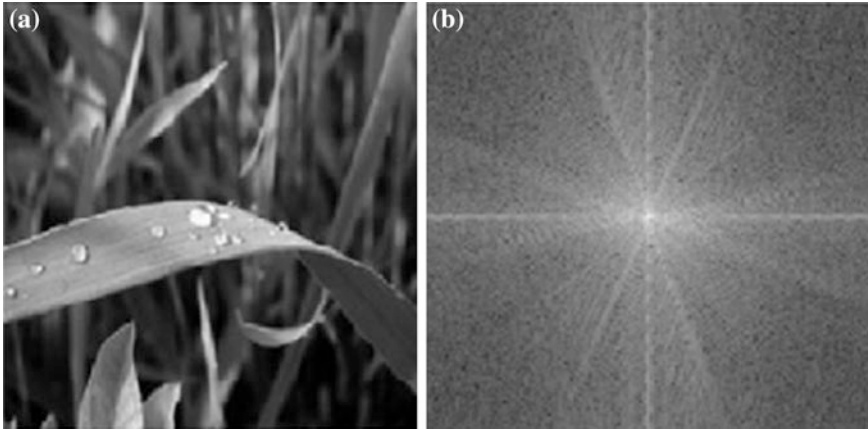
The Fourier transform is generally used for domain transformation from the spatial domain to the frequency domain. Wavelet transform is also a currently popular technique in image processing.

Fast Fourier transform (FFT) and inverse fast Fourier transform (IFFT) algorithms are developed to calculate the Fourier transform and inverse transform step efficiently, using the following equations:

$$\text{Spatial domain : } g(x, y) = f(x, y) * h(x, y) \tag{4.11}$$

$$\text{Frequency domain : } G(w_1, w_2) = F(w_1, w_2)H(w_1, w_2) \tag{4.12}$$

where  $h(x, y)$  and  $H(w_1, w_2)$  are filter transfer functions.



**Fig. 4.13** Image in frequency domain [6] **a** Original image **b** Image in frequency domain

The image at right of Fig. 4.13 shows the frequency image of the original image on the left.

Another advantage of using the frequency domain is that it can help us to understand the filter more easily by means of frequency analysis. It can manipulate images with different frequencies, like the low pass filter or high pass filter. The operation of filtering in the frequency domain consists of [2]

Step 1: transforming the image data to the frequency domain via FFT

Step 2: multiplying the image's spectrum with a filtering mask

Step 3: transforming the spectrum back to the spatial domain.

In the following section, we consider two major filters in the frequency domain: smoothing filters and sharpening filters.

#### 4.1.3.1 Smoothing Filter in the Frequency Domain

Smoothing is achieved in the frequency domain by high frequency attenuation, which is low pass filtering. There are three types of low pass filters: ideal, Butterworth, and Gaussian. These three categories cover the range from very sharp (ideal) to very smooth (Gaussian) filtering. The Butterworth filter is viewed as a transition between the two extremes.

The ideal low pass filter cuts off all high frequency components outside a circle of a specified radius  $D_0$  from the origin of the transform. It is specified by the following equation [1]:

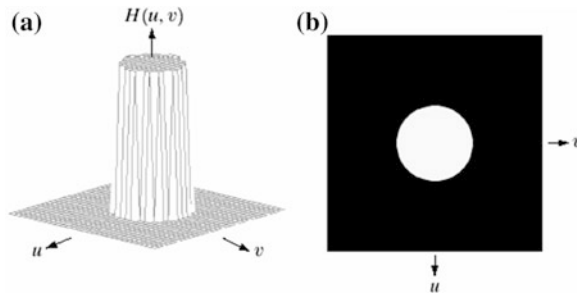


Fig. 4.14 a Perspective plot of an ideal low pass filter transfer function and b the filtered image [1]

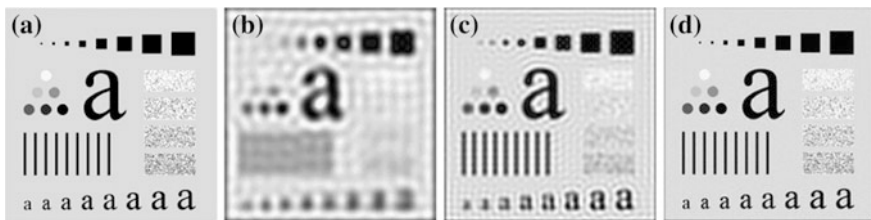


Fig. 4.15 Results of ideal low pass filtering with different cutoff frequencies [1] a Original image b  $D_0 = 30$  case c  $D_0 = 60$  case d  $D_0 = 160$  case

$$H(u, v) = \begin{cases} 1 & \text{if } D(u, v) \leq D_0 \\ 0 & \text{if } D(u, v) > D_0 \end{cases} \quad (4.13)$$

where  $D_0$  is a positive constant and  $D(u, v)$  is the distance between a point  $(u, v)$  in the frequency domain and the center of the frequency rectangle; that is,

$$D(u, v) = [(u - P/2)^2 + (v - Q/2)^2]^{1/2} \quad (4.14)$$

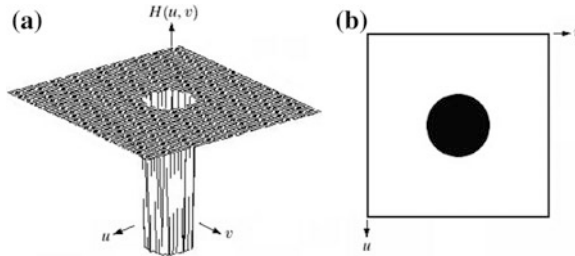
Figure 4.14 shows a perspective plot of an ideal low pass filter and filter, displayed as an image.

Figure 4.15 shows the results of applying ideal low pass filters with different cutoff frequencies.

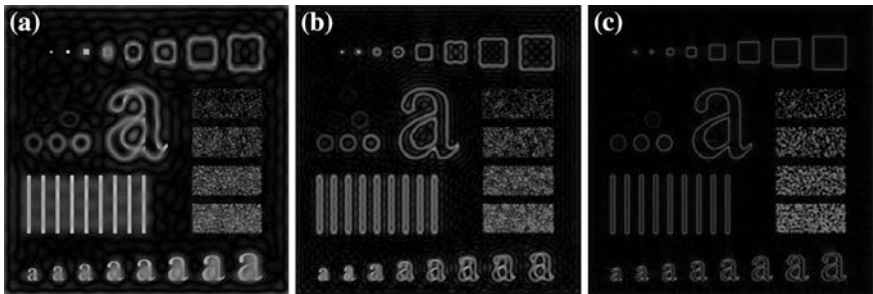
As shown in this figure, when the radius of the filter increases, the blurring of the resulting image diminishes. More details about further examples of low pass filters can be found at section 4.8 in reference [1].

### 4.1.3.2 Sharpening Filter in the Frequency Domain

Image sharpening can be achieved in the frequency domain by high pass filtering, which attenuates the low frequency components without disturbing high frequency information in the Fourier transform. As noted above, there are three types of high



**Fig. 4.16** **a** Perspective plot of an ideal high pass filter transfer function and **b** the filtered image [1]



**Fig. 4.17** Results of ideal high pass filtering with different cutoff frequencies [1] **a**  $D_0 = 30$  case **b**  $D_0 = 60$  case **c**  $D_0 = 160$  case

pass filters: ideal, Butterworth, and Gaussian. In this section, we deal with the ideal high pass filter.

The ideal high pass filter stresses edges and abrupt changes in an image. It performs the reverse operation of the ideal low pass filter. A high pass filter is obtained from a given ideal low pass filter using the equation

$$H_{hp}(u, v) = 1 - H_{lp}(u, v) \tag{4.15}$$

where  $H_{lp}(u, v)$  is the transfer function of the low pass filter. That is, when the low pass filter attenuates frequencies, the high pass filter lets them pass, and vice versa.

A 2-D ideal high pass filter is defined as

$$H(u, v) = \begin{cases} 0 & \text{if } D(u, v) \leq D_0 \\ 1 & \text{if } D(u, v) > D_0 \end{cases} \tag{4.16}$$

Figure 4.16 (new) shows a perspective plot of an ideal high pass filter and image representation.

Figure 4.17 shows the results of ideal high pass filtering.

At the images with small radius  $D_0$ , ringing effects are observed. More details about further examples of high pass filters can be found at section 4.9 in reference [1].



**Fig. 4.18** Examples of road images taken at night and in rainy conditions **a** Image taken at night. **b** Image taken in rainy conditions

## 4.2 Image Enhancement Techniques for Automobile Application

### 4.2.1 Outline

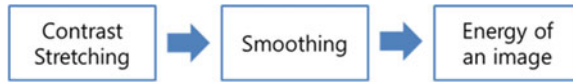
Every image serves a purpose, and image enhancement should be used appropriate to that purpose. Image enhancement of the image of a person aims to make the image clearer to the human eye. The primary goal of image enhancement for application in automobiles, however, is to modify the image to achieve better object recognition. This is because most automotive applications are based on recognition of objects such as pedestrians, lanes, other vehicles detection, and signals.

Image-based pedestrian/vehicle detection is a pattern recognition problem. Feature extraction is a most important step in determining the performance (positive and false detection rate) of the pattern recognition system. Feature extraction is essentially the detection of edges and boundaries. For example, it is well known that horizontal and vertical edges are strong cues for hypothesizing vehicle detection.

A visual processing system needs to function well under a wide dynamic range of visibility conditions, including overcast sky, strong highlights, low visibility due to inclement weather, change of context, and daytime and nighttime driving [7]. The driving environment is also influenced by weather conditions such as rain, fog, or shadow. The images acquired in these conditions are quite different from images taken on a normal light and sunny day. Figure 4.18 shows examples of road images acquired at night and in rainy conditions.

When one tries to recognize objects from these low contrast images, the detection rate is very low, certainly inadequate for auto navigation. Image enhancement is necessary to make the edges of objects clear and improve the detection rate of objects in the driving environment.





**Fig. 4.19** Image enhancement process in the spatial domain for vehicular application

There are three types of enhancement algorithm for automobile application. The first is image enhancement in the spatial domain. The second is image enhancement in the frequency domain, which requires domain transformation from the spatial domain to the frequency domain, using Fourier transform or wavelet transform. The third is the high dynamic range (HDR) imaging processing algorithm.

## 4.2.2 Image Enhancement in the Spatial Domain

Research on image processing for the vehicular environment is still in its infancy. Several types of applications exist, each requiring a different kind of processing. In this chapter, a pre-processing technique for object recognition will be introduced as an example of vehicular image processing. The object recognition rate is the main goal of this enhancement process.

Figure 4.19 shows an example of image enhancement steps in the spatial domain. As a first step, the contrast stretching technique is applied to enhance contrast in the image. In the next, a smoothing process is performed to remove the artifacts of the image. The edge strength of the image is then calculated to produce clear edges of objects in the image.

### (1) Step 1: Contrast stretching

Contrast stretching is a technique to improve the quality of an image by extending the contrast (intensity range) of the input image. If the brightness of an image is dark or too bright overall, visibility is quite low. The contrast stretching technique makes the edges of the objects clearer.

Figure 4.20 shows a typical example of a low contrast image. The brightness of the image on the left is excessive, and it is not easy to detect the white line on the road. Applying the contrast stretching algorithm yields the image on the right, in which the white line becomes more visible.

### (2) Step 2: Smoothing process

The smoothing process is a technique that can remove image artifacts, which are mainly high frequency components of an image. When an image is enhanced by a smoothing algorithm, peak points and thermal noise are eliminated or reduced. In the enhanced image, a blurring effect occurs, which is one characteristic of the smoothing algorithm. Figure 4.21 shows the result of applying a smoothing algorithm to the contrast stretched image.



**Fig. 4.20** Low contrast image and contrast stretched image **a** Low contrast image **b** Contrast stretched image

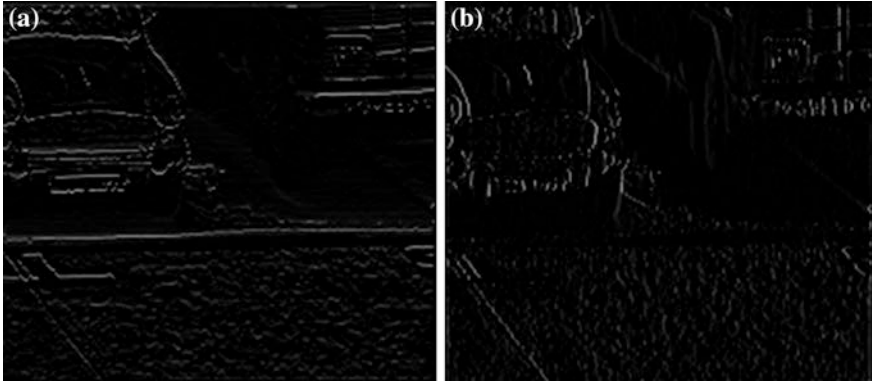


**Fig. 4.21** Contrast stretched image and smoothing filter adjusted image **a** Contrast stretched image. **b** Smoothed image

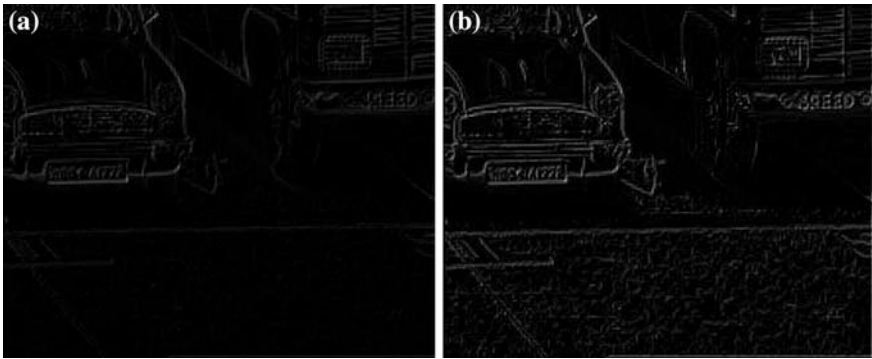
In the spatial domain, the smoothing algorithm is realized by applying a mask. There are several kinds of masks for smoothing, including the Gaussian mask and the averaging mask.

### (3) Step 3: Calculation of edge strength

Edge (or contour information) is one of the most useful means of representing the characteristic of an object. Some calculation of the strength of edge is therefore important, for which many algorithms are available. Determining the energy of an image is a typical method for calculating the edge strength. The first step in this calculation is obtaining the vertical and horizontal gradients of the image. Gradient means the difference between pixel values. Equations (4.17) and (4.18) are equations of horizontal and vertical gradient, respectively.



**Fig. 4.22** Vertical and horizontal gradients of an image **a** Vertical gradient image. **b** Horizontal gradient image



**Fig. 4.23** Energy of original and enhanced images **a** Energy of the original image. **b** Energy of the enhanced image

$$g_x = n(x + 1, y) - n(x - 1, y) \quad (4.17)$$

$$g_y = n(x, y + 1) - n(x, y - 1) \quad (4.18)$$

where  $n(x, y)$  is the intensity of target location for gradient calculation. The energy of an image is calculated by adding the absolute values of each gradient.

$$E = |g_x| + |g_y| \quad (4.19)$$

The vertical and horizontal gradients of Fig. 4.19 b are shown as Fig. 4.22.

Figure 4.23 shows the difference between two energy images. In the energy of the original image shown in Fig. 4.21a, most of the edges are not clear. On the other hand, the energy of the enhanced image, which is processed in the spatial domain, shows clearer edges and is much better for object recognition.

### 4.2.3 Image Enhancement in the Frequency Domain

Restoration of an image may be difficult when a vehicular-mounted camera captures an image in a road environment of rapid variation of luminance, such as the presence of the high beam of a car on the opposite side of the road or an irregular change of luminance. These kinds of illuminance problem make image enhancement processing in the spatial domain difficult. As a result, image enhancement in the frequency domain is widely used for vehicular images. To enhance an image in the frequency domain, domain transformation is required in advance. The most familiar transformation method is the Fourier transform. However, unlike other applications, real-time processing of vehicular images requires information from both the spatial and the frequency domain. The Fourier transform loses all information from the spatial domain, which makes it unsuitable for automobile application. Wavelet transform is proposed to satisfy this requirement. Wavelet transform is also much faster than discrete Fourier transform (DFT), and hence is more attractive for vehicular application.

#### 4.2.3.1 Wavelet Transform

Wavelet transform is one of the most popular methods to transform from the spatial to the frequency domain. For the continuous, square-integrable function  $f(x)$ , the definition of wavelet transform is defined as Eq. (4.20):

$$W_{\Psi}(s, \tau) = \int_{-\infty}^{\infty} f(x)\Psi_{s,\tau}(x)dx \quad (4.20)$$

where wavelet  $\Psi_{s,\tau}(x)$  is defined as

$$\Psi_{s,\tau}(x) = \frac{1}{\sqrt{s}}\Psi\left(\frac{x-\tau}{s}\right) \quad (4.21)$$

and  $s$  and  $\tau$  are called *scale* and *translation* parameters, respectively.

#### 4.2.3.2 Wavelet Series Expansion

Wavelet transform can be derived from wavelet series expansion. The wavelet series expansion of function  $f(x)$  is related to the scaling function  $\varphi(x)$  and the wavelet  $\Psi(x)$ . The scaling function is controlled by the variable  $j$ . Consider the set of expansion functions composed of integer translations and binary scaling of the real, square-integrable function  $\varphi(x)$ ; this is the set  $\{\varphi_{j,k}(x)\}$ , where

$$\varphi_{j,k}(x) = 2^{j/2} \varphi(2^j x - k) \quad (4.22)$$

Here,  $k$  determines the position of  $\varphi_{j,k}(x)$  along the  $x$ -axis, and  $j$  determines the width of  $\varphi_{j,k}(x)$ . The term  $2^{j/2}$  controls the amplitude of the function. The wavelet  $\Psi(x)$  is the basis function of wavelet series expansion.

The equation for the wavelet series is defined as

$$f(x) = \sum_k c_{j_0}(k) \varphi_{j_0,k}(x) + \sum_{j=j_0}^{\infty} \sum_k d_j(k) \Psi_{j,k}(x) \quad (4.23)$$

where  $j_0$  is an arbitrary starting scale, the  $c_{j_0}(k)$  are scaling coefficients, and  $d_j(k)$  are referred to as wavelet coefficients. If the expansion functions form an orthonormal basis or tight frame, the expansion coefficients are calculated as

$$c_{j_0}(k) = \int f(x) \varphi_{j_0,k}(x) dx \quad (4.24)$$

$$d_j(k) = \int f(x) \Psi_{j,k}(x) dx \quad (4.25)$$

#### 4.2.3.3 Discrete Wavelet Transform (DWT)

Like the Fourier series expansion, the wavelet series expansion maps a function of a continuous variable into a sequence of coefficients. If the function being expanded is discrete, the resulting coefficients are called the discrete wavelet transform. If  $f(n) = f(x_0 + n\Delta x)$  for some  $x_0$ ,  $\Delta x$  and  $n = 0, 1, 2, \dots, M-1$ , the wavelet series expansion coefficients for  $f(x)$  [defined by Eqs. (4.24) and (4.25)] become the forward DWT coefficients for sequence  $f(n)$ :

$$W_\varphi(j_0, k) = \frac{1}{\sqrt{M}} \sum_n f(n) \varphi_{j_0,k}(n) \quad (4.26)$$

$$W_\Psi(j, k) = \frac{1}{\sqrt{M}} \sum_n f(n) \Psi_{j,k}(n) \quad j \geq j_0 \quad (4.27)$$

The  $\varphi_{j_0,k}(n)$  and  $\Psi_{j,k}(n)$  in these equations are sampled versions of basis functions  $\varphi_{j_0,k}(x)$  and  $\Psi_{j,k}(x)$ . In accordance with Eq. (4.23), the complementary inverse DWT is Eq. (4.28).

$$f(n) = \frac{1}{\sqrt{M}} \sum_k W_\varphi(j_0, k) \varphi_{j_0,k}(n) + \frac{1}{\sqrt{M}} \sum_{j=j_0}^{\infty} \sum_k W_\Psi(j, k) \Psi_{j,k}(n) \quad (4.28)$$

#### 4.2.3.4 Wavelet Basis Function

For the wavelet transformation, the wavelet is used as the basis function. The wavelet is a small wave that has varying frequency and limited duration [1]. Many kinds of wavelet basis functions exist, such as the Morlet wavelet, Shannon wavelet, Daubechise wavelet, and Haar wavelet [8–10].

The Haar wavelet is a powerful wavelet and the most widely used. It is a sequence of rescaled “square-shaped” functions, which together form a wavelet family or basis. Haar used these functions to give an example of an orthonormal system for the space of square-integrable functions on the unit interval  $[0, 1]$ . The technical disadvantage of the Haar wavelet is that it is not continuous and therefore not differentiable. This property can, however, be an advantage for the analysis of signals with sudden transitions, such as monitoring tool failure in machines. The Haar wavelet’s mother wavelet function  $\Psi(t)$  can be described as follows:

$$\Psi(t) = \begin{cases} 1 & (0 \leq t \leq 1/2) \\ -1 & (1/2 \leq t \leq 1) \\ 0 & \textit{otherwise} \end{cases} \quad (4.29)$$

#### 4.2.3.5 Image Enhancement in the Wavelet Domain

Wavelet transform allows the original image to be divided into several sub images. If the wavelet transform is performed once, then the result will be four sub images, and each sub image is filtered twice. The filtered images will be, from the top left in clockwise order, low pass-low pass (LL), low pass-high pass (LH), high pass-low pass (HL), and high pass-high pass (HH). Figure 4.24 shows an example of the result of wavelet transform. If more sub images are needed, additional wavelet transform can be performed on each sub image.

The LL image has low frequency components of the original image and still permits the shape of the image to be seen. The LL image thus indicates which part of the original image has low frequency. By the same principle, because the HH image has high frequency components, it indicates which part of the original image has high frequency. An important fact is that most energy is concentrated in the LL region. If more precise information of the image is required, two or more stages of wavelet transform can be performed.

Sub images from DWT can be composed into one image with the inverse DWT. If there was no change of information, the restored image will be the same as the original image. However, if there were changes in any of the sub regions, the restored image will not be the same. Hence, if the proper filtering method is adjusted for each sub image, an enhanced image can be acquired by inverse DWT of the filtered images.

Figure 4.25 is an example of a flow chart for an image enhancement technique using DWT and inverse DWT [11]. This non-linear technique for image enhancement is one of the most recent developments based on the characteristics

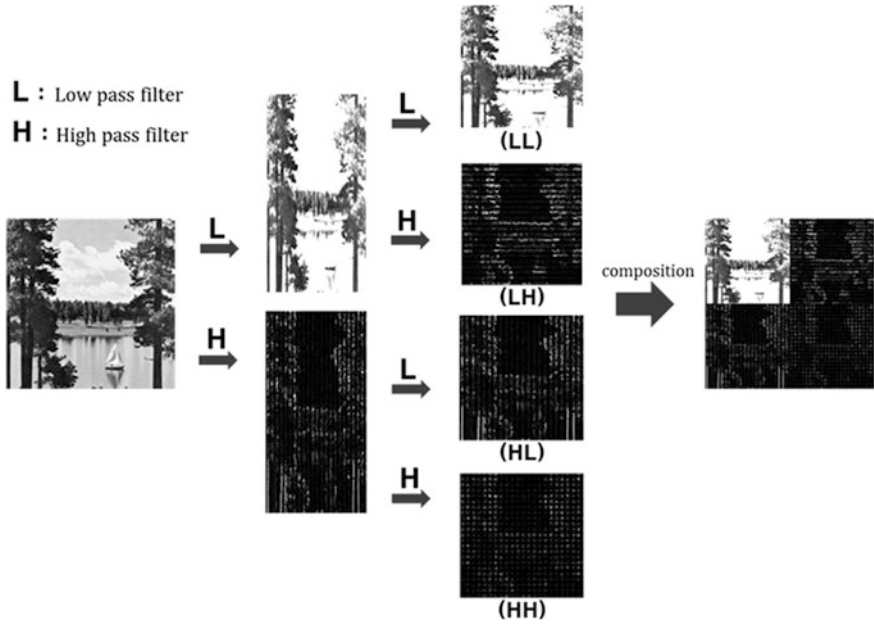
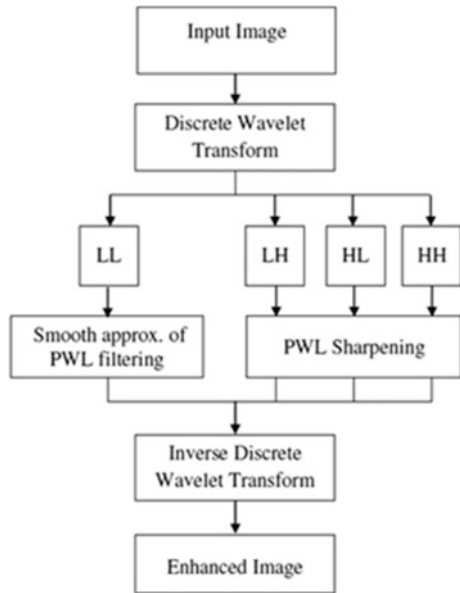
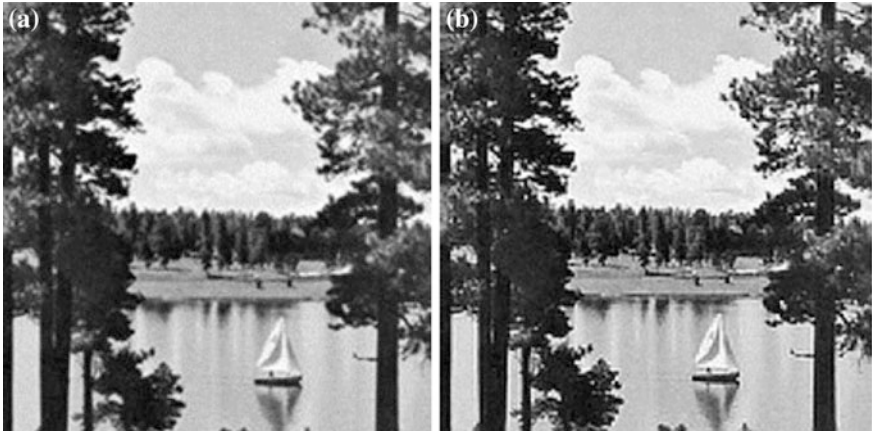


Fig. 4.24 Example of wavelet transform

Fig. 4.25 Example of image enhancement using DWT





**Fig. 4.26** Comparison between the original image and the enhanced image with DWT and PWL filtering **a** Original image **b** Enhanced image

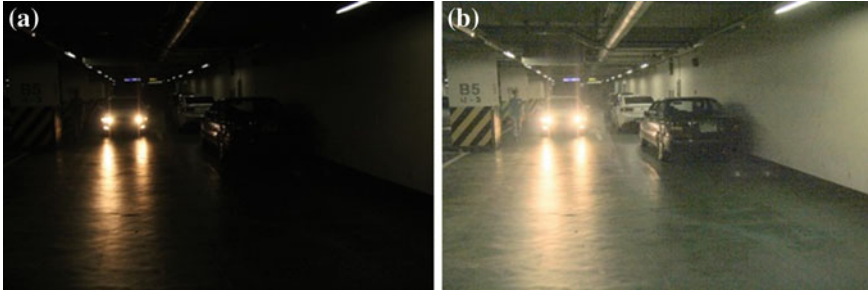
of DWT. The algorithm adopts piecewise linear (PWL) filtering with discrete wavelet transform. The algorithm was designed to suppress noise in the low frequency region. Thus, the LL region is filtered with a smooth PWL filter and other regions are filtered with a PWL sharpened filter. This process removes noise in the low frequency region and sharply enhances the high frequency regions. All regions are then combined with inverse discrete wavelet transform.

Figure 4.26 shows DWT and PWL filtering adjusted images [11]. The original image is divided into four sub images with one stage DWT. All sub images are enhanced with the PWL filtering algorithm described in Fig. 4.23. In the enhanced images, a remarkable improvement can be observed, allowing, for example, the leaves of the trees, the edges of the boat, and other details to be clearly distinguished.

#### ***4.2.4 High Dynamic Range Tone-Mapping Technique***

In the driving environment, it is hard to recognize objects when there is interference from the high beam of a vehicle on the opposite side of the road or when the vehicle is passing through a tunnel. This situation is caused by the common camera's limited capacity for detecting luminance. Dynamic range is defined as the ratio between the largest and smallest luminance values [12]. When the scene has high dynamic range (HDR), this range must be reduced in order to improve object recognition, which is typically done with HDR tone-mapping techniques.





**Fig. 4.27** Original and tone-mapped images **a** Original image **b** Tone-mapped image

#### 4.2.4.1 Concept of Tone-Mapping

The specific algorithm that performs the reduction of dynamic range is referred to as tone-mapping (or tone reproduction). The display of a tone-mapped image should perceptually match the depicted scene. Since dynamic range reduction requires the preservation of certain scene characteristics, it is important to study how people perceive scenes and images [3].

When there is a high beam on a vehicle, as shown in Fig. 4.25a, the quality of image captured by a vehicle-mounted camera is not adequate for recognition. Tone-mapping reduction yields the kind of image shown in Fig. 4.27b, which produces much clearer recognition.

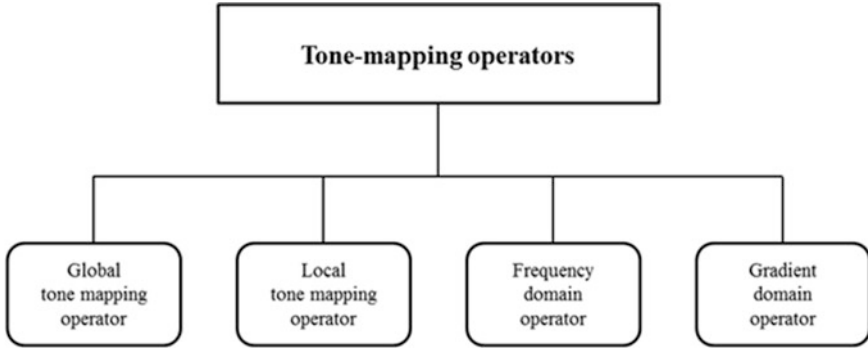
Nowadays, this kind of tone-mapping technique is widely used for object recognition in the automobile industry

#### 4.2.4.2 Tone-Mapping Operators

Tone-mapping operators can be classified into four types: the global tone-mapping operator and the local tone-mapping operator in the spatial domain, and, in the non-spatial domain, an operator that manipulates images in the frequency domain and another that manipulates images in the gradient domain. Figure 4.28 shows the classification of tone-mapping operators.

The global tone-mapping operator compresses the entire image using an identical mapping curve for each pixel. The local tone-mapping operator computes a local adaptation level for each pixel, based on the pixel value itself, as well as a neighborhood of pixels surrounding the pixel of interest [3].

The frequency domain operator requires a domain conversion, which is commonly performed by FFT and IFFT. The gradient domain operator also requires domain conversion, which is commonly performed by gradient calculation and the Poisson equation solver.



**Fig. 4.28** Classification of tone-mapping operators

(a) Global tone-mapping operator

The global tone-mapping operator compresses the entire image using a single compression curve. This technique is the simplest one to reduce an image's dynamic range intuitively.

A well-known global tone-mapping operator is Stockham's algorithm [13], which uses Eq. (4.30) as a mapping function [3].

$$L_d(x, y) = \frac{\log_b(1 + L_w(x, y))}{\log_b(1 + L_{w,max})} \quad (4.30)$$

where the displayed luminance  $L_d$  is derived from the ratio of real world luminance  $L_w$  and maximum luminance in the scene  $L_{w,max}$ . This mapping ensures that, whatever the dynamic range of the scene, the maximum value is remapped to one (white), and other luminance values are smoothly incremented [13].

The operator effectively applies a logarithmic compression to the input luminances, but the base of the logarithm is adjusted according to each pixel's value. The base is varied between 2 and 10, allowing contrast and detail preservation in dark and medium luminance regions, while still compressing light regions by larger amounts.

Drago's logarithm tone-mapping [14] is proposed to interpolate smoothly between different bases. Drago uses Eq. (4.31) as a mapping function.

$$L_d(x, y) = \frac{L_{d,max}/100}{\log_{10}(1 + L_{w,max})} \cdot \frac{\log_{10}(1 + L_w(x, y))}{\log_{10}(2 + 8\left(\frac{L_w(x, y)}{L_{w,max}}\right) \log_{10}(p) / \log_{10}(0.5))} \quad (4.31)$$

where  $L_{d,max}$  is the maximum display luminance, which is display dependent and should be specified by the user. In most cases, a value of  $100 \text{ cd}/\text{m}^2$  would be appropriate for  $L_{d,max}$ . This algorithm leaves the bias parameter  $p$  to be specified by the user. For many applications, a value between 0.7 and 0.9 produces plausible



**Fig. 4.29** Images resulting from application of Drago's tone-mapping algorithm with different bias values [3] **a**  $p = 0.6$ . **b**  $p = 0.7$ . **c**  $p = 0.8$ . **d**  $p = 0.9$ . **e**  $p = 1.0$

results,  $p = 0.85$  being a good initial value. Figure 4.29 shows images created with different bias values [3].

The computation cost of global tone-mapping operators is very low, and many may be executed in real time. However, global tone-mapping operators are weaker at preserving visibility when the scene's dynamic range is extremely high and local characteristics are not well depicted.

## (b) Local tone-mapping operator

The local tone-mapping operator computes a local adaptation level for each pixel, based on the pixel value itself, as well as a neighborhood of pixels surrounding the pixel of interest. This local adaptation level drives the compression curve for this pixel. Because the neighborhood of a pixel helps determine how the pixel is compressed, a bright pixel in a dark neighborhood will be treated differently from a bright pixel in a bright neighborhood. A similar situation pertains for dark pixels with bright and dark neighborhoods [3]. This process is effective for preserving the locality of images.

Local tone-mapping techniques have been studied in various ways: to determine how many neighboring pixels need to be included in the computation, how to weight each neighboring pixel's contribution to the local adaptation level, and how to use this adaptation level within a compression function [3].

Chiu proposes a local tone-mapping operator [15]. This algorithm uses the following equations as a mapping function [3]:

$$L_d(x, y) = s(x, y)L_w(x, y) \quad (4.32a)$$

$$s(x, y) = \frac{1}{k \bullet L_w^{blur}(x, y)} \quad (4.32b)$$

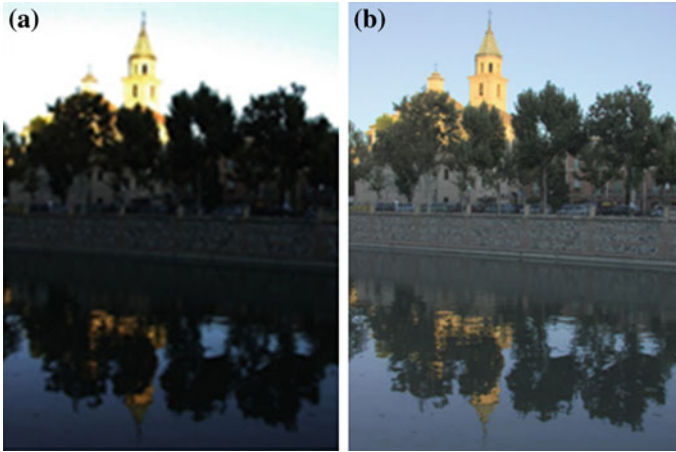
Where  $L_d$  is displayed value,  $L_w$  is the real world scene's luminance, and  $s(x, y)$  is a low pass filtered version of the input image [3]. Here,  $k$  is the user-defined parameter that controls the  $s(x, y)$ , and  $L_w^{blur}$  is the luminance of the blurred image. Usually  $k$  is bigger than 1.  $s(x, y)$  can be controlled by changing the blur kernel size and changing the user-defined parameter  $k$ .

Another well-known local tone-mapping operator, based on Chiu's work, is the retinex algorithm developed by Rahman [16]. This algorithm applies a Gaussian-blurred kernel to input the image in the log domain to reduce a halo effect. The blurred images can be generated as two different forms: single-scale retinex or multi-scale retinex. The single-scale retinex form uses the following equation as a mapping function [3]:

$$I_d(x, y) = \exp(\log(I_w(x, y)) - k \log(I_w^{blur}(x, y))) \quad (4.33)$$

Where  $I_d$  is display luminance value in the log space,  $I_w$  is world luminance value, and  $I_w^{blur}$  is the blurred image's luminance. In single-scale retinex form, this algorithm operates in the log domain. However, the placements of the logarithms are somewhat peculiar (namely, after the image is convolved with a Gaussian filter kernel). The multi-scale retinex form uses the following equation as a mapping function [3]:

$$I_d(x, y) = \exp\left(\sum_{n=0}^N w_n (\log(I_w(x, y)) - k \bullet \log(I_w^{blur}(x, y)))\right) \quad (4.34)$$



**Fig. 4.30** Original and Rahman tone-mapped image **a** Original image. **b** Rahman tone-mapped result

where  $w_n$  is a weight function. The multi-scale retinex form is simply the weighted sum of a set of single scale retinexed images. The weight given to each scale is determined by the user. For the purposes of experimentation, it is convenient to weight each level by a power function, which gives straightforward control over the weights. For an image stack with  $N$  levels, the normalized weights are then computed by the following equation [3]:

$$w_n = \frac{(N - n - 1)^f}{\sum_{m=0}^N (N - m - 1)^f} \quad (4.35)$$

where  $N$  is the number of scaled images. Figure 4.30 shows the result of using the Rahman retinex tone-mapping algorithm.

There are two user-defined parameters in the retinex algorithm:  $k$ , which is the scale of the second log term, and the weight function,  $w_n$ . The user-defined parameters have a trade-off relationship. Larger values of  $k$  will cause the compression to be more dramatic, but also create larger halo effects [3].

A local tone-mapping operator produces results that are more natural and represent details well. However, it creates a halo effect and the computing cost is very high because it applies a different curve for every single pixel.

There are also other local tone-mapping operators, such as the Fairchild iCAM algorithm [17], which uses a color appearance model, and the Pattanaik multi-scale observer model [18], which is an improved algorithm of Fairchild iCAM's.

### (c) Frequency domain operator

The frequency domain operator is a frequency-dependent compressor that converts the spatial domain data into frequency domain data using FFT and IFFT.

One such operator is the Oppenheim operator [19], the basic concept of which is that low frequency components are attenuated more than higher frequency components in images. As the high frequency component is related to edge information, this algorithm preserves edge information well.

The frequency-based operator starts by converting images in the spatial domain into the frequency domain in the log space. In the frequency domain, tone reproduction is performed as follows:

$$I_d(f) = s(f) \bullet I_w(f) \quad (4.36)$$

where  $I_d(f)$  is log luminance value in the frequency domain,  $s(f)$  is an attenuating factor that depends on frequency  $f$ , and  $I_w(f)$  is the real world scene's log luminance value in the frequency domain.  $s(f)$  is calculated by the following equation [3]:

$$s(f) = (1 - c) + c \frac{k \bullet f}{1 + k \bullet f} \quad (4.37)$$

where  $c$  and  $k$  are the user-defined parameters. The parameter  $c$  controls the maximum amplitude of attenuation applied to the DC components, and the parameter  $k$  controls the slope of the attenuating factor for a specific frequency.

Figure 4.31 shows the results depending on different  $c$  values in Oppenheim's algorithm. The value of  $c$  is sequentially 0.1, 0.3, 0.5, and 0.9.

Oppenheim suggests the reasonable default value for  $c$  is 0.5.

Further research based on Oppenheim's work has been conducted by Choudhury [20], who proposed trilateral filtering to overcome the drawbacks of bilateral filtering, which is poor in high-gradient and high-curvature regions.

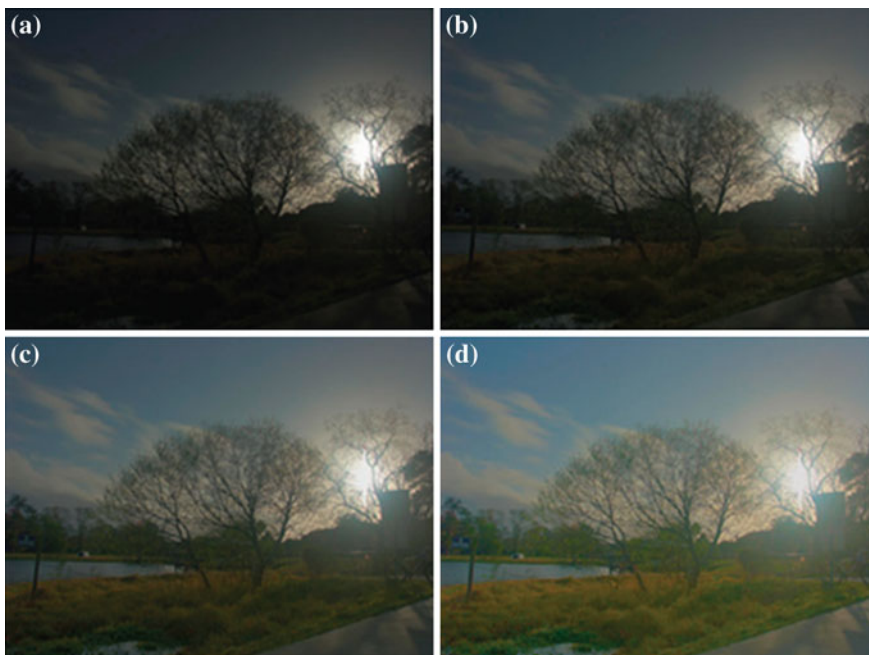
The frequency domain operator is not widely used because of its poor performance. However, it contains several key ideas that have found their way into numerous other tone-reproduction operators.

#### (d) Gradient domain operator

The gradient domain operator is a gradient-dependent compressor, which converts spatial domain data into frequency domain data using simple subtraction and the Poisson equation solver.

High-frequency components in an image cause rapid changes from one pixel to the next. On the other hand, low-frequency features cause the differences between neighboring pixels to be relatively small. It is possible to partially distinguish between illuminance and reflectance in a different way by considering the gradients in the image [3].

The first to explore this notion was Berthold Horn [21], who outlined a computational model of human lightness perception, which is perceptual quantity that correlates with surface reflectance. This work assumes that each pixel of an image is formed as the product of illumination and surface reflectance, as follows:



**Fig. 4.31** Images resulting from Oppenheim's algorithm with different values for  $c$  [3] **a**  $c = 0.1$ , **b**  $c = 0.3$ , **c**  $c = 0.5$ , **d**  $c = 0.7$

$$L_v(x, y) = E_v(x, y)r(x, y) \quad (4.38)$$

where  $L_v(x, y)$  is the pixel's luminance, and  $E_v(x, y)$  and  $r(x, y)$  are illuminance and reflectance components, respectively.

When we take the logarithm value of  $L_v(x, y)$  as density image  $D(x, y)$ ,

$$D(x, y) = \log(L_v(x, y)) \quad (4.39)$$

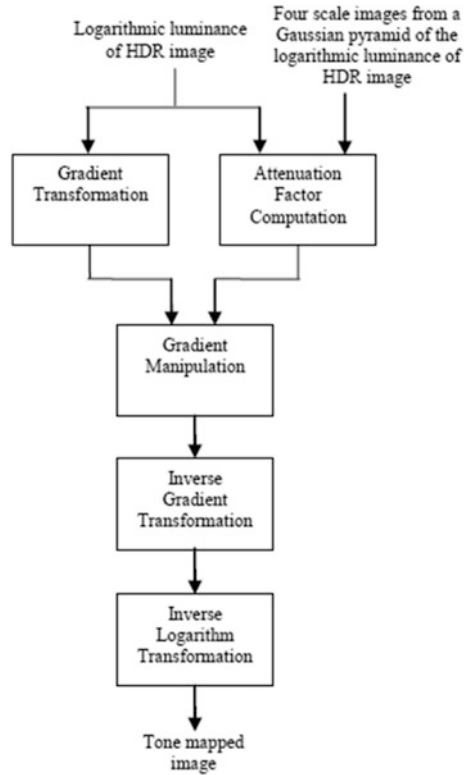
The gradient field of an image can be computed as follows [3]:

$$\begin{aligned} \nabla G(x, y) &= (G_x(x, y), G_y(x, y)) \\ &= (D(x+1, y) - D(x, y), D(x, y+1) - D(x, y)) \end{aligned} \quad (4.40)$$

The compressed gradient field is then expressed by multiplying each gradient by a compressive function  $\varphi(x, y)$ , as follows:

$$\nabla G'(x, y) = \nabla G(x, y) \varphi(x, y) \quad (4.41)$$

**Fig. 4.32** Procedure of Fattal tone-mapping algorithm using four scaled images



A compressed density image  $D'(x, y)$  is constructed by solving the Poisson equation as follows [3]:

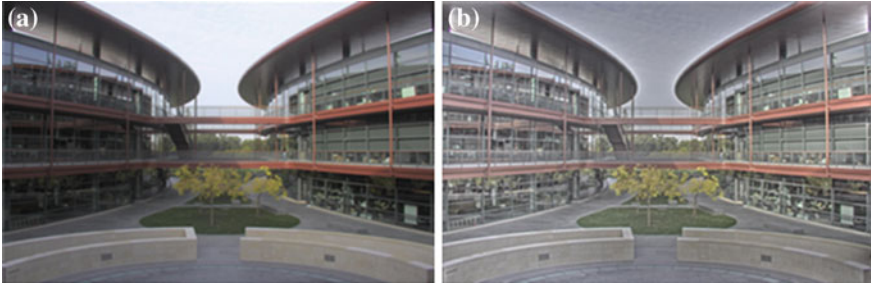
$$\nabla^2 D'(x, y) = \text{div } G'(x, y) \tag{4.42}$$

Finally, the tone-mapped image  $L_d$  is computed using the following equation [3]:

$$L_d(x, y) = \exp\left(D'(x, y)\right) \tag{4.43}$$

Although Horn was largely interested in computational models of human lightness perception, Fattal [22] proposed a gradient domain operator based on Horn’s model. This algorithm uses Gaussian pyramid images [23] when the gradient field is manipulated [3]. Gaussian pyramid images are repeatedly filtered and subsampled images obtained from the original image. The procedure of the Fattal tone-mapping algorithm using four scaled images is shown in Fig. 4.32 [24].





**Fig. 4.33** Original and gradient tone-mapped images **a** Original image **b** Gradient tone-mapped image

As the first step, a Gaussian pyramid  $D_0, D_1, \dots, D_d$  is constructed from the density image  $D(x, y)$ . At each level  $s$ , a gradient field  $\nabla G_s$  is computed as follows [3]:

$$\nabla G_s(x, y) = \left( \frac{D_s(x+1, y) - D_s(x-1, y)}{2^{s+1}}, \frac{D_s(x, y+1) - D_s(x, y-1)}{2^{s+1}} \right) \quad (4.44)$$

where  $D_s$  is a density image for level  $s$ .

As the next step, a scale factor  $\varphi_s(x, y)$  is computed based on the magnitude of the gradient, as follows:

$$\varphi_s(x, y) = \frac{\alpha}{\|\nabla G_s(x, y)\|} \left( \frac{\|\nabla G_s(x, y)\|}{\alpha} \right)^\beta \quad (4.45)$$

where  $\alpha$  and  $\beta$  are user-defined parameters. Gradients larger than  $\alpha$  are attenuated provided that  $\beta < 1$ , whereas smaller gradients are not attenuated and in fact may even be somewhat amplified. After calculating each attenuation factor, compressed gradient fields are accumulated by linear interpolation.

Figure 4.33 shows a gradient tone-mapped image. Although the gradient tone-mapped image looks unnatural, edges are emphasized.

The Fattal algorithm is known as an effective technique for rendering the dynamic range of edge-enhanced HDR images. However, it has a high computational complexity because of the domain conversion procedure. Performing inverse gradient transformation, which is solved by the Poisson equation solver, is the process that takes the most time.

To solve the Poisson equation effectively, many alternative Poisson solver algorithms have been developed, several of which are shown in Fig. 4.34 [24], which includes time complexity for image size and type of solver. Image size  $N$  is the total number of pixels for an image.

Algorithm	Time complexity for image size ( $N$ )	Type of Solver
Gaussian Elimination	$N^3$	Direct
Inverse( $\nabla^2$ ) $\times$ div $\hat{G}$	$N^2$	Direct
Jacobi	$N^2$	Indirect
Conjugate Gradient (CG)	$N^{3/2}$	Indirect
Successive Over Relaxation (SOR)	$N^{3/2}$	Indirect
Multigrid	$N$	Indirect
FFT-based	$N \log N$	Direct

Fig. 4.34 Several Poisson solver algorithms

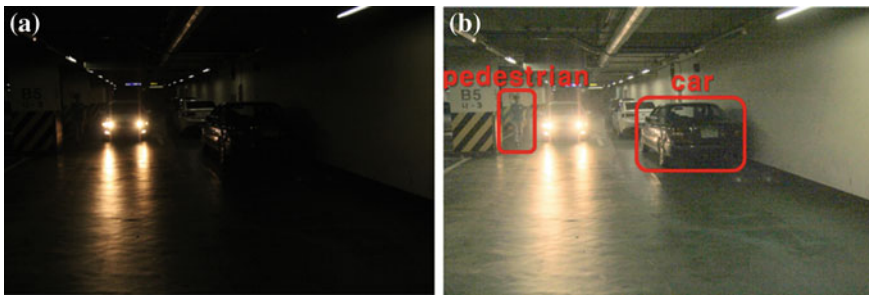


Fig. 4.35 Original and Fattal’s algorithm applied image **a** Original image **b** Fattal tone-mapped image

In 2013, Lavanya [24] proposed a local Poisson solver that provides a direct solution for inverse gradient transformation in real time. This algorithm is easy to implement on hardware and does not need whole pixels of the image.

Although the gradient domain operator does not preserve color information, it emphasizes the edges of objects. Furthermore, it also can be processed in real time using Lavanya’s algorithm. This suggests that the Fattal gradient tone-mapping algorithm is a good candidate for object recognition in a driving environment.

We performed a simulation of Fattal’s algorithm applied to object recognition in a driving environment. The experimental result is shown in Fig. 4.35.

When we applied a vehicle/pedestrian detection algorithm to the original image, detection failed. On the other hand, when we applied the same algorithm to the Fattal tone-mapped image, both car and pedestrian could be readily detected. This result, shown in Fig. 4.35b, indicates that Fattal’s tone-mapping algorithm is effective for object recognition in the driving environment.

## References

1. R. Gonzales, R. Woods, *Digital Image Processing*, (Prentice Hall, 2010)
2. R. Crane, *Simplified Approach to Image Processing*, (Prentice Hall, 1997)
3. E. Reinhard, G. Ward, S. Pattanaik, P. Debevec, *High Dynamic Range Imaging: Acquisition, Display and Image-Based Lighting*, (Morgan Kaufman, 2005)
4. S.-D. Chen, R. Ramli, Contrast enhancement using recursive mean-separate histogram equalization for scalable brightness preservation. *IEEE Trans. Consum. Electron.* **49**(4), (2003)
5. M. Ahmad, D. Sundararajan, A fast algorithm for Two-Dimensional median Filtering. *IEEE Trans. Circuits Syst.* **34**(11), 1364–1374 (1987)
6. <http://blogmolinodeagua.blogspot.kr/>
7. A. Shashua, Y. Gdalyahu, G. Hayun, Pedestrian detection for driving assistance systems: single-frame classification and system level performance. *IEEE intelligent vehicles symposium*, pp. 1–6 (2004)
8. K.K. Mohanty, The wavelet transform for local image enhancement. *Int. J. Remote Sens.* **18**(1), 213–219 (1997)
9. S. Qian, *Introduction to Time-Frequency and Wavelet Transforms*, (Prentice Hall, 2002)
10. C. Sidney Burrus, R. A. Gopinath, H. Guo, *Introduction to Wavelets and Wavelet Transforms: A Primer*, (Prentice Hall, 1998)
11. S. Gopinathan, P. Thangavel, A non linear technique for image enhancement based on discrete wavelet transform. *Eur. J. Sci. Res.* **79**(3), 328–336 (2012)
12. [http://en.wikipedia.org/wiki/Dynamic\\_range](http://en.wikipedia.org/wiki/Dynamic_range)
13. T. G. Stockham. Image processing in the context of a visual model. *Proceedings of the IEEE* **60**:828–842
14. F. Drago et al., *Adaptive Logarithmic Mapping for Displaying High Contrast Scenes*. *Computer Graphics Forum*, vol. 22, no. 3 (Blackwell Publishing, Inc 2003)
15. K. Chiu, M. Herf, P. Shirley, S. Swamy, C. Wang, and K. Zimmerman., Spatially nonuniform scaling functions for high contrast images, in *Proceedings of Graphics Interface '93*, Toronto, 245–253 May 1993
16. Rahman, Zia-ur, D. J. Jobson, G. A. Woodell., *Multiscale retinex for color rendition and dynamic range compression*. in *SPIE's 1996 International Symposium on Optical Science, Engineering, and Instrumentation*. International Society for Optics and Photonics, (1996)
17. M. D. Fairchild and G. M. Johnson., *Meet iCAM : An Image Color Appearance Model*, in *IS&T/SID 10th Color Imaging Conference*, pp. 33–38, Scottsdale : IS&T, (2002)
18. S. N. Pattanaik, J. A. Ferwerda, M. D. Fairchild, and D. P. Greenberg., *A Multiscale Model of Adaptation and Spatial Vision for Realistic Image Display*, in *SIGGRAPH 98 Conference Proceedings (ACM SIGGRAPH)*, 287–298 July 1998
19. A. V. Oppenheim, R. Schaffer, and T. Stockham., Nonlinear filtering of multiplied and convolved signals. *Proceedings of the IEEE* **56**(8), 1265–1291 (1968)
20. Choudhury, Prasun, J. Tumblin. The trilateral filter for high contrast images and meshes. (*ACM SIGGRAPH 2005 Courses*. ACM), (2005)
21. B.K.P. Horn, Determining lightness from an image. *CVGIP* **3**(4), 277–299 (1974)
22. Fattal, Raanan, D. Lischinski, M. Werman. Gradient domain high dynamic range compression. *ACM Trans. Graphics (TOG)* **21**(3), 2002
23. Adelson, H. Edward., et al., Pyramid methods in image processing. *RCA engineer* 29.6, 33–41 (1984)
24. L. Vytla, F. Hassan, J. E. Carletta, A real-time implementation of gradient domain high dynamic range compression using a local Poisson solver. *J. Real-Time Image Proc.* **8**(2), 153–167 (2013)

# Chapter 5

## Detection of Vehicles and Pedestrians

Hyunchul Shin and Irfan Riaz

**Abstract** Vehicle and pedestrian detection has gained the attention of researchers in the past decade because of the increasing number of on road vehicles and traffic accidents. Vehicle and pedestrian detection system is of utmost importance since it can be used to take instantaneous and calculated decisions where human failure might occur resulting in reduction of road mishaps. But designing a detection system which is robust to various shapes of vehicles, different human postures/clothing, and weather/environment conditions is a challenging problem. In the following sections, the state of the art methods in vehicle and pedestrian detection are discussed.

### 5.1 Introduction to Vehicle/Pedestrian Detection

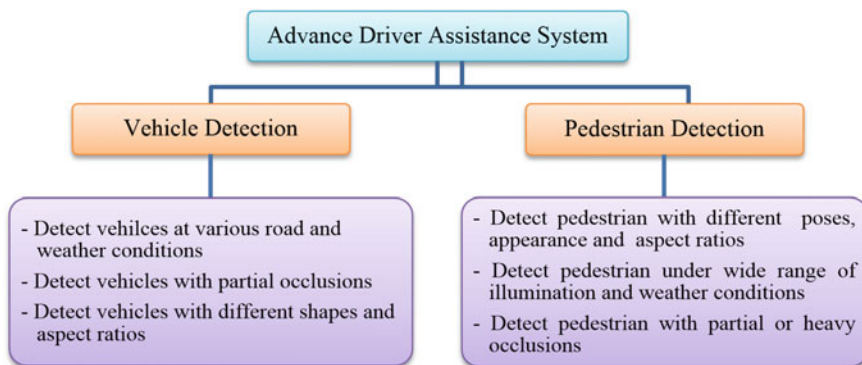
Vehicle and pedestrian detections have gained great attention of researchers in the past decade because the Advanced Driver Assistance Systems (ADAS) is mainly based on these technologies.

There are primarily three big reasons for blooming research in vision-based vehicle detection techniques. The most prominent being the startling losses both to human lives and money, caused by vehicle accidents. Secondly, technology has improved considerably within the last 30 years and thirdly, the exponential growth in processing speed has paved the way for viable implementation of computation intensive vision algorithms.

---

H. Shin (✉) · I. Riaz  
Department of Electronics and Communication Engineering, Hanyang University ERICA  
Campus, Ansan, South Korea  
e-mail: shin@hanyang.ac.kr

I. Riaz  
e-mail: irfancra@digital.hanyang.ac.kr



**Fig. 5.1** Vision for an advance driver assistance system

Pedestrian detection is one of the key problems in computer vision, with several applications that have the potential to greatly enhance the quality of life. In recent years, the number of approaches to detect pedestrians in monocular images has grown steadily. Detecting pedestrians has advanced the frontiers of this problem in many aspects, e.g., features, classifiers, testing speed, night vision, and occlusion handling. Recent experiments show that despite significant progress, there is still much room for improvement in performance and accuracy. Especially, detection rate is disappointing for low resolution images and partially occluded pedestrians [1].

Some of the vital factors in performance evaluation of a vehicle/pedestrian detection systems are presented in Fig. 5.1.

Vision-based Vehicle/Pedestrian detection is a **Pattern Recognition** problem in computer vision. The performance of the detection system depends on two factors: the **features** used for object representation and the **machine learning** method used for classification.

### 5.1.1 Feature Extraction

Features are functions of the original measurement variables which are useful for classification and/or pattern recognition. Feature extraction is the process of defining a set of features, or image characteristics, which will most efficiently or meaningfully represent the information that is important for analysis and classification.

Normally some pre-processing procedures are necessary, like filtering, normalization, histogram equalization and tone-mapping [2], to adjust the overall and local contrast for satisfactory recognition.

We classify the various features currently employed as follows [3]:

### (1) **General Features**

Application independent features such as color, texture, and shape. According to the abstraction level, they can be further divided into:

- (1) Pixel-level features: Features calculated at each pixel, e.g. intensity, color, location.
- (2) Local features: Features calculated over the results of subdivision of the image through segmentation, edge detection, gradient or Local Binary Patterns (LBP) [4].
- (3) Global features: Features calculated over the entire image or just on a regular sub-area of an image.

### (2) **Domain Specific Features**

Application dependent features such as human, faces, fingerprints, and conceptual features. These features are often a synthesis of low-level features for a specific domain.

On the other hand, all features can be coarsely classified into low-level features and high-level features. Low-level features can be extracted directly from the original images, whereas high-level feature extraction must be based on low level features.

## ***5.1.2 Classification***

In the field of machine learning, the goal of statistical classification is to use object's characteristics to identify which class (or group) it belongs to. The term "classifier" refers to the mathematical function, implemented by a classification algorithm, which maps input data to a category.

In machine learning we normally encounter two types of classification procedures, classification is considered an instance of supervised learning, i.e. learning where a features extracted from a training set along with labels is used. Once a classifier is trained it can classify the test data to the nearest feature matching class as shown in Fig. 5.2. The unsupervised learning procedure is known as clustering (or cluster analysis), and involves grouping data into categories based on some measure of inherent similarity (e.g. the distance between instances, considered as vectors in a multi-dimensional vector space).

In the pedestrian/vehicle detection the following classifiers are normally used.

### (1) **Cascade Classifier**

Cascading is based on the concatenation of several classifiers, using all information collected from the output of a given classifier as additional information for the next classifier in the cascade, as shown in Fig. 5.3. The key idea of cascading is that we select a threshold with high recall (i.e. we make our classifier more permissive so it doesn't miss even weak positive candidates) for each stage and we

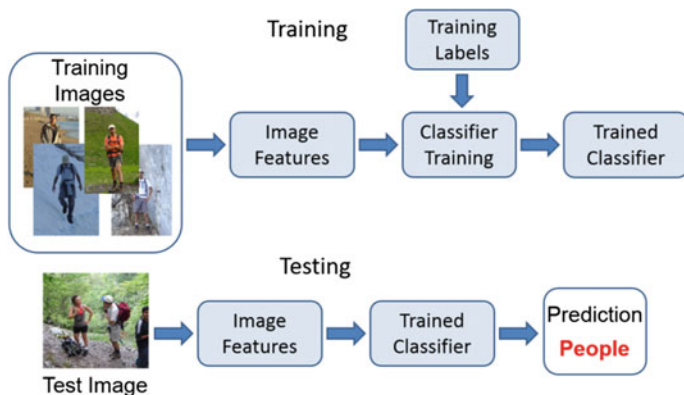
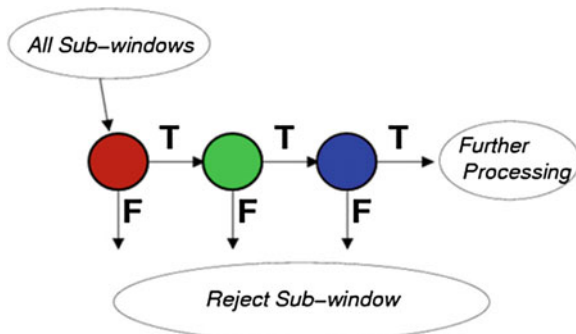


Fig. 5.2 Supervised classifier training and testing

Fig. 5.3 Cascade classifier



increase precision using the cascade. Hence the pure negative candidates get rejected in the early stage and the detection efficiency is improved.

(2) **Bag-of-Words**

Bag-of-words model (BoW model) [5] can be applied to image classification, by treating image features as words. A bag of visual words is a sparse vector of histogram of a vocabulary of local image features. To represent an image using BoW model, an image can be treated as a document. Similarly, “words” in images need to be defined too. To achieve this, it usually includes following three steps: *feature detection*, *feature description*, and *visual words codebook generation*, as shown in Fig. 5.4.

The first step of the algorithm is to collect a series of feature points from training images and cluster them into a visual vocabulary of so-called “visual words.” These features are gathered by examining various attributes such as intensity, color, texture, edge and corner distribution of an image. After feature detection, each image is abstracted by several local patches.

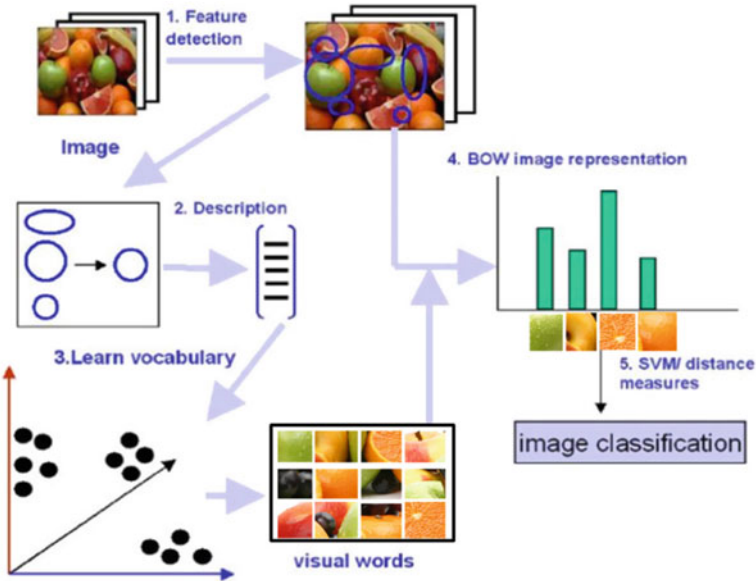


Fig. 5.4 Bag-of-words description

Feature description methods deal with how to represent the patches as numerical vectors. These vectors are called feature descriptors. A good descriptor should have the ability to handle intensity, rotation, scale and affine variations to some extent. One of the most famous descriptors is Scale-invariant feature transform (SIFT) [6]. SIFT converts each patch to 128-dimensional vector. After this step, each image is a collection of vectors of the same dimension (128 for SIFT), where the order of different vectors is of no importance. This allows features to be represented similarly among images with different lighting, size, etc.

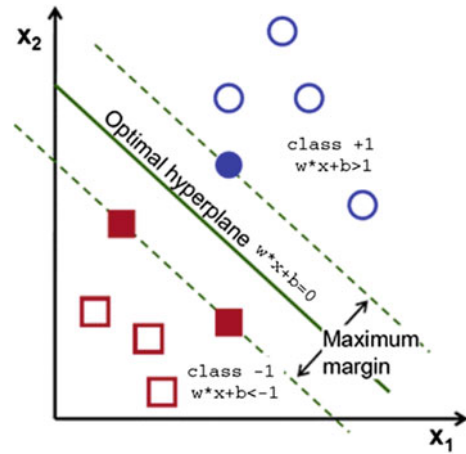
After all of these features are extracted from the training data, they are then clustered using a clustering algorithm, e.g. k-means. These cluster centers can be thought of as “visual words” or “codewords” which produce “codebook” of size equal to the number of clusters formed. A codeword can be considered as a representative of several similar patches. This is where the “bag of words” model comes into play; these “visual words” are not dependent on spatial information in the images. One can abstractly think of it as chopping up an image into many little segments, taking the most relevant ones, and throwing them randomly into a bag.

### (3) Support Vector Machine (SVM)

Support vector machines (SVMs) are supervised learning models with associated learning algorithms that analyze data and recognize patterns, used for classification and regression analysis [7]. SVM is a famous technique for high speed and accurate data classification.



Fig. 5.5 SVM boundary



The basic SVM takes a set of input data and predicts, for each given input, which of two possible classes forms the output, making it a non-probabilistic binary linear classifier. Given a set of training examples, each marked as belonging to one of two categories, an SVM training algorithm builds a model that assigns new examples into one category or the other. An SVM model is a representation of the examples as points in space, mapped so that the examples of the separate categories are divided by a clear gap that is as wide as possible. New examples are then mapped into that same space and predicted to belong to a category based on which side of the gap they fall on.

Formally, the SVM binary classifier algorithm looks for an optimal hyperplane as a decision function in a high-dimensional space, as shown in Fig. 5.5. Thus, consider one has a training data set  $\{x_k, y_k\} \in y_k \times \{-1, 1\}$  where  $x_k$  are the training example feature vector and  $y_k$  the class label. At first, the method consists in mapping  $x_k$  in a high dimensional space owing to a function  $\Phi$ . Then, it looks for a decision function of the form:  $f(x) = w \cdot \Phi(x) + b$  and  $f(x)$  is optimal in the sense that it maximizes the distance between the nearest point  $\Phi(x_i)$  and the hyperplane as shown in the figure. The class label of  $x$  is then obtained by considering the sign of  $f(x)$ .

#### (4) Active Learning

Active learning is very promising in reducing the amount of training data needed and has been applied to various tasks [8]. Instead of assuming that all of the training examples are given at the start, active learning algorithms interactively collect new examples, typically by making queries to a human user. Often, the queries are based on unlabeled data, which is a scenario that combines semi-supervised learning with active learning.

The typical components of active learning setting are shown in Fig. 5.6. The data are divided into (typically few) labeled instances  $L$  and pool of unlabeled instances  $U$ . There is also a learner which is trained on the labeled data and a query

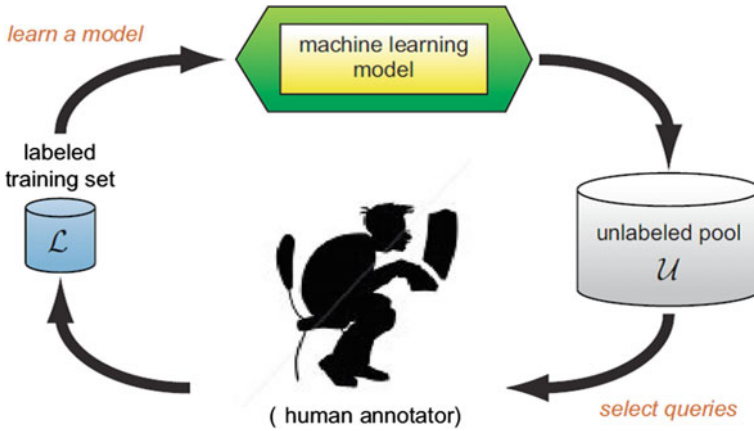


Fig. 5.6 Active learning illustration

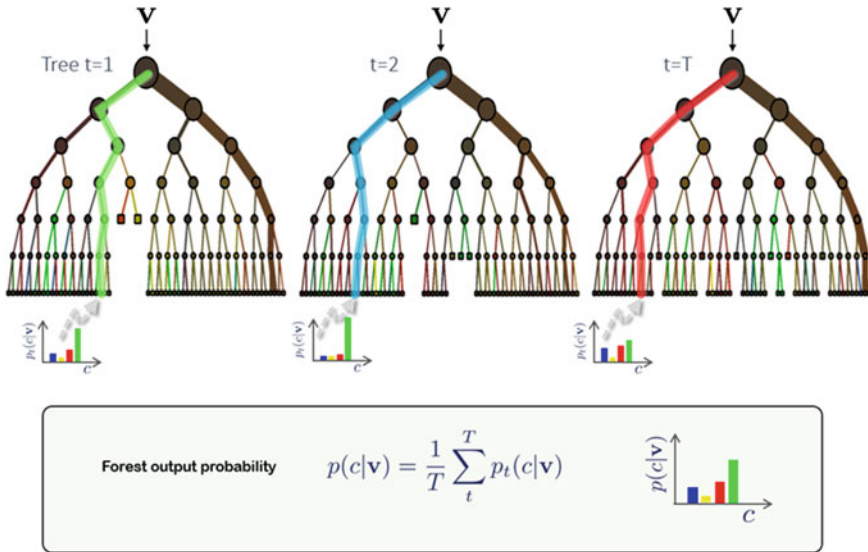
module  $q$ . The module  $q$  decides which instances of  $U$  will be selected to be labeled and added in  $L$ , which in turn will be used to train the learner. In a passive learning setting,  $q$  selects instances randomly, as opposed to active learning where the most informative instances are chosen.

### (5) Random Forest

Random forest is a technique, capable of performing regression and classification analysis of the test data based on the model learned through the training data [9]. It can handle a large number of features, and it's helpful for estimating which of the variables are important in the underlying data being modeled. It belongs to a larger class of machine learning algorithms, called ensemble methods. Ensemble learning involves the combination of several models to solve a single prediction problem. It works by generating multiple classifiers/models which learn and make predictions independently. Those predictions are then combined (by averaging or voting) into a single (mega) prediction that should be as good or better than the prediction made by any one classifier, as shown in Fig. 5.7.

Random forest is a brand of ensemble learning, as it relies on an ensemble of decision (Classification or Regression) trees. A decision tree is composed of a series of decisions that can be used to classify an observation in a dataset. The algorithm to induce a random forest will create a bunch of random decision trees automatically. Since the trees are generated at random, most won't be all that meaningful to learning the classification/regression problem. But what is helpful are the few really good decision trees that are also generated along with the bad ones.

To make a prediction, the new observation goes down through each decision tree which assigns a predicted value/label. Once each of the trees in the forest has reported its predicted value/label, the predictions are tallied up and the mode vote of all trees is returned as the final prediction.



**Fig. 5.7** Typical structure of random forest

Simply, the 99.9 % of trees that are irrelevant make predictions that are all over the map and cancel each other out. The predictions of the minority of trees that are good dominates that noise and yield a good prediction.

## 5.2 Vehicle Detection

### 5.2.1 Challenges

Vehicle detection using optical sensors is very challenging due to huge ‘within class variations’ in vehicle appearance.

- Vehicles may vary in shape (Fig. 5.8a), size, and color. The appearance of a vehicle depends on its pose and is affected by nearby objects (Fig. 5.8b).
- Complex outdoor environments (e.g., illumination conditions (Fig. 5.8c)), unpredictable interaction between traffic participants, cluttered background (Fig. 5.8d) are difficult to control.
- On-road vehicle detection also requires faster processing than other applications since the vehicle speed is bounded by the processing rate.
- Another key issue is the robustness of the system to vehicle’s movements and drifts.



Fig. 5.8 Different cases in vehicle detection [10]

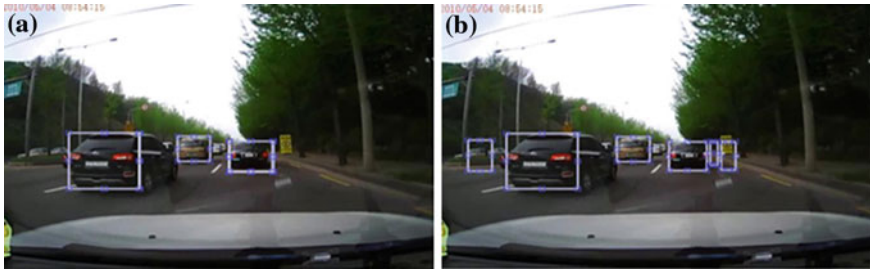
### 5.2.2 Research and Industrial Status

With the ultimate goal of building autonomous vehicles, many government institutions, automotive manufacturers and suppliers, and R&D institutions have launched various projects worldwide, involving a large number of research units working cooperatively.

### 5.2.3 State of the Art

On-road vehicle detection systems have high computational requirements as they need to process the acquired images in real-time or close to real-time to save time for driver reaction. Searching the whole image to locate potential vehicle locations is impractical for real-time applications. The majority of methods reported in the literature follows two basic steps [5], as shown in Fig. 5.9:

- Hypothesis Generation (HG), where the locations of possible vehicles in an image are hypothesized.
- Hypothesis Verification (HV), where tests are performed to verify the presence of vehicles in the image.



**Fig. 5.9** Two steps of vehicle detection. **a** HG. **b** HV

### (1) HG Methods

The objective of the HG step is to find candidate vehicle locations in an image quickly for further exploration. The hypothesized locations from the HG step form the input to the HV step, over which tests are performed to verify the correctness of the hypotheses. The following are the signatures that hold importance in HG methods:

#### 5.2.3.1 Symmetry

As one of the main signatures of man-made objects, symmetry is often used for object detection and recognition in computer vision. Images of vehicles observed from rear or frontal views are in general symmetrical in the horizontal direction. An important issue which arises while computing symmetry from intensity is that symmetry estimations are noise sensitive. In [11], the authors formulated symmetry detection as an optimization problem that was solved using Neural Networks (NNs).

#### 5.2.3.2 Shadow

Using shadow information as a sign pattern for vehicle detection was discussed initially in [12]. By investigating image intensity, it was found that the area underneath a vehicle is distinctly darker than any other areas on asphalt road. The intensity of the shadow depends on the illumination of the image. To segment the shadow area, low and high thresholds are required. However, it is obvious that it is hard to find a low threshold for a shadow area.

In [13], a method was proposed to determine the threshold values. Specifically, a normal distribution was assumed for the intensity of a free driving space. The mean and variance of the distribution was estimated using Maximum Likelihood (ML) estimation. But the assumption about the distribution of the road pixels might not always be true.

### 5.2.3.3 Vertical/Horizontal Edges

Different views of a vehicle, especially rear/frontal views, contain many horizontal and vertical structures, such as rear-window, bumper, etc. Using constellations of vertical and horizontal edges has proved to be a strong cue for hypothesizing vehicle presence.

In [14], the authors utilized the edge information to detect distant cars. They proposed a coarse-to-fine search method looking for rectangular objects. The coarse search looked through the whole edge maps for prominent edges, such as long uninterrupted edges. Whenever such edges were found, the refined search process was started in that region. In [15], vertical and horizontal edges were extracted separately using the Sobel operator. Then, two edge-based constraint filters (i.e., rank filter and attached line edge filter) were applied on those edges to segment vehicles from the background.

#### (2) HV Methods

The input to the HV step is the set of hypothesized locations from the HG step. During HV, tests are performed to verify the correctness of a hypothesis. Approaches to HV can be mainly classified into two categories: (1) template-based and (2) appearance-based.

### 5.2.3.4 Template Based Methods

Template-based methods use predefined patterns of the vehicle class and perform correlation between the image and the template. In [16], the authors proposed a template based on the observation that the rear/frontal view of a vehicle has a “U” shape (i.e., one horizontal edge, two vertical edges, and two corners connecting the horizontal and vertical edges). During verification, they considered a vehicle to be present in the image if they could find the “U” shape.

### 5.2.3.5 Appearance Methods

HV using appearance models is treated as a two-class pattern classification problem: vehicle versus non-vehicles. Building a robust pattern classification system involves searching for an optimum decision boundary between the classes to be categorized. Given the huge inter-class variations of the vehicle class, we can imagine that this is not an easy task. One feasible approach is to learn the decision boundary based on training a classifier using the feature set extracted from the training images. Usually, the variations in non-vehicle class are also modeled to improve the performance. Each training image is represented by a set of local or global features. Then, the decision boundary between the vehicle and non-vehicle classes is learned either by training a classifier (e.g., NNs, SVMs) or by modeling the probability distribution of the features in each class (e.g., using the Bayes rule

assuming a Gaussian distribution). Using Gabor filters for vehicle feature extraction was investigated in [17]. Gabor filters provide a mechanism for obtaining orientation and scale tunable edge and line detectors. Vehicles contain strong edges and lines at different orientation and scales; thus, these features are very effective for vehicle detection. The hypothesized vehicle sub-images were subdivided into nine overlapping sub-windows. Gabor filters were then applied on each sub-window separately.

A “vocabulary” of information-rich vehicle parts was constructed automatically by applying the Forstner interest operator [18] onto a set of representative images, together with a clustering method [19]. Each image was represented in terms of parts from this vocabulary to form a feature vector, which was used to train a classifier to verify the hypotheses.

Based on the previous active learning framework, a real-time detector was introduced [20], in which the authors independently detected vehicle parts using strong classifiers trained with active learning. They matched part responses using a learned matching classification. The learning process for part configuration leveraged user input regarding full vehicle configuration. Part configurations were evaluated using the SVM classification.

### ***5.2.4 Typical Methods***

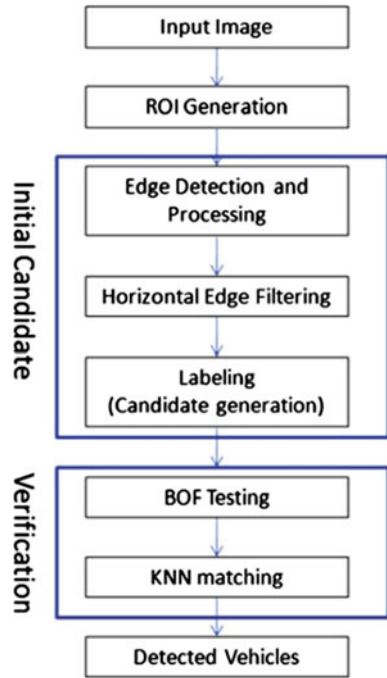
#### **(1) Single Camera Vehicle Detection Using Edges and Bag-of-Features**

The methodology presented in [5] involves two main parts, initial candidate generation using edge based method and verification using Bag-of-Features (BoF) algorithm. Initial candidate locations or Region of Interest (ROI) are those regions where presence of vehicle is most likely and hence they should be verified. Cascading of the two methods helps to achieve accurate vehicle detection at lower computational cost.

Edges are one of the main characteristics of an object, which carries most of the information about an object in an image. In [5], it was observed that horizontal edges are strong features for vehicle detection. Therefore, initial candidate were generated using Horizontal Edge Filtering (HEF) on the Canny edge map. These initial candidates are further verified using the BoF with K Nearest Neighbor (KNN) algorithm. A threshold is used on differences of histograms of training and test images for matching the vehicles. The combination of edges (initial candidate) and bag-of-features (final verification) improves the detection rate significantly. Figure 5.10 shows the flow chart of main methodology.

Initial candidate generation is very important for vehicle detection as the detection accuracy largely depends on it. In this chapter, for an initial candidate generation, the authors used HEF of canny edge map. The Canny edge detector is

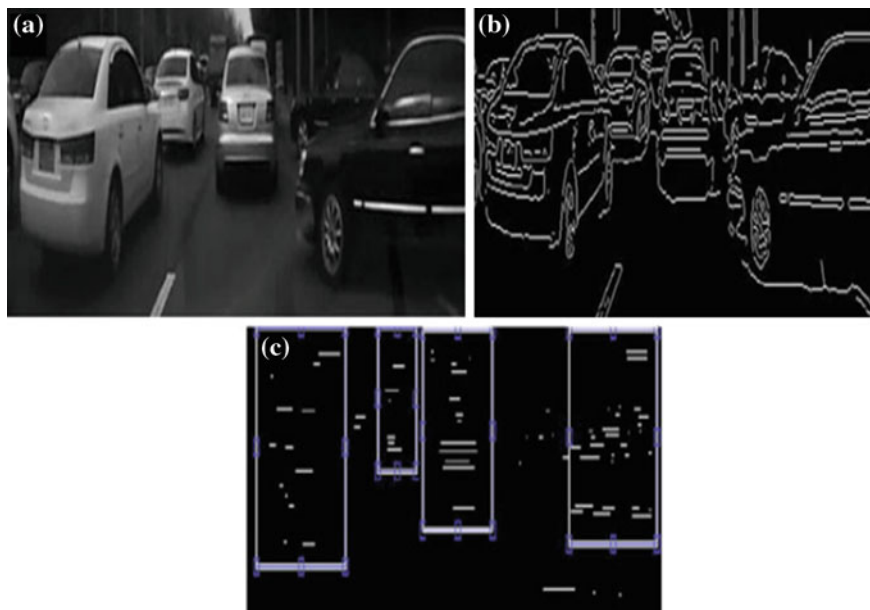
**Fig. 5.10** Procedure for vehicle detection



applied on the ROI of the input image to generate an edge map as shown in Fig. 5.11b. Through rear-view images of various vehicles it was observed that long connected horizontal edges usually belong to a vehicle. Based on this observation, HEF is applied on Canny edge response for detection of vehicles. In HEF, each row of the whole edge map is searched for at least ten consecutive 1's. Whenever a number of consecutive ones are found, they are preserved as a long connected horizontal edge. Following this criterion, the whole image is scanned and only long connected edges are set to 1, while small unconnected edges are set to 0. As a result, small unconnected edges are neglected and only long connected edges are used. The edge based method is very fast and permissive (almost zero miss rate). It generates a sparse search space for verification stage by filtering out most of the unworthy candidates.

BoF algorithm is used for the initial candidate verification. This reduces the number of false matches and improves detection accuracy. This cascaded technique for vehicle detection reports 98 % detection accuracy on the highways and 96 % on urban roads.





**Fig. 5.11** **a** Region of interest (ROI) generation. **b** Canny edge map of ROI image. **c** Results of horizontal edge filtering on an image

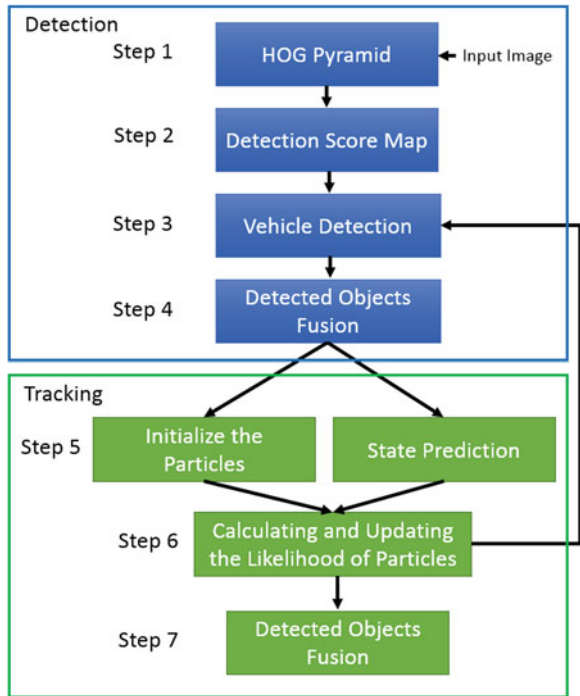
## (2) A General Active-Learning Framework for On-Road Vehicle Recognition and Tracking

In [21], the authors introduced a general active-learning framework for robust on-road vehicle recognition and tracking. This framework takes an active-learning approach for building vehicle-recognition and tracking systems. A passively trained recognition system is built using conventional supervised learning. Using the Query and Archiving Interface for Active Learning (QUAIL), the passively trained vehicle-recognition system is evaluated on an independent real-world data set, and informative samples are queried and archived to perform selective sampling.

For the task of identifying vehicles, a boosted cascade of simple Haar-like rectangular features has been used, since Rectangular features are sensitive to edges, bars, vertical and horizontal details, and symmetric structures. The algorithm also allows for rapid object detection that can be exploited in building a real-time system, partially due to fast and efficient feature extraction using the integral image. The resulting extracted values are effective weak classifiers, which are then classified by Adaboost.

They integrate a particle filter for vehicle tracking. The probability densities of possible predictions of the state of the system are represented by a randomly generated set, and multiple hypotheses are used to estimate the density of the tracked object.

**Fig. 5.12** Procedure for multivehicle detection and tracking [22]



**(3) On-Road Multi-Vehicle Tracking Using Deformable Object Model and Particle Filter with Improved Likelihood Estimation**

In [22], a multi-vehicle detection and tracking method was proposed using a vehicle-mounted monocular camera. In the proposed method, the vehicle features are learned as a deformable object model through the combination of a Latent Support Vector Machine (LSVM) [23] and Histograms of Oriented Gradients (HOGs) [24]. The detection algorithm combines both global and local features of the vehicle as a deformable object model. Detected vehicles are tracked through a particle filter, which estimates the particles’ likelihood by using a detection score map and template compatibility for both root and parts of the vehicle while considering the deformation cost caused by the movement of vehicle parts. Tracking likelihoods are iteratively used as a priori probability to generate vehicle hypothesis regions and update the detection threshold to reduce false negatives.

This method combines a deformable object model with particle filter to improve the detection accuracy according to the procedure shown in Fig. 5.12. At the detection stage, vehicles are detected using the deformable object model algorithm. It uses a scanning window approach, and a model for the vehicle consists of a global “root” filter and six part filters and a spatial model of part filters. The spatial model defines a set of allowed placements for a part relative to a detection window and a deformation cost for each placement. The score of a detection

window is the score of the root filter on the window plus the sum over scores of part filters minus the deformation cost.

To reduce the false negatives without increasing the false positives, tracking information from the previous frame has been used as a priori probability for updating the threshold in step 3. In step 6, the vehicle “hypothesis” regions are generated, and the threshold has been set to a lower value in the next frame to reduce false negatives.

At step 4, the new detected vehicles at time  $t$  are fused with tracking information from the previous frame. If a vehicle is detected for the first time,  $M$  particles are initialized according to a Gaussian distribution. If the detected vehicle is recognized as a “side” view, the initial value for the horizontal variance of the particle distribution is set to a higher value than those of the “front” and “rear” views. On the other hand, if the detected vehicle has been detected previously, a Kalman filter has been used to generate prediction according to the vehicle’s movement model at step 5. The likelihood of particles is updated by integrating the scores from the detection score map and intensity correlations for vehicle’s root and parts between two sequential frames while considering the deformation costs at step 6 and 7.

## 5.3 Pedestrian Detection

### 5.3.1 Challenges

The appearance of a pedestrian exhibits very high variability since they can change pose, wear different clothes, carry different objects, and have a considerable range of sizes, e.g. as shown in Fig. 5.13.

Pedestrians have to be identified in the context of a cluttered background, under a wide range of illumination and weather conditions, as can be seen in Fig. 5.14.

Other pedestrians or common urban elements can partially or heavily occluded pedestrians (Fig. 5.15).

Pedestrians are to be identified in highly dynamic scenes since both of the pedestrians and the camera are in motion. The required performance is quite demanding in terms of system reaction time and robustness.

### 5.3.2 Pedestrian Detection in a Typical Urban Scenario

The Caltech Pedestrian Dataset [1] is large, realistic and well annotated, allowing us to study the statistics of the size, position, and occlusion of pedestrians in urban scenes and also to accurately evaluate the state of the art in pedestrian detection.



Fig. 5.13 Variability in pose, clothes, etc.



Fig. 5.14 Illumination and weather conditions



Fig. 5.15 Occlusions [25, 26]

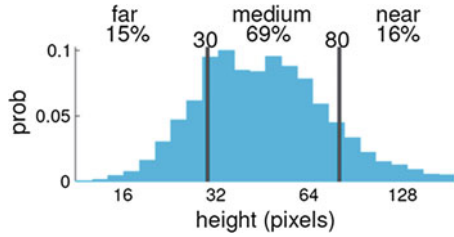


Fig. 5.16 Pixel distribution of pedestrians [1]

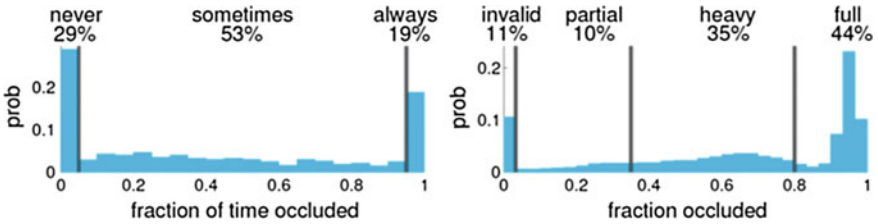


Fig. 5.17 Occlusion statistics [1]

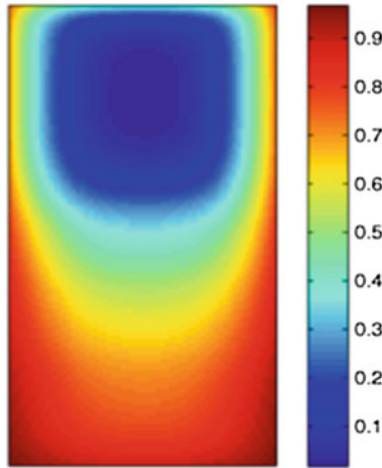
The Caltech Pedestrian Dataset has a CCD video resolution of  $640 \times 480$ ,  $27^0$  vertical field of view, and focal length fixed at 7.5 mm. But, the overall image quality is lower than that of still images of comparable resolution. The next section presents terminologies and statistical analysis of pedestrian appearance in a typical roadside scenario.

### (1) Scale Statistics

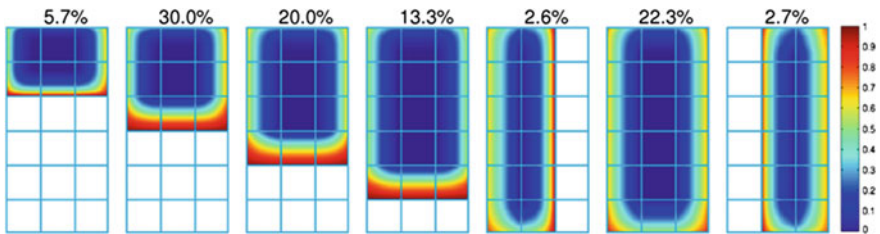
Distribution of pedestrian's pixel height: The near scale is defined as to include pedestrians over 80 pixels, the medium scale as 30–80 pixels and the far scale as under 30 pixels. Most observed pedestrians (69 %) are at the medium scale as shown Fig. 5.16.

### (2) Occlusion Statistics

- **Fraction of time occluded:** the fraction of frames in which the pedestrian was at least partially occluded.
  - Most pedestrians (70 %) are occluded in at least one frame, emphasizing on the importance of detecting occluded people, see Fig. 5.17a.
- **Fraction occluded:** defined as one minus the visible pedestrian area divided by total pedestrian area. It is further categorized in three parts as shown in Fig. 5.17b.
  - **Invalid** e.g. a diagonal occlusion.
  - **Partial occlusion** (1–35 % area occluded).
  - **Heavy occlusion** (35–80 % occluded).



**Fig. 5.18** Heat map probability of occlusion for each pixel (conditioned on the person being partially occluded) [1]



**Fig. 5.19** Types of occlusions and their probability of occurrence [1]

Figure 5.18 presents the probability of occlusion for each pixel (conditioned on the person being partially occluded). Observe the strong bias for the lower portion of the pedestrian to be occluded, particularly the feet. And there is a strong bias for the top portion, especially the head, to be visible.

To observe further structures in the types of occlusions that actually occur, in [1] the authors quantize occlusion into a fixed number of types. Together, there are 7 types of occlusions that account for nearly 97 % of all occlusions in the dataset. As can be seen from Fig. 5.19, pedestrians are almost always occluded from either below or the side; more complex occlusions are rare.

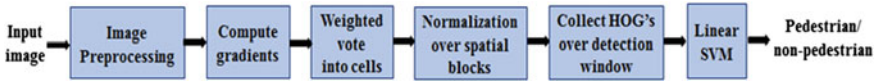


Fig. 5.20 HOG feature extraction block diagram

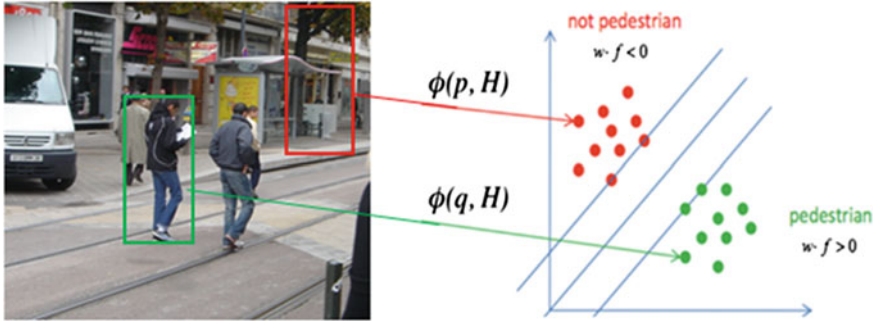


Fig. 5.21 Binary classification

### 5.3.3 State-of-the-Art Methods

Detecting pedestrians in images is a challenging task owing to various styles of clothing in appearance and huge possible postures. Significant research has been devoted to detecting, locating, and tracking people in images and videos. In recent years, the number of approaches to detect pedestrians in monocular images has grown steadily. Among all the aspects, the performance of a pedestrian detection system is mainly determined by two key factors: the learning algorithm and the feature representation. In the following we will introduce the state of the art methods of pedestrian detection.

#### (1) HOG, SVM (Milestone)

In HOG algorithm [24], each detection window is divided into cells of size  $8 \times 8$  pixels and each group of  $2 \times 2$  cells is integrated into a block in a sliding fashion, so blocks overlap each other. Each cell consists of a 9-bin Histogram of Oriented Gradients (HOG) and each block contains a concatenated vector of all its cells. Each block is thus represented by a 36-D feature vector which is normalized to an L2 unit length. Each  $64 \times 128$  detection window is represented by  $7 \times 15$  blocks, giving a total of 3,780 features per detection window. These features are then used to train a linear SVM classifier as shown in Fig. 5.20. For testing, the detector follows a sliding window paradigm which entails feature extraction, binary classification (Fig. 5.21), and dense multiscale scanning of detection windows followed by non-maximum suppression.

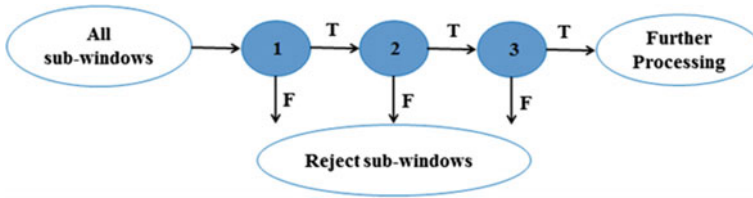


Fig. 5.22 Algorithm description

- **Strong point:**

- HOG is a very informative feature for human representation till now;
- Linear SVM achieves fast evaluation and training speed.

- **Weak point:**

- HOG is not robust under challenging conditions, e.g. occlusions, cluttered backgrounds;
- Classification accuracy of Linear SVM is not as good as kernelized SVM.

## (2) HOG, SVM, AdaBoost

In [27], the authors integrate the cascade-of-rejectors (Fig. 5.22) with the Histograms of Oriented Gradients (HOG) features to build a fast and accurate human detection system. The features used in their system are HOGs of variable-size blocks which capture salient features of pedestrians automatically. Using AdaBoost for feature selection, the appropriate set of blocks is identified, from a large set of possible blocks. In their system, they use the integrated image representation and a rejection cascade which significantly speeds up the computation. For a  $320 \times 280$  image depending on the density settings for scale pyramid and spatial stride, the system can process 5–30 frames per second while maintaining an accuracy level similar to existing methods.

During feature extraction, for a  $64 \times 128$  detection window, they consider all blocks whose size ranges from  $12 \times 12$  to  $64 \times 128$ . The ratio between block width and block height can be any of the following ratios (1:1) (1:2) and (2:1). Moreover, they choose a small step-size, which can be any of {4, 6, 8} pixels depending on the block size, to obtain a dense grid of overlapping blocks. In total, 5,031 blocks are defined in a  $64 \times 128$  detection window, each of which contains a 36-D histogram vector of concatenating the nine orientation bins in  $2 \times 2$  sub-regions.

In order to achieve good classification result, they construct rejection cascade. Each feature in their scheme corresponds to the 36-D vector used to describe a block. The weak classifiers they use are the separating hyperplane computed using a linear SVM. For each level of the cascade they construct a strong classifier consisting of several weak classifiers (linear SVMs in our case). In each level of the cascade they keep adding weak classifiers until the predefined quality requirements are met. In their case they require the minimum detection rate to be 0.9975 and the maximum false positive to be 0.7 in each stage.



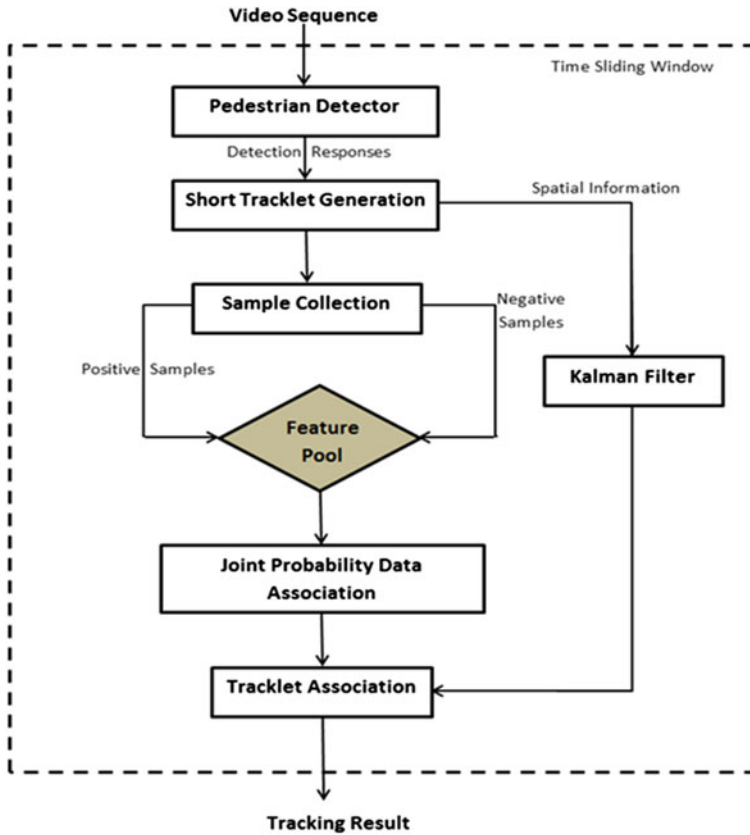


Fig. 5.23 Overview of our framework

- **Strong point:**

- Integration of the cascade-of-rejecters approach with HOG features;
- Achievement of a fast and accurate human detection system.

- **Weak point:**

- It is real time only for single scale detection;
- Not robust in challenging conditions, e.g. occlusions, cluttered backgrounds

### (3) HOG, LAT-SVM, Kalman Filtering

In [28], a new approach for multi-person tracking-by-detection using deformable part models in Kalman filtering framework was proposed. The authors used a pedestrian detection system based on mixtures of multi-scale deformable part models [23]. The core ideas of deformable part-based models boil down to three factors:

- (1) A deformable part representation for pedestrian.
- (2) An efficient matching process.
- (3) Latent SVM, a discriminative training method.

The overview of this method is shown in Fig. 5.23. A star-structured part-based model is defined which is composed of a root filter,  $n$  (usually six) part filters, and associated deformation parameters. An efficient matching process based on dynamic programming and generalized distance transforms is proposed. To detect objects in an image we compute an overall score for each root location according to the best possible placement of the parts. Finally, a latent SVM training process is formulated to train a mixture of star models from bounding box ground truth.

The Kalman filter is used to keep track of each person and a unique label is assigned to each tracked individual. Based on this approach, people can enter and leave the scene at random. We test and demonstrate our results on the Caltech Pedestrian benchmark, which is the largest available dataset and consists of pedestrians varying widely in appearance, pose and scale. Complex situations such as people merging together are handled gracefully and individual persons can be tracked correctly after a group of people split. The tracking accuracy can be increased at the cost of computational complexity by using particle filter instead of Kalman filter.

- **Strong point:**

- One of the most successful approaches for general object detection and tracking;
- Accepted to be one of the most successful methods at higher resolutions.

- **Weak point:**

- Deformable part models are time consuming;
- HOG method is not robust to extract informative features in challenging conditions.

#### (4) **Integral Channel Features**

The general idea behind integral channel features [29] is that multiple registered image channels are computed using linear and nonlinear transformations of the input image, and then features such as local sums, histograms, and their various generalizations are efficiently computed using integral images. Although integral channel features have proven effective, little effort has been devoted to analyzing or optimizing the features themselves. In this work they present a unified view of the relevant work in this area and perform a detailed experimental evaluation. They demonstrate that when designed properly, integral channel features not only outperform other features including histogram of oriented gradient (HOG), they also

- (1) Naturally integrate heterogeneous sources of information.
- (2) Have few parameters and are insensitive to exact parameter settings.
- (3) Allow for more accurate spatial localization during detection.
- (4) Result in fast detectors when coupled with cascade classifiers.

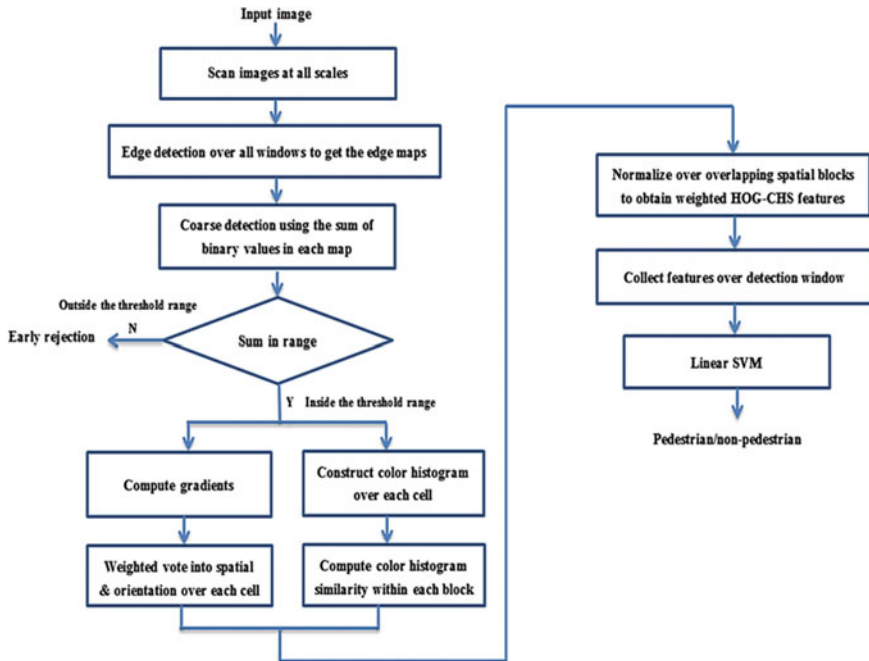


Fig. 5.24 The flow diagram of the proposed detection method

They evaluated combinations of three types of channels: gradient histograms, color (including grayscale, RGB, HSV and LUV), and gradient magnitude. These channels can be computed efficiently and capture diverse information. From the performance of different combinations of features, they found that LUV + Grad + Hist achieve best result. By default, eight channels were used, including gradient magnitude, grayscale, and six gradient histogram channels with no pre or post-smoothing. They generate a large pool of candidate features randomly rather than through careful design. They generate candidates by randomly choosing both the channel index and the rectangle (enforcing a minimum area of 25 pixels). Considering boosting offers a convenient, fast approach to learning given a large number of candidate features, they test with AdaBoost algorithm.

- **Strong point:**

- Integral channel features coupled with a standard boosting algorithm outperforms existing features for pedestrian detection.

- **Weak point:**

- It is not real time and still is not quite robust on the challenge datasets.

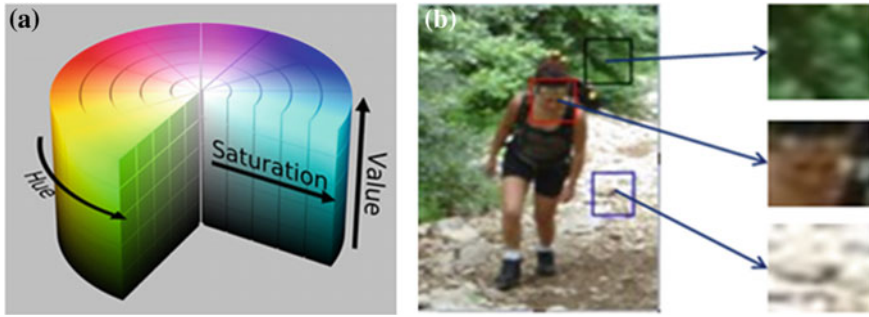


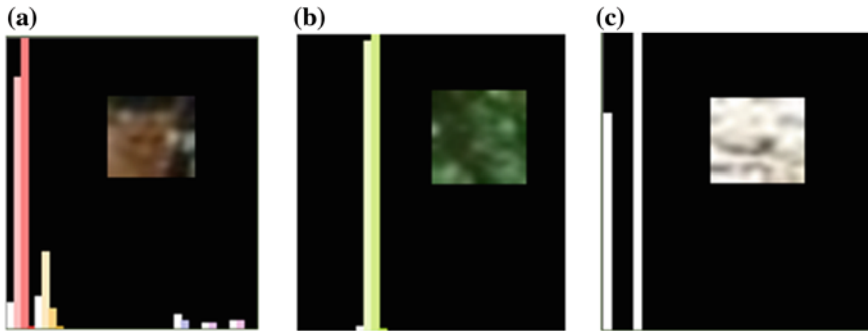
Fig. 5.25 a HSV cylinder. b Selected regions from an input image

### (5) Coarse-to-fine Detection, Hog Features, Color Histogram Similarity (CHS)

In [30], the key contributions are from the observation that edge information can be used as coarse human detection by rejecting a large part of the background windows and that color cues are informative for representing humans. The authors proposed a new coarse-to-fine human detection method in order to achieve efficient detection with high accuracy, as shown in Fig. 5.24. The color information is represented by using color histogram similarity within each HOG block, which is referred as CHS feature, then HOG and CHS are weighted and combined into a vector as a feature.

In the sliding window detection approach, all the sub-windows are scanned at all scales. Since most of the sub-windows are background, evaluating these windows with complex descriptors is time consuming. Thus, to find an efficient way to reject these negative windows while preserving the positive windows. Edges are quite useful cues for image segmentation tasks. Since windows with humans usually have more edges than background windows, it is possible to use edge maps for coarse human detection. Based on this analysis, the sum of binary values in each edge map is compute. The gradient magnitude of each detection window is computed and it is thresholded to generate a binary edge map. Then the sum of binary values is computed to see whether the sum value belongs to the threshold region found during the training procedure. If the sum of binary values of the edge map is not in the threshold region then it is rejected as a negative window.

Color by itself is of limited use, because colors vary across the entire spectrum both for people (respectively their clothing) and for the background, and because of the essentially unsolved color constancy problem. However, people do exhibit some structure, in that colors are locally similar—for example the skin color of a specific person is similar on their two arms and face. HSV color histogram is constructed in hue, saturation, and value space, which is the most common cylindrical-coordinate representation of points in an RGB color model. Based on the HSV cylinder (Fig. 5.25a), the color histogram by making use of hue and saturation information in a selected image region is constructed (Fig. 5.25b), the



**Fig. 5.26** HSV color histograms of the selected image regions

resulting histogram consists of nine bins of hue interval with four levels of saturation within each bin. In this process, brightness information is ignored since brightness may vary with the lighting conditions when the testing images are captured.

Figure 5.26 shows an example of color histograms of different regions with the size of  $16 \times 16$  pixels extracted from Fig. 5.25b. As expected, one can see that histogram of human size has different prominent color bins in comparison with the histogram of background. Thus, color histogram is informative for representing humans. Color histogram itself is not sufficient and robust when it is used alone for pedestrian detection. Thus finding an efficient way to combine the color histogram with HOG features is required. Color histogram can be used as a complement to encode the smooth areas where HOG may not work well. The color histogram in each cell ( $8 \times 8$  pixel) is constructed and the color histogram similarity in each HOG block ( $16 \times 16$ ) is computed using histogram intersection to obtain a 9-D CHS feature. So the HOG-CHS feature of a block is 45-D ( $36 + 9$ ) vector with L2 normalization. After collecting features over the detection window, linear SVM is used to perform the binary classification.

- **Strong point:**

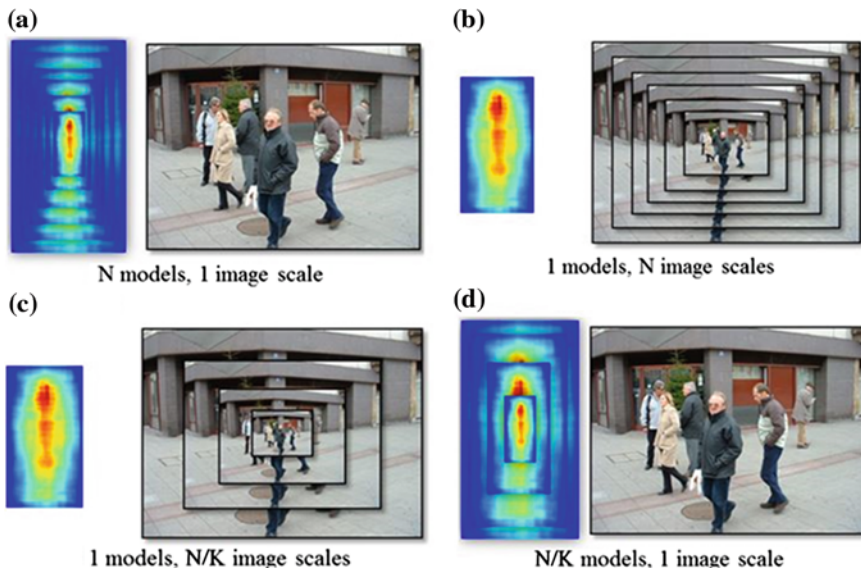
- It offered the method for computing CHS, which indeed complementary to gradient information and achieved promising accuracy.

- **Weak point:**

- Detection time is increased compared with HOG methods.

#### (6) Integral Channel Features, Multi-Scale Classifier, Depth Information

In [31], the authors proposed a novel method to improve the detection speed by efficiently handling different scales and transferring computation from test time to training time, which was based on Integral Channel Features [29]. Furthermore, they also proposed a new method for exploiting geometric context extracted from stereo images to further improve the detection speed and accuracy.



**Fig. 5.27** Different approaches to detecting pedestrians at multiple scales. **a** Naive method. **b** Traditional method. **c** FPDW method. **d** Reference method

They first introduced different approaches to detecting pedestrians at multiple scales. As shown in Fig. 5.27a, a naive approach would create a classifier for each position and scale, and make them compete against each other. The strongest responses would then be selected. Assuming that object appearance is invariant to translations in the image, to implement the naive approach we should train as many models as there are scales, which is a daunting task. The traditional approach for object detection at multiple scales, is to train a single model for one canonical scale, and then rescale the image  $N$  times, as seen in Fig. 5.27b. It poses two problems: 1. Training a canonical scale is delicate, as one needs to find the optimal size and learn a model that will trade-off between the rich high resolution scales and the blurry low resolution scales; 2. At run-time one needs to resize the input image  $N$  times, and re-compute the image features  $N$  times too. The Fastest Pedestrian Detector in the West (FPDW) approach [32], shown in Fig. 5.27c, is proposed for fast pedestrian detection. Instead of rescaling the input image  $N$  times ( $N \sim 50$ ), they propose to rescale it only  $N/K$  times. Each rescaled image is used to compute the image features, and these image features are then in turn used to approximate the feature response in the remaining  $N-N/K$  scales ( $K \sim 10$ ). The core idea of this chapter is to move the resizing of the image from test time to training time. They use the insight of the FPDW [32] detector and reverse it. They proceed to train  $N/K$  classifier, and use the described approximation used in [32] to transform  $N/K$  classifiers into  $N$  classifiers, as shown in Fig. 5.27d.

Depth information is known to be a strong cue for detections. Using scene geometry as prior for object detection can improve both the quality (by re-weighting the detection scores) and speed. For the first time, they show that a recently introduced fast depth information method [33] can be used to accelerate objects detection in practice. It assumes that the ground is locally flat and that all objects can be described as flat “sticks” rising vertically above the ground. The key feature of this approach is that the stixel world model can be estimated directly from the stereo images quickly, without having to compute the full depth map. It would provide a faster use of stereo images to reduce the set of candidate detection windows, which will also reduce the false positives.

- **Strong point:**

- It provides high quality pedestrian detection at 100 fps by using stixel world model and the insight of FPPW.

- **Weak point:**

- They should apply their approach to the multi-class/multi-view detection of other datasets.

### 5.3.4 Related Research Trends

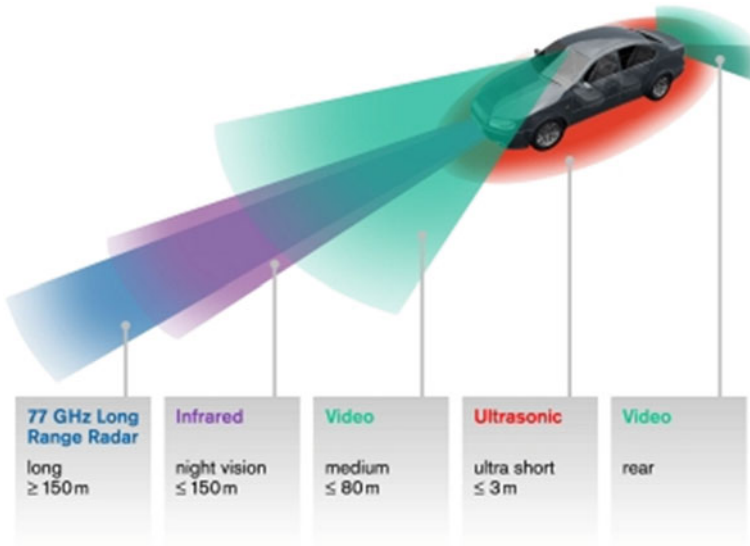
Considerable progress has been made in pedestrian detection over the past decade, which has advanced the frontiers of this problem in many aspects, e.g. features, classifiers, testing speed, and occlusion handling, while there is still a lot of work to do in order to achieve real-time high accuracy pedestrian detection in complex real world.

- Develop robust detection methods for small sized pedestrian in the 30–80 pixel range.
- Develop effective features for pedestrian in occlusion conditions.
- The combination of multi features to form effective descriptors.
- Stereo-based pedestrian detection.

## 5.4 Night-Time Pedestrian Detection

### 5.4.1 Why Infrared Imaging?

Driving at night, a stressful experience for many people, is also more dangerous than daytime motoring. According to the USA federal Department of Transportation, slightly more than 20 % of all fatal accidents in the United States in 2004



**Fig. 5.28** Different sensors and their ranges

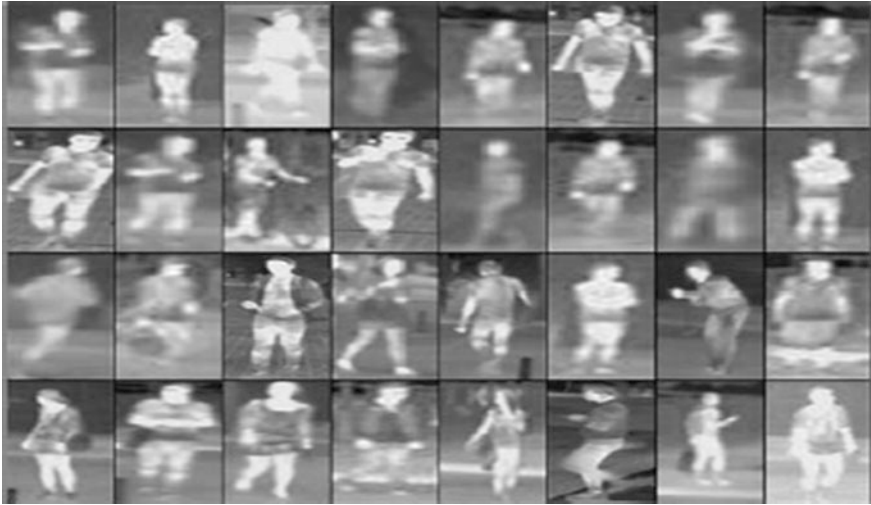
occurred between midnight and 6 a.m., a period that accounts for only about 2.4 % of daily traffic volume [34].

Aside from engineering factors such as the construction and maintenance of vehicles and roads, a large percentage of road injuries are attributed to human perceptual error. Of course drinking and fatigue are big factors. But inadequate illumination also ranks high: headlights provide about 50 m of visibility on a dark road, but it takes nearly 110 m to come to a full stop from 100 km/h. At that speed, you may not respond fast enough to an unexpected event, simply because the bright spot provided by your headlights doesn't give you enough time.

Figure 5.28 shows commonly used sensors in a typical vision system. Each of the sensors has its own advantages and disadvantages for example radar sensor provides object detection in long ranges in almost all weather conditions but it is very expensive typically around \$10,000 [35] and it also has low resolution when it comes to object identification.

Infrared imaging can be a remedy for relieving the intrinsic problems of visible imaging techniques and it can help in building a pedestrian detector that can work under various lighting conditions and weather conditions. Infrared imaging systems detect infrared (IR) light waves not visible to the human eyes. They work by either detecting near infrared (NIR) light waves or far infrared (FIR) light waves. Both types use special cameras that are able to collect small amounts of infrared light and process it so that it can be seen with human eyes. A signal processor translates the IR data to an image suitable for display on a monitor.





**Fig. 5.29** Thermal views of the pedestrians [36]

### (1) Near Infrared (NIR)

Night vision enhancement systems that use NIR technology (also called active vision systems) require “IR floodlamps” to project invisible NIR light onto the surrounding area. The camera captures the reflected NIR light from the surrounding area and intensifies it to create an enhanced (brightened) image of the normally dark area.

**Pros:** higher resolution image, superior picture of inanimate objects, works better in warmer conditions; smaller sensor can be mounted to rearview mirror, cost around \$300 [35].

**Cons:** does not work as well in fog or rain, lower contrast for animals, shorter range of 150–200 m or 500–650 ft.

### (2) Far Infrared (FIR)

Far infrared (FIR) light waves are further away from the visible portion of the spectrum and are created by the heat emitted from objects. This type of technology may also be referred to as thermal imaging (also called passive vision systems). It is ideal for detecting living things like people and animals.

**Pros:** Only shows relevant object (hot objects) on the screen, hence causes less distraction for driver, has greater range of about 300 m or 1,000 ft, shows higher contrast for living objects, and does not require “IR flood lamps”

**Cons:** grainy, lower resolution image, not great for warmer weather conditions, larger sensor, cost around \$1,500 [35].



Fig. 5.30 Successful detections via IR image [37]

### 5.4.2 FIR Pedestrian Detection

FIR sensor records electromagnetic radiations emitted by objects in a scene as a thermal image whose pixel intensity values represent the temperature of the respective bodies as in the Fig. 5.29.

In a thermal image consisting of humans in the scene, human silhouettes can be generally extracted from the background regardless of lighting conditions, even when the color of human clothing or skin, and the background is same. This is because the temperature of the human body and the background is different in most situations and this makes FIR sensors particularly suitable for night-time pedestrian detection. In Fig. 5.30, the pedestrians were not detected in visible image but they were successfully detected in the IR image.

### 5.4.3 State-of-the-Art

#### (1) HoG, SVM, Kalman Filter Tracking

In [38], a method for pedestrian detection and tracking for a vehicle mounted monocular thermal camera is presented. To deal with the non-rigid nature of human appearance on the road, a two-step detection/tracking method is proposed.

Their method first detects hotspots, then estimates the size of pedestrians, then clips corresponding image regions as pedestrian candidates. Once that is done, classification of candidate regions as pedestrians or non-pedestrians is done using SVM.

The tracking of the recognized pedestrians is done using Kalman filter prediction and mean shift tracking of heads or bodies of pedestrians. They make use



Fig. 5.31 Recognition results [38]

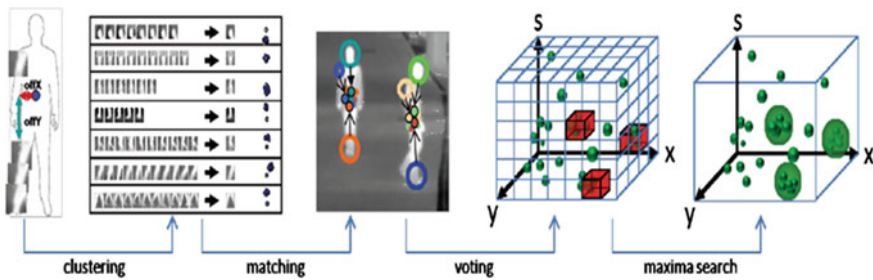


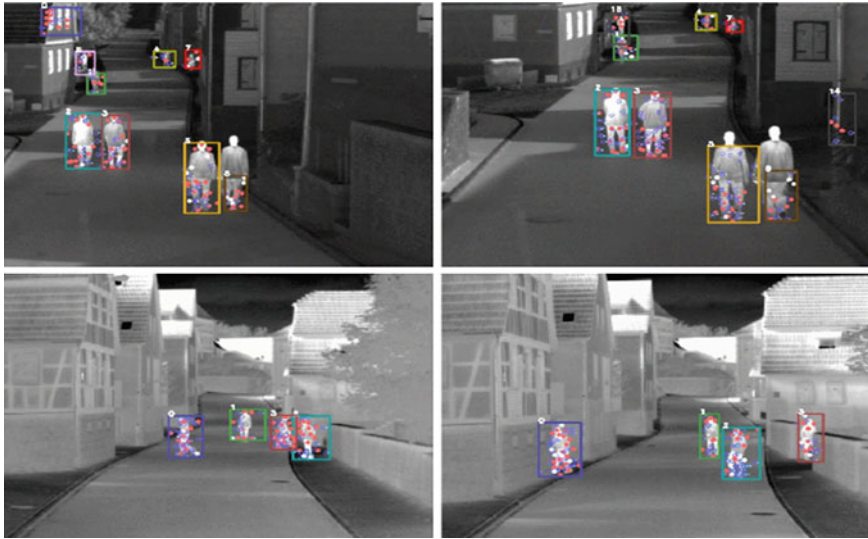
Fig. 5.32 Methodology [39]

of a combination of appearance-based detection and tracking methods to benefit from the strengths of different techniques and to overcome their respective limitations. The detection results are shown in Fig. 5.31.

(2) **Implicit Shape Model (ISM) Based Objects Detector**

In [39] another approach for detecting and tracking multiple pedestrians in real world environments from a moving, monocular infrared camera is presented. The authors focus on pedestrian tracking in infrared images and specifically address the problem of tracking under strong ego motion of the camera bearing vehicle.

In the training stage, a codebook is created by clustering the features (SURF) followed by building Implicit Shape Model (ISM), which describes the spatial configuration of features relative to the object center. In the detection stage, SURF features are first located in each image. Then, matching between the features and the codebook is conducted to vote the object center. Finally, the voted maxima's



**Fig. 5.33** Recognition results [39]

are refined by the mean shift algorithm [40] to accurately identify the object center location. Figure 5.32 shows the whole detection procedure.

They showed that, this feature matching based approach is specifically suited to detect and track people in infrared image sequences. Figure 5.33 shows some of the detection results. For the case of strong camera motion, they replace the explicit motion model by a feature shift vector based motion compensation strategy, which easily fits into the existing detection and tracking paradigm on the level of image features. They also show how their approach is able to track persons over short term occlusions and provide a tracking situation where the Kalman filter dynamics model comes to its limit.

#### **5.4.4 Related Research Trends**

The research done till now cover only a subset of the possible challenges in pedestrian detection. For example, night-time pedestrian detection has only barely been addressed. However drivers need more support at night-time because humans do tend to get visually impaired under inadequate light conditions.

Visible cameras are the most widely used sensors, due to the high potential of visual features, high spatial resolution and richness of texture and color cues. However, this analysis is far from simple: cluttering and illumination, among many other factors, affect the performance of visible camera, while FIR is influenced by other hot objects. A fusion of VS/FIR sensors and active sensors, which

is used to obtain complementary information, is being investigated in the context of on-board pedestrian detection. The strengths and weaknesses of different kinds of sensors can be complemented in order to improve the performance. Active sensors are based on technologies that emit signals and observe their reflection from the objects in the environment, for example, radars emitting radio waves or laser scanners emitting infrared light. In general, these sensors are convenient for detecting objects and providing superior range estimates out to larger distances when compared to relative to passive sensors.

With further introspection, it can be concluded that the gap between the current and the desired performance is large and unlikely to be reached without major leaps in our understanding.

**Small scales:** Better performance is needed in the 30–80 pixel range and below, while most research has been focused on pedestrians over 100 pixels.

**Occlusion:** Performance degrades rapidly even under mild occlusion, including for part based detectors.

**Temporal integration:** Although full systems often utilize tracking, a comparative study of approaches for integrating detector outputs over time has not been carried out. Note that full tracking may be unnecessary and methods that integrate detector outputs over a few frames may suffice.

**Context:** The ground plane assumption can reduce errors to some extent; however at low resolutions, more sophisticated approaches for utilizing context are needed.

**Novel features:** The best detectors use multiple feature types in combination with gradient histograms. Additional gains from continued research on improving feature extraction are expected.

## References

1. P. Dollar, C. Wojek, B. Schiele, P. Perona, Pedestrian Detection: An Evaluation of the State of the Art. *IEEE Trans. Pattern Anal. Mach. Intell.* **34**(4), 743, 761 (2012)
2. H. Shin, T. Yu, Y. Ismail, B. Saeed, Rendering high dynamic range images by using integrated global and local processing. *Opt. Eng.* **50**(11), 117002 (2011)
3. R.S. Choras, Feature extraction for CBIR and biometrics applications, in *7th WSEAS International Conference on Applied Computer Science*, vol. 7 (ACS'07) (2007)
4. T. Ojala, M. Pietikäinen, D. Harwood, Performance evaluation of texture measures with classification based on Kullback discrimination of distributions, in *Proceedings of the 12th IAPR International Conference on Pattern Recognition (ICPR)* (1994)
5. S.J.H. Pirzada, E.U. Haq, H. Shin, in *Single Camera Vehicle Detection Using Edges and Bag-of-Features*. Computer Science and Convergence Springer (Netherlands, 2012)
6. D.G. Lowe, Object recognition from local scale-invariant features, in *Proceedings of the International Conference on Computer Vision 2* (1999)
7. C. Cortes, V.N. Vapnik, Support-vector networks. *Mach. Learn.* **20**(3), 273–297 (1995)
8. M.-F. Balcan, A. Beygelzimer, J. Langford, Agnostic active learning, in *Proceedings of the 23rd international conference on Machine learning (ICML '06)*, New York (2006)
9. L. Breiman, Random forests. *Mach. Learn.* **45**(1), 5–32 (2001)

10. E.M Van Gool, L. Williams, J.C.K.I. Winn, A. Zisserman, The PASCAL Visual Object Classes Challenge 2012 (VOC 2012). Available at <http://www.pascal-network.org/challenges/VOC/voc2012/workshop/index.html>
11. W. Von Seelen, C. Curio, J. Gayko, U. Handmann, T. Kalinke, Scene analysis and organization of behavior in driver assistance systems, in *Proceedings International Conference on Image Processing*, vol. 3 (2000), pp. 524, 527
12. J. Crisman, C. Thorpe, Color vision for road following, in *Proceedings of SPIE Conference on Mobile Robots*, pp. 246–249 (1988)
13. C. Tzomakas, W. Seelen, Vehicle detection in traffic scenes using shadows, Technical report 98-06, Institut für Neuroinformatik, Ruhr-Universität, Bochum, (1998)
14. M. Betke, E. Haritagu, L. Davis, Real-time multiple vehicle detection and tracking from a moving vehicle. *Mach. Vis. Appl.* **12**(2), 69–83 (2000)
15. N. Srinivasa, “A Vision-Based Vehicle Detection and Tracking Method for Forward Collision Warning,” in *Proc. IEEE Intelligent Vehicle Symp.*, pp. 626–631, 2002
16. U. Handmann, T. Kalinke, C. Tzomakas, M. Werner, W. Seelen, An image processing system for driver assistance. *Image Vis. Comput.* **18**(5), 367–376 (2000)
17. Z. Sun, G. Bebis, R. Miller, On-road vehicle detection using gabor filters and support vector machines, in *Proceedings of IEEE International Conference on Digital Signal Processing* (2002)
18. W. Förstner, E. Gülch, A fast operator for detection and precise location of distinct points, corners and centers of circular features, in *Proceedings of the ISPRS Intercommission Workshop on Fast Processing of Photogrammetric Data* (1987)
19. S. Agarwal, D. Roth, Learning a sparse representation for object detection, in *Proceedings of European Conference on Computer Vision* (2002)
20. S. Sivaraman, M.M. Trivedi, Real-time vehicle detection using parts at intersections, in *Proceedings of 15th International IEEE Conference on Intelligent Transportation Systems Anchorage* (2012)
21. S. Sivaraman, M. Trivedi, A general active-learning framework for on-road vehicle recognition and tracking. *IEEE Trans. Intell. Transp. Syst.* **11**(2), 267–276 (2010)
22. H. Niknejad, A. Takeuchi, S. Mita, D. McAllester, On-road multivehicle tracking using deformable object model and particle filter with improved likelihood estimation. *IEEE Trans. Intell. Transp. Syst.* **13**(2), 748, 758 (2012)
23. P. Felzenszwalb, D. McAllester, Object detection with discriminatively trained part-based models. *IEEE Trans. Pattern Anal. Mach. Intell.* **32**(9), 1267–1645 (2010)
24. N. Dalal, B. Triggs, Histograms of oriented gradients for human detection, in *Computer Vision and Pattern Recognition* (2005)
25. M. Andriluka, S. Roth, B. Schiele, TUD-crossing sequence dataset (2008). Available at <http://www.d2.mpi-inf.mpg.de/sites/default/files/people/andriluka/tud-crossing-sequence.tgz>
26. N. Dalal, INRIA person dataset (2005). Available at <http://pascal.inrialpes.fr/data/human/>
27. Z. Qiang, Y. Mei-Chen, Fast human detection using a cascade of histograms of oriented gradients, in *IEEE International Conference on Computer Vision and Pattern Recognition* (2006)
28. S. Mittal, T. Prasad, H. Shin, Pedestrian detection and tracking using deformable part models and Kalman filtering, in *International SoC Design Conference* (2012)
29. P. Dollár, Z. Tu, P. Perona, S. Belongie, Integral channel features, in *British Machine Vision Conference* (2009)
30. Y. Teng, F. Xue, H. Shin, An efficient pedestrian detection method by using coarse-to-fine detection and color histogram similarity, in *Proceedings of International Conference on Hybrid Information Technology* (2012)
31. R. Benenson, M. Mathias, Pedestrian detection at 100 frames per second, in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition* (2012)
32. D. Piotr, S.B., The fastest pedestrian detector in the west, in *Proceedings of British Machine Vision Conference* (2009)

33. R. Benenson, R. Timofte, L. Van Gool, Stixels estimation without depth map computation, in *Proceedings of IEEE Conference on Computer Vision Workshops* (2011)
34. IRTAD Road Safety Annual Report 2013, International traffic safety data and analysis group (2013). Available at <http://internationaltransportforum.org/irtadpublic/pdf/13IrtadReport.pdf>
35. W.D. Jones, Safer driving in the dead of night (2006). Available at <http://spectrum.ieee.org/green-tech/advanced-cars/safer-driving-in-the-dead-of-night>
36. I. Riaz, J. Piao, H. Shin, Human detection by using CENTRIST features for thermal images, in *International Conference Computer Graphics, Visualization, Computer Vision and Image Processing* (2013)
37. BMW Night-Vision System. Available at [https://www.bmw.co.uk/bmw/marketEV/bmw\\_next/en\\_DE/footer/q-and-a/technology-guide/bmw-night-vision.html](https://www.bmw.co.uk/bmw/marketEV/bmw_next/en_DE/footer/q-and-a/technology-guide/bmw-night-vision.html)
38. F. Xu, X. Liu, K. Fujimura, Pedestrian detection and tracking with night vision, in *IEEE Transactions on Intelligent Transportation Systems* (2005)
39. K. Jungling, M. Arens, Pedestrian tracking in infrared from moving vehicles, in *IEEE Intelligent Vehicles Symposium (IV)* (2010)
40. Y. Cheng, Mean shift, mode seeking, and clustering. *IEEE Trans. Pattern Anal. Mach. Intell. PAMI* **17**(8), 790–799 (1995)

# Chapter 6

## Monitoring Driver's State and Predicting Unsafe Driving Behavior

Hang-Bong Kang

**Abstract** In recent years, driver drowsiness and distraction have been important factors in a large number of accidents because they reduce driver perception level and decision making capability, which negatively affect the ability to control the vehicle. One way to reduce these kinds of accidents would be through monitoring driver and driving behavior and alerting the driver when they are drowsy or in a distracted state. In addition, if it were possible to predict unsafe driving behavior in advance, this would also contribute to safe driving. In this chapter, we will discuss various monitoring methods for driver and driving behavior as well as for predicting unsafe driving behaviors. In respect to measurement methods of driver drowsiness, we discussed visual and non-visual features of driver behavior, as well as driving performance behaviors related to vehicle-based features. Eye related measurements such as PERCLOS, yawning detection and some limitations in measuring visual features are discussed in detail. As for non-visual features, we explore various physiological signals and possible drowsiness detection methods that use these signals. As for vehicle-based features, we describe steering wheel movement and the standard deviation of lateral position. To detect driver distraction, we describe head pose and gaze direction methods. To predict unsafe driving behavior, we explain predicting methods based on facial expressions and car dynamics. Finally, we discuss several issues to be tackled for active driver safety systems. They are (1) hybrid measures for drowsiness detection, (2) driving context awareness for safe driving, (3) the necessity for public data sets of simulated and real driving conditions.

---

H.-B. Kang (✉)

Department of Digital Media, Catholic University of Korea, Seoul, South Korea  
e-mail: hbkang@catholic.ac.kr



## 6.1 Introduction

Recently, the total number of serious car crashes is still increasing regardless of improvements in road and vehicle design for driver safety. The U.S. National Highway Traffic Safety Administration (NHTSA) data indicate that more than 40,000 Americans suffer serious injuries from 56,000 sleep related road crashes annually [1]. According to a study by the Sleep Research Center (UK), driver drowsiness at the wheel causes up to 20 % of accidents on monotonous roads [2]. Several studies have produced various estimates of the level of sleep deprivation as it relates to road accidents. In addition, driver distraction or inattention is another critical problem for safe driving [3]. In summary, driver drowsiness and distraction are major causal factors behind road accidents.

To reduce the number of road accidents, it is necessary to monitor driver and driving behavior and alert the driver when he or she is drowsy or in distraction state. In addition, if it were possible to predict unsafe driving behaviors in advance, this would contribute to safe driving. According to one report [4], the amount of car crashes would be reduced by 10–20 % by monitoring and predicting driver and driving behaviors. A reliable and robust driver drowsiness and distraction detection system would send an alert to the driver and thus reduce the number of hazardous situations on the road. If it were possible to predict unsafe driving behavior in advance, this would also be helpful in preventing road accidents. Thus, it is desirable to design a framework consisting of two phases, that is, both monitoring and predicting driver and driving behavior. Figure 6.1 shows such a framework.

For a driver monitoring system, two issues such as driver fatigue measurement and distraction detection should be solved. Usually, driver fatigue or drowsiness may be related with symptoms including eye movement, facial expression, heart and breathing rate, and brain activity [5–9]. To detect driver drowsiness, visual features such as eye movement and facial expression are very important. The frequency of eye blinking and degree of eyelid opening are a good index of the tiredness level [10]. Yawning measurement is also good indicator of a driver's drowsiness [11]. As non-visual features, heart rate variability (HRV), galvanic skin response (GSR) and conductivity, steering-wheel grip pressure, and body temperature are possible candidates for estimating the driver's fatigue level indirectly [12]. Electroencephalogram (EEG) and Electro-oculogram (EoG) give additional psychophysiological information about drowsiness or emotional reactions [13]. Driving behavior information such as steering wheel movement, lane keeping, acceleration pedal movement and braking, etc., should also be considered to detect driver drowsiness.

A good first step in detecting driver distraction or inattention is to monitor the driver head pose and gaze direction. A forward warning system [14] uses driver behavioral information to determine driver distraction and to determine whether the driver is looking straight ahead. Murphy-Chutorian et al. [15] use head pose information extracted from a localized gradient histogram and support vector regressors (SVRs) to recognize driver awareness. Kaminski et al. [16] propose a system to estimate continuous head orientation and gaze direction. Recently,

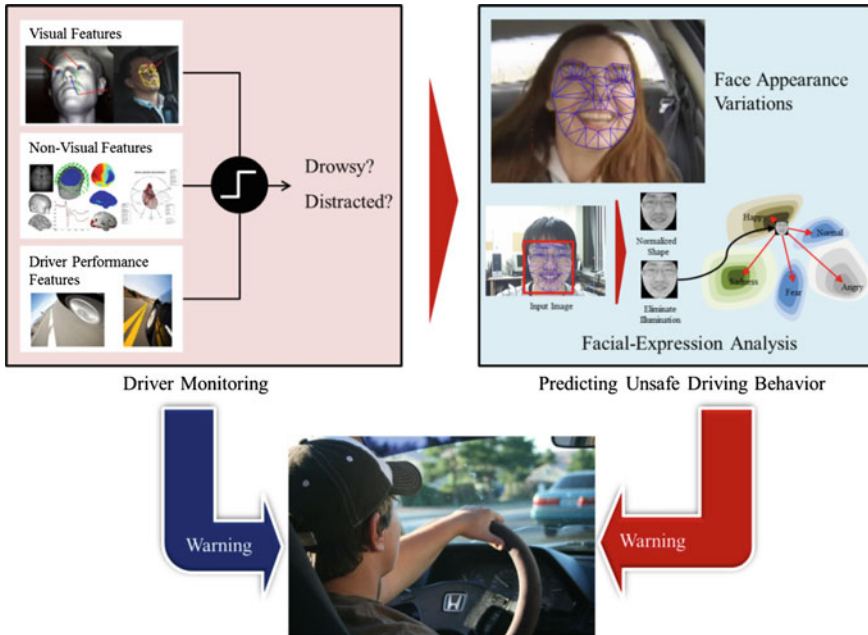


Fig. 6.1 General framework of a monitoring and predicting unsafe behavior system

excessive uses of in-vehicle information systems such as navigation systems and mobile phones induce visual and cognitive distraction in the driver. Visual distraction refers to the state of “eye-off-road”, and cognitive distraction is described as “mind-off-road” [17]. Liang [3] proposes a method to detect the interactions of visual and cognitive distractions.

By carefully monitoring driver and driving performance behavior, it is possible to predict minor and major accidents. In particular, the progress of pervasive computing technology with integrated sensors and networking has made it possible to build an ideal platform to predict accidents. Jabon et al. [18] identify a comprehensive set of driver’s key facial features at various pre-accident intervals and use them to predict minor and major accidents. This approach is very important for active driver-safety systems designed to prevent accidents. The Signal Processing 5 Laboratory of the EPFL in Switzerland [19] is doing much work in this area. They analyze facial expressions and muscle movements to detect distraction as well as emotions that could indicate that the driver is not up to the task at hand.

The aim of this chapter is to discuss monitoring the driver’s state as well as predicting unsafe driving behaviors. We explain several issues in developing a framework consisting of two phases: monitoring and predicting driver and driving behavior. The organization of this chapter is as follows: Section 6.2 discusses driver drowsiness detection. We discuss driver behavior features such as visual and non-visual features, and driving performance behaviors related to vehicle-based features. Eye related measurement like PERCLOS, yawning detection and some

current limitations in measuring visual features are discussed in detail. As for non-visual features, we explore various physiological signals and drowsiness detection methods that use these signals. As for vehicle-based features, we describe steering wheel movement and the standard deviation of lateral position. [Section 6.3](#) describes some issues related to driver distraction measurement, in particular, head pose and gaze direction methods. [Section 6.4](#) presents prediction methods for unsafe driving behaviors. We explain predicting methods based on facial expression and car dynamics. Finally, [Sect. 6.5](#) discusses some issues for active driver safety systems. They are (1) hybrid measures for drowsiness detection, (2) driving context awareness for safe driving, (3) the necessity for public data sets of simulated and real driving conditions.

## 6.2 Driver Drowsiness Measurement

For safe driving, it is necessary to construct a reliable driver monitoring system which could alert the driver when he or she is drowsy or a state of inattention. [Figure 6.2](#) shows a monitoring system for driver and driving behavior. In this section, we will discuss drowsiness measurement methods.

The word “drowsy” simply refers to an inclination to fall asleep. A drowsy driver who falls asleep at the wheel can be characterized by diminished alertness compared to a normal state. Sometimes a driver experiences sleep for a few seconds and may not even realize it. This is called micro-sleep. The duration of micro-sleep can last between a few seconds and as long as 30 s or even more. This is sufficient time to drift out from one’s traffic lane and crash into a tree or another car. Therefore, the driver’s drowsiness state, in which a transition occurs from awake to asleep, should be monitored.

To detect the drowsiness level of the driver, we have to extract driver behavior information as well as driving behavior information as shown in [Table 6.1](#). Driver behavior information consists of both visual and non-visual features. Visual features include eye closure, eye blinking, yawning, head pose, facial expression etc. [[10](#), [11](#), [20](#)]. Non-visual features consist of heart rate, pulse rate and brain activity. These physiological signals (electrocardiogram (ECG), electromyogram (EMG), electrooculogram (EoG) and electroencephalogram (EEG)) are used to detect driver drowsiness [[21–26](#)]. Driving behavior information includes deviations from lane position, vehicle speed, steering movement, pressure on the acceleration pedal, etc. [[27](#), [28](#)].

### 6.2.1 Visual Features

Usually, facial movements such as eye blinking, frequent yawning and nodding or swinging head are key elements among visual features used for detecting drowsiness. Much research work is focused on eye behaviors in particular to

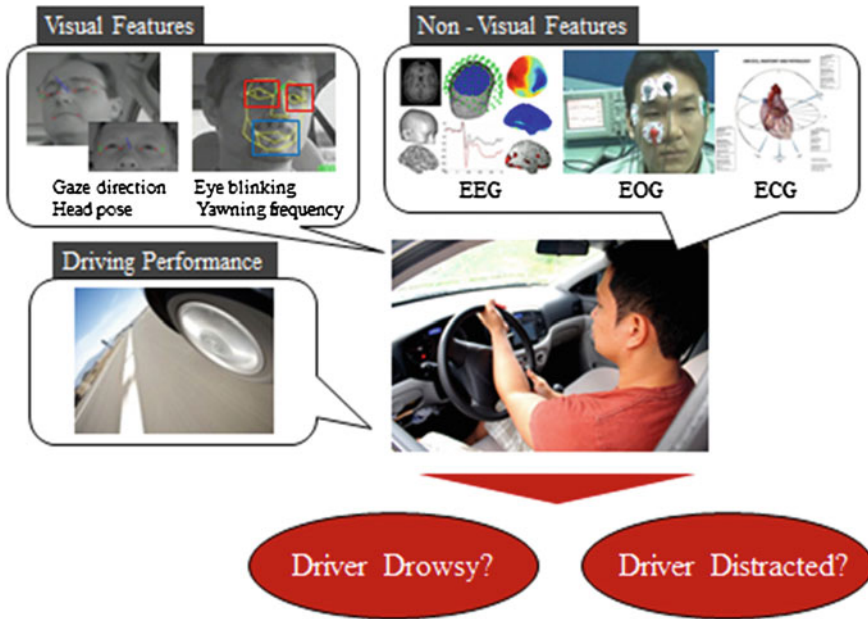


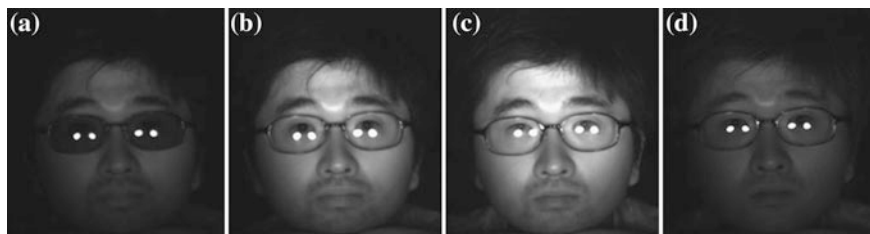
Fig. 6.2 Driver monitoring system

Table 6.1 Summary of various features for detecting driver drowsiness

Category	Features	Measurement	Function	Characteristics
Driver behavior	Visual feature	PERCLOS, yawn detection facial expression	Monitoring monitoring, prediction	Non-intrusive
	Non-visual feature	EEG(electroencephalogram) ECG(electrocardiogram) EOG(electro-oculogram) PPG(Photoplethysmography)	Monitoring	Intrusive
Driving behavior	Vehicle-based feature	Steering wheel movement Standard deviation of lateral position	Monitoring	Variations from vehicle type and individual

determine a driver's alertness [29, 30]. A reliable and valid determination of a driver's alertness level is known as PERCLOS (Percent Eye Closure) [31–33]. PERCLOS is the percentage of total time that the driver's eyelid is closed 80 % (or more) over the pupil and also reflects slow eyelid closure. When PERCLOS exceeds a predetermined threshold, the proposed system generates a drowsiness warning. However, one disadvantage of PERCLOS is that sometimes a driver who is trying to stay awake is able to fall asleep with his eyes open.

To calculate PERCLOS, we have to extract the eye region including the pupil area. However, there are some limitations in extracting those visual features. One of them is the problem of proper lighting. Drowsiness should be detected under



**Fig. 6.3** Images of a driver wearing black sunglasses at various wavelengths (a 700 nm, b 750 nm, c 850 nm, d 900 nm) from [37]

real conditions, i.e., throughout daytime and night, and regardless of whether the driver is wearing glasses or sunglasses. Usually, a simple CCD or web camera is used during the day, and an IR camera is used at night [34–37]. Moreover, for eye detection with a driver who is wearing sunglasses, it is necessary to find a proper wavelength of Near IR (NIR) illumination. Figure 6.3 shows an example of images of the driver wearing black sunglasses at various wavelengths [37]. One possible candidate wavelength is 850 nm. In a real automotive environment, reflected sunlight is also generated on the outer surface of the eyeglasses. To diminish the reflection effect, Jo et al. [37] used a NIR illuminator with a narrow bandpass filter which restricts the incoming wavelength of light to 850 nm. This is because the high-power LED illuminator is more powerful than the sunlight in the car. Figure 6.4 shows that the reflected sunlight is removed by an NIR illuminator with narrow bandpass filter [37].

Another method is faceLAB used in the commercial product such as the Seeing Machine [38]. A passive pair of video cameras is used and the video images are processed in real-time to determine the 3D position of each feature. The system is able to determine a precise 3D head pose and computes eye gaze direction. Its advantages include coping well with low light conditions and head movements while the driver is wearing sunglasses. Figure 6.5 shows faceLab 5 cameras [38].

Yawning is another sign of driver drowsiness. This is detected from measuring both the rate and the amount of changes in the driver's mouth contour [7, 11]. Figure 6.6 shows an example of a yawning detection system [7]. Head pose estimation and head motion detection of movements such as nodding are also important in monitoring driver alertness [39, 40]. In addition, facial wrinkles of the driver appearing on the brows, mouth and nasolabial fold are good physical signs that drowsiness is being resisted, and that therefore it is present [41].

## 6.2.2 Non-Visual Features

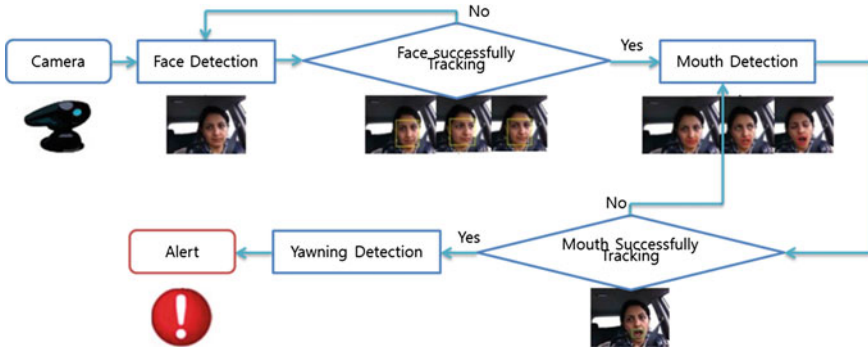
Non-visual features or physiological signals such as heart rate and brain activity are useful in predicting drowsiness, with fewer false positives compared to visual features because the determination of a drowsy state from visual features can be



**Fig. 6.4** Reflected sunlight is dealt with by an NIR illuminator with a narrow bandpass filter (a NIR illuminator only, b NIR illuminator with bandpass filter) in [37]



**Fig. 6.5** Commercial product faceLab 5 in [38]



**Fig. 6.6** Yawning detection system in [7]

possible only after the driver is well on the way to sleep. In other words, the prediction of drowsiness based on these physiological signals makes it possible to warn a drowsy driver in a timely manner. Electrocardiogram (ECG), electroencephalogram (EEG), electromyogram (EMG), electro-oculogram (EoG), and Photoplethysmography (PPG) may all be used as physiological signals.

From the ECG signal, heart rate (HR) can be extracted; the heart rate can be used to detect drowsiness because it varies significantly between alertness and drowsiness states [25, 42]. Heart rate variability (HRV) which measures the beat-to-beat changes in the heart rate is also used to detect drowsiness. As the driver goes from an alert to a drowsy state, the ratio of low frequency to high frequency beats in the ECG signal progressively decreases [23, 43].

The EEG signal is commonly used to detect drowsiness by detecting power changes in the alpha band (8–13 Hz) and theta band (4–8 Hz) [44–46]. Drowsiness is determined by a decrease in alpha frequency band and an increase in the theta frequency band. The combination of EEG and EMG signals is used to detect drowsiness and has obtained high success rate compared to the frequency signal alone [21].

The EoG signal is related to eye movement [24, 47]. The EoG signal is obtained from the electric field generated by the electric potential difference between the cornea and the retina. To measure eye movement, pairs of electrodes are placed above and below the eye or to the left and right of the eye. Rapid eye movements (REM) and slow eye movements (SEM) are detected easily using the EoG signal to differentiate alertness and drowsiness [48].

Photoplethysmography (PPG) is a non-invasive technique associated with changes in peripheral blood flow circulation [49]. PPG signals feature peaks and valleys representing maximum blood volume changes, and minimum blood volume changes corresponding to the beginning of and the end of blood ejection [50]. Fluctuation in these patterns is significant because it correlates to arousal events such as drowsiness; sympathetic motions provoke the PPG baseline to wander or drift [51]. Thus, changes in the PPG reflected wave pattern are important features in drowsiness classification.

**Fig. 6.7** ECG measurements on driver's seatback from [55]



One critical issue in handling physiological signals is to eliminate noise and artifacts inevitable in real environment driving conditions. Following effective filtering, various feature extraction techniques such as Fast Fourier Transform (FFT) and Discrete Wavelet Transform (DWT) are used. Then, the extracted features are classified using Support Vector Machine (SVM), Artificial Neural Networks (ANN), Linear Discriminant Analysis (LDA), etc. [52–54].

Even though the reliability and accuracy in detecting driver's drowsiness based on physiological signals is high when compared to visible features, an important limitation of physiological signal measurement is its intrusive nature. One possible way to solve this limitation is to use wireless technologies such as Zigbee and Bluetooth for measuring physiological signals in a non-intrusive way by placing the electrodes on the steering wheel or in the driver's seat [55, 56]. Figure 6.7 shows ECG measurement on the driver seatback [55]. Finally, the signals are handled by smart phones and driver drowsiness is determined [57]. However, this kind of non-intrusive system is less accurate compared to intrusive systems due to improper electrode contact.

### **6.2.3 Driving Behavior Features**

Driving behavior features or driving performance measures include steering wheel movement, lane keeping, acceleration pedal movement and braking, etc. [58–60]. These features correlate to vehicle type and variability among drivers in their driving habits, skills and experience. The two most commonly used driving behavior measures for detecting the level of driver drowsiness are the steering wheel movement and the standard deviation in lateral position.

Steering Wheel Movement (SWM) is measured using steering angle sensor mounted on the steering column. When the driver is drowsy, the number of micro-corrections to the steering wheel, which are necessary in normal driving, is



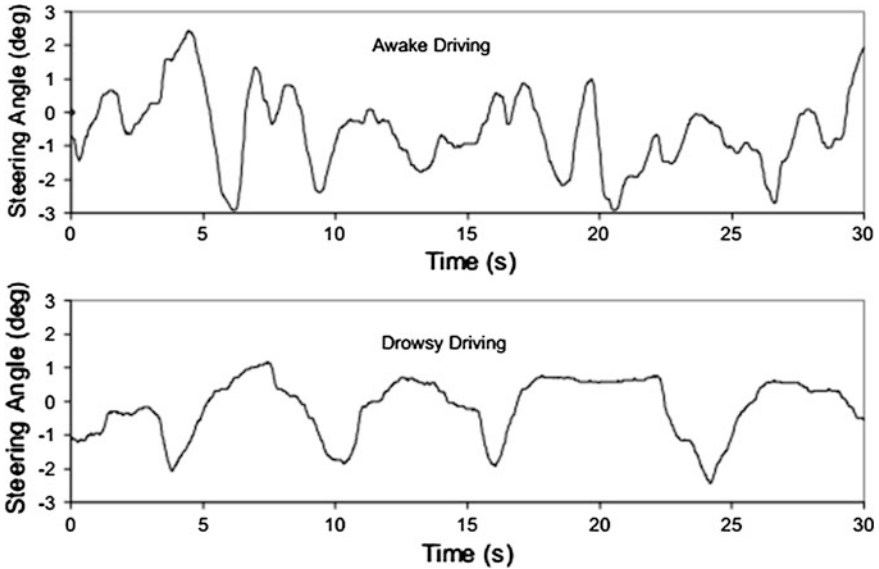


Fig. 6.8 Comparison of steering wheel angle for awake and drowsy drivers in [61]

Table 6.2 Karolinska sleepiness scale (KSS)

Scale	Descriptions
1	Extremely alert
2	Very alert
3	Alert
4	Fairly alert
5	Neither alert nor sleepy
6	Some signs of sleepiness
7	Sleepy, but no effort to keep alert
8	Sleepy, some effort to keep alert
9	Very sleepy, great effort to keep alert, fighting sleep

reduced [61]. Figure 6.8 shows two typical steering wheel angles, for an awake and a drowsy driver. The driver’s drowsiness state is determined from small SWM’s of between 0.5 and 5°. SWM’s are being adopted by car companies such as Nissan and Renault, but work in very limited situations due to their reliability only in particular environments [26].

Standard Deviation of Lateral Position (SDLP) is another sleepiness sensitive continuous performance measure. Ingre et al. [62] found that SDLP is correlated with the Karolinska Sleepiness Scale (KSS), a nine-point scale that has verbal anchors for each step shown in Table 6.2. However, SDLP is dependent on external factors such as road markings, lighting and climatic conditions. Sometimes, these driving performance measures are not specific to the driver’s drowsiness. In particular, these kinds of driving behavior measures are dependent on the vehicle type, driver experience, and conditions of the road.

### 6.3 Driver Distraction Detection

Distraction is another important factor causing impairment of driver attention, involving a driver not paying sufficient attention to the road in spite of the presence of obstacles or other people. In particular, there is a trend toward increasing use of in-vehicle information systems, which also leads to driver distraction. Distractions can be categorized as visual, that is, “eye-off-road”, and cognitive, that is, “mind-off-road” [17]. Driver distraction may lead to larger lane variation, slower response to obstacles, and more abrupt steering control, and its monitoring should be a feature of a safer driver-monitoring system.

To detect driver distraction, it is necessary to extract head pose or gaze information [63, 64]. Head pose estimation provides a driver's field of view and current focus of attention. It is intrinsically linked to visual gaze direction. When the eyes are not visible, head pose is used to estimate the gaze direction. The combination of both head pose and eye direction provides a person's gaze information [65].

Murphy-Chutorian et al. [63] categorize eight conceptual approaches to head pose estimation. These are appearance template methods, detector array methods, nonlinear regression methods, manifold embedding methods, flexible models, geometric methods, tracking methods, and hybrid methods. Feature-based methods are the most commonly used gaze direction estimation methods [64]. With these methods, local features such as contours, eye corners, and reflections from the eye images are used. However, in vehicular environments, general gaze direction is sometimes good enough to reduce false warnings in forward collision warning (FCW) systems [66]. General gaze direction can be approximated by using only head orientation, which is computed by shape features with/without eye position, texture features or hybrid features consisting of shape and texture features [67].

### 6.4 Predicting Unsafe Driving Behavior

Monitoring driver state, driving behavior performance and vehicle state is very important for improving active driver safety systems. The driver state is monitored by measuring drowsiness, fatigue or stress levels [68–71]. Driving behavior performance and vehicle state are also monitored by analyzing the information regarding driving speed, steering wheel angle, braking, and acceleration [58–60, 72–74]. After detecting drowsiness or distraction, an alert is sent to the driver.

Another important issue for an active driver safety system is to develop a mechanism to predict minor and major accidents in advance. Jabon et al. [18] used facial features to aid in driver accident prediction. They combined both vehicle dynamics and driver face analysis for accident prediction. First, a comprehensive set of 22 raw facial features are analyzed. Then, the most valuable statistics for predicting accidents are extracted from a range of time and frequency domain values, so that both major and minor accidents are can be predicted. Even though

the experimental results of Jabon et al. [18] are not based on real road situations, it has been found that facial features show most predictive accuracy four seconds prior to accidents and are more helpful in predicting minor accidents than major ones. This is because predictive accuracy for major accidents comes primarily from vehicle features rather than facial features.

EPFL and PSA Peugeot Citroen [19] are developing a technology to detect the driver distraction as well as emotions that could indicate that the driver is not up to the task at hand. In other words, facial expressions and muscle movements are important in analyzing whether the driver is too distracted, too tired or even too angry to safely control their vehicle.

Although facial features have proven to aid in predicting minor accidents, their predictive efficacy should be improved. To predict accidents more accurately, it is necessary to capture other physiological signals either from the driver or from some other part of the driver-environment system. Based on these data, a new model can be generated for predicting impending driver accidents. In particular, various populations of participants, traffic cultures and driving contexts should be handled in constructing a more extensive and general accident prediction model.

## 6.5 Discussion

For active driver safety systems, we have discussed various topics such as driver drowsiness detection, driver distraction detection, and prediction of unsafe driving behavior. To develop a better driver safety system, several other issues should be addressed. The most important ones are (a) hybrid measures for drowsiness detection, (b) driving context-awareness for safe driving, (c) the necessity for public data sets for simulation and real driving conditions.

### 6.5.1 *Hybrid Measures for Drowsiness Detection*

Among driving behavior and driver behavior features in detecting driver drowsiness, driving behavior may sometimes not detect a driver's drowsiness reliably. Driver behavior features are better than driving behavior features, but visual features have sometimes limitations due to illumination conditions and driver posture [41]. Non-visual features such as physiological features are reliable and accurate, but their nature is intrusive. This should be solved before they can be used in real vehicular environments. Even though a less intrusive measurement of ECG has been developed [56, 75, 76], EEG and EoG still require electrodes placed on the scalp or eye area in an intrusive manner. However, non-intrusive measurement of physiological signals may be developed in the near future [76, 77].

Some attempts at a fusion of various measurements has already been done [23, 77, 78]. A mixture of PERCLOS, ECC and EEG were used to detect driver

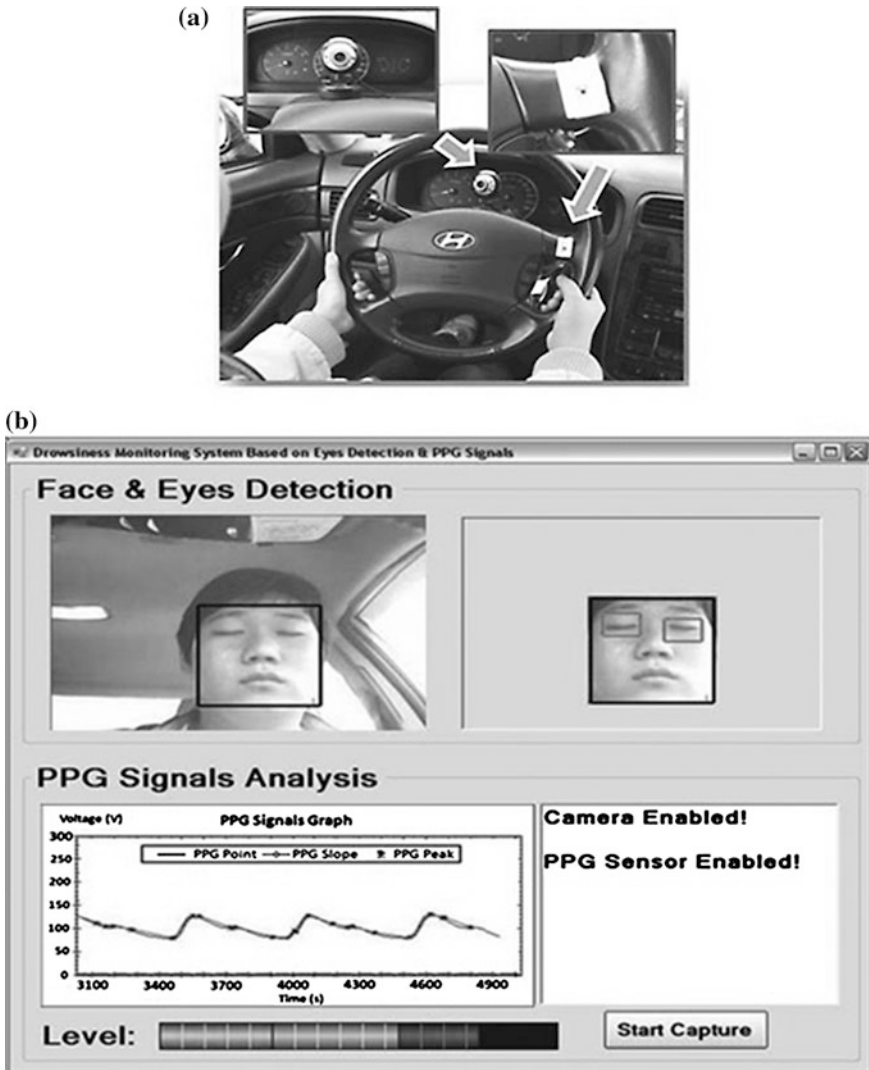


Fig. 6.9 a Web cam and PPG PCB. b drowsiness detection monitoring system [49]

drowsiness and resulted in a higher success rate than individual measures [23]. Lee et al. [49, 79] proposed a real-time physiological and vision monitoring method for drowsiness detection. Photoplethysmography (PPG) signal and eye pattern data are combined to detect driver fatigue. Figure 6.9 shows an example of web cam and PPG system from [49]. In another example, Cheng et al. [78] used fusion of PERCLOS, blink rate, maximum close duration and percentage of non-steering measures to detect drowsiness.

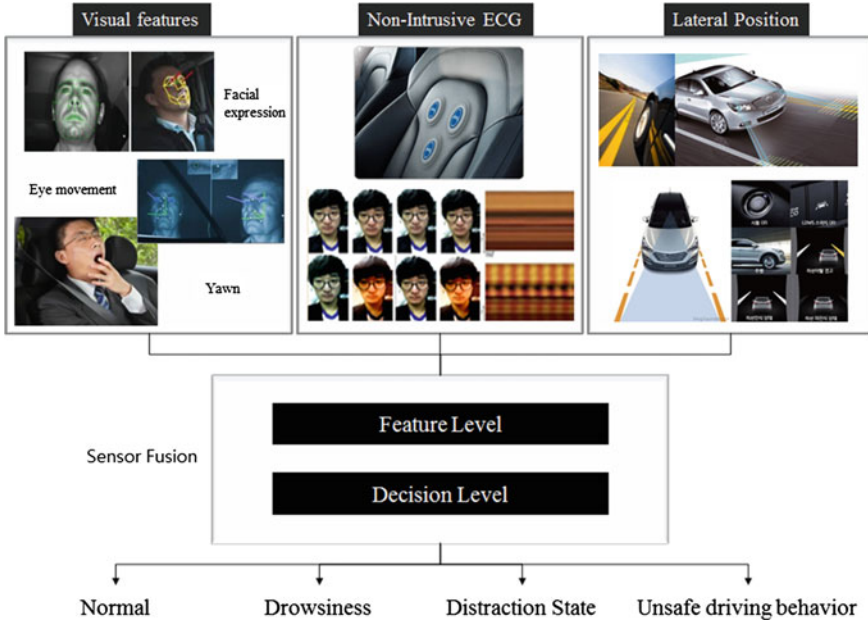


Fig. 6.10 Hybrid measurement for detecting driver state

The fusion method usually shows good performance in drowsiness detection even when certain sensors lose validity. A desirable hybrid measure is to fuse visual, physiological and driving behavior features. Figure 6.10 shows an example of hybrid measurement for driver state detection. One issue is developing data fusion methods such as feature-level and decision-level fusion [23].

### 6.5.2 Driving Context-Awareness for Safe Driving

For safe driving, driving context awareness is necessary because various information related to driving conditions and environment should be fused effectively. We can divide driving context into global and local. The global driving context refers to driving environment parameters such as vehicle type, road type, driving time, driving circumstances, road conditions, etc. The local driving context refers to the driver status. In other words, local context is related to the driver’s visual and cognitive perceptiveness and its deterioration because of distraction, drowsiness and/or emotions.

In particular, a driving context-based computational model combining driving environment and driver status seems to be a promising avenue of development. Using such a model, the detection rate for distraction and drowsiness will increase, which will be helpful in predicting unsafe driving behavior.

### ***6.5.3 Necessity for Public Data Sets for Simulation and Real Driving Conditions***

To monitor driver and driving behavior, various hardware and software algorithms are being developed, but they are tested mostly in the simulated environments instead of real driving ones. This is due to the danger of testing drowsiness in real driving environments. Philp et al. [80] found that reaction times and sleepiness as derived from self-evaluations increase in a simulated environment compared to a real driving environment due to the monotony of the experience. Engstorm et al. [81] stated that physiological workload and steering activity were both higher under real driving conditions compared to simulated environments. In real driving conditions, various factors including variations in lighting and noise can also affect the driver's attention. Thus, it is necessary to make simulated environments look more like realistic.

Even though various kinds of methods for drowsiness and distraction detection are proposed and tested, it is very difficult to compare them directly. This is because there are no benchmark data sets available. To develop reliable drowsiness detection systems, public data sets covering simulated and real driving environments should be released in the near future.

## **6.6 Conclusions**

In this chapter, we have reviewed the various methods available to determine the drowsiness and distraction state of the driver. Driver behavior such as visual features, non-visual features and driving performance behavior are explored to detect driver drowsiness. PERCLOS, eye-closure duration (ECD), frequency of eye closure (FEC) are visual feature-based systems used to detect driver drowsiness. Among these, PERCLOS shows good performance in detecting drowsiness but has some limitations, such as illumination conditions. To overcome this problem, an 850 nm IR illuminator is used. Physiological signals such as ECG, EEG, EoG and PPG signals are used as non-visual features to detect driver drowsiness. Even though physiological signals show better performance than visual features, they have some limitations, particularly their intrusive nature. To overcome this problem, less intrusive sensors should be developed. Currently, ECG signals can be captured using a less intrusive manner. Driving performance behavior such as steering wheel movement and standard deviation of lateral position are also used to detect drowsiness.

Driver distraction is detected using head pose and gaze direction. Driver distraction may lead to larger lane variation, slower response to obstacles, and more abrupt steering control. Thus, distraction should be monitored for developing a safer driver-monitoring system.

For active driver safety systems, it is desirable to predict unsafe driving behavior. We have explained prediction methods based on facial expression and car dynamics. Based on facial expression, the driver's emotion is detected, which is helpful in predicting driving behavior.

Finally, we have discussed several issues to tackle in future development of active driver safety systems. They are (a) hybrid measures for drowsiness detection, (b) driving context-awareness for safe driving, (c) the necessity of public data sets for simulation and real driving conditions.

## References

1. U.S. Department of Transportation, Traffic Safety Facts 2006: A compilation of motor vehicle crash data from the fatality analysis reporting system and the general estimates system. Technical report DOTHS 810 818, National Highway Traffic Safety Administration, 2006
2. <http://www.lboro.ac.uk/departments/ssehs/research/behavioural-medicine/sleep-research-centre/>
3. Y. Liang, Detecting driver distraction, Ph.D. thesis, University of Iowa, 2009
4. M. Bayly, B. Fildes, M. Regan, K. Young, Review of crash effectiveness of intelligent transport system, TRAFFIC Accident Causation in Europe (TRACE), 2007
5. E. Rogado, J. Garcia, R. Barea, L. Bergasa, E. Lopez, Driver fatigue detection system. in *Proceedings of IEEE International Conference on Robotics and Biomimetics 2009*
6. T. Nakagawa, T. Kawachi, S. Arimitsu, M. Kanno, K. Sasaki, H. Hosaka, Drowsiness detection using spectrum analysis of eye movement and effective stimuli to keep driver awake. *DENSO Tech. Rev.* **12**, 113–118 (2006)
7. B. Hariri, S. Abtahi, S. Shirmohammadi, L. Martel, A yawning measurement method to detect driver drowsiness. Technical Papers, 2012
8. C. Lin, L. Ko, I. Chung et al., Adaptive EEG-based alertness estimation system by using ICA-based fuzzy neural networks. *IEEE Trans. Circ. Syst.* **53**(11), 2469–2476 (2006)
9. H. Caim, Y. Lin, An experiment to non-intrusively collect physiological parameters towards driver state detection, in *Proceedings of the SAE World Congress*, Detroit, 2007
10. Q. Ji, Z. Zhu, P. Lan, Real-time nonintrusive monitoring and prediction of driver fatigue. *IEEE Trans. Veh. Technol.* **53**(4), 1052–1068 (2004)
11. S. Abtahi, Driver drowsiness monitoring based on yawning detection, MS thesis, University of Ottawa, 2012
12. F. Nasoz, O. Ozyer, C. Lisetti, N. Finkelstein, Multimodal affective driver interfaces for future cars, in *Proceedings of ACM International Multimedia Conference Exhibition*, pp. 319–322, 2002
13. Y. Lin, H. Leng, G. Yang, H. Cai, An intelligent noninvasive sensor for driver pulse wave measurement. *IEEE Sens. J.* **7**(5), 790–799 (2007)
14. A. Hattori, S. Tokoro, M. Miyashita, I. Tanakam, K. Ohue, S. Uozumi, Development of forward collision warning system using the driver behavioral information, in *Proceedings of 2006 SAE World Congress*, Detroit, 2006
15. E. Murphy-Chutorian, A. Doshi, M. Trivedi, Head pose estimation for driver assistance systems: a robust algorithm and experimental evaluation, in *Proceedings of 10th International IEEE Conference on Intelligent Transportation Systems*, pp. 709–714, 2007
16. J. Kaminski, D. Knaan, A. Shavit, Single image face orientation and gaze detection. *Mach. Vis. Appl.* **21**, 85–98 (2009)
17. T. Victor, J. Harbluk, J. Engström, Sensitivity of eye-movement measures to in-vehicle task difficulty. *Trans. Res. Part F* **8**, 167–190 (2005)

18. M. Jabon, J. Bailenson, E. Pontikakis, L. Takayama, C. Nass, Facial-expression analysis for predicting unsafe driving behavior. *Pervasive Comput.* **10**(4), 84–94 (2011)
19. <http://phys.org/news/2012-11-tracking-facial-features-safer-comfortable.html#inRlv>
20. B. Yin, X. Fan, Y. Sun, Multiscale dynamic features based driver fatigue detection. *Int. J. Pattern Recogn. Artif. Intell.* **23**, 575–589 (2009)
21. M. Akin, M. Kurt, N. Sezgin, M. Bayram, Estimating vigilance level by using EEG and EMG signals. *Neural Comput. Appl.* **17**(3), 227–236, (2008)
22. A. Kokonozi, E. Michail, I. Chouvarda, N. Maglaveras, A study of heart rate and brain system complexity and their interaction in sleep-deprived subjects, in *Proceedings of the Conference Computers in Cardiology*, Bologna, 2008
23. Y. Guosheng, L. Yingzi, B. Prabir, A driver fatigue recognition model based on information fusion and dynamic Bayesian network. *Inform. Sci.* **180**, 1942–1954 (2010)
24. R. Khushaba, S. Kodagoda, S. Lal, G. Dissanayake, Driver drowsiness classification using fuzzy wavelet-packet-based feature-extraction algorithm. *IEEE Trans. Biomed. Eng.* **58**(6), 1855–1864 (2011)
25. W. Liang, J. Yuan, D. Sun, M. Lin, Changes in physiological parameters induced by indoor simulated driving: effect of lower body exercise at mid-term break. *Sensors* (2009)
26. A. Sahyadehas, K. Sundarajm, M. Murugappan, Detecting driver drowsiness based on sensors: a review. *Sensors* (2012)
27. K. Torkkola, N. Massey, C. Wood, Driver inattention detection through intelligent analysis of readily available sensors, in *Proceedings Of IEEE Conference on intelligent transportation systems*, Washington DC, pp. 326–331, 2004
28. C. Liu, S. Hosking, M. Lenné, Predicting driver drowsiness using vehicle measures: recent insights and future challenges. *J. Saf. Res.* **40**, 239–245 (2009)
29. L. Bergasa, J. Nuevo, M. Sotelo, R. Barea, M. Lopez, Real-time system for monitoring driver vigilance. *IEEE Trans. Intell. Trans. Syst.* **7**(1), 63–77 (2006)
30. T. D’Orazio, M. Leo, C. Guaragnella, A. Distanto, A visual approach for driver inattention detection. *Pattern Recog.* **40**(8), 2341–2355 (2007)
31. W.W. Wierwille, L.A. Ellsworth, S.S. Wreggit, R.J. Fairbanks, C.L. Kim, Research on vehicle based driver status/performance monitoring: development, validation, and refinement of algorithms for detection of driver drowsiness. National Highway Traffic Safety Administration Final Report, DOT HS 808 247 (1994)
32. D. Dinges, R. Grace, PERCLOS: a valid psychophysiological measure of alertness as assessed by psychomotor vigilance. U.S. Department of Transportation, Federal highway Administration. Publication Number FHWA-MCRT-98-006
33. R. Grace et al., A drowsy driver detection system for heavy vehicle, in *Proceedings of 17th Digital Avionics Systems Conference*, Bellevue, vol. 2, pp. I36/1–I36/8, 1998
34. C. Yan, Y. Wang, Z. Zhang, Robust real-time multi-used pupil detection and tracking under various illumination and large-scale head motion. *Comput. Vis. Image Underst.* 1223–1338 (2011)
35. W. Shen, H. Sun, E. Cheng, Q. Zhu, Q. Li, Effective driver fatigue monitoring through pupil detection and yawing analysis in low light level environments. *Int. J. Digit. Technol. Appl.* **6**, 372–383 (2012)
36. M. Flores, J. Armingol, A. de la Escalera, Driver drowsiness warning system using visual information for both diurnal and nocturnal illumination conditions. *EURASIP J. Adv. Sign. Process.* **2010**, 1–20 (2010)
37. J. Jo, S. Lee, H. Jung, K. Park, J. Kim, Vision-based method for detecting driver drowsiness and distraction in driver monitoring system. *Opt. Eng.* **20**(12), 127202 (2011)
38. Seeing Machines Driver State Sensor, <http://www.seeingmachines.com/product/dss/>
39. T. Xue, N. Nan, M. Fan, J. Yong, Head pose estimation using isophote features for driver assistance systems, in *Proceedings of the IEEE Intelligent Vehicles Symposium*, Xi’an, 3–5 June 2009
40. E. Murphy-Chutorian, M. Trivedi, Head pose estimation and augmented reality tracking: an integrated system and evaluation for monitoring driver awareness. *IEEE Trans. Intell. Trans. Syst.* **11**(2), 300–311 (2010)



41. T. Nakamura, T. Matsuda, A. Maejima, S. Morishima, Driver drowsiness estimation using facial wrinkle feature. *Siggraph poster* (2013)
42. M. Miyaji, H. Kawanaka, K. Oguri, Driver's cognitive distraction detection using physiological features by the adaboost, in *Proceedings of the 12th International IEEE Conference on Intelligent Transportation Systems*, St. Louis, 2009
43. M. Patel, S. Lal, D. Kavanagh, P. Rossiter, Applying neural network analysis on heart rate variability data to assess driver fatigue. *Exp. Syst. Appl.* **38**(6), 7235–7242 (2011)
44. T. Chin, J. Che, S. Bor, H. Shao, F. Chih, I. Wang, A real-time wireless brain-computer interface system for drowsiness detection. *IEEE Trans. Biomed. Circ. Syst.* (2010)
45. J. Liu, C. Zhang, C. Zheng, EEG-based estimation of mental fatigue by using KPCA-HMM and complexity parameters. *Biomed. Sign. Process. Contr.* **5**, 124–130 (2010)
46. C. Fu, W. Li, H. Chun, P. Tung, T. Chin, Generalized EEG-based drowsiness prediction system by using a self-organizing neural fuzzy system. *IEEE Trans. Circ. Syst.* (2012)
47. M. Kurt, N. Sezgin, M. Akin, G. Kirbas, M. Bayram, "The ANN-based computing of drowsy level." *Exp. Syst. Appl.*, 2009
48. S. Lal, A. Craig, A critical review of the psychophysiology of driver fatigue. *Biol. Psychol.* **55**(3), 173–194 (2001)
49. B. Lee, S. Jung, W. Chung, Real-time physiological and vision monitoring of vehicle driver for non-intrusive drowsiness detection. *IET Commun.* **5**(17), 2461–2469 (2011)
50. R. Enriquez, M. Castellanos, J. Rodriguez, J. Caceres, Analysis of the photoplethysmographic signal by means of the decomposition in principal components. *Phys. Meas.* **23**(3), N17–N29 (2002)
51. H. Shin, C. Lee, M. Lee, Adaptive threshold method for the peak detection of photoplethysmographic waveform. *Comput. Biol. Med.* **44**, 331–337 (2009)
52. V. Vapnik, in *Statistical Learning Theory*. Support vector estimation of functions, (Wiley, Hoboken, 1998), pp. 375–570
53. S. Hu, G. Zheng, Driver drowsiness detection with eyelid related parameters by support vector machine. *Exp. Syst. Appl.* **36**, 7651–7658 (2009)
54. M. Kurt, N. Sezgin, M. Akin, G. Kirbas, M. Bayram, The ANN-based computing of drowsy level. *Exp. Syst. Appl.* **36**, 2534–2542 (2009)
55. X. Yu, Real-time nonintrusive detection of driver drowsiness. Technical Report for University of Minnesota, Minneapolis (2009)
56. J. Hyun, S. Gih, K. Ko, S. Kwang, A Smart health monitoring chair for nonintrusive measurement of biological signals. *IEEE Trans. Inform. Technol. Biomed.* (2012)
57. B. Lee, W. Chung, Multi-classifier for highly reliable driver drowsiness detection in android platform. *Biomed. Eng. Appl. Basis Commun.* **24**, 147–154 (2012)
58. C. Wylie, T. Shultz, J. Miller, M. Mitler, R. Mackie, Commercial motor vehicle driver fatigue and alertness study: technical summary. FHWA-MC-97-001 (1996)
59. H. Uno, Detection decline in arousal level using combined physiological and behavioral measures. *JARI Res. J.* **25**(8), (2003)
60. S. Otmami, T. Pebayle, J. Roge, A. Muzet, Effect of driving duration and partial sleep deprivation on subsequent alertness and performance of car drivers. *Physiol. Behav.* **84**, 715–724 (2005)
61. F. Ruijia, Z. Guangyuan, C. Bo, "An on-Board System for Detecting Driver Drowsiness Based on Multi-Sensor Data Fusion Using Dempster-Shafer Theory," In *Proceedings of the International Conference on Networking, Sensing and Control*, Okayama, Japan, 2009
62. M. Ingre, T. Åkerstedt, B. Peters, A. Anund, G. Kecklund, Subjective sleepiness, simulated driving performance and blink duration: examining individual differences. *J. Sleep Res.* **15**(1), 47–53 (2006)
63. E. Murphy-Chutorian, M. Trivedi, Head pose estimation in computer vision: a survey. *IEEE Trans. Pattern Anal. Mach. Intell.* **31**(4), 607–626 (2009)
64. W. Hansen, Q. Ji, In the eye of the beholder: a survey of models for eyes and gaze. *IEEE Trans. Pattern Anal. Mach. Intell.* **32**, 478–500 (2010)

65. S. Langton, H. Honeyman, E. Tessler, The influence of head contour and nose angle on the perception of eye-gaze direction. *Percept. Psychophys.* **66**(5), 752–771 (2004)
66. K. Ohue, Y. Yamada, S. Uozumi, S. Tokoro, A. Hattori, T. Hayashi, Development of a new pre-crash safety system, presented at the Society of Automotive Engineering World Congress, SAE Technical paper series, Paper 2006-01-1461, Detroit, 2006
67. S. Lee, J. Jo, H. Jung, K. Park, J. Kim, Real-time gaze estimator based on driver's head orientation for forward collision warning system. *IEEE Trans. Intell. Trans. Syst.* **12**(1), 254–267 (2011)
68. M. Trivedi, T. Gandhi, J. McCall, Looking-in and looking-out of a vehicle: computer-vision-based enhanced vehicle safety. *IEEE Trans. Intell. Trans. Syst.* **8**(1), 108–120 (2007)
69. A. Williamson, T. Chamberlain, *Review of On-Road Driver Fatigue Monitoring Devices* (NSW Injury Risk Management Research Centre, University of New South Wales, NSW, 2005)
70. J. Healy, R. Picard, Detecting stress during real-world driving tasks using physiological sensors. *IEEE Trans. Intell. Trans. Syst.* **6**(2), 156–166 (2005)
71. L. Fletcher, L. Petersson, A. Zelinsky, Road scene monotony detection in a fatigue management driver assistance system, in *Proceedings of IEEE Intelligent Vehicles Symposium*, IEEE Press, 2005
72. J. McCall, M. Trivedi, Video-based lane estimation and tracking for driver assistance: survey, system, and evaluation. *IEEE Trans. Intell. Trans. Syst.* **7**(1), 2037 (2006)
73. M. Bertozzi et al., Knowledge-based intelligent information and engineering systems, ed. by B. Appolloi et al. (Springer, Berlin, 2007)
74. L. Li et al., IVS 05: new developments and research trends for intelligent vehicles. *IEEE Intell. Syst.* **20**(4), 10–14 (2005)
75. B. Lawrence, P. Stephen, H. Howarth, *An Evaluation of Emerging Driver Fatigue Detection Measures and Technologies* (Volpe National Transportation Systems Center Cambridge, Cambridge, 2009)
76. S. Ikeda, H. Ishimura, M. Mastumura, Non-restrictive measurement of pulse transit time using ECG sensor and PPG sensor mounted on the neckband, in *Proceedings of 35th Annual International IEEE EMBS Conference*, 2013
77. M. Poh, D. McDuff, R. Picard, Non-contact, automated cardiac pulse measurements using video imaging and blind source separation. *Opt. Express* **18**(10), 10762–10774 (2010)
78. B. Cheng, W. Zhang, Y. Lin, R. Feng, X. Zhang, Driver drowsiness detection based on multisource information. *Hum. Factors Ergon. Manuf. Serv. Ind.* (2012)
79. B. Lee, W. Chung, Driver alertness monitoring using fusion of facial features and bio-signals. *IEEE Sens. J.* **12**, 2416–2422 (2012)
80. P. Philip, P. Sagaspe, N. Moore, J. Taillard, A. Charles, C. Guilleminault, B. Bioulac, Fatigue, sleep restriction and driving performance. *Accid. Anal. Prev.* **37**(3), 473–47 (2005)
81. J. Engström, E. Johansson, J. Östlund, Effects of visual and cognitive load in real and simulated motorway driving. *Trans. Res. Traffic Psychol. Behav.* **8**, 97–120 (2005)

# Chapter 7

## SoC Architecture for Automobile Vision System

Kyounghoon Kim and Kiyoung Choi

**Abstract** Advanced Driver Assistance System (ADAS) is becoming more and more popular and increasing its importance in a car with the advancement in electronics and computer engineering that provides key enabling technologies for such a system. Among others, *vision* is one of the most important technologies since the current practice of automotive driving is mostly, if not entirely, based on vision. This chapter discusses architectural issues to be considered when designing Systems-on-a-Chip (SoC) for automobile vision system. Various existing architectures are introduced together with some analysis and comparison.

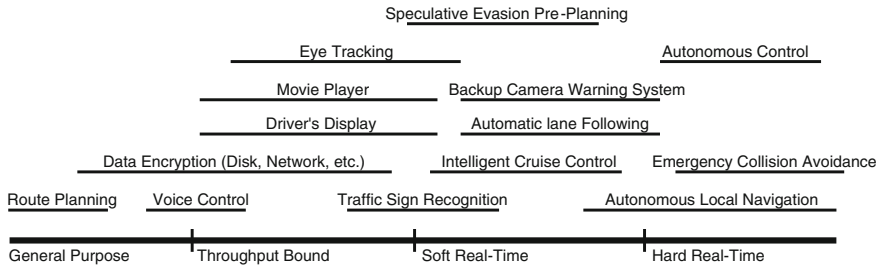
### 7.1 Automotive Applications

An ADAS may consist of many different applications including emergency collision avoidance, backup warning, adaptive cruise control, self-parking, lane departure warning, route planning, etc. Some of those applications are very time critical. For example, emergency collision avoidance should be fast enough so that the system can give timely control over the steering and/or braking of the car. Such applications have deadlines to be surely satisfied. If they fail to meet the deadlines, then serious problems such as traffic accident can occur. Thus the applications must be implemented with high priority such that the latency constraints are to be definitely satisfied during the execution of the applications. In this case, the implementation should consider the system performance in terms of latency, i.e., the delay from the input event to the output or response. On the other hand, some

---

K. Kim · K. Choi (✉)  
Seoul National University, Seoul, Korea  
e-mail: kchoi@snu.ac.kr

K. Kim  
e-mail: khkim@dal.snu.ac.kr



**Fig. 7.1** Spectrum of possible temporal requirements for a number of automotive applications [1] (the term ‘Real-Fast’ in [1] has been replaced with ‘Throughput Bound’ by the author’s preference)

applications do not have stringent deadline constraints. For example, route planning applications sometimes take time to find an optimal path to the destination. It could be annoying, but does not directly lead to a serious problem. Some applications require high throughput but may not care much about the latency. For example, video display requires high throughput computing for processing motion image data of large bandwidth, but the latency is in general not an issue. That makes pipelined architectures very attractive for such applications. Thus the design of an ADAS system should take into account such temporal requirements of applications that are to be integrated into the system. Figure 7.1 shows various automotive applications and their temporal requirements in terms of criticality in response time [1].

## 7.2 Architectural Consideration for Vision

The architecture design for an automobile vision system may need considering requirements in many aspects including accuracy, performance, energy consumption, reliability, and cost. For example, the system may need accurate detection of objects such as pedestrians. It should be fast enough for timely warning or emergency stop of the car. Even when not driving the car, the user may want to keep the system turned on for security purposes and, in such a case, energy consumption matters. Reliability is another important issue especially for autonomous operation of a car. Those requirements can be considered as constraints to be satisfied or as objective functions to be optimized during the architecture design.

Designing a vision system architecture that satisfies such constraints can be very challenging. First of all, the architecture needs to handle heavy computations on a huge amount of data. For example, the calculation of disparity map needs one billion operations per stereo VGA ( $640 \times 480$ ) image [2]. Thus processing 30 video frames per second requires 30 billion operations per second (30 GOPS) only for disparity map calculation. It also requires lots of data requiring large memory

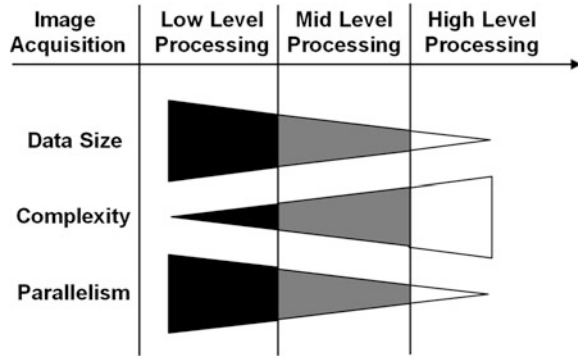
space and bandwidth. In the VGA case, the required memory capacity for a frame of stereo image is  $640 \times 480 \times 3$  (RGB)  $\times 8$  (color)  $\times 2 = 14$  Mbits (1 M = 1,048,576). Increasing the resolution to SVGA (800  $\times$  600) and 1080p (1920  $\times$  1080) increases the memory capacity to 22 Mbits and 95 Mbits, respectively. Consider fetching 30 frames of video data per second. It amounts to 420 Mbps, 660 Mbps, and 2.85 Gbps for VGA, SVGA, and 1080p, respectively, only for fetching raw video data. The amounts of computation, memory capacity, and communication depend on the applications, the constraints given, and the algorithms used for implementing the applications. In general, however, the amounts are relatively big and can be handled efficiently only by careful architecture design.

Also to be considered are real-time issues. For hard real-time systems, it is important to meet the deadlines. Just improving the average performance may not help but making the performance predictable helps. Assigning more time-critical tasks with higher priority allows using slower/cheaper hardware/processor while satisfying real-time constraints. Pipelining can improve throughput for streaming applications but may not help satisfying real-time constraints.

Flexibility is another issue for architecture design. Compared to software implementation, hardware implementation can achieve much higher performance at much lower energy consumption and area cost. However, it requires much more effort and time for the design. What makes it worse is that it is not flexible at all; if it is to be modified for a bug-fix or upgrade, it should be entirely re-implemented. On the other hand, software implementation is relatively easy, provided that the processor architecture is given. Bug-fix or upgrade is also relatively easy if that can be done by only software modifications. Performance improvement may be achieved by just replacing existing processor cores with faster ones and porting the existing software to the new processor cores. However, such an approach may have limitation in performance and energy consumption. Thus a good balance between hardware and software is important in designing the architecture.

The image signal processing for vision can be divided into three different levels: low-level, mid-level, and high-level. Each level deals with different data size, different algorithm complexity, and different levels of parallelism. Figure 7.2 shows how the characteristics change as the processing moves from low-level to high-level [3]. Low-level processing includes relatively simple filtering operations such as noise cancellation and image enhancement. It typically has large input data (pixels to be processed) and large output data (processed pixels). Although it needs to process a large amount of data, since the processing over a set of data is in general independent of that over a different set of data, massively parallel processing can be done to speed up the process. Thus it can benefit from hardware implementation. Mid-level processing includes operations such as segmentation, region description, and object classification. It typically takes large amount of input data and produces relatively small amount of output data. The algorithms used for this level of processing have complexity higher than that of simple filtering and have parallelism to some extent. High-level processing includes intelligent analysis such as cognitive vision. It takes small amount of input data

**Fig. 7.2** Characteristics of three levels of image processing [3]



extracted from the image signal and generates results of small data size. The algorithms used at this level are typically very complicated and sequential and thus software implementation fits better.

Since vision systems handle a lot of data (mostly for the low-level and mid-level processing), memory and communication architectures are also very important. The architectures should be designed based on the memory access patterns (refer to Sect. 7.7 for more discussions on architecture design based on memory access patterns). Low-level processing involves pixel to pixel transformation and mid-level processing involves pixel to symbol transformation, while high-level processing performs reasoning upon symbols without accessing memory much. Following are various operations typically performed in vision processing and the corresponding memory access patterns, which are also pictorialized by Fig. 7.3 [4].

- Point Operation (PO) gets a source pixel from the memory, processes it to generate a destination pixel, and writes it back to the memory. Examples that perform such operations include thresholding and color conversion.
- Local Neighborhood Operation (LNO) reads pixel values in a local area of an image from the memory, processes it to obtain the pixel value at the center of the area, and writes it back to the memory. 2D image filtering such as smoothing is an example.
- Global Operation (GIO) needs to read source pixels in the global area to generate a destination pixel. Fourier transform is an example, where each point in the frequency domain is computed based on all values in the spatial domain.
- Statistical Operation (SO) reads source pixels to generate scalar or vector data. Examples include histogram calculation and Hough transform. Hough transform is a technique used to detect lines or shapes in the image.
- Geometrical Operation (GeO) reads data in the source area/location to generate data in the destination location. Examples include affine transform and image transposition.

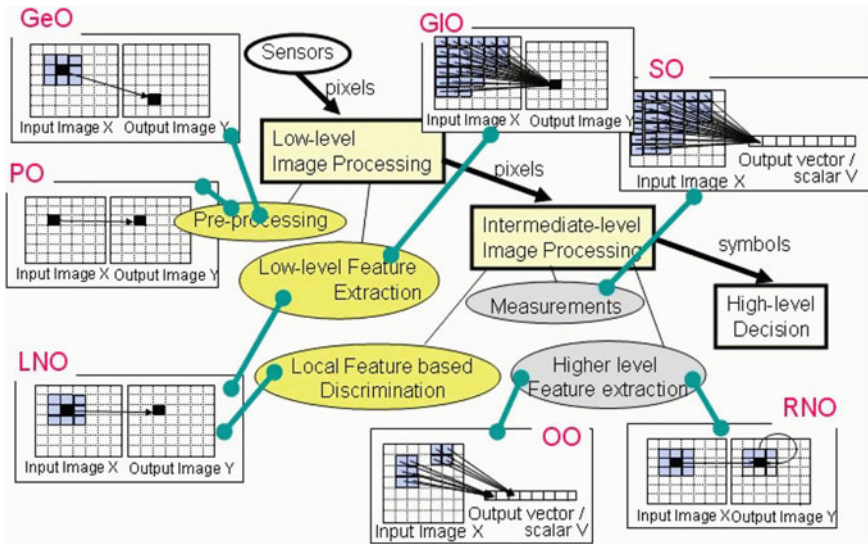


Fig. 7.3 Image processing operations and the corresponding memory access patterns [4]

- Recursive Neighborhood Operation (RNO) reads source pixels as well as previous destination pixels to determine the current destination pixel. Examples are distance transform and dither transform.
- Object Operation (OO) reads data in a local area and generates scalar or vector data corresponding to that area. Examples are region growing and connected component labeling.

## 7.3 Example: Pedestrian Detection

### 7.3.1 Detection Flow and Preprocessing

Pedestrian Protection System (PPS) is an active research area aimed at improving traffic safety. Pedestrian detection is one of the key processes in the PPS and is typically performed by a general module-based processing. As shown in Fig. 7.4 [5], the module-based processing flow may include six cascaded modules: pre-processing, foreground segmentation, object classification, verification/refinement, tracking, and application. The first three modules can be considered as low-level and mid-level processing, and thus we focus on those three modules since efficient implementation of them needs architectural consideration of hardware, which is our main interest in this chapter. The remaining three modules require complicated algorithms that may not be well parallelized and thus can be better implemented with software.

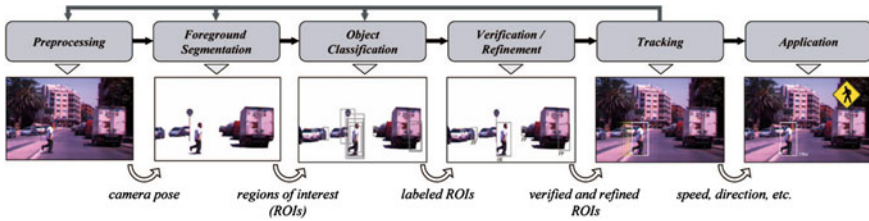


Fig. 7.4 Module-based pedestrian detection flow [5]

Among the first three stages, the preprocessing module has simple functions such as correction of camera distortion and rectification. Camera distortion includes lens distortion, sensor tilting, and offset from focal axis. Rectification is to align two corresponding epipolar lines from two stereo images into a straight line on a common image plane. For this, the epipolar lines on stereo image planes are rotated into a straight horizontal line on the common image plane. The resulting rectified pixels are typically obtained through bilinear interpolation. In this section, we focus more on the second and third stages (foreground segmentation and object classification) and explain them in the following subsections.

### 7.3.2 Foreground Segmentation

Foreground segmentation is to extract regions of interest (ROIs) in a frame of image so that they can be used as inputs to the next vision processing step for object classification. One primitive way of generating ROIs is through exhausted scan. In that case, however, huge amount of data will be generated forcing subsequent processing steps consume a lot of time and memory space. The foreground segmentation step must deselect as many background regions as possible for efficiency of the subsequent steps for pedestrian detection, whereas it should collect foreground ROIs if they have any possibility to include a target object, which may be more important. Figure 7.5 shows (a) an original image (b) only 10 % of the ROIs obtained by exhaustive scan, and (c) extracted ROIs by a more efficient way. It is not difficult to see the difference in the number of ROIs between the exhausted search and a smart search at first glance.

Due to foreground segmentation, the amount of computation can be reduced dramatically. In addition, once classified as having a target object in the previous video frame, the object can be tracked in order to enhance the accuracy of segmentation. This is a good example to explain how each step of vision processing affects other steps.

There are many techniques used for foreground segmentation including stereo vision, laser scanning, optical flow, and combination thereof. Among them, stereo vision is the most useful technique that can be combined with other techniques to



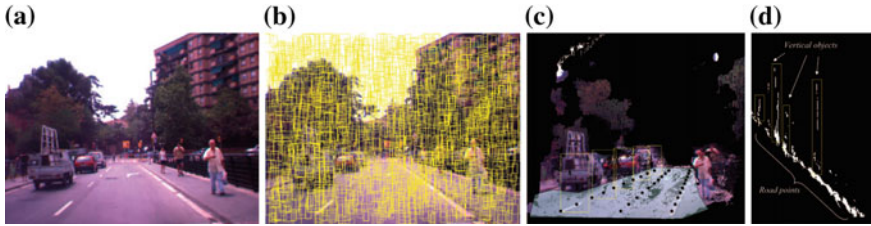


Fig. 7.5 Foreground segmentation [5]

build a powerful tool for pedestrian detection. It can extract ROIs with perspective views similar to the behavior of animals with two eyes.

To find ROIs using stereo images, a  $v$ -disparity image can be obtained from the disparity map of two stereo images [6]. Figure 7.5d show the  $v$ -disparity image obtained for the original scene in Fig. 7.5a. In the  $v$ -disparity image, abscissas represent disparity values and ordinates represent row numbers in the original image. The intensity of a point  $(x, y)$  in the  $v$ -disparity image represents the number of points that have the same disparity value  $x$  on row  $y$  in the disparity map. Thus the slanted line appearing on the  $v$ -disparity image typically represents the road (horizontal plane) and the vertical lines typically represent obstacles (vertical planes or ROIs).

Another way of finding ROIs using stereo images (used for extracting depth information) is to perform road plane fitting directly in the Euclidean space instead of obtaining the  $v$ -disparity image (to avoid inefficiency due to non-linear relationship between disparity and depth) [7, 8]. Once the road plane fitting is done, a set of ROIs are obtained by placing boxes sized by pedestrian size constraint (PSC) at equispaced grid points on the road as shown in Fig. 7.5c.

### 7.3.3 Object Classification

In the object classification step, machine learning and pattern recognition techniques are used in order to classify the input ROIs into two categories: pedestrian and non-pedestrian, which is the objective of the entire system. Thus, object classification is a very important step in a pedestrian detection system. The objective of classifier itself is to classify input objects into relative categories with some criteria, which is similar to what human beings do. A classifier generally works in two phases: training phase for building up the classifier and recognition phase for the actual use of it. In the former, the classifier is trained with a set of data to adjust its internal parameters to enhance the accuracy of classification; in the latter, it is used to make a decision on whether an input data object belongs to a category or not. This section introduces three classification techniques—support vector machine, adaptive boosting, and neural network—that are relatively well

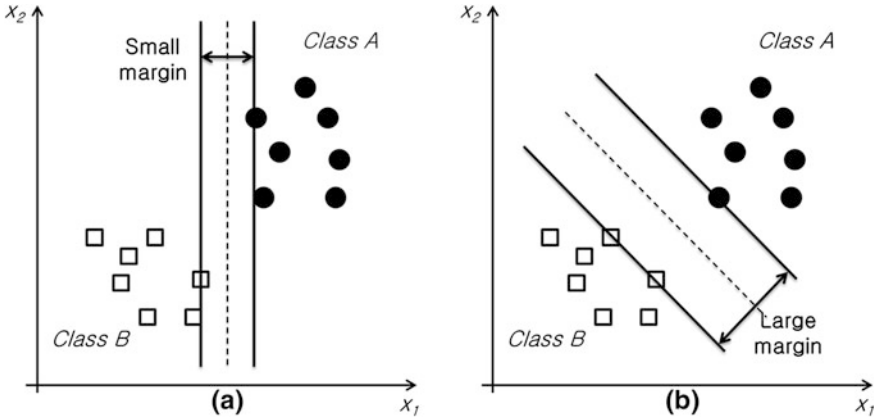


Fig. 7.6 SVM classifier

known due to achieving good performance thereby being adopted as classifiers in the contemporary automobile vision systems.

Support Vector Machine (SVM) invented by Vapnik [9], is one of the most popular methods for supervised learning. It is popular mainly due to its performance. The main idea of SVM is to maximize the margin between classes in order to increase the capability of separation. Maximizing the margin helps the classifier not to generate an error. Figure 7.6 shows an example where input data are grouped into two classes—circles and squares—which should be identified by the SVM classifier. It is easy to see that (b) has a larger margin than (a), implying that the SVM classifier represented by the separation line in (b) is more stable than that in (a). Thus, SVM concentrates on how to maximize the margin between classes.

In order to obtain the maximum margin, mathematical analysis and induction can be adopted with Lagrange multiplier. However, because this is beyond our scope, this section just briefly mentions the theoretical view. As shown in Fig. 7.7, the hyper-plane separating the two classes can be represented by  $\mathbf{w}^T \mathbf{x} - b = 0$ , where  $\mathbf{x}$  is a vector indicating a point on the hyper-plane, and  $\mathbf{w}$  is the normal vector to the hyper-plane. The parameter  $b/\|\mathbf{w}\| = b/\sqrt{(\mathbf{w}^T \mathbf{w})}$  determines the offset of the hyper-plane from the origin along the normal vector  $\mathbf{w}$ . The set of input data points  $\mathbf{x}_i$  belonging to one class (circles) satisfies the constraint given by  $\mathbf{w}^T \mathbf{x}_i - b \leq -1$ , and the other set of data points  $\mathbf{x}_j$  (squares) satisfies the constraint given by  $\mathbf{w}^T \mathbf{x}_j - b \geq 1$ . Then the margin is given by  $2/\sqrt{(\mathbf{w}^T \mathbf{w})}$ , and thus  $\|\mathbf{w}\|^2 = \mathbf{w}^T \mathbf{w}$  is minimize (to maximize the margin) by adjusting  $\mathbf{w}$  and  $b$  in the training process while satisfying the constraints. The data points on the boundary of the constraints (i.e., the vectors  $\mathbf{x}_k$  satisfying  $\mathbf{w}^T \mathbf{x}_k - b = -1$  or  $\mathbf{w}^T \mathbf{x}_k - b = 1$ ) are called support vectors; this is the reason why the classifier is called support vector machine.

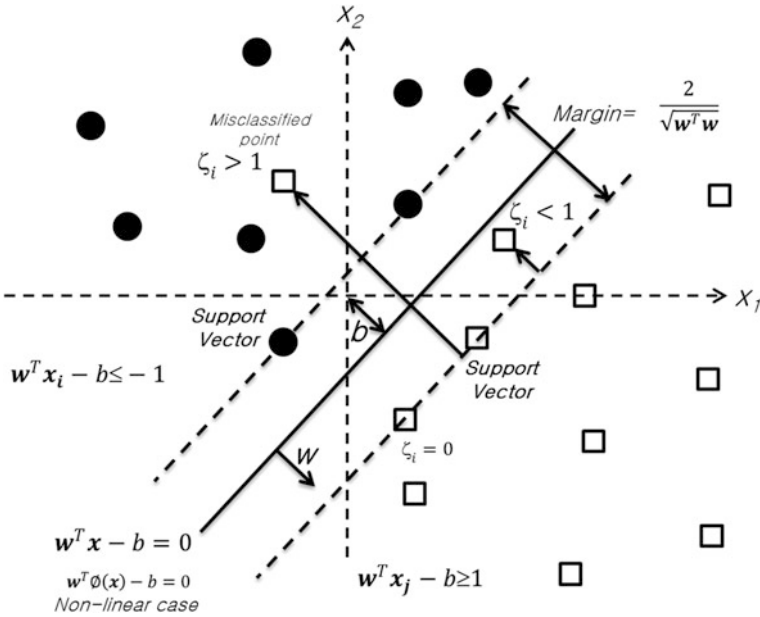
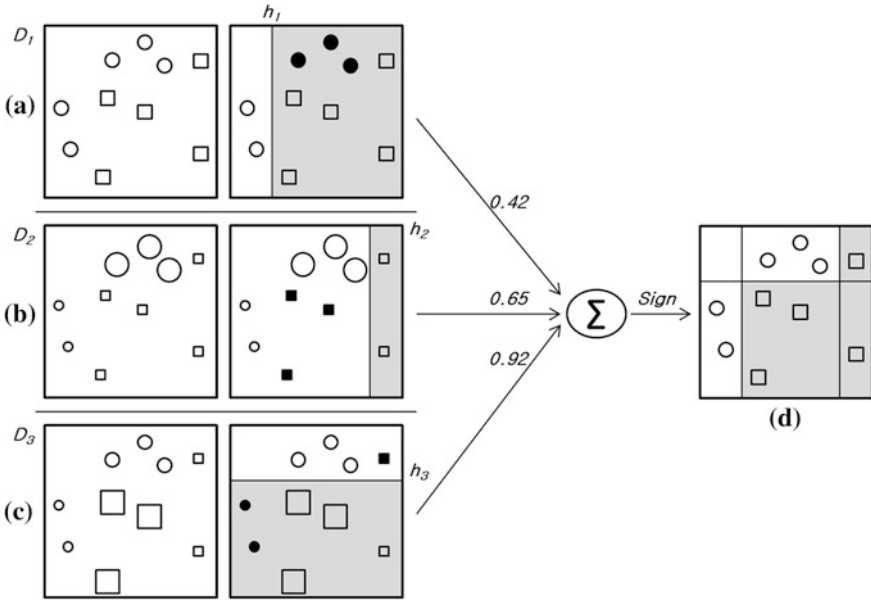


Fig. 7.7 Maximum-margin hyper-plane

There are cases where no hyper-plane can divide the space cleanly; some misclassified points have to be allowed as shown in Fig. 7.7. Also, some points are allowed to be placed within the margin area. Of course the number of such points and the degree of misclassification should be minimized. For this, the concept of soft margin hyper-plane is introduced [10]. The constraints are replaced by  $w^T x_i - b \leq -(1 - \zeta_i)$  and  $w^T x_j - b \geq (1 - \zeta_j)$ , where  $\zeta_i \geq 0$  represents the degree of misclassification of data point  $x_i$ . In this case,  $w$ ,  $b$ , and  $\zeta_i$  are adjusted to minimize  $\|w\|^2 + C \sum_i \zeta_i$  (the objective function to be minimized takes this form for mathematical convenience but can take a different form).

Although the SVM mentioned here divides two classes in a linear manner, non-linear SVM can also be implemented by doing transformation using a function  $\phi(x)$  and applying the standard SVM formulation to obtain  $w^T \phi(x) - b = 0$ . However, the details are omitted in this section.

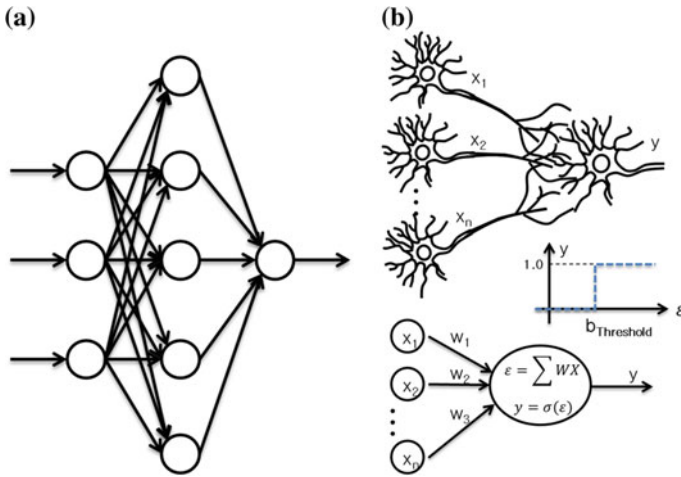
AdaBoost, which stands for Adaptive Boosting, was invented by Freund and Schapire [11]. The main idea is to combine a set of weak classifiers, because generating a single strong classifier is much more difficult in many cases. In other words, combining a set of weak classifiers is more efficient in terms of training effort and accuracy enhancement. However, there also occur some complicated issues such as diversity problem among the classifiers. AdaBoost focuses on the input set on which a classifier made a wrong decision, and gives a higher weight to it when a new classifier is generated so that it can recover from the failure of the previous classifier. Figure 7.8 depicts an overview of this mechanism including three training



**Fig. 7.8** AdaBoost classifier

steps and one evaluation step. Given a set  $D_1$  of data as shown in Fig. 7.8a with circles and squares, the objective is to divide them into two classes, one for circles and the other for squares. White region represents the class of circles, while shaded region represents that of squares. In the first training step, the left part of  $D_1$  is separated by the  $h_1$  classifier, which results in three errors for the circles filled up with black. In order to recover from this failure, the weights of those data points are increased to obtain a new set  $D_2$  and a new classifier  $h_2$  is generated as shown in Fig. 7.8b. Due to the increased weights,  $h_2$  may better classify those points. However, it also generates three new errors for the squares filled up with black. In the same way as the previous process, the weights of the data points corresponding to the three new errors are adjusted in  $D_3$  so that they can be classified correctly in the next step. Finally, a new classifier  $h_3$  is generated from  $D_3$  as shown in Fig. 7.8c. The total combination of the three classifiers is shown in Fig. 7.8d, in which each classifier has its own weight. The classifier-weight is determined based on the ratio of the amount of correct classification to that of incorrect classification achieved by the classifier. The calculation of the amount of correct/incorrect classification also considers the weight of each data point. Thus, if a classifier correctly classifies more data points with higher weights, then it gets larger classifier-weight.

Neural network has been created with the intension of mimicking the mechanism of learning and calculation in a human brain. Due to the non-linear separable property as well as the highly parallelizable structure, it has been adopted as one of the most popular classifiers in the area of pattern recognition and classification. Many variations of neural networks have been invented from traditional multi-layer



**Fig. 7.9** A neural network in the form of multi-layer perceptron and a perceptron model of a neuron

perceptron (MLP) to convolution neural networks (CNN), motivated by the visual cortex of human. Even, a spiking neural network can be built following actual communications of neurons by modeling biological electric signals [12].

In the aspect of vision systems, neural networks can also be used as a classifier that decides whether an object is a person/vehicle or not. Multi-layer perceptron, a typical neural network, is shown in Fig. 7.9a, which has three layers: an input layer, an output layer, and a hidden layer. Each layer is composed of conceptual building blocks called a neuron that are represented as a circle in (a); it models biological neurons that are connected with each other with dendrites and synapses as shown in (b), where neuron  $y$  receives electric signals from previous neurons  $x_1, x_2, \dots, x_n$ . Each signal line has a weight value to reflect the significance of it. The model of neuron  $y$  calculates the weighted sum of those signals, and if it is bigger than given threshold value  $b$ , the output is set to one by the transfer function shown in Fig. 7.9b. However, various transfer functions such as sigmoid, linear, Gaussian function can be used. The characteristics of a neural network are determined by how to combine neurons, select transfer functions, and adjust weight values. Similar to SVM and Adaboost, the weights of neurons are adjusted in response to the input training data so that the neural network can properly recognize objects.

### 7.3.4 Post Processing

Post processing is composed of verification/refinement and tracking. Although these steps do not affect the quality of the entire vision system that much, compared with the previous steps, they still have some influence on it. As mentions in

**Sect. 7.2**, they are typically based on control intensive algorithms and fit well with software implementations. Since just conventional processors with super-scalar or VLIW architecture are in general good enough to execute them, this subsection provides only a brief explanation of what is done in those steps. In the verification/refinement step, the judgments made in the object classification step are inspected again to prevent false positive and true negative with some criteria. Examples of the criteria include vertical to horizontal length ratio of walking people or the height of people relative to the distance from camera. Refer to [5] for other techniques. In case of tracking, the object is monitored over time in order to predict future positions and help extracting ROIs in an early processing step for segmentation.

So far, we have overviewed each processing step for pedestrian detection as an example of vision system. In the following sections, hardware architectures for the implementation of those processes are introduced, which include commercial off-the-shelf (COTS) machines, specific hardware such as ASIC, and architectures based on biological recognition as peculiar approaches.

## 7.4 Comparison of COTS Architectures

Ranft et al. [13] studied vision-processing systems having a stereo camera, covering steps from preprocessing to foreground segmentation mentioned in the previous section. They adopted COTS devices including X86-compatible CPU, Tiler many-core, and Nvidia GPU in order to show that COTS devices have performance good enough to run the vision applications at reasonable power consumption. First, they tried two kinds of X86-compatible CPU: two Intel Xeon X5660 with six-cores and four AMD Opteron 6172 with twelve-cores. Thus they have in total 12 and 48 cores, respectively. The architectures have multi-level caches and can execute SIMD instructions. They also support multi-threading based on parallel programming models such as Pthread and OpenMP. The second case they tried is a many-core system that has a TILEPro64 processor with 64 cores made by Tiler. Figure 7.10 shows the cores connected by a network-on-chip with mesh topology; external memory controllers are placed at the top and bottom; other peripherals such as Ethernet interface, serial interface, and PCIe interface are placed on the left and right side. It also supports Pthread and OpenMP. The last architecture used for the comparison is NVidia GeForce GTX 470, a GPU that supports CUDA programming model. It has 14 streaming multi-processors (SMs), each of which has 32 CUDA cores, a register file, caches, and a shared memory as shown in Fig. 7.11 (two SMs are disabled in GeForce GTX 470).

With those architectures, the comparison of performance and power consumption was conducted for disparity map generation as well as motion estimation. Major processing steps include camera undistortion and rectification (see Sect. 7.3.1), cropping and scaling, Difference of Gaussian (DoG) preprocessing, block matching for calculating disparity and static scene optical flow (SSOF)

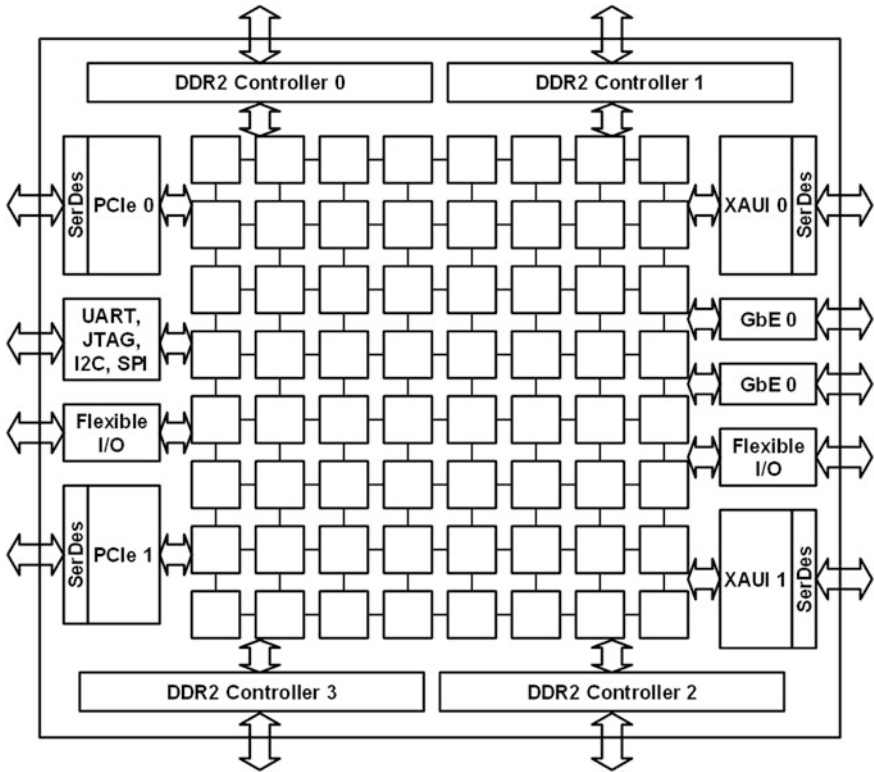


Fig. 7.10 Tiler TILEPro64 processor [20]

estimation, and consistency and texturedness filtering. Those processing steps take  $1344 \times 391$  stereo images as the input. Figure 7.12 shows overall processing steps where each step has one or more groups of threads for parallel processing, and the results are kept in concurrent queues in between steps. Figure 7.13a depicts example images after cropping (top) and DoG pre-processing (bottom); the DoG pre-processing is for removing variability in brightness by subtracting Gaussian blur from the original image. Over the pre-processed images, block matching is applied to obtain disparity map and SSOF. Sum of absolute differences (SAD) is the most popular operation used for block matching. The disparity map is obtained by applying block matching between left image and right image and the SSOF (bottom) is obtained by applying block matching between new image and old image. The consistency and texturedness filtering is to calculate the confidence level of the results by checking consistency of the results in the two images and also by checking texturedness to see if there is a large homogeneous region where SAD operations do not work well. Figure 7.13b shows the corresponding disparity map (top) and the SSOF (bottom) obtained by the overall system. Here, brightness encodes the confidence.

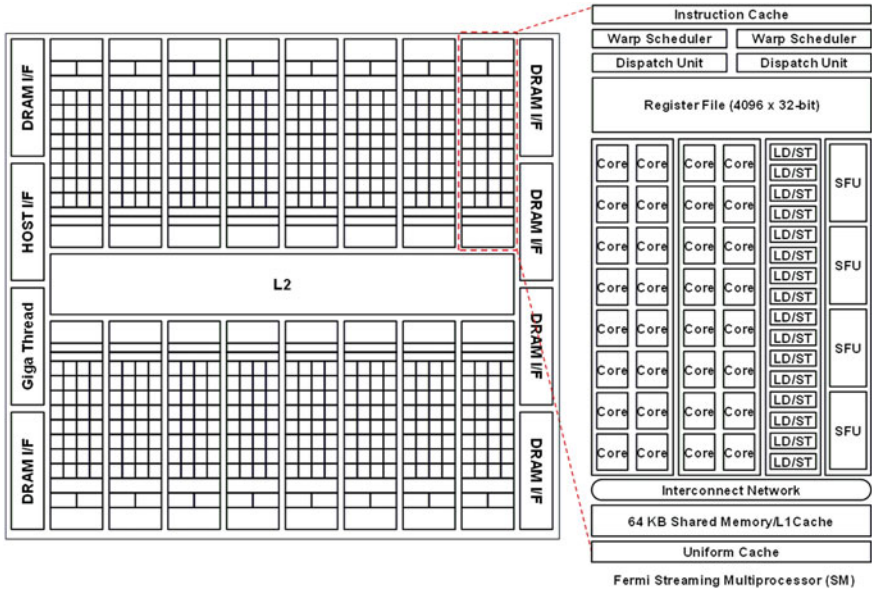


Fig. 7.11 Nvidia GPU Fermi architecture [21]

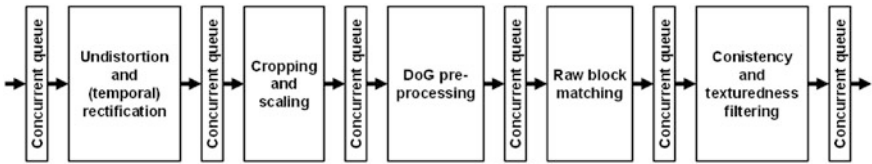


Fig. 7.12 Processing steps [13]

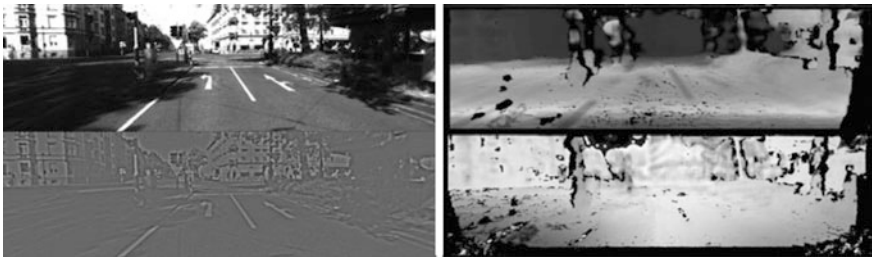


Fig. 7.13 Vision processing. a Image after cropping (top) and DoG pre-processing (bottom). b Results of disparity (top) and SSOF (bottom) [13]



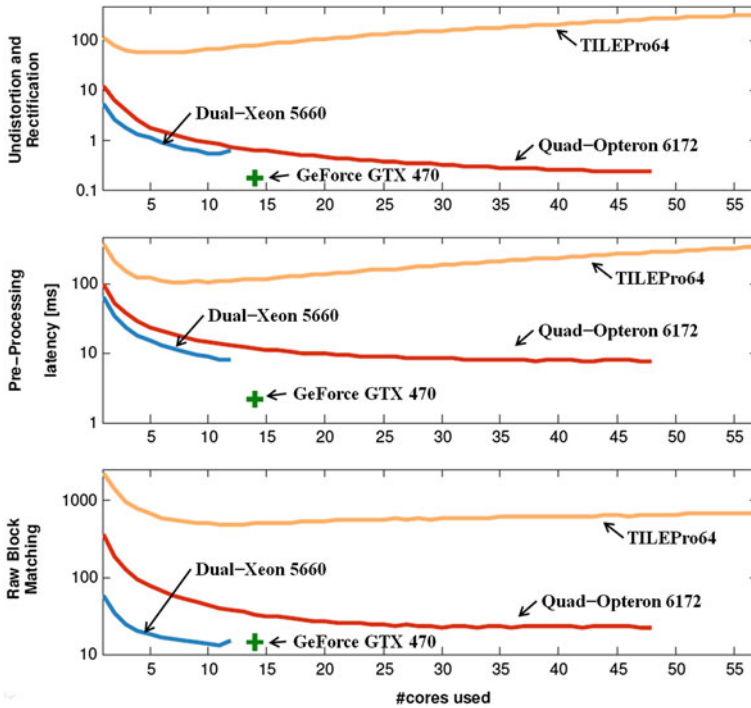
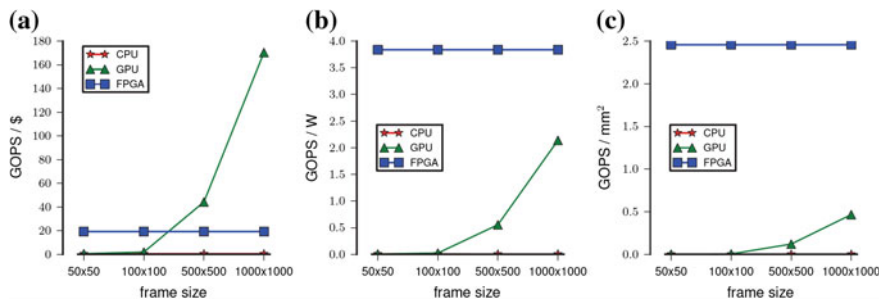


Fig. 7.14 Single steps' processing time about number of cores used [13]

The performance of those systems in undistortion and rectification, pre-processing, and raw block matching steps is shown in Fig. 7.14 where ordinate is for latency in milliseconds and abscissa is for number of used cores. The graph shows that GeForce GTX 470 gives the best performance, which is explained by the fact that the implemented steps are for low-level processing with a lot of data parallelism (refer to Sect. 7.2), which enables GPU (it actually has 448 CUDA cores in 14 SMs) to achieve the best performance. X86 compatible CPUs (Dual-Xeon and Quad-Opteron) outperforms TILEPro64 due to their SIMD capability for data parallelism, which is not in TILEPro64. Dual-Xeon processor outperforms Quad-Opteron for raw block matching due to the additional SIMD instructions supported by Xeon. In terms of energy consumption, GeForce GTX 470 and TILEPro64 performs better than Dual-Xeon and Quad-Opteron.

Another comparison among Intel Xeon E5405, Nvidia Tesla C1060 GPU, and Xilinx Virtex-5 LX330 FPGA was made for a vision system [14]. As shown in Fig. 7.15, FPGA is regarded as the best choice in the aspect of performance for unit power and area, although it is inferior to other devices in terms of flexibility and reusability. It is interesting to see that, as the frame size of vision processing increases, only GPU shows rapid increase in performance for unit cost, power, and area.



**Fig. 7.15** Comparison among CPU, GPU, and FPGA [14]. **a** Performance per USS. **b** Performance per watt. **c** Performance per mm<sup>2</sup>

## 7.5 More on GPU

As shown in the previous section, GPU provides a good opportunity in the design of a vision system. This section investigates more on the use of GPU for pedestrian detection based on histogram of oriented gradients (HoG) and SVM classifier (refer to Sect. 7.3.3). HoG is a popular feature descriptor used for object detection in a vision system [15]. Figure 7.16a shows how HoG feature vector is generated. Briefly speaking, after dividing a detection window (each ROI can be considered as a detection window) into overlapped blocks consisting of multiple cells, HoG is obtained by calculating gradients of each pixel in a cell along specific directions and building a histogram for each cell through orientation binning. To reduce the effect of varying illumination and shadowing across cells, the HoG values are normalized within a block. Then the normalized histogram of each cell becomes a building block for the HoG feature of the entire detection window. The histogram of an overlapped cell can contribute to the feature multiple times but with different normalizations. Thus, the feature vector of the detection window is made by collecting HoGs for all the cells as shown at the bottom of Fig. 7.16a. The vector is used for classifying the image into pedestrian or non-pedestrian group by using SVM. The collected set of HoG data has useful information about the image including the distribution of local intensity gradients and edge directions. SVM classifier takes the information as a feature to discriminate a pedestrian from others. Figure 7.16b shows the overall process of construction of HoG data and classification by SVM.

The tasks for generating HoG data and classifying by SVM have data-parallelism that can be utilized by GPU. For example, the HoG operation belongs to low-level processing mainly for pixel-level computing as mentioned in Sect. 7.2. Classifying an object with trained SVM is mainly conducted by inner product operations. Now, let's see how we can implement the system on a GPU. The ROI image from a camera is first sent to the global memory of the GPU so that the gradient value of each pixel can be calculated by a GPU thread. The threads are allocated as shown in Fig. 7.17a (note that Block(*i*, *j*) in the figure represents a

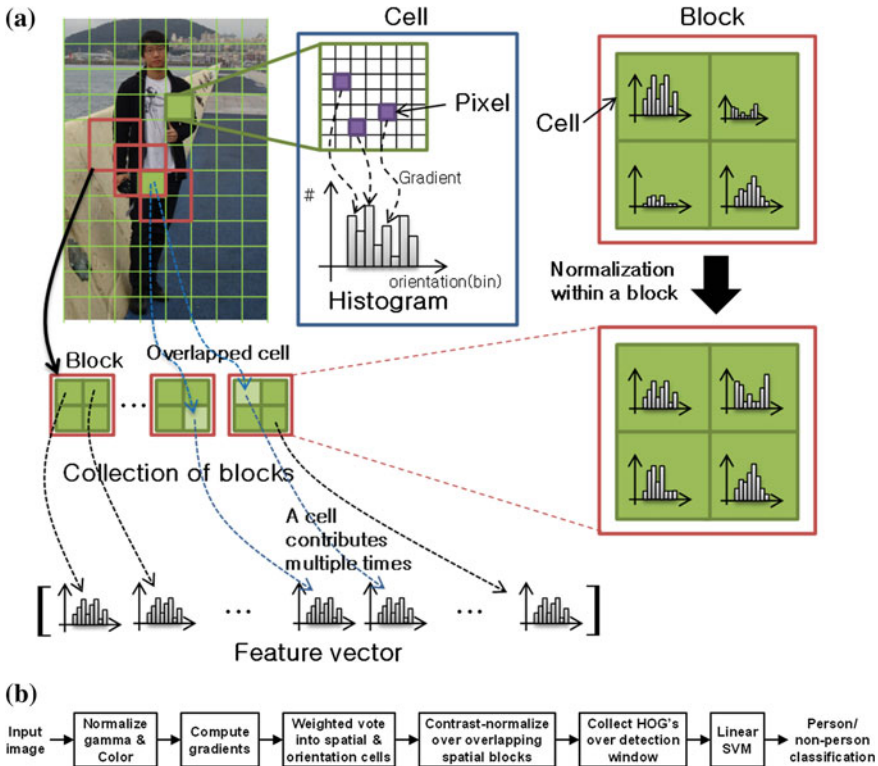


Fig. 7.16 Overview of HoG feature and SVM classification [15]

GPU block), where the gradient data computed by each thread is stored into a space of the shared memory such that multiple threads can access the data. All the processing is done in a parallel manner. Similarly, the histogram for a cell is calculated by multiple threads as shown in Fig. 7.17b. In case of a cell containing  $8 \times 8$  pixels, 64 threads per cell co-work to build a histogram. To normalize the histograms within a block, the threads in a HoG block are mapped to a GPU block because the threads belonging to the same GPU block can communicate fast with each other through data shared among them. If a block consists of four cells and each cell has a histogram having nine bins,  $9 \times 4$  threads are used in the histogram normalization step, since the threads can concurrently operate on the bins with synchronization and locking method.

In the training phase of SVM, the parallelism in the algorithm may not be sufficient for full utilization of the GPU. However, the classifying phase contains plenty of parallel executions. The classifying phase can be divided into two steps: computation of inner products of the coefficient vector of SVM and the HOG vector of entire ROI and the reduction of the inner products. The inner product is calculated by multiple threads that generate product values from SVM coefficients

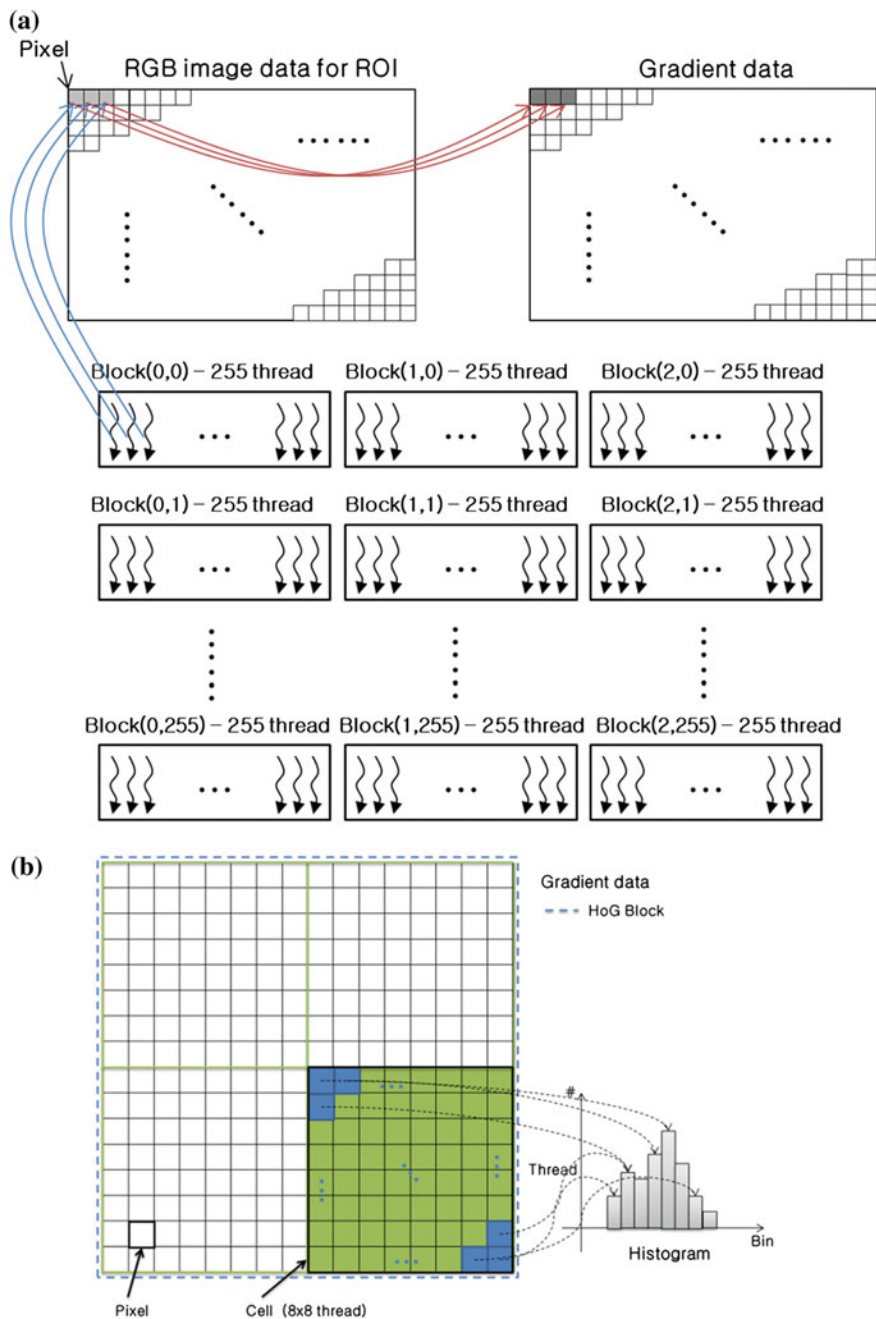
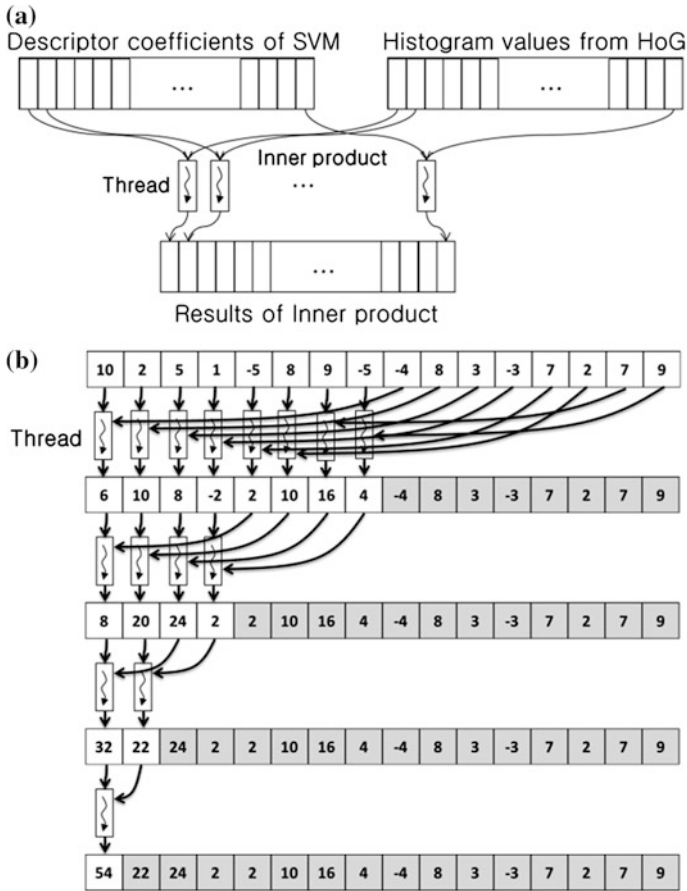


Fig. 7.17 Generating HoG data using GPU. a Gradient and b Histogram computation



**Fig. 7.18** SVM calculation using GPU. **a** Inner product for SVM classification. **b** Data reduction of product values

and HoG values as shown in Fig. 7.18a. Then data reduction (summation) is done over the set of products in order to get the final value for decision as shown in Fig. 7.18b, where the number of data is reduced to a half every step, and in each step, synchronization operations are also executed in order to align the processing status of all the threads.

As a result, the performance improvement obtained by using GPU (GTX 560Ti) was 13 times compared to CPU (Core i7@3.2 GHz) only implementation. Figure 7.19 shows original images from a camera and the human detection results. The detection with HoG and SVM using the GPU processed 36.5 fps on average, while the CPU achieved 2.8 fps.

When using a GPU, a real-time issue can be raised. Because GPU architecture is concentrated on throughput-bound applications, it may not be a good choice for handling the real-time properties in automobile vision system, which was

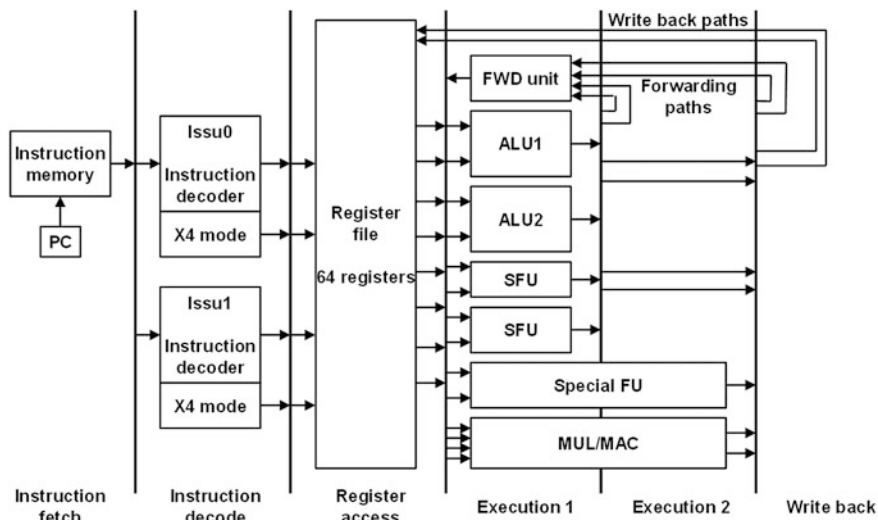


**Fig. 7.19** Results of a GPU-based human detection system

mentioned in Sect. 7.1. Even if a real-time operation system (OS) is used, it is hard to control timing when GPU is involved, since it is not a system programmable device, and even the GPU driver is treated as a black box. The OS cannot directly schedule or control the execution of threads in a GPU; preemptive scheduling is not allowed. There have been attempts to overcome this issue. The approach in [1], for example, groups all GPU-using tasks into a container that models a logical processor. The tasks in the container are then scheduled in first-in-first-out (FIFO) order so that there will be no contention among multiple tasks. Some GPU-using tasks may also require CPU execution time, during which the GPU will be in idle state. To better utilize the GPU by allowing other GPU-using tasks to exploit the idle time, the GPU is treated as a shared resource protected by a real-time mutex.

## 7.6 Comparison of VLIW and COTS Architecture

There is another research that compares VLIW, FPGA, and COTS architectures for a vision system calculating disparity map for stereo vision with video size of VGA ( $640 \times 480$ ) [2]. As the COTS architectures, three GPU architectures and AMD Opteron CPU having 2.8 Ghz frequency have been selected. The three GPU architectures include Nvidia Quadro FX5600, GeForce 7900 GTX, and GeForce 8800 Ultra. As the FPGA, Xilinx Virtex-5 has been selected. As the VLIW



**Fig. 7.20** Pipeline structure of a vector unit that can be replicated to build a VLIW architecture [2]

architecture, the authors have designed a generic VLIW architecture with vision-specific customized features such as scalable control/data-paths, a basic media instruction set architecture, and a flexible register file structure. The VLIW processor is based on a vector unit (VU) that can be replicated to increase the number of parallel executable instructions. Figure 7.20 illustrates the pipeline structure of a VU. In a VU, using identical functional units (FUs) also helps to enhance instruction parallelism, which can magnify the speed of the rank transformation and the semi-global-matching algorithm as explained in the following paragraph.

The architectures mentioned above are compared in terms of performance of computing disparity map, which is an essential component in stereo vision processing. The comparison takes the rank transform and the semi-global matching method, which are known to be very efficient for disparity computation. The processing steps are shown in Fig. 7.21, where rank transformations are applied to the two stereo images, left and right, in the first stage. Rank transformation for a pixel  $\mathbf{p} = [p_x, p_y]^T$  is given by

$$R(\mathbf{p}) = \|\{\mathbf{p}' \in A(\mathbf{p}) \mid I(\mathbf{p}') < I(\mathbf{p})\}\|, \tag{7.1}$$

where  $A(\mathbf{p})$  is the set of pixels in the neighborhood of  $\mathbf{p}$  and  $I(\mathbf{p})$  is the intensity of  $\mathbf{p}$ . It helps alleviate the effect of different background scenes (from the two images) at the boundary of the same object. It also removes the effect of differences in the gain and bias of the two cameras (note that DoG can also be used to remove bias in the intensity or brightness as explained in Sect. 7.4).

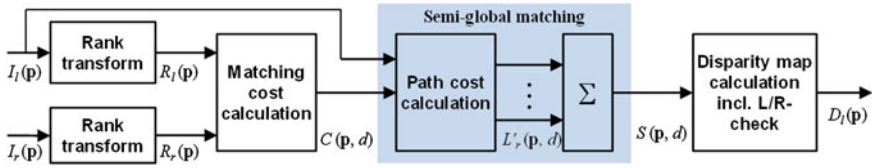


Fig. 7.21 Disparity calculation with rank transformation and semi-global matching [2]

Stereo matching based on semi-global matching is performed on the results of rank transformation. The matching cost for pixel  $\mathbf{p}$  and disparity  $d$  can be given by

$$C(\mathbf{p}, d) = |R_l(x, y) - R_r(x - d, y)| \quad (7.2)$$

Semi-global matching, however, uses the concept of path cost, which is recursively calculated by

$$L_r(\mathbf{p}, d) = C(\mathbf{p}, d) + \min \left[ \begin{array}{l} L_r(\mathbf{p} - \mathbf{r}, d), L_r(\mathbf{p} - \mathbf{r}, d - 1) \\ + P_1, L_r(\mathbf{p} - \mathbf{r}, d + 1) \\ + P_1, \min_i L_r(\mathbf{p} - \mathbf{r}, i) + P_2 \end{array} \right] - \min_l L_r(\mathbf{p} - \mathbf{r}, l) \quad (7.3)$$

where  $\mathbf{r}$  is a vector indicating a path such that  $\mathbf{p} - \mathbf{r}$  represents a previous pixel in that direction. Multiple paths are considered to cover the 2D image as shown in Fig. 7.22. The second term adds the path cost along  $\mathbf{r}$ , while considering a small change or discontinuity in disparity.  $P_1$  and  $P_2$  are penalties for the small change and discontinuity, respectively. The last term is just to prevent the cost from increasing too much along the path due to adding the second term. Then disparity values are obtained by  $\min_d S(\mathbf{p}, d)$ , where  $S(\mathbf{p}, d) = \sum_r L_r(\mathbf{p}, d)$ . Finally, as shown in Fig. 7.21, left/right check is performed to see if the result based on the left image is the same as (or different by only one pixel from) that on the right image and thus to ensure that the disparity value is valid. Figure 7.23b shows the disparity map obtained from the input image in (a), where brighter points in (b) have higher disparity values (closer).

The processing steps in Fig. 7.21 require 3,093 basic operations per pixel, for  $9 \times 9$  rank kernel, 64 pixel disparity range, and 4 paths, and thus require about one billion operations per pair of VGA ( $640 \times 480$ ) images. For those operations, AMD Opteron with SIMD takes 1.8 s to process a  $450 \times 375$  image. Table 7.1 compares the performance of various architectures. Virtex-5 FPGA shows the best performance of 66–167 fps, while the next best one is VLIW architecture that can execute rank transformation (RT) and semi-global matching (SGM) with 30 fps at 400 MHz clock frequency. Nvidia GPU and AMD Opteron CPU show lower



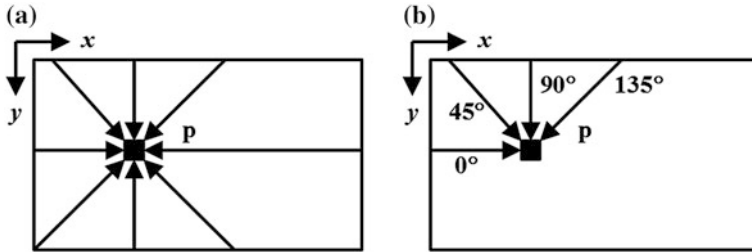


Fig. 7.22 Possible path orientations for eight and four paths [2]

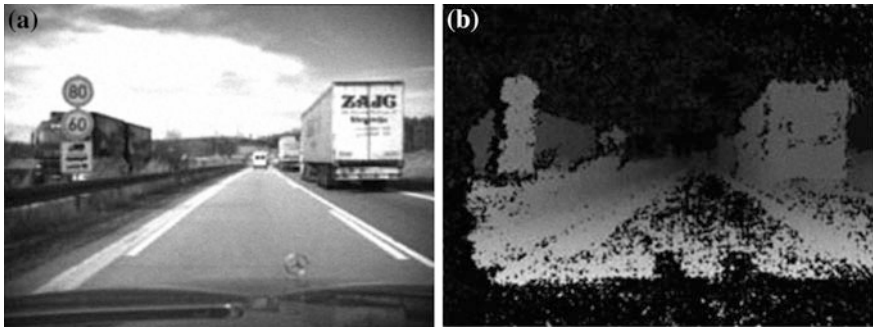


Fig. 7.23 Input image and disparity map [2]

Table 7.1 Performance comparison among different architectures for stereo matching [2]

Architecture	Frame size (Disp. range)	Performance (fps@MHz)	Algorithm
Generic VLIW	640 × 480 (64)	30 fps@400 MHz	RT&SGM(4x)
Nvidia Quadro FX5600	450 × 375 (64)	5.85 fps	BT&SGM(8x)
Nvidia GeForce 7900 GTX	320 × 240 (64)	8 fps	AD&SGM(8x)
Nvidia GeForce 8800 Ultra	320 × 240 (64)	13 fps	HMI&SGM(8x)
AMD Opteron	450 × 375 (64)	0.56 fps@2.8 GHz	SGM
Virtex-5-FPGA	640 × 480 (64)	66–167 fps@133 MHz	RT&SGM(4x)

performance. Note, however, that the comparison was not conducted under the same condition. For example, the GPUs used smaller image sizes but 8 paths for SGM; they used different matching cost functions including Birchfield and Tomasi (BT), absolute difference (AD), and hierarchical mutual information (HMI).

## 7.7 Memory/Bus Requirement

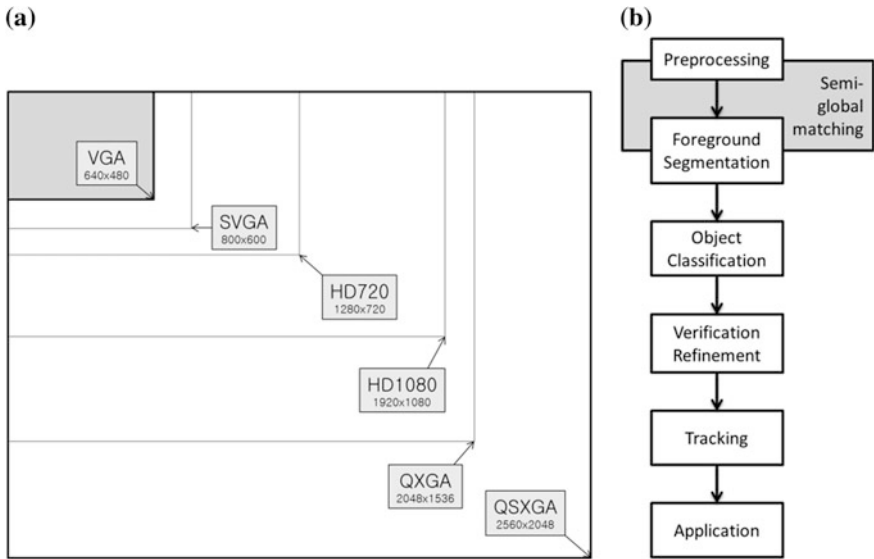
So far, vision systems have been looked into from the viewpoint of computation, considering that they require huge amount of computation (e.g., as mentioned in the previous example, stereo matching requires about one billion operations to compute disparity map for  $640 \times 480$  images). Many researches have also concentrated only on computation to improve the overall performance. However, memory and bus subsystems are also very important to achieve real-time performance for automobile vision system.

Let's take the example of calculating disparity map for stereo images in Sect. 7.6 to see how much memory and bus bandwidths are required in a typical stereo vision system. Suppose the system has 64 disparity range for eight paths for semi-global matching and thus one pixel generates 64 matching cost values. The work in [16] tried to reduce memory footprint for semi-global matching and enhance system performance. Their algorithm requires the memory size of  $w \times h \times 18 + 3 \times w \times d_{\max} + d_{\max}$ , where  $w$  and  $h$  are width and height of the image respectively, and  $d_{\max}$  is disparity range. VGA gray stereo image with 64 disparity claims about 5.7 MBytes ( $640 \times 480 \times 18 + 3 \times 640 \times 64 + 64$ ). It is not difficult to guess that the requirement of memory and bandwidth increases dramatically when high resolution image is required and the entire automobile vision processing is considered (see Fig. 7.24).

Table 7.2 shows that the cache memory size of popular up-to-date embedded processors is much smaller than the memory requirement estimated above for stereo vision. Without any architectural support, the system performance will be degraded significantly due to frequent memory spill. Considering that an automobile vision system should operate as a real-time system and 30 fps is commonly required, such degradation of performance can be critical. In case of bus subsystem, total required bandwidth is roughly  $30 \text{ (fps)} \times 14 \text{ (Mbits)} \times 64 \text{ (disparity range)} \times 2 \text{ (read and write)} = 53.8 \text{ Gbps}$ . A typical 64 bit bus matrix must operate at clock frequency higher than 800 MHz to satisfy the required bandwidth. This example ignores any contention among IP blocks and considers only disparity map calculation, and thus the real situation would get much worse. Therefore, both bus and memory sub-system must be managed together with computation in order to achieve real-time performance required for automobile vision system.

## 7.8 Vision Processors

Because computer vision involves considerable amount of computation compared with other applications, it is challenging to achieve its real-time requirement. There have been many researches for decades in order to overcome the challenges. Most of them focus on low level processing for data parallelism as mentioned in Sect. 7.2, because a typical vision system has huge amount of data to process and



**Fig. 7.24** SGM for VGA image. **a** Display image resolution. **b** Positioning of semi-global matching in the detection processing

**Table 7.2** Cache size of embedded processors

	ARM11	ARM Cortex-A8	ARM Cortex-A9	Qualcomm Scorpion	Qualcomm Krait
L1 cache	Varying, typically 16 + 16 kB	32 + 32 kB	32 + 32 kB	32 + 32 kB	16 + 16 kB 4-way set associative
L2 cache	Varying, typically none	512 kB	1 MB	256 kB (Single-core) 512 kB (Dual-core)	1 MB 8-way set associative (Dual-core) 2 MB (Quad-core)

thus requires massively parallel processing. This section introduces two such researches together with the architectures proposed by them.

IMAP-CE, an implementation of IMAP architecture, was designed as highly parallel SIMD architecture in order to utilize the parallelism of vision processing [4]. As shown in Fig. 7.25, the overall architecture is composed of a control processor (CP) and an array of 128 PEs. The array has 16 PE groups, each of which has eight PE units. CP is a 16-bit RISC that has a 6-stage pipeline, 32 KB instruction cache, and 2 KB data cache. Each PE has 4-way VLIW architecture based on 8-bit RISC that supports 3-stage pipeline, 24 general-purpose registers with 8-bit width, an 8-bit ALU, an  $8 \times 8$ -bit multiplier, a load-store unit (LSU), a reduction unit (RDU) for communication with CP, and a communication unit between PEs. Each PE is attached to a locally addressable 2 KB on-chip RAM

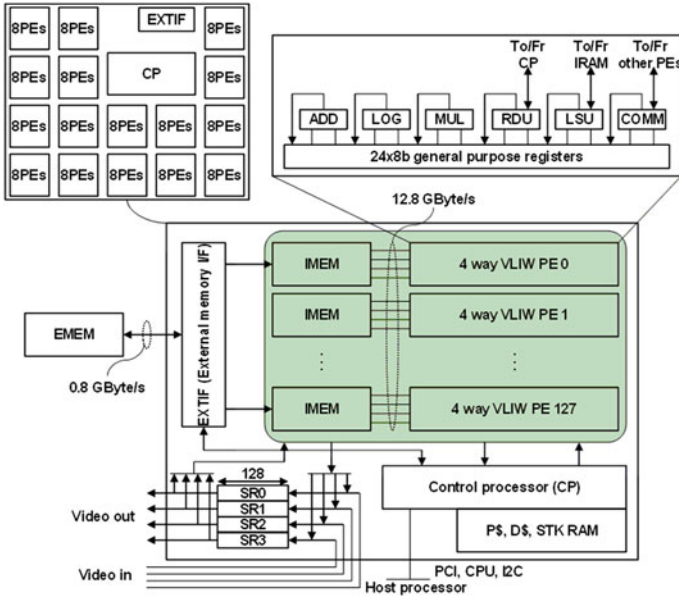


Fig. 7.25 IMAP-CE block diagram and die-photo [4]

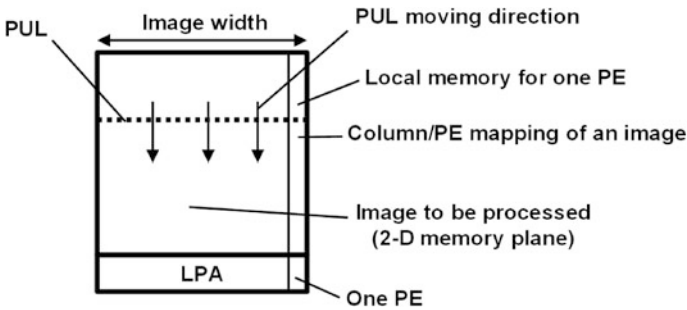


Fig. 7.26 IMAP working model [4]

(IMEM: internal memory) for data caching and can communicate with other PEs through a bus in a ring topology, thus named as integrated memory array processor (IMAP). It also has an external interface (EXTIF) containing DMA block to transfer data between IMEM and EMEM (external memory). The chip was fabricated using 0.18  $\mu\text{m}$  CMOS process with  $11 \times 11 \text{ mm}^2$  die area containing 32.7 M transistors and consumed average power of about 2 W.

Figure 7.26 shows an example of processing image data on IMAP, where each column corresponds to a local memory of each PE and thus each row of the

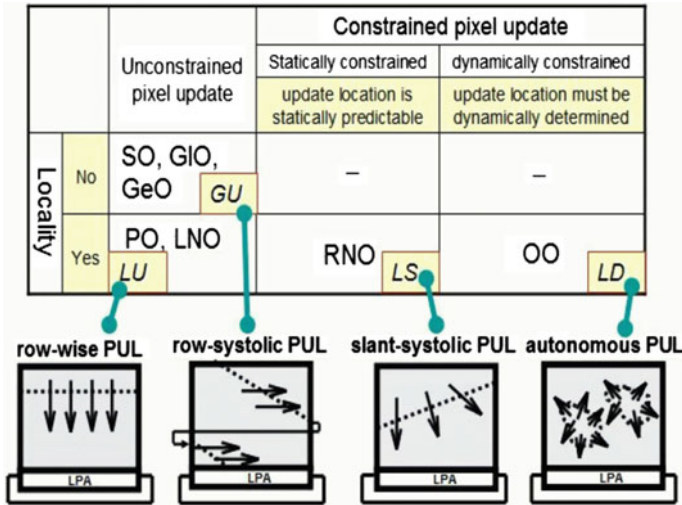


Fig. 7.27 Correspondence between pixel operation groups and parallelizing techniques [4]

column corresponds to an address of the local memory. The example assumes that the image width is the same as the number of PEs so that each PE can process one pixel on a horizontal line of the image (each local memory stores the source and destination pixel data). The dotted line in the figure is called pixel-updating line (PUL), which represents the set of pixels processed in parallel. As explained in Sect. 7.2, the low-level processing for computer vision involves pixel operations that can be grouped into Point Operation (PO), Local Neighborhood Operation (LNO), Global Operation (GIO), Statistical Operation (SO), Geometrical Operation (GeO), Recursive Neighborhood Operation (RNO), and Object Operation (OO). IMAP can effectively manage such pixel operations with the notion of PUL including row-wise PUL, row-systolic PUL, slant-systolic PUL, and autonomous PUL as shown in Fig. 7.27. The experimental result in Fig. 7.28 shows that IMAP-CE operating at 100 MHz is about three times faster than Intel P4 general purpose processor operating at 2.4 GHz clock frequency. This comparison was conducted under  $128 \times 240$  source image with 8 bits per pixel.

Another architecture proposed in [17] is based on multi-SIMD, which exploits the parallelism in vision processing using SIMD arrays. As shown in Fig. 7.29, the architecture consists of two SIMD modules sharing a memory; one consists of a two-dimensional (2D) array of  $16 \times 16$  PEs and the other has an one-dimensional (1D) array of 16 PEs. There are also separate communication channels between those two modules and an external CPU. The 1D SIMD module calculates column or row data of an image in parallel, while the 2D SIMD module computes 2D pixel data of the image in parallel. The PEs in the 2D SIMD module are connected in a mesh topology, and thus each PE can communicate with four neighboring PEs.

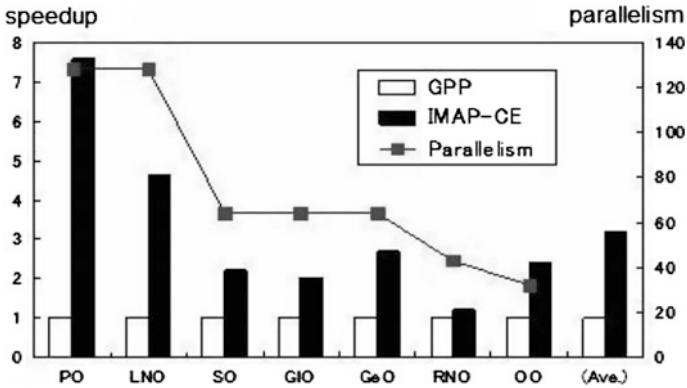


Fig. 7.28 Performance compared with 2.4 GHz Intel P4 [4]

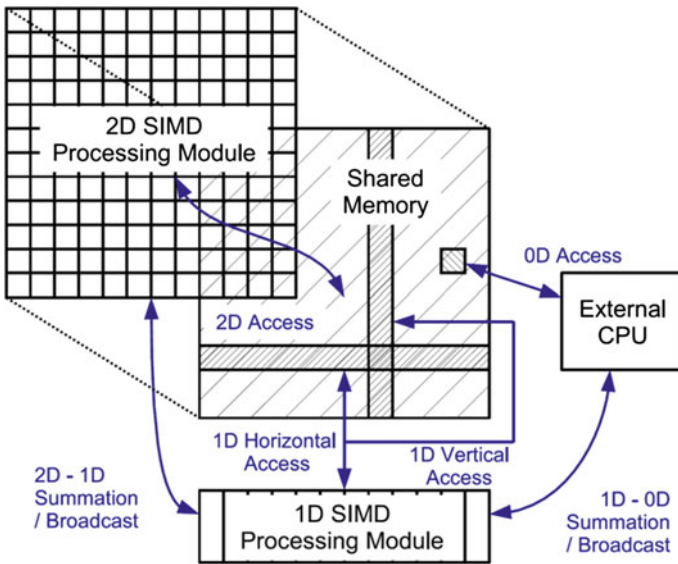


Fig. 7.29 Conceptual Image of Multi-SIMD [17]

Although 1D SIMD modules cannot communicate with its neighbor, they are designed to control longer data words. This architecture supports summation and broadcast; the former is to accumulate data using the 2D SIMD and send the result to the 1D SIMD; the latter is to distribute the data treated by the 1D SIMD into the 2D SIMD. Through this method, the Multi-SIMD architecture can execute complicated operations, which conventional SIMD architecture cannot do easily. In case of shared memory, the architecture has four access patterns: 2D pixel access from the 2D module, 1D column access from the 1D module, 1D row access from

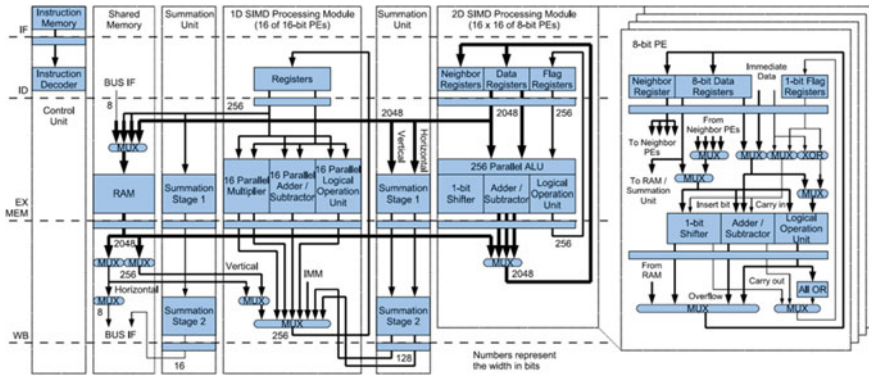


Fig. 7.30 Data path of the vision processor [17]

the 1D module, and 0D sequential access from the external CPU. Figure 7.30 shows the data path of the proposed architecture. Both 1D and 2D SIMD modules can execute two-stage pipeline, while the summation logic has three-stage pipeline paths. The shard memory consists of  $16 \times 16$  memory elements, each of which can store 4 Kbits and perform 8-bit access at once. Each PE of the 2D module is composed of 8-bit general-purpose registers, 1-bit flag registers, an adder, a subtractor, a 1-bit shifter, and a bitwise logical operation unit. Each PE in the 1D module has 16-bit registers, an adder, a subtractor, a multiplier, and a bitwise logical operation unit. Each summation unit is implemented with tree-structured adders.

### 7.9 Yet Another Approach

In the aspect of conventional computation, it is true that computer vision has been regarded as a difficult area even until these days. However, biological systems such as human brain not only achieve miraculous recognition ratio compared with artificial vision systems, but also recognize complex objects instantly. Simply, they are accurate and fast; they even consume very low power. With this observation, some researches have focused on a direction to mimic what human brain does. Figure 7.31 shows a visual cortex system and its building block, neuron, of a human brain. As illustrated in Sect. 7.3.3, a neuron consists of neuron cell body, dendrites, and axon. Each neuron connects to other neurons through a synapse that is a junction between a dendrite and an axon tip with a tiny gap. Visual recognition in the brain involves two parts called the dorsal stream and the ventral stream. The dorsal stream is associated with motion, object locations, and control of the eyes, while the ventral stream is concerned with object recognition and representation.

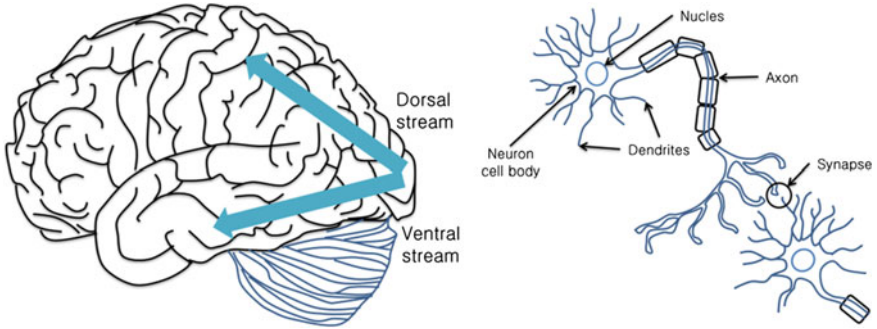


Fig. 7.31 Visual cortex and neurons of human brain

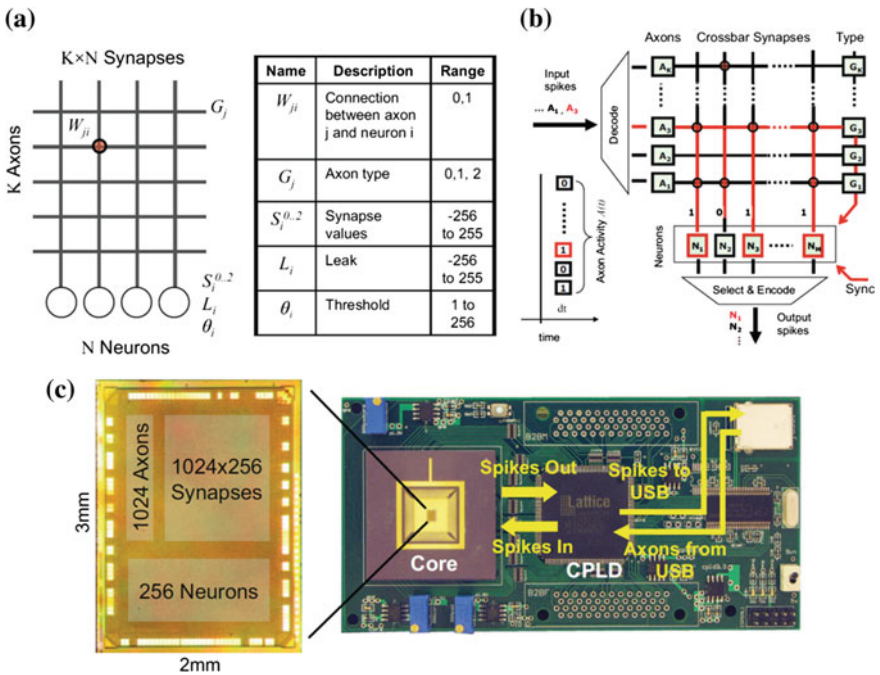


Fig. 7.32 Neurosynaptic core architecture and its implementation [18]

The research presented in [18] suggested a brain-inspired architecture called digital neurosynaptic core that achieves real-time response, low power consumption, and compact chip size. The chip has 256 integrate-and-fire neurons, 1,024 axons, and a  $1,024 \times 256$  SRAM crossbar memory for synapses to connect neurons, while keeping flexibility to configure in terms of neuron parameter, axon types, and synapse states. The research also suggested an abstract neural



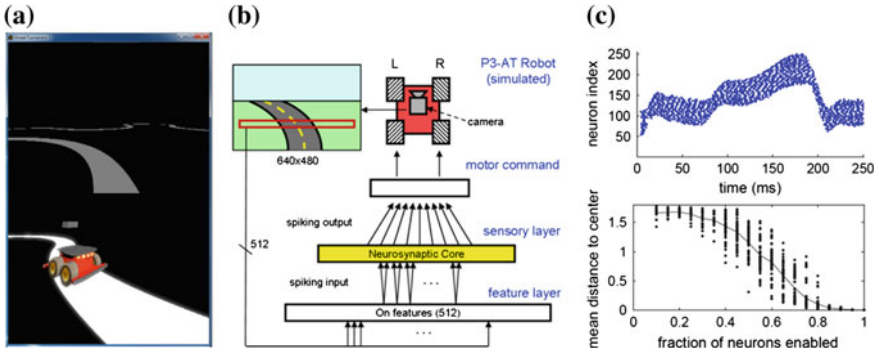


Fig. 7.33 Application for autonomous virtual robot driver [18]

programming model to execute applications in the area of control system, classification, and computer vision. As shown in Fig. 7.32a, axons are modeled by rows while dendrites are modeled by columns; synapses are modeled by row-column junction points; neurons illustrated as circles in the bottom receive signals from the dendrites. Figure 7.32b shows a snapshot of operation right after the firing of axon 3, which activates the corresponding row wire. The row has three synapses represented by circles, and thus  $N_1$ ,  $N_3$ , and  $N_M$  are eventually updated. Figure 7.32c shows the photo of the neurosynaptic core implemented on a  $2 \times 3$  mm die and the test board.

Figure 7.33 shows an autonomous virtual robot driver application using the neurosynaptic core, where the chip receives input data from scenery and generates commands to move the robot as an action of the driver. For simplicity, the scenery inputs in the virtual environment are generated in the host machine (Fig. 7.33a). 512 feature data extracted from the current scene are mapped to the 1,024 axons (half excitatory and half inhibitory), and then neurons produce spiking outputs (sensory layer in Fig. 7.33b). Then the motor command processing block collects the outputs and steers the robot toward left or right. The top of Fig. 7.33c shows spike activity of neurons during driving; the bottom depicts average distance from the center of the driving track.

Another application using the neurosynaptic core is to recognize a handwritten numeric digit composed of  $22 \times 22$  pixel units. It uses Restricted Boltzmann Machine (RBM) as the classifier in order to recognize digit numbers. Using 256 hidden units, each pixel unit is mapped to two axons: excitatory and inhibitory. Figure 7.34 depicts that (a) the digit value is recognized as three by the neurosynaptic core using 968 axons and 256 neurons, and (b) the average accuracy for digit recognition is 89 %. Actually, it achieved 94 % accuracy when classifying 10,000 images provided by MNIST dataset [19].

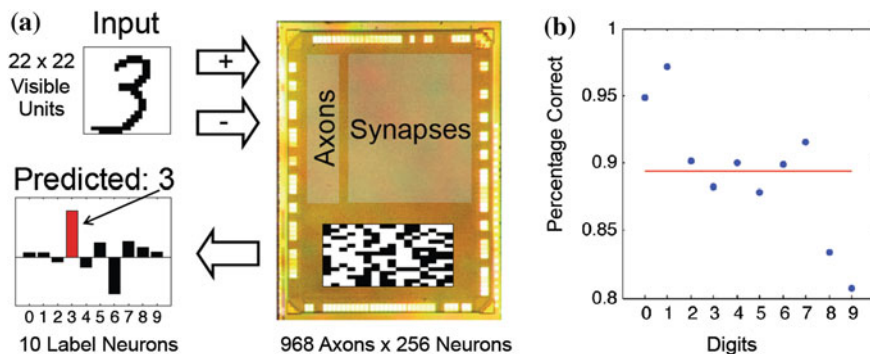


Fig. 7.34 Application for visual digit recognition [18]

## 7.10 Conclusions

In an Advanced Driver Assistance System (ADAS), vision is one of the most important key technologies, because most important information that is necessary when driving a car can be generated from visible scenes such as pedestrians, vehicles, lanes, traffic signs, etc. Since automobile vision systems are directly coupled with human safety, its design has stringent requirements on various constraints such as high accuracy, real-time performance, low power, area, etc. The challenges are doubled because of huge amount of data to compute and transfer.

In summary, this chapter has presented key characteristics of vision systems and the architectural considerations for them. It has taken pedestrian detection as an example for the application and reviewed the processing steps comprising the detection flow including preprocessing, foreground segmentation, object classification, verification, and tracking. Several system architectures for implementing the vision-processing steps have been introduced. They include conventional COTS machines, vision-specific architectures, and even a human-brain-inspired system. Those architectures have been studied mostly to overcome the problem of huge amount of computation required to realize high accuracy as well as real-time property. However, it should be pointed out that bus and memory subsystems for communication and storage are also important to further enhance the performance of such systems.

## References

1. G.A. Elliott, J.H. Anderson, Real-world constraints of GPUs in real-time systems, in *Proceedings of IEEE 17th International Conference on Embedded and Real-Time Computing Systems and Applications (RTCSA)* (2011), pp. 48–54
2. G. Paya-Vaya, J. Martin-Langerwerf, C. Banz, F. Giesemann, P. Pirsch, and H. Blume, VLIW architecture optimization for an efficient computation of stereoscopic video

- applications, in *Proceedings of International Conference on Green Circuits and Systems* (2010), pp. 457–462
3. W. Nanjian, High speed CMOS vision chips, in *Proceedings of IEEE 54th International Midwest Symposium on Circuits and Systems* (2011), pp. 1–4
  4. S. Kyo, S. Okazaki, T. Arai, An integrated memory array processor architecture for embedded image recognition systems, in *Proceedings of 32nd International Symposium on Computer Architecture* (2005), pp. 134–145
  5. D. Geronimo, A.M. Lopez, A.D. Sappa, T. Graf, Survey of pedestrian detection for advanced driver assistance systems. *IEEE Trans. Pattern Anal. Mach. Intell.* **32**, 1239–1258 (2010)
  6. R. Labayrade, D. Aubert, J. Tarel, Real time obstacle detection in stereovision on non flat road geometry through ‘V-disparity’ representation, in *Proceedings of IEEE Intelligent Vehicles Symposium*, pp. 646–651, Versailles, France, June 2002
  7. A. Sappa, F. Dornaika, D. Ponsa, D. Gerónimo, A. López, An efficient approach to onboard stereo vision system pose estimation. *IEEE Trans. Intell. Transp. Syst.* **9**(3), 476–490 (2008)
  8. D. Gerónimo, A. Sappa, A. López, D. Ponsa, Adaptive image sampling and windows classification for on-board pedestrian detection, in *Proceedings of International Conference on Computer Vision System*, Bielefeld, Germany, 2007
  9. V. Vapnik, *Estimation of Dependences Based on Empirical Data (in Russian)*, Nauka, Moscow, 1979 (*English Translation*) (Springer, New York, 1982)
  10. C. Cortes, V. Vapnik, Support-vector networks. *Mach. Learn.* **20**(3), 273–297 (1995)
  11. Y. Freund, Robert E. Schapire, A decision-theoretic generalization of on-line learning and an application to boosting. *J. Comput. Syst. Sci.* **55**, 119–139 (1997)
  12. J. Misra, I. Saha, Artificial neural networks in hardware: a survey of two decades of progress. *Neurocomputing* **74**, 239–255 (2010)
  13. B. Ranft, T. Schoenwald, B. Kitt, Parallel matching-based estimation—a case study on three different hardware architectures, in *Proceedings of IEEE Intelligent Vehicles Symposium* (2011), pp. 1060–1067
  14. V. Sriram, D. Cox, T. Kuen Hung, W. Luk, Towards an embedded biologically-inspired machine vision processor, in *Proceedings of International Conference on Field-Programmable Technology* (2010), pp. 273–278
  15. N. Dalal, B. Triggs, Histograms of oriented gradients for human detection, in *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)* (2005), pp. 886–893
  16. H. Hirschmüller, I. Ernst, M. Buder, Memory efficient semi-global matching. *Int. Ann. Photogramm. Remote Sens.* **3**, 371–376 (2012)
  17. K. Yamaguchi, Y. Watanabe, T. Komuro, M. Ishikawa, Design of a massively parallel vision processor based on multi-SIMD architecture, in *Proceedings of IEEE International Symposium on Circuits and Systems* (2007), pp. 3498–3501
  18. J.V. Arthur, P.A. Merolla, F. Akopyan, R. Alvarez, A. Cassidy, S. Chandra, S.K. Esser, N. Imam, W. Risk, D.B.D. Rubin, R. Manohar, D.S. Modha, Building block of a programmable neuromorphic substrate: a digital neurosynaptic core, in *Proceedings of International Joint Conference on Neural Networks* (2012), pp. 1–8
  19. Y. LeCun, C. Cortes, The MNIST database of handwritten digits, <http://yann.lecun.com/exdb/mnist/>
  20. Tiler, TILEPro64 Processor, [http://www.tilera.com/sites/default/files/productbriefs/TILEPro64\\_Processor\\_PB019\\_v4.pdf](http://www.tilera.com/sites/default/files/productbriefs/TILEPro64_Processor_PB019_v4.pdf)
  21. NVIDIA, NVIDIA’s next generation CUDA compute architecture: Fermi, [http://www.nvidia.com/content/PDF/fermi\\_white\\_papers/NVIDIA\\_Fermi\\_Compute\\_Architecture\\_Whitepaper.pdf](http://www.nvidia.com/content/PDF/fermi_white_papers/NVIDIA_Fermi_Compute_Architecture_Whitepaper.pdf)

# Chapter 8

## Hardware Accelerator for Feature Point Detection and Matching

Jun-Seok Park and Lee-Sup Kim

**Abstract** Recently, many vehicle manufacturers have adopted a vision processing based driver assistance system (DAS) for safety. Since vehicles cannot allow any crash or collision even when running very fast, all vision algorithms should be operated in real-time without any error. Since many algorithms such as object recognition/tracking, image matching, and simultaneous localization and mapping (SLAM) are based on the interest point detection and matching algorithm, interest point detection and matching algorithms should be accelerated to provide result data for overall vision system in real-time. However, they are one of the most compute-intensive operations in DAS and even the state-of-the-art hardwired accelerators hardly achieve 60 frames per second (fps) only in VGA resolution (640 × 480). They suffer from tremendous hardware overhead because they are implemented based on heterogeneous many-core system. To overcome these limitations, we aims to implement hardware which achieve more than 90 frames per second in full HD resolution (1080p) only with 30 % of logic gates compared to the state-of-the-art object recognition processors. In this chapter, we introduces three techniques to design this hardware : (1) Joint algorithm-architecture optimizations for exploiting bit-level parallelism, (2) A low-power unified hardware platform for interest point detection and matching, and (3) scalable hardware architecture.

---

J.-S. Park (✉) · L.-S. Kim  
School of EE, KAIST, Daejeon, South Korea  
e-mail: jspark@mvlsi.kaist.ac.kr

L.-S. Kim  
e-mail: lskim@ee.kaist.ac.kr

## 8.1 Introduction to Interest Point Detection and Matching Hardware

Compute-intensive vision algorithms and applications are widely studied to improve robustness and operation speed. At the same time, people are becoming more familiar with mobile devices, which have three major features: (1) integrated camera, (2) various embedded vision algorithms according to the enhancement in process technology, and (3) easiness of implementing vision applications based on programmable hardware. As the demand for high-quality visual contents processing on mobile devices increases, computer vision algorithms such as face detection and augmented reality (AR) become essential in order to design instinctive human interactive solution. There are several complex vision algorithms focused on mobile environment, such as marker-less AR, object recognition and object tracking introduced for mobile devices [1, 2].

Recently, vision algorithms and applications have been further applied and embedded in vehicles, as well as mobile devices. Many vehicle manufacturers have adopted a driver assistance system (DAS) based on vision algorithms for safety [3, 4]. Since vehicles cannot allow any crash or collision, accurate distance between the vehicle and other obstacles should be calculated correctly in real-time even when running very fast. However, 3D depth estimation and object (other vehicles or obstacles) detection are one of the most complex vision algorithms.

Although the high-end PC platform includes large amount of computing resources, running complex vision algorithms on PC platform in real-time is difficult. Thus, accuracy/performance loss is inevitable on embedded environment because of limitations of embedded systems such as implementation area and battery lifetime. Many researchers tried to accelerate various vision algorithms by implementing hardwired logics. Interest point detection and matching are basic and one of the most compute-intensive operations in general vision tasks such as object recognition/tracking, image matching/stitching, and simultaneous localization and mapping (SLAM). Object recognition based on scale invariant feature transform (SIFT) is stable and robust from various noises, so it is widely used to extract and match feature points. However, real-time processing is difficult because of tremendous amounts of computations [5]. Many object recognition processors are proposed to resolve this problem [6, 7], but they achieve real-time performance (60 frames per second (fps)) only in VGA resolution ( $640 \times 480$ ). In this work, features from accelerated segment test (FAST) [8] and binary robust independent elementary features (BRIEF) algorithms [9] are used for interest point detection and matching respectively as shown in Fig. 8.1. These two algorithms are widely used because they achieve better performance than SIFT in general. Although these algorithms can completely replace SIFT with good performance and robustness [2], they are optimized more for PC platform. For example, the decision tree in FAST is not appropriate for mobile environment, because the decision tree requires large on-chip memory resources. Furthermore, it requires many branch (sequential) operations while degrading processing performance.

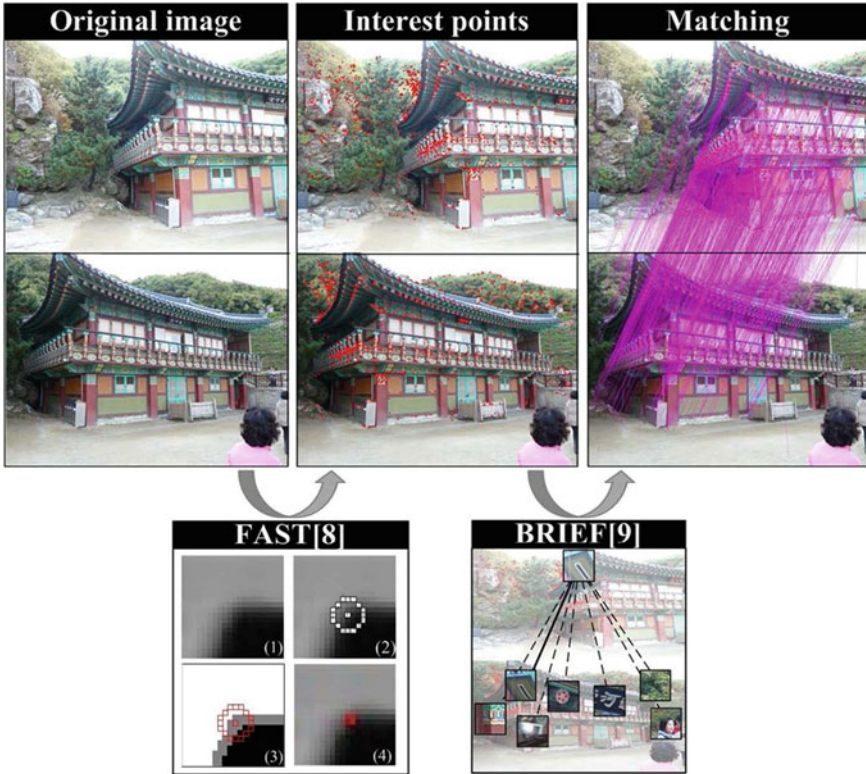


Fig. 8.1 General recognition process based on feature detection and matching

To meet the requirement of modern vision based DAS, a practical solution for real-time processing of vision tasks with high-resolution video stream should be designed. In [10], we proposed an interest point detector, which accelerates the segment test of FAST. The system is implemented as a small IP with joint algorithm-architecture optimization techniques in order to exploit bit-level parallelism. It has  $8\times$  higher throughput than state-of-the-art object recognition processors which are implemented based on many-core system [6, 7]. Furthermore, the unified hardware platform for interest point detection and matching is proposed to improve area-efficiency and workload balancing.

### 8.2 Interest Point Detection Hardware with Joint Algorithm-Architecture Optimization

Since many algorithms are optimized in PC platform not in dedicated hardware, it is hard to guarantee that an algorithm, which operates well in PC platform, can be implemented in hardware efficiently. To implement it in hardware, designer should

consider about parallelism and some overhead factors such as area and power and modify it. In this chapter, we introduce interest point detection and matching algorithm and hardware accelerators. Then, we introduce interest point detection hardware which is optimized by joint algorithm-architecture optimization process.

## ***8.2.1 Introduction to Algorithm***

### **8.2.1.1 Interest Point Detection Algorithm**

Interest points are sparsely distributed but extracted repeatedly on 2D images of an object. For example, if you have some pictures taken a book, each corner of the book would be an interest point. Since corners of the book are detected repeatedly with a plurality of images and these points represent the book, it is possible to express a large image of the object using very little data for interest points. In general, the number of interest point is less than 1 % of the number of the whole pixels on the image.

SIFT is one of the most popular interest point detectors [5]. Scale invariance is guaranteed in SIFT by convolving the image with a Difference of Gaussian (DoG) kernel at multiple scales. SIFT uses a DoG kernel instead of a Laplacian of a Gaussian (LoG) kernel because DoG is a good approximation of LoG and much faster.

A feature (interest point) in SIFT is defined as a local maximum or minimum of DoG that occurs in multiple scales of images. Because it is a blob detector, the each feature point is not located on the edge or corner of an object in general. This is done by comparing each pixel in the DoG images with its 26 neighbors (eight at the same scale, two sets of nine neighbor pixels in upper and lower scales). Because features are extracted from multiple scales of images, feature is scale-invariant. Furthermore, since high-frequency noise is removed during image scaling, it is robust to noise. To make a rotation invariant feature, the descriptor of an interest point is recorded according to the direction of the signal which has maximum intensity.

The extracted SIFT feature is accurate, stable, and scale-/rotation-invariant. However, it requires data-intensive operations because the number of Gaussian convolved images are much more than the other interest point detectors. Obtaining local maximum or minimum also has heavy computational overhead by the same reason.

Speed up robust features (SURF) is an enhanced feature extraction algorithm in terms of processing speed compared to SIFT [11]. It uses box filters and utilizes a fast Hessian operator to approximate the Hessian-Laplace detector. Since box filters are computed by a few add and subtract operations based on integral image, it is faster than the SIFT feature detector which is based on Gaussian convolution operations. Although fast Hessian requires integral images which demand extra

storage and computation, it achieves over  $3\times$  of performance gain compared to DoG.

Edward Rosten proposes a novel interest point detection method, FAST [8]. The segment test of FAST criterion operates as shown in Fig. 8.1. It requires a circle of 16 pixels around the corner candidate pixel  $p$  with intensity  $I_p$ . The status of each pixel on the circle is determined as white, black, or gray. The white, black, and gray states mean that the target pixel is brighter than  $I_p + \text{threshold } t$ , darker than  $I_p - t$ , and otherwise, respectively. The segment test determines  $p$  as a corner when a set of  $n$  consecutive pixels with all white or all black states exists on the circle. Rosten uses an optimized decision tree trained by ID3 algorithm to increase performance of the detector in a CPU-based system.

According to Rosten et al. [8], FAST-9 (FAST with  $n = 9$ ) is the best in terms of processing speed and repeatability among various feature detectors such as Harris, SUSAN, or DoG. Furthermore, FAST-9 achieves more than  $15\times$  performance enhancement compared to the others. It means that FAST algorithm detects interest points with low computational overhead. However, FAST is not appropriate in scale-invariant environments, because it is an algorithm based on a single-scaled image.

Adaptive and generic accelerated segment test (AGAST), a faster and more robust feature detection method based on segment test in FAST, is proposed [12]. The main difference between AGAST and FAST is the decision tree construction method [13].

### 8.2.1.2 Interest Point Matching Algorithm

Interest point matching is a process which finds the same interest points among a set of interest points and group them. Since the relationship among matched interest points is used for image processing, interest point extraction and matching algorithms are used as a pair in most cases. Interest point matching determines whether a pair of the interest points is same or not according to the similarity of the pixel values around the interest points. The pixel data around the interest points, called an image patch, should be stored with its own interest points because the image patch is loaded whenever the interest point are compared with another point. But the comparison based on the raw pixel data of image patches causes two critical problems. Firstly, the image patches require a lot of memory. Suppose that 1 % of overall pixels in image are interest points and the size of an image patch is  $16 \times 16$ . It means that  $2.5\times$  larger memory space is necessary for storing image patch than for the original image. Secondly, if the raw data is used for interest point matching, redundant operations are inevitable. General interest point matching algorithms transform the pixel data to another data format such as binary vector in order to calculate the similarity between interest points. Each interest points are compared with more than one point in interest matching process, redundant transformation occurs. Descriptor replaces the image patch to resolve these problems. It is an encoded data of the image patches. Many researchers find a



way to make efficient descriptors which have low memory requirement, good matching result, and low operation overhead of encoding and comparison.

The SIFT descriptor encodes the distribution of pixel intensities around the feature instead of raw pixel intensities [5]. In general, it offers higher accuracy than previous approaches such as sum of absolute differences (SAD), sum of squared differences (SSD), or zero-mean normalized cross-correlation (ZNCC). But it is slow because SIFT requires a 128-dimensional vector (512 byte) for its descriptor size. The descriptor is important especially in complex applications such as SLAM and object recognition because they should handle many feature points in real-time.

The SURF descriptor is one of the widely used descriptors, like the SIFT descriptor [11]. It relies on local gradient histograms. The SURF descriptor achieves similar or better accuracy than the SIFT descriptor, enhancing processing speed. In order to speed up the performance, it uses integral images.

Both SIFT and SURF descriptors are created by sampling image gradient magnitudes and orientations around each interest point and accumulating them into orientation histograms. In order to achieve rotation invariance, the orientation histograms are rotated relative to local image gradient directions.

Since SIFT and SURF descriptors use 128- and 64-dimensional vectors with floating point values, they require more than 256 bytes for one descriptor. Although many researchers focused on descriptor size reduction [14–16], a substantial amount of computations is additionally required to reduce the size of long descriptor. In [9], Calonder proposed a new method (BRIEF) to build a descriptor based on the intensity comparison method in [17]. This method does not create a long floating point vector because it directly builds short descriptors by comparing the intensity of pairs of points.

An  $n_d$ -bit descriptor is generated by  $n_d$  intensity tests. Each test compares the intensity of pixels located at  $(x_i, y_i)$  in a small image, which is called as a patch, and encodes the result on  $i$ th bit on the descriptor. Choosing a set of  $n_d$   $(x, y)$  location pairs uniquely defines a set of binary tests. 256 and 512 are widely used for  $n_d$  because the descriptor yields near optimal results when the  $n_d$  is larger than 256.

Similarity between descriptors can be measured based on the Hamming distance between corresponding binary strings. This is very fast because the Hamming distance can be computed efficiently using a bitwise XOR operation followed by a bit count [9].

BRIEF achieves  $4\times$ ,  $23\times$  performance improvements in generating and matching descriptors, respectively compared to the previous state-of-the-art methods, while reducing the size of descriptor less than one-fourth of the SURF descriptor.

CHoG is a SIFT compression scheme for computing low bit-rate feature descriptor [18]. It divides a patch into soft log polar spatial bins and the gradient histogram in each spatial bin is captured directly into the descriptor. For compression the descriptor, CHoG quantizes the gradient histogram in each cell individually and maps it to an index. The indices are encoded with fixed length.

The 60-bit CHoG descriptor matches the performance of the uncompressed 1024-bit SIFT descriptor. CHoG is one of the widely used interest point descriptors because of its descriptor size and robustness [19].

### 8.2.1.3 Comparison to Robustness

Since SURF-SURF achieves reasonable performance on PC and it is one of the most robust algorithm, it is considered as de facto standard in interest point detection and matching. To quantify the robustness, we calculate the recognition rate of FAST-BRIEF, SURF-BRIEF and SURF-SURF based on six publicly available test image sets<sup>1</sup>. The open source code of FAST and BRIEF and the latest OpenCV implementation of the SURF are used for comparison.

In graffiti image set, BRIEF decreases robustness. Graffiti dataset requires strong rotation invariance but BRIEF is not designed to be rotationally invariant. To overcome this problem, FAST-BRIEF algorithm with orientation correction is recently proposed in [20]. It achieves similar rotation invariance to SIFT or SURF even in a strong rotating environment.

In bike image set, the major reason of decreasing robustness is the weakness of FAST in blurring. FAST cannot extract corners from strongly blurred images because FAST is a kind of finite impulse response (FIR) filter. Since cameras are set to obtain well-focused images for target applications, extremely blurred images do not frequently occur in real-time video stream. According to Ebrahimi et al. [2], the robustness of FAST-BRIEF is on par or even slightly better than SIFT's or SURF's when they are used for feature detection, tracking and recognition in a mobile platform.

FAST-BRIEF algorithm has similar robustness compared to SURF algorithm in many aspects such as lighting and perspective distortion.

## 8.2.2 Interest Point Detection and Matching Hardware

Many hardware accelerators were proposed to perform SIFT in real-time [6, 7]. Despite the computation complexity of SIFT, they achieved 60 fps in VGA resolution. They used circuit-level techniques to reduce the total area of region-of-interest (ROI) and exploited data-level parallelism based on many-core architecture. Since ROI reduction decreases the amount of image data to be processed, total execution latency is also decreased. However, they required large hardware size (50 mm<sup>2</sup>). The implementation area and required hardware resources increase super-linearly as the target resolution increases.

---

<sup>1</sup> <http://www.robots.ox.ac.uk/~vgg/research/affine>.

Yushi Moko et al. implemented FAST on an MX-G embedded processor which is a SIMD parallel processor [21]. They developed a method to parallelize FAST by using both the parallel processor and a co-processor (the host CPU processor). They achieved  $5\times$  improved performance compared to the CPU-based implementation without parallel processing components, but the frame rate is 27 fps in  $512 \times 512$  images.

Jason Clemons et al. proposed a feature extraction accelerator [22]. They proposed an embedded heterogeneous multi-core processor with special functional units. In [22], they evaluated various feature extraction algorithms, such as FAST, SIFT, and HoG. The implemented processor achieves  $14\times$  performance gain for HoG, but only  $4\times$  speed up for FAST. It requires 32 cycles for one segment test.

Marek Kraft et al. presented an SoC coprocessor for accelerating feature detection (FAST) and matching (SURF) [13]. They did not focus on the continuity test for FAST which determines where a set of nine consecutive pixels with all white or all black states on the circle is. Under well-optimized pipeline condition, the implemented hardware achieved 21 fps in VGA at 50 MHz operating frequency. The hardware architecture for FAST is not much different from [21, 22], except for the non-max suppression.

Jan Svab et al. showed several simplification techniques to implement SURF on a FPGA [23]. Their implementation achieves about 10 fps at HD ( $1024 \times 768$ ) resolution with similar distinguishability and robustness as GPU-SURF while consuming less than 10 W.

Keisuke Dohi et al. represented a FPGA implementation of FAST corner detection [24]. Because of a large number of corner patterns, the look-up table for corner pattern was too large for implementation with an FPGA. They propose corner pattern compression methods focusing on discriminant division and pattern symmetry for rotation and inversion. The FPGA implementation of FAST achieved real-time throughput when consuming only 529 slices in Virtex5 FPGA. We estimate that it requires 1 or 2 cycles for one segment test because it just reads the look-up table in order to get the result.

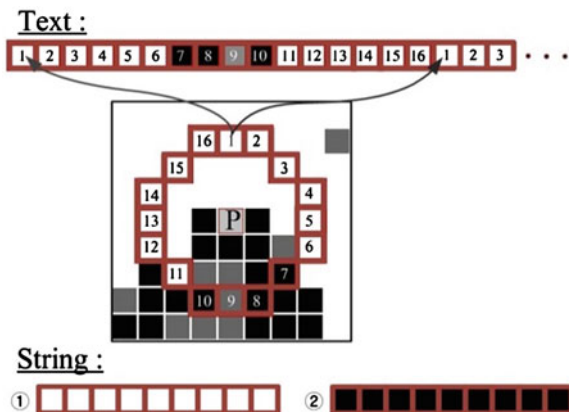
## 8.2.3 Proposed Method

### 8.2.3.1 Fast Corner Detection Hardware

#### Basic Concept

To design a high-performance interest point detection accelerator within small implementation area, a segment test method based on a string searching algorithm which searches a target *string* in long *text* is proposed. The proposed segment test method replaces the conventional decision tree method as shown in Fig. 8.2. In interest point detection process, 16 pixels around the target pixel are classified into three states (white, gray, and black) as the same as [8]. In this case, *text* is replaced

**Fig. 8.2** Proposed segment test method for exploiting bit-level parallelism



with an infinite repetition of 1-D array of the encoded 16 pixels' states, and *string* is replaced with  $N$  ( $8 < N < 13$ ) continuous white (or black) states based on a string searching algorithm as shown in Fig. 8.2.

It is logically equivalent to the original segment test and gives the same result as the original. Thus, any string searching algorithm can be used to replace the conventional decision tree based segment test.

### Proposed Algorithm

The Boyer-Moore-Horspool (BMH) algorithm [25] is a simplified version of the Boyer-Moore (BM) algorithm [26] which is the standard benchmark for the practical string search algorithm. Not only BMH but also all string searching algorithms such as Shift-OR [27], Turbo-BM [28], BM and Karp-Rabin [29] are possible to execute segment test for FAST. In general string searching environment, BMH would be slower than new complex string searching algorithms. However, the segment test of FAST has three special conditions : (*Condition A*) there are only three alphabets. (white, gray, and black), (*Condition B*) a *string* consists of all black or all white words. (*Condition C*) the length value of *string* and *text* is fixed and small. Since the segment test under those three conditions is very simple compared to the general string searching algorithms, it is hard to assure that new complex string searching algorithms are always faster than BMH.

BMH become useful as the number of used alphabets increase, because average processing time of BMH is inversely proportional to the cardinality of alphabets [30]. Since there are only three alphabets (white, black, and gray) in the segment test, BMH is not the best solution for the segment test. Based on the fact that the string used in the segment test consists of all black or all white words as shown in Fig. 8.2, the modified BMH is proposed to improve the segment test performance.

The pseudo code of the proposed method is described in Fig. 8.3. Suppose that partial *text* within given range, called window, compares to *string*. The size of

---

 Pseudo code of proposed segment test
 

---

```

text : Infinite repetition of the encoded 16 pixels' states
string : N (8<N<13) continuous white (or black) states
string_length : The length of string.

segment test (text, string, string_length)
  scan = 0 ;
  window_head = 0 ;
  last = string_length - 1;
  shift = 0;
  // Segment test
  while( window_head <= 16 )
  begin
    //Window matching test
    // ㉓ Equality comparison
    for(scan = last; text[scan] == string[scan];scan = scan -1)
      if(scan == 0) return CORNER;
    // ㉔ Window shifting amount calculation
    shift = string_length - scan;
    window_head = window_head + shift;
    text = text + shift;
  end
  return NOT CORNER;

```

---

**Fig. 8.3** Pseudo code of the proposed method

window is the same as the size of *string*. The leftmost position of window in *text* is defined window head. Since the window can be moved in *text*, the location of window in *text* is defined as the window head position. Window matching test consists of two functions; window equality check and window shifting amount calculation. Window equality check tests whether partial *text* within a given window and *string* are exactly same or not. Window shifting amount calculation determines the next window head position for saving redundant window matching tests as much as possible.

The proposed algorithm begins with aligning the left edges of *text* and window. At this time, window head position is set to 0. If window equality check finds a matched window, the target pixel **p** is classified as a corner by the definition of the segment test, and the segment test is terminated. However, when there is a mismatched word in the window, we cannot decide whether the target pixel **p** is a corner or not. Thus, we should perform another window matching test after shifting the window. At that time, window head position is increased as many as the result of window shifting amount calculation and the window slides from left to right in *text*. Window matching test is repeatedly executed until (1) window

equality check finds a match or (2) window head position is larger than the length of *text* (number of pixels on the circle; 16 in FAST). Case 2 means that there is no matched *string* in *text* after checking all possibilities. In this case, the segment test algorithm returns fail.

Window equality check compares *string* and *text* within a given window from right to left. Assume that there is a word in *text* at which the comparison process failed. Since *string* consists of the same words (all white or all black), it is impossible that the mismatched word in *text* occurs to the left in *string*. Thus, window shifting amount for next window matching test is as same as the number of words from the left-end of string to the right-most mismatched word.

The proposed algorithm has two advantages compared to BMH. First, pre-processing in BMH is not necessary, so the implementation cost can be saved. Second, the proposed algorithm is simple to be implemented in hardwired logic and it performs the segment test more efficiently than BMH. These two advantages will be discussed in detail in the section below.

### Joint Algorithm-Architecture Optimization

General string searching algorithms can be simplified based on these strong conditions. We aim to implement area and power efficient hardware for interest point detection and matching algorithm. If any string searching algorithm has similar performance (speed) and gives exactly the same operation results in segment test, it is preferable to choose simple and hardware-familiar algorithm.

Original BM has two phases (pre-processing phase and searching phase). Pre-processing is required to reduce redundant character comparison in searching phase. Searching phase of BM algorithm consists of two parts: good-suffix shift and bad-character shift. Bad-character shift is simpler but always slower than good-suffix shift in *condition B* as mentioned in above section. We want to achieve performance as same as the good-suffix shifts with simple hardware like the accelerator for bad-character shift.

The proposed algorithm considers the character in text at which the comparison process failed. The next occur of that character to the left in string is found, and a shift which brings that occurrence in line with the mismatched occurrence in text is proposed. By *condition B*, it is impossible that the mismatched character occurs to the left in string. Many parts of BM operations are removed as described below.

- (1) Bad-character shift process can be removed. The result of good suffix shift is always less than or equal to the result of proposed method in *condition B*. Good-suffix shift process also simplified based on *condition B*, because the result of good suffix shift always equal to the result of proposed method in *condition B*. So, we can save redundant processing resources and time which would be used for this operation.
- (2) Pre-processing phase is omitted. The result of pre-processing is used for finding the next occurrence of the mismatched character to the left in *string*.

However, since the process to find the next occurrence of the mismatched character is simplified, the result of pre-processing is not required anymore.

Proposed algorithm has many benefits. (1) It doesn't request any pre-processing. (2) It consists of simple operation and can be accelerated by bit-level parallel processing. (3) It requires no additional memory to store temporal data.

In order to estimate the average execution time of string searching algorithms, cycle-accurate simulation is performed with the same test-benches used in [8]. Open source codes<sup>2</sup> of string searching algorithms are used to evaluate the performance of the segment test based on various string searching algorithms. We assume that each window matching test is accelerated by hardware and it can be executed in a cycle. The performance is represented by the number of window matching test for each segment test.

### **Turbo-BM**

Since Turbo-BM modifies only good-suffice shift process of BM, it gives same result as our proposed algorithm after modification.

### **Rabin-Karp**

Brute force string searching algorithm is slow because it compares every single character from the string and the text. The performance of Brute-Force algorithm is  $O(\mathbf{n}*\mathbf{m})$ , where  $\mathbf{n}$  and  $\mathbf{m}$  is the length of *string* and *text*, respectively. Karp-Rabin algorithm reduces the length of *string* using rolling hash algorithm.

Since it is possible to compare a *string* and a part of *text* in a single cycle based on parallel hardware, reduction the length of *string* using hashing algorithm does not affect to enhance the performance of hardware. Besides, the hardware size can be fixed because of **condition 3**. Performance overhead for generating hash values and hardware overhead to store pre-processed hash value should be considered.

Unfortunately, it does not have any shifting amount calculation process to save redundant window-level comparison. If the *string* and partial *text* in window is mismatched, it shifts the window one by one like brute-force algorithm. As a result, the modified version of Karp-Rabin is same as brute-force algorithm in hardware.

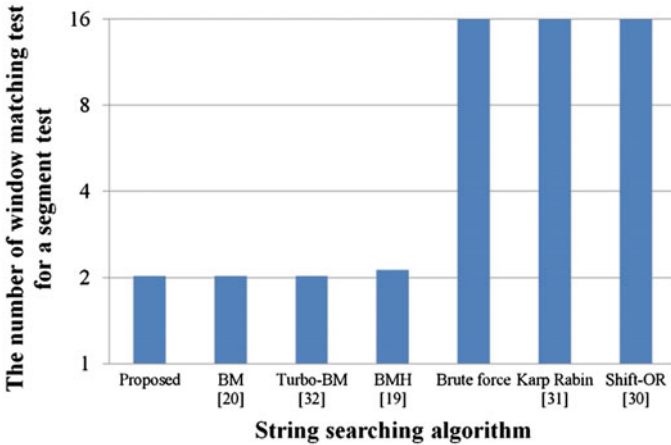
### **SHIFT-OR(BITAP)**

SHIFT-OR algorithm executes comparison between *string* and partial *text* in window in a cycle. It is noticeable because it is achieve to exploiting bit-level parallelism in CPU based system. As we mentioned in the explanation of Rabin-Karp, it is easy to execute word-wise comparison within a window in a dedicated hardware.

It does not have any shifting amount calculation process to save redundant window level comparison. So, the modified version of SHIFT-OR is same as brute-force algorithm in hardware.

---

<sup>2</sup> <http://www-igm.univ-mlv.fr/~lecroq/string/>.



**Fig. 8.4** The number of the window matching test for a segment test according to various string searching algorithms

Shift-OR, Karp-Rabin and brute-force shows similar performance. BM family such as BM, BMH, and Turbo-BM shows better performance than the others as shown in Fig. 8.4. BM and Turbo-BM have slightly better performance than BMH because of an additional complex searching process called good-suffix shift. The proposed algorithm achieves exactly the same performance as BM and Turbo-BM while maintain its simplicity. The proposed algorithm is hardware-friendly and it is developed by joint algorithm-architecture optimization.

## Hardware Implementation

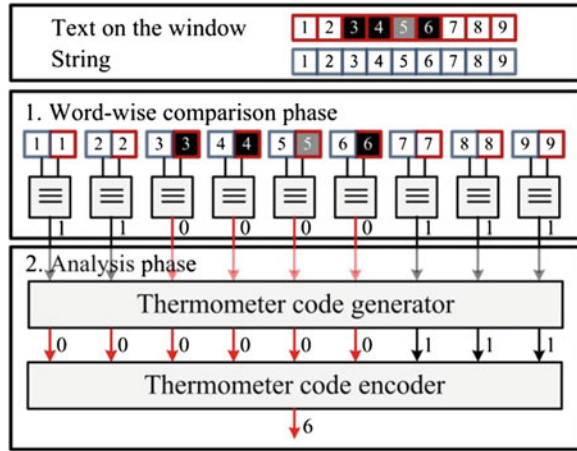
### (1) Window Matching Accelerator (WMA)

WMA consists of two phases; the word-wise comparison phase and the analysis phase as shown in Fig. 8.5. In word-wise comparison phase, *string* is compared with a partial *text* within window word-by-word manner in order to find the first mismatching position in a current window. To support FAST-9, 10, 11, and 12, the length of the window and *string* is less than or equal 12. In this case, only twelve 2-bit equality comparators (EC) are required for window equality check. ECs generate a 12-bit result code which is called as an equality code. If the words are matched, “1” is encoded in an equality code. In other cases, it encodes “0” as shown in Fig. 8.5.

In analysis phase, WMA finds the location of the right most “0” from the left-end in the equality code. This result is defined as a shift number. If the shift number is 0, it means that “0” does not exist in the equality code. In other word, the equality code consists of only “1” and window matching test finds a match. Otherwise, WMA calculates window shifting amount for the next test. According



**Fig. 8.5** The block diagram of window matching accelerator (WMA) and an example of its functionality



to the proposed segment test algorithm, WMA shifts the window as many as the shift number. Consequently, the shift number has all information to control WMA.

To simplify the analysis phase, we use a thermometer code generator (TG) and a thermometer code encoder (TE) as shown in Fig. 8.5. TG fills “0” in all position between the left-end of the equality code and the position of the right-most “0”. TE counts the number of “0” in the thermometer code and converts it into a binary number. The output of TE is the shift number.

Since the equality comparator in the first phase only requires some comparator logics and the implementation overhead of TG and TE is not critical [31], the proposed WMA is appropriate for mobile environments.

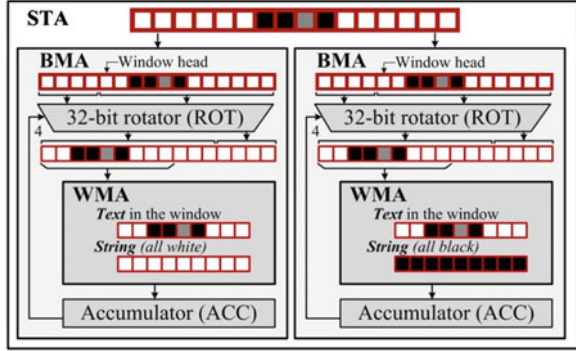
(2) *Bit-level Matching Accelerator (BMA) and Segment Test Accelerator (STA)*

Bit-level matching accelerator (BMA) fully supports the proposed algorithm. It includes WMA and a dedicated hardware for cropping partial *text* within the window to generate input data of WMA.

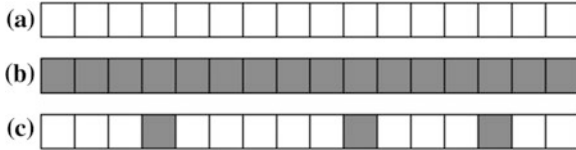
Figure 8.6 shows the block diagram of BMA and a segment test accelerator (STA) for segment test of FAST. BMA includes three operation stages; (1) a 32-bit rotator (ROT), (2) WMA, and (3) an accumulator (ACC). ROT rotates *text* for matching test as shown in Fig. 8.6. WMA determines the amount of additional rotations. ACC accumulates the number of rotations which is the input value for ROT. BMA operates in a loop-like manner from ROT to ACC until the accumulated value is larger than 16.

The segment test for a given *text* should be performed twice using both white and black *string* in order to detect a corner. Since BMA executes the segment test with only one string at a time, STA requires two BMAs to exploit parallelism. If one of BMA detects a corner, the whole segment tests in STA are terminated, and STA returns true. Otherwise, STA waits to receive new input data until both BMAs report fail.

**Fig. 8.6** Block diagram of bit-level matching accelerator (BMA) and segment test accelerator (STA). Each BMA has one WMA and a STA consists of two BMA



**Fig. 8.7** **a** The best case of *CORNER* case **b** The best case of *NOT\_CORNER* case **c** The worst case of BMA



Performance Estimation

Performance estimation of the proposed hardware is done by performing the best, worst, and average case analysis. We use an STA (a pair of BMAs) to execute a segment test for both black and white *strings*, simultaneously. Each window matching test requires only one execution cycle.

The total execution time is defined as accumulated execution cycles until meeting the termination condition. If one of two BMAs detects a match, the segment test is finished and reports *CORNER*. In the best case, the segment test can be done only with 1 cycle as shown in Fig. 8.7a. If the window head position of both BMAs is larger than 16 without any matching during the segment test, the segment test returns *NOT\_CORNER*. Since the window head position can be shifted only 9 ~ 12 at each execution cycle, *NOT\_CORNER* case requires at least 2 cycles as shown in Fig. 8.7b.

Execution time of *CORNER* cases is always less than or equal to of *NOT\_CORNER* cases. In other words, once *CORNER* cases are decided, the rest of operations are not required anymore. However, *NOT\_CORNER* cases can be determined only when BMA checks all possibilities (Fig. 8.8). Thus, it is reasonable to focus on *NOT\_CORNER* cases to estimate the upper bound of performance in BMA.

Suppose that *string* with nine white words (states) is used for BMA. The shift amount of the window head position at time  $T_{\#}$  is  $n_{\#}$  as shown in Fig. 8.9. In this example, BMA reports fail (window is not matched) at  $T_0$  and shifts the window for the next window matching as many as  $n_0$ . In other words, the  $9-n_0$  words from  $n_0$ th word of *text* inside the window are composed of only white states. At  $T_1$ , if  $n_1$  is less than or equal to  $9-n_0$ , it is a *CORNER* case because it means that a pattern with nine continuous white states exists in a given *text*. In the worst case (four

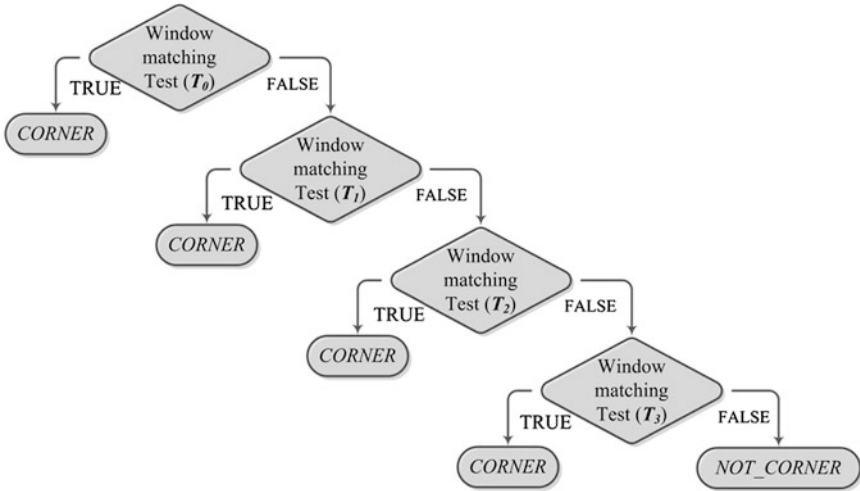


Fig. 8.8 CORNER case is always terminated earlier than the NOT\_CORNER case in given text

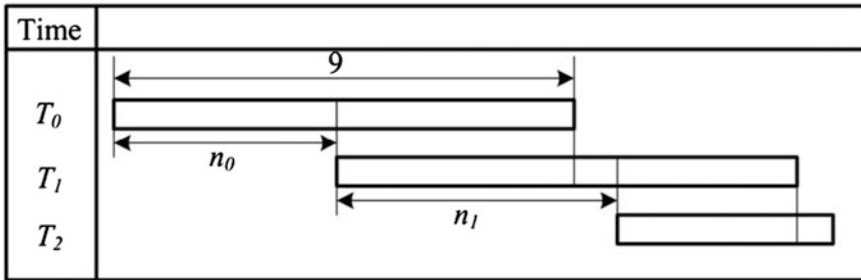


Fig. 8.9 At  $T_0$ , BMA returns fail and shift  $n_0$ . At  $T_1$ ,  $n_1$  should be larger than  $9 - n_0$  to become NOT\_CORNER. Since the accumulated shift is always larger than 9 during 2 cycles, the execution time of BMA is less than or equal to 4 cycles

consecutive FALSEs described in Fig. 8.8),  $n_1$ ,  $n_2$ , and  $n_3$  are larger than  $9 - n_0$ ,  $9 - n_1$ , and  $9 - n_2$ , respectively ( $n_0 + n_1 > 9$  and  $n_2 + n_3 > 9$ ). Since the number of pixels on the circle for the segment test is 16 and  $n_0 + n_1 + n_2 + n_3$  is larger than 18, the worst case execution time of BMA is four cycles. Figure 8.7c shows the worst case example with four cycles of execution time.

Note that the worst cases which require four execution cycles always result in NOT\_CORNER. In all worst cases, *text* has Match-Mismatch patterns as shown in Fig. 8.10, because this is only way to execute 4th window matching test in a segment test. We describe detailed conditions for this situation and prove that there is no chance to pass the window matching test. Since  $n_0 + n_6$  is always smaller than six, at least one mismatched word always exists on the window at the 4th cycle. Thus, three cycles are sufficient to determine whether target pixel  $p$  is a corner or not.

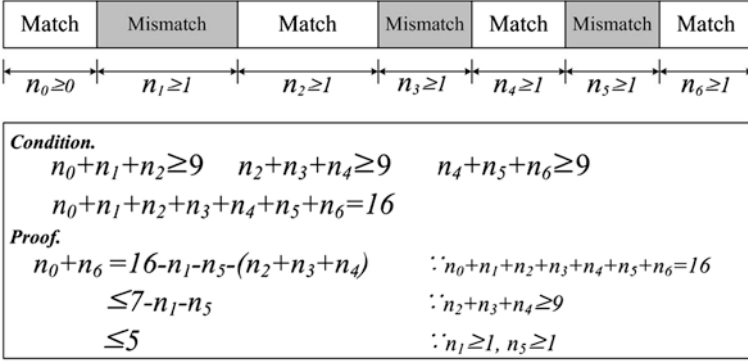
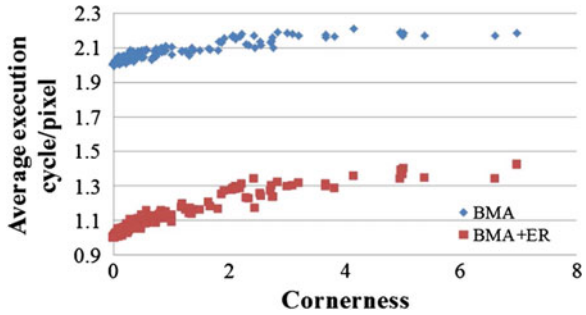


Fig. 8.10 The condition and proof of the worst case of BMA

Fig. 8.11 The performance of (1) the BMA only, (2) the BMA with the early rejection scheme according to the cornerness

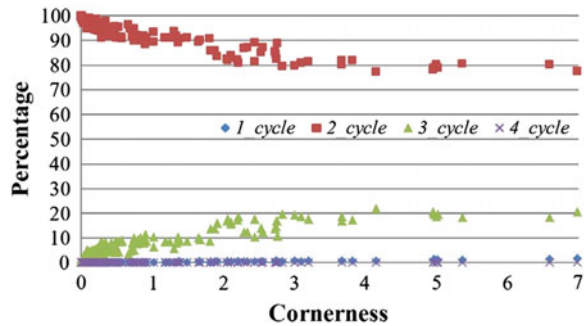


The segment test should be performed twice with two strings as described in the previous subsection. Since two segment tests with two different strings are independent each other, we can execute two segment tests in parallel with two sets of hardware accelerators. With two BMAs, one complete corner detection operation can be performed within three execution cycles.

In order to estimate the average execution time of the proposed hardware, cycle-accurate simulation is performed with the same test-benches used in [8]. Average execution time directly depends on the input images and the threshold value used in FAST. It is difficult to quantify the effects of the performance according to input test images. In this work, we quantify the effects as the cornerness. Cornerness is the ratio of the number of corners with respect to the number of pixels in the input image. Complex input images with lower threshold value lead to higher cornerness output. The input images with about 1 % of cornerness are used in previous works [2, 8].

Figure 8.11 shows performance (average execution time per pixel) of the proposed BMA according to the cornerness. If the cornerness is less than 1 %, average performance is 2.09 cycles/pixel. Even in the complex image with 7 % of cornerness, the average performance is less than 2.2 cycles/pixel. As a result, STA

**Fig. 8.12** The breakdown of execution cycle for each pixel when uses only BMA



achieves 43.8 fps with full HD ( $1920 \times 1080$ ) images at 200 MHz operation frequency on the average.

Since *CORNER* cases are always finished earlier than *NOT\_CORNER* cases, total execution time decreases as the number of corners increases. However, Fig. 8.11 shows the opposite results such that the average execution time increases as the cornerness increases. In other words, cornerness and average execution time have a positive correlation with a near-linear relationship. Higher cornerness results in the increment of the number of pixels to be classified into a corner. In general, images with more high frequency components give higher cornerness. In this case, the segment test reacts more sensitively to the difference of the pixel intensity as threshold value decreases. This increases the complexity of *text* for each pixel. In this case, BMA requires additional execution time due to the change of *text*.

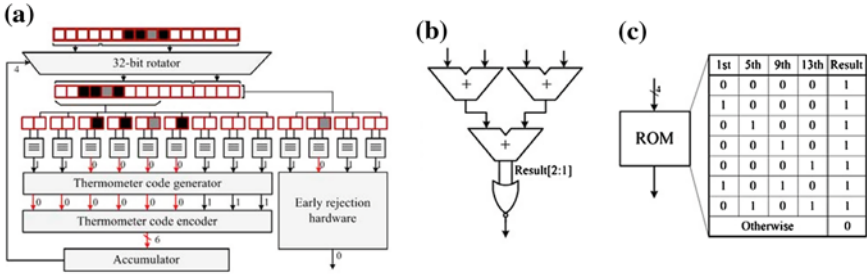
The breakdown of execution cycles according to the cornerness is shown in Fig. 8.12. As the cornerness increases, *3\_cycle* case encroaches on *2\_cycle* cases. It means that the high-frequency region of images affects the performance of BMA.

### 8.2.3.2 Early Rejection Scheme

#### Introduction

Interest points detected by the segment test are located at the point where the pixel value changes sharply. However, the most part of an ordinary image has similar values to their neighbor pixels. In this case, the number of gray words in a *text* approaches to 16. Since *text* should have gray words less than seven to be detected as *CORNER*, the case is obviously *NOT\_CORNER*. But an STA always requires at least 2 execution cycles to classify any *NOT\_CORNER*. If we have another dedicated hardware to reject the redundant *NOT\_CORNER* case in parallel, we will save a lot of time which may be spent for unnecessary window matching test.

A rapid rejection method is proposed in order to skip redundant segment tests of FAST-12 [32]. It examines only four pixel values at 1, 5, 9, and 13 (the four compass directions) as described in Fig. 8.2. And, it classifies target pixel  $p$  with



**Fig. 8.13** a Modified BMA with early rejection scheme. b Early rejection hardware for Condition 1. c Early rejection hardware for Condition 2

intensity  $I_p$  as *NOT\_CORNER* directly unless at least three of the four pixel intensities are all brighter than  $I_p + t$  or darker than  $I_p - t$ . It requires only four intensity comparisons to exclude *NOT\_CORNER* cases, such as a *text* with all gray states. If the target pixel  $p$  is not filtered by rapid rejection method, a test is continued to execute checking that a set of 12 continuous pixels exists on the circle for the segment test of FAST-12. It means that the information obtained from the rapid rejection scheme is only useful when  $p$  is discarded. However, since this method detects most *NOT\_CORNER* cases by using simple operations, performance of the overall segment test would be increased.

The rapid rejection scheme can be applied only on FAST-12 [32]. To overcome this limitation, a machine learning technique based on decision tree is adopted to generalize and optimize overall segment test [8].

Since the proposed segment test based on a string searching algorithm and a rejection scheme is independent each other, any rejection method and the proposed segment test can be operated in parallel as shown in Fig. 8.13a. While STA classifies *NOT\_CORNER* cases after checking all possibilities to detect *CORNER* as mentioned in the previous section, a rejection method just focuses on excluding *NOT\_CORNER* as soon as possible with simple operations. If the rejection method detects *NOT\_CORNER*, it is obvious to save redundant execution time. Even in the other cases, additional execution cycle is not required, because the rejection method runs with STA in parallel.

Early rejection method simply judges  $p$  as *NOT\_CORNER* when  $p$  does not satisfy the necessary condition for the segment test of FAST. A lot of necessary conditions can be set up according to implementation environments such as hardware architecture, performance, area overhead, and power consumption. For example, two types of simple necessary conditions can be proposed. Assume that these conditions use four pixel values at 1, 5, 9, and 13 like the rapid rejection method; *Condition 1*. At least two states among four pixels are all white or black; *Condition 2*. At least 2 consecutive states among four pixels are all white or black. *Condition 2* requires more hardware resources compared to *condition 1*, but *condition 2* rejects *NOT\_CORNER* cases more efficiently than *condition 1*. Both *condition 1* and *condition 2* can be used for FAST-9, 10, 11, and 12.

## Hardware Implementation

Various implementations are possible according to the necessary condition. For example, *condition 1* is simply implemented by an adder tree to just count white (or black) states, while *condition 2* is implemented by small look-up table as shown in Fig. 8.13b and c, respectively. Since many necessary conditions are implemented by their own hardware, we can select a necessary condition at design time according to target performance and hardware architecture.

In most cases, the size of early rejection hardware is small, because early rejection method just excludes some candidates which do not satisfy a simple necessary condition with limited variables. For example, both *condition 1* and *condition 2* require tens of logic gates.

## Performance Estimation

Since the early rejection scheme detects redundant cases and excludes them from further tests, it affects directly the best case of *NOT\_CORNER* cases and average performance. However, the early rejection scheme does not change performance of *CORNER* cases because the rejection method works independently with STA and detecting *CORNER* is a unique job of STA. Since complex pattern which requires 3 cycles of execution time in STA might not be filtered by the rejection method, the worst case of *NOT\_CORNER* cases does not change.

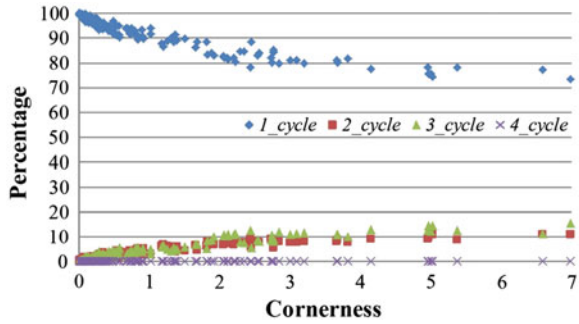
To show the effect of the early rejection, cycle-accurate simulations are performed with early rejection on *condition 1* and *condition 2* based on the same environment of the previous analysis. Figure 8.11 shows the average execution cycle per pixel of the proposed hardware with respect to the cornerness. If the cornerness is less than 1 %, average cycle is up to 1.15 cycle/pixel. Even at 7 % cornerness, the average cycle is less than 1.42.

Since the theoretical performance limitation of the interest point detection is 1 cycle/pixel and the average performance of the STA converges to the 1 cycle at the nominal cornerness (less than 1 %), our hardware based on STA and the early rejection scheme is well-optimized. STA with early rejection scheme achieves more than 68.52 fps with Full HD resolution at 200 MHz operation frequency.

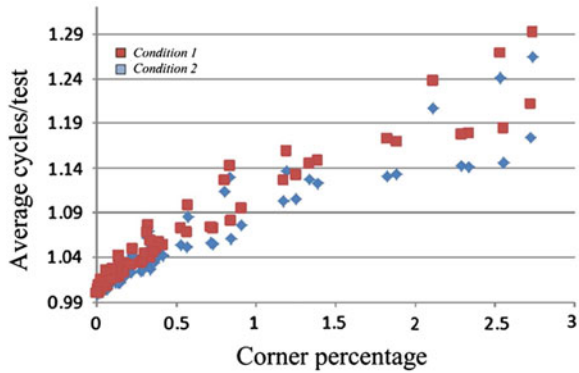
Figure 8.14 shows that the break-down of the execution cycles per test. Note that major part of *2\_cycle* cases are handled in *1\_cycle* as we expected. It is major reason of performance improvement.

Performance varies slightly according to the necessary condition to be used as shown Fig. 8.15. Performance results of the proposed system with *condition 1* and *condition 2* are similar each other with small difference (about 5 %). The proposed technique including STA and the early rejection method can be used for acceleration of the segment test such as FAST and AGAST. This scheme achieves near-ideal performance with identical corner detection result as the original one.

**Fig. 8.14** The breakdown of execution cycle for each pixel when uses BMA with the early rejection scheme



**Fig. 8.15** Performance comparison between condition 1 and condition 2



### 8.2.4 Conclusions

Previous SIFT-based processors used floating point number [6, 7]. For example, convolving with Gaussian window translates 8-bit pixel data to floating point number. They exploited data parallelism by many-core architecture, but the total area of hardware is too large to be used in mobile systems.

Previous FAST-based accelerators [13, 21, 22] are well implemented, but they did not focus on the continuity test, despite of performance overhead. They spent more than 14.7 cycles for a continuity test with simple or unknown hardware.

FAST accelerator proposed in [24] requires 1-2 cycles for one segment test based on compressed decision trees. It stores the results of all cases in a look-up table and read the look-up table according to input vectors. But, it can be burdensome as it requires a look-up table for each FAST-n test.

BMA requires *text*, a 32-bit code including 16 total states, as input. Each state is encoded in 2-bit, because there are only three kinds of state (white, black and gray). Reduction of the size of input data resolves the hardware overhead, such as data-path and internal memory.

Since STA exploits bit-level parallelism, it performs the segment test efficiently with a given *text*. Both the word-wise comparison and analysis phases in WMA



take care of the input data with 16 states in parallel. Furthermore, each BMA in STA executes the segment test with a given text and each own string (all white and all black string) in parallel.

STA can be easily extended to multi-core systems according to target performance. In many kind of vision and image processing, each segment test for different target pixel  $p$  is independent each other. Since each *text* is treated as an independent instruction for STA, critical issues of multi-core system such as data-dependency and atomic operation are not faced. In other words, it is possible to exploit data-level parallelism with many STAs. The performance increases linearly with respect to the number of STAs. In other words, we easily estimate the number of STAs in order to achieve the target performance at design time.

In conclusion, the proposed STA can increase performance per area and performance per power by exploiting high-parallelism and small hardware. STA requires only 1–3 cycles per segment test and it is implemented only with 1,600 logic gates. Although the proposed algorithm based on BMH is much slower than the original segment test based on decision tree on the CPU system, it can be implemented very efficiently with a dedicated hardware.

## 8.3 Unified Datapath

### 8.3.1 Introduction to Unified Datapath

Both interest point detection and matching operations are required for general recognition process. These two operations are functionally independent, so different hardware should be implemented for complete recognition process. This causes two critical problems; (1) area-efficiency loss and (2) unbalanced workload.

- (1) Interest point matching is sequentially executed after interest point detection per every image frame. While a detector extracts points from a given image frame, a matcher should be in the idle state, and vice versa. The hardware resources in the idle state reduce the area-efficiency.
- (2) Hardware pipelining method may be a good solution for (1). With the pipelining, the detector can operate in parallel with a matcher, while they treat different image frames. But, hardware suffers from unbalanced workloads as a side-effect. Load balancing is important in a pipelined hardware because the performance of overall hardware is the same as the performance of the slowest pipeline stage that is a bottle-neck stage. Unfortunately, computation load per each operation (feature detection or matching) varies according to the target environments such as image resolution, the number of detected feature points, and scene complexity. Since hardware architecture is fixed at design time, unbalanced workloads between the matcher and detector stages degrade the performance.

Previous vision processors [6, 7, 13, 23] based on massively parallel multi-core architecture also face this problem. In this work, unified hardware platform is proposed to share the same hardware between interest point detection and matching.

### 8.3.2 Proposed Hardware

#### 8.3.2.1 BRIEF Based Interest Point Matching Hardware

It is necessary to design an interest point matcher based on STA in order to realize the unified hardware platform. BRIEF is one of the state-of-the-art interest point matching algorithms. BRIEF is widely used with FAST in [2, 20]. Furthermore, although BRIEF was designed to reduce the size of descriptor, it can be implemented in similar data-path to our modified FAST accelerator.

Figure 8.16 shows BRIEF mapped on FAST hardware. BRIEF is divided into B-a (Pair loader), B-b (descriptor generator) and B-c (Similarity checker) stages. B-a stage loads the intensity of pixel pairs located at a given position, B-b stage compares the loaded pixel's intensities and generates a descriptor. After that, to find the best matching result between the generated descriptors, B-c stage measures the hamming distance to find the most similar descriptor among a set of descriptors.

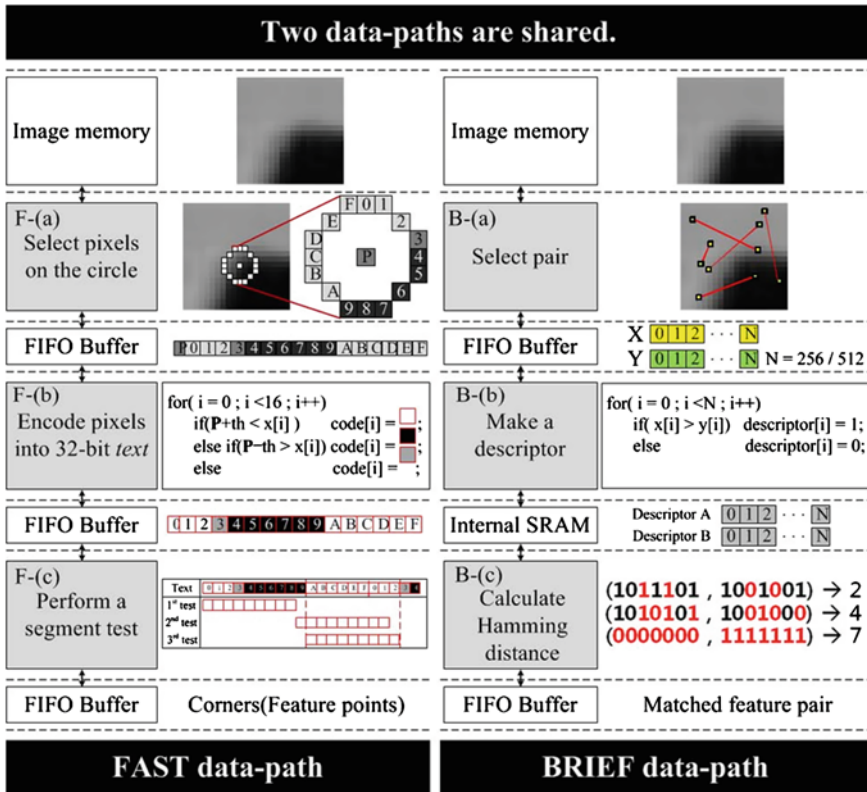
They are completely mapped on F-a (pattern loader: loading image pixels from an image buffer), F-b (*text* generator: encoding the pixel data as a short binary *text*), F-c (BMA) in FAST implementation. F-a and B-a stages load pixel intensities from arbitrary positions. F-b and B-b stages compare pixel intensities and make a bit-vector. F-c and B-c stages perform bit-parallel operations with the vectors obtained from the previous stages.

The proposed FAST algorithm has a similar operation flow to BRIEF, and next subsection will show that they can be merged into the same data-path

#### 8.3.2.2 Overall Hardware

We implement a hardware accelerator to show performance of the proposed STA. To estimate performance clearly, we remove some factors, such as I/O bandwidth and miss rate of memory access, which cause overall performance degradation of the implemented hardware. We use a descriptor buffer and a pattern loader to reduce the effect from I/O.

All descriptors in two different images should be prepared before the matching test. In general, descriptors are stored in an external memory due to the large size of the required memory. For example, 128 KB memory space is required to store 4,096 32-byte descriptors. The limitation of the external memory bandwidth causes serious problems. Assume that 2,048 descriptors per each image are stored in external memory. To achieve 60 fps based on the exact NN matching algorithm [9],



**Fig. 8.16** Proposed unified hardware platform for interest point detection and matching based on FAST and BRIEF

external memory bandwidth reaches 128.8 G bit/sec. Since 128.8 G bit/sec is hard to be supported in a mobile environment, performance decrease according to limited memory bandwidth is inevitable. We use 128 KB internal SRAM, called descriptor buffer, to resolve it.

FAST and BRIEF loads pixel values at arbitrary location from an image patch. Miss-rate of memory access might be increased, because traditional memory architecture is optimized for exploiting spatial locality. Miss-rate increases average memory access time and it causes performance degradation. We assume that input data comes from external device, which loads pixel value on the complex pattern. The input buffer, which is called as pattern loader, just stores loaded data and puts it in the data-path when a request signal is received.

The text generator gets input data from pattern loader and encodes *text* of FAST or descriptor of BRIEF. Pattern matching based recognition accelerator (PRA) executes segment test or calculates the hamming distance between two descriptors for matching with four modified STAs as shown in Fig. 8.17.

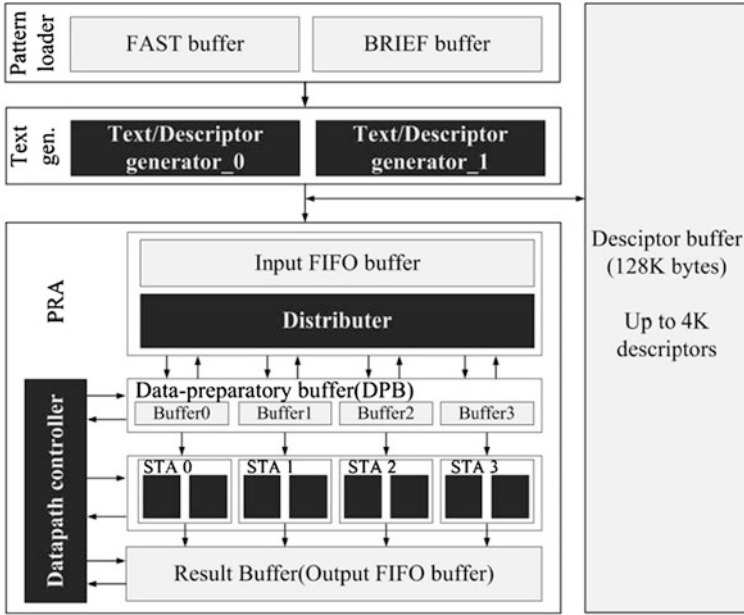


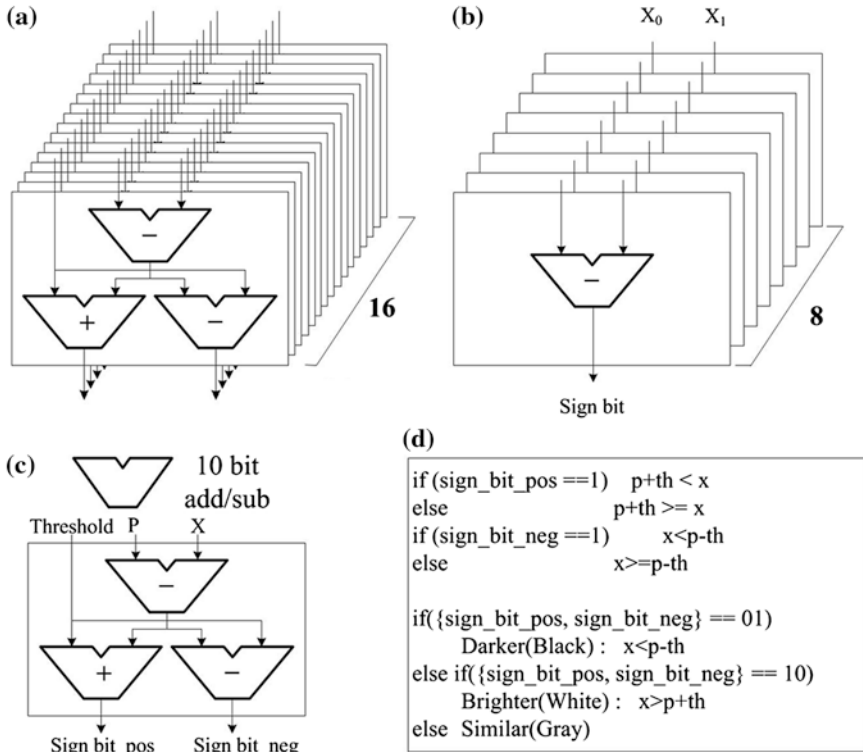
Fig. 8.17 System architecture of the scalable recognition accelerator

### Text Generator and Descriptor Generator

Figure 8.18a shows the text generator for FAST and Fig. 8.18b represents the descriptor generator for BRIEF. Text generator consists of sixteen state classifiers depicted in Fig. 8.18c. A state classifier consists of a 10-bit adder and two 10-bit subtractors and the entire operation flow is shown in Fig. 8.18d. A pair of sign bits means three states; white (10), black (01), and gray (00, 11). Inequality sign is slightly different from the original test in [8], however, it can be simply calibrated by adjusting threshold value. The descriptor generator consists of eight 10-bit adders based on the intensity test of BRIEF. Both the text generator and the descriptor generator get the input data from the pixel selector, and we take one of the results according to target application (FAST or BRIEF).

The descriptor generator can be implemented with text generator. It requires additional hardware logics for selecting input data with respect to the FAST or BRIEF. However, this method does not have any benefit in terms of area-overhead and performance.

The descriptor generator generates only an 8-bit vector, while the text generator generates full *text* (32-bit vector) at once. Since BRIEF uses a 256/512-bit descriptor in general, 32/64 cycles are required for encoding a descriptor. The descriptor buffer gathers partial vectors and stores the completed vector in internal SRAM.

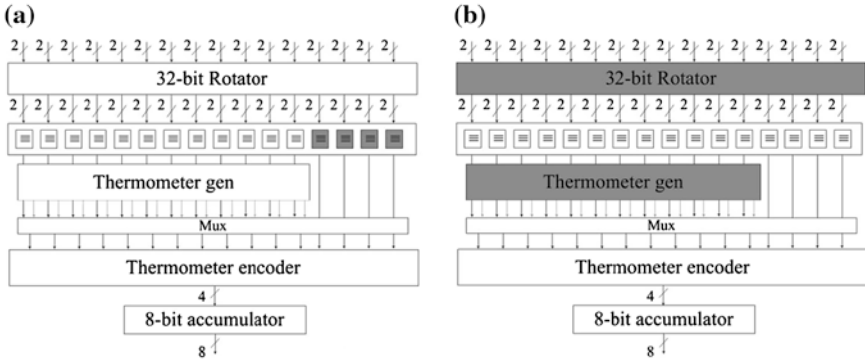


**Fig. 8.18** **a** Text generator for FAST. **b** Descriptor generator for BRIEF. **c** A state classifier for text generator. **d** The pseudo-code describing the functionality of the state classifier

Two sets of text and descriptor generators are integrated for balancing the workloads between pipeline stages. Since each STA does not include early rejection hardware, each STA requires 2.1 cycles on average. To utilize four STAs fully in parallel, we need two sets of text and descriptor generators.

### Extended STA

WMA is modified in order to support both FAST and BRIEF as shown in Fig. 8.19. Figure 8.19a and b show the activation of WMA in a detection mode and a matching mode, respectively. Dark gray colored cells imply inactivated stages and the others are activated stages. Four XOR gates and sixteen 1-bit multiplexers are added in the previous WMA without significant changes. Actually, modified WMA in detection mode is operated almost in the same way as the previous WMA.



**Fig. 8.19** Modified WMA in STA. Dark gray colored blocks imply that the block is inactivated at each mode; **a** FAST and **b** BRIEF

In a matching mode, WMA calculates the Hamming distance between two descriptors. Since the size of input data of WMA is 32-bit, WMA should support 16 XOR operations in the execution cycle. Twelve equality comparators (EC) are modified to perform both equality comparison and XOR operation according to operation modes. And then, four XOR gates are added to achieve target performance. Note that four modified ECs are added rather than XOR gates when the early rejection scheme is also implemented.

The result of comparisons directly goes to thermometer code encoder (TE). TE is implemented with an adder tree in order to count the number of “1” from binary number. So, TE can be shared for both FAST and BRIEF operations. Multiplexer selects appropriate data between the result from ECs and the result from TG according to operation mode.

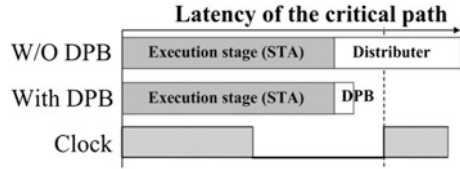
STA can execute thirty two XOR operations in the execution cycle and the descriptor size is 256 or 512 bits. An 8-bit accumulator accumulates partial bit counts which come from TE for 8 or 16 cycles.

### PRA

PRA consists of four STAs, input and output buffers, a controller, and an internal buffer, which is called as a data preparatory buffer (DPB) as shown in Fig. 8.17. Since the execution pattern changes according to FAST or BRIEF, the controller and DPB should be able to control execution flow in order to fully utilize the STA.

Although STA requires from 1 to 3 cycles to perform a segment test for FAST, it is impossible to predict the execution time exactly before finishing the segment test. STA receives new data from a distributor as shown in Fig. 8.17. The latency of distributor is a few nanoseconds because the distributor apportions data among four STAs according to the number of available data in FIFO, the number of requests from STAs, and operation modes. STA and distributor should be merged

**Fig. 8.20** Latency of the critical path with or without DPB



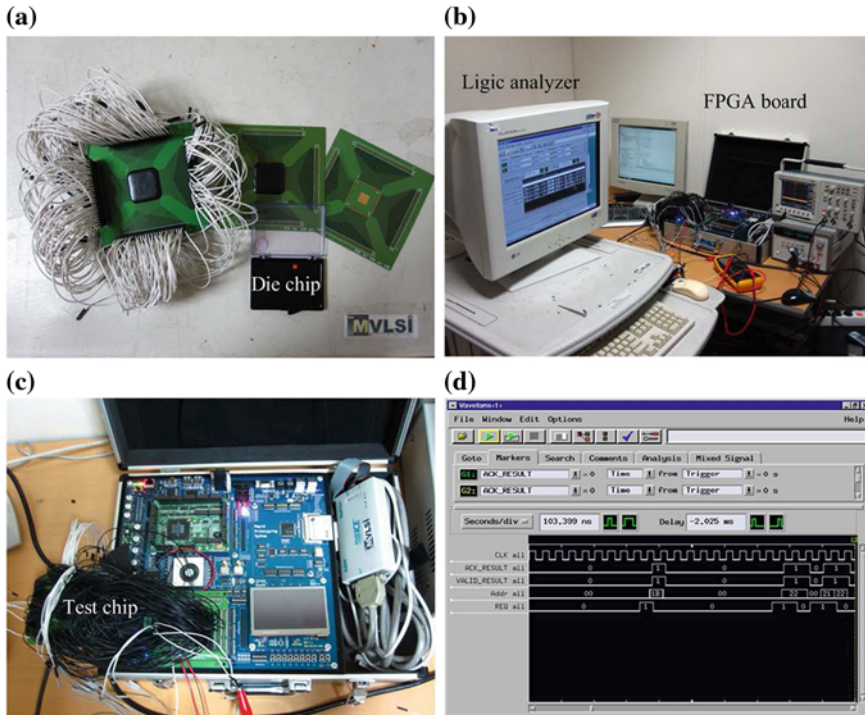
in an operation stage in order to get the next data without any idle time. However, the latency of the critical path is longer than 5 ns as shown in Fig. 8.20. So, PRA cannot be operated at 200 MHz.

To achieve target frequency (200 MHz), we use DPB, added between a distributor and multiple cores as shown in Fig. 8.17. DPB is a kind of pipeline register in order to store four vectors per STA. STA seems to already have the current and next vectors, so it simply switches to the next one after termination of test without any request through memory interface. When DPB has an unoccupied buffer space, the controller detects it and requests an order to fill the buffer immediately. Since DPB requires much shorter latency than the controller, the distributor, or STA in general, additional latency for filling DPB is hidden by the latency of the critical path as shown in Fig. 8.20. As a result, it is possible to support input data without any idle time and additional latency on the critical path.

In a matching mode, STA perform a Hamming distance calculation between two descriptors for 8 or 16 cycles. Since the required execution time is fixed according to the descriptor size, STA can execute the calculation without any idle time and additional latency of the critical path.

## 8.4 Chip Implementation

The proposed unified hardware is fabricated in 0.13 um standard complementary metal-oxide-semiconductor (CMOS) technology. Figure 8.21 shows test environment and measured waveform, and the chip specification is summarized in Table 8.1. The chip size is  $4 \times 4$  mm<sup>2</sup> including 861 K logic gates. The overall system operates at 200 MHz. The unified datapath consists of 78.3 k logic gates without a descriptor buffer (less than 3 % area compared to previous works [6, 7]). A large descriptor buffer which is implemented with 128 KB SRAM is integrated in the test chip for testing the proposed IP. This on-chip buffer can store up to 4,096 descriptors (256 bit) generated from feature detection process, and the stored descriptors are directly used in matching process without off-chip communication. Figure 8.22 shows an area break-down of the top module and PRA. The major part of the hardware is an internal memory, such as a descriptor buffer, input and output buffers, and DPB, and it is possible to reduce the size of them according to target design.



**Fig. 8.21** Test environment and measured waveforms. **a** Chip and PCB board for measurement. **b** Measurement environment. **c** FPGA board. **d** Measured waveforms for PRA

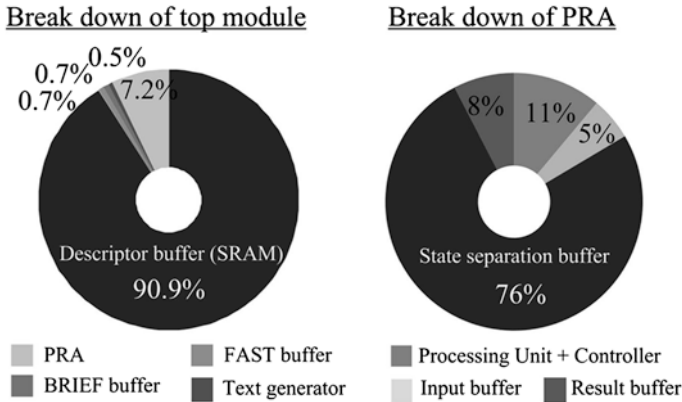
**Table 8.1** Implementation summary

Process technology	0.13 um CMOS (1P6 M)
Supply voltage	Core 1.2 V; I/O 3.3 V
Chip size	4.0 × 4.0 mm <sup>2</sup>
Core size	3.2 × 3.20 mm <sup>2</sup>
Logic gate count	861 K (2-input NAND gate) including SRAM
On-chip SRAM	128 Kbyte (Descriptor buffer)
Operating frequency	200 MHz
Power consumption	182 mW

STA only requires 1.15 and 2.09 cycles (maximum 3 cycles) per one segment test with/without the early rejection scheme, respectively. And it requires 8 cycles per one feature matching test (32-byte descriptor). Furthermore, frame rate increases linearly according to the number of STAs by the proposed scalable/reconfigurable architecture.

Table 8.2 shows performance on various parameters such as the number of descriptor, the size of descriptor, and the performance of FAST algorithm. The entire hardware can achieve 235 fps in full HD (1920 × 1080) resolution at





**Fig. 8.22** Area break-down of the proposed hardware

**Table 8.2** Performance (frame rate) of unified data-path with respect to descriptor size, the number of descriptors, and FAST algorithm

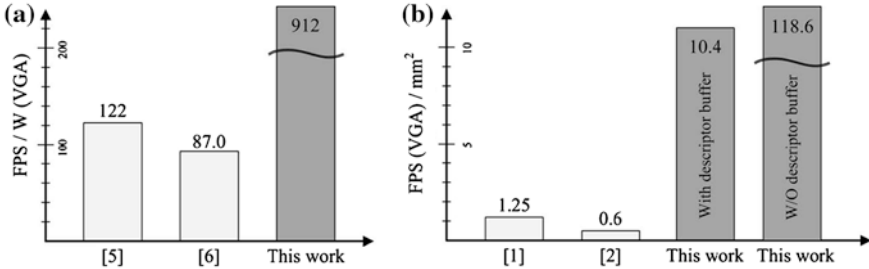
	FAST (Worst)	FAST (w/o ER)	FAST (with ER)
BRIEF (256 point, 32 byte)	1080p 111 fps	1080p 150 fps	1080p 235 fps
BRIEF (256 point, 64 byte)	1080p 96.9 fps	1080p 125 fps	1080p 179 fps
BRIEF (512 point, 32 byte)	1080p 77.3 fps	1080p 94.3 fps	1080p 122 fps
BRIEF (512 point, 64 byte)	1080p 55.0 fps	1080p 63.1 fps	1080p 74.4 fps
BRIEF (1024 point, 32 byte)	1080p 34.9 fps	1080p 38.0 fps	1080p 41.8 fps
BRIEF (1024 point, 64 byte)	1080p 20.1 fps	1080p 21.1 fps	1080p 22.3 fps
BRIEF (2048 point, 32 byte)	1080p 10.9 fps	1080p 11.2 fps	1080p 11.5 fps

200 MHz operating frequency with 256 descriptors per image. The required execution time for matching increases quadratically as the target resolution increases, because the exact NN is used for a matching algorithm [9]. When we use additional techniques such as approximate nearest neighbor to train k-dimensional tree or vocabulary tree, the comparison number for matching test can be reduced [2, 33]. We evaluate the raw performance of the proposed hardware without additional hardware.

Due to the lack of previous works which implement FAST and BRIEF at the same time, we compared our chip to [6, 7] having equivalent objectives. It is important to determine the number of interest points for comparison, but the average number of interest points is not written explicitly in [6, 7]. Since 16,382 vectors are used to represent 50 objects in database [6], we can estimate 328 vectors per object. Many techniques are proposed to predict region for each target object in a given image and detect interest points in the region [6, 7]. Since the selected region and the image used to generate vectors for database might be similar, it is reasonable to estimate that the number of interest points per each

**Table 8.3** Comparison of performance and power efficiency with [6] and [7]

	Technology ( $\mu\text{m}$ )	Power (mW)	Frame rate (FPS)	Frame rate/ power(fps/W)
ISSCC 2009 [6]	0.13	496	60 fps @ VGA	122
ISSCC2010 [7]	0.13	345	30 fps @ VGA	87.0
This work (512 interest points)	0.13	182	94.3 fps @ HD 166 fps @ VGA	518 912

**Fig. 8.23** Performance improvements with respect to **a** power-efficiency and **b** area-efficiency

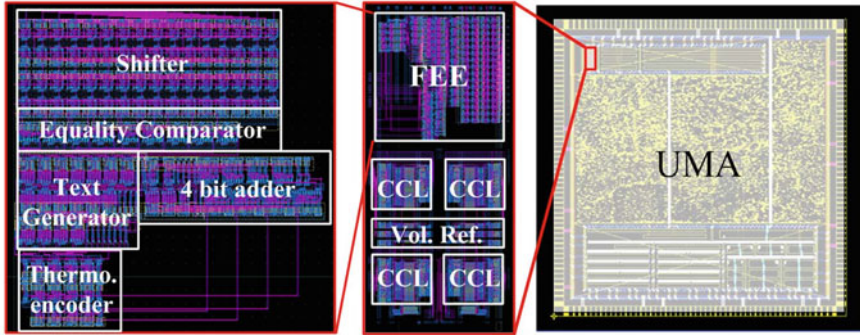
object is about 330. In a conservative way, we assume that 512 interest points are detected in each frame.

The entire hardware achieves 166 and 94.3 fps in VGA and full HD resolution at 200 MHz operating frequency, respectively as shown in Table 8.3. The total chip power consumption is about 182 mW at 1.2 V power supply where the unified datapath uses 8.75 % of the die area ( $1.4 \text{ mm}^2$ ).

Figure 8.23 shows the improvement of power- and area-efficiency. The proposed high-performance and small-area recognition accelerator achieves  $7.45\times$  higher fps/Watt and  $85.2\times$  higher fps/area compared to the previous object recognition processors [6, 7].

## 8.5 Application

A unified media application processor (UMAP) has used the proposed interest point detector, named a mixed-mode feature extraction engine (FEE), for real-time and energy-efficient vision processing [34]. Previous media application processor detects corner point using general purpose parallel processing cores. Although the cores are widely used for various applications such as vision and graphics processing, they causes serious power/area overhead compared to dedicated hardware. Since FEE is well optimized dedicated hardware with small power/area overhead, it can replace the cores when corner detection is required.



**Fig. 8.24** Layout result implemented in 90 nm CMOS process within  $120 \times 280$   $\mu\text{m}$

Since the application of UMAP is specified as it detects only black corner points on white background, FEE considers only white string case as shown in Fig. 8.2. Only 2 alphabets (white and gray) are sufficient for the segment test and the size of string and text is reduced to 9-bit and 16-bit, respectively. Furthermore, FEE is custom-designed in mixed-mode integrating four analog current contention logics (CCLs) for low-power high-speed corner detection. A set of hardware including FEE, memory buffer and dedicated controller uses only 4,500 logic gates while FEE consists of 300 logic gates as the result of area optimization. Figure 8.24 shows a result of layout with a custom-designed FEE implemented in 90 nm CMOS process within  $120 \times 280$   $\mu\text{m}$ .

FEE saves significant amount of dynamic power while improving system energy efficiency in UMAP. FEE replaces the parallel processing core cluster which is the most power hungry IP in augmented reality processing (target application), since it consumes only 2.25 mW on the average in corner detection while parallel processing cores require 69 mW. UMAP achieves 1 mJ/frame energy-efficiency which is up to 2.8 better than the state-of-the-art AR processors and 96.7 % of cluster power and 99.1 % of target detection time are saved in real operation.

FEEs, implemented by synthesized and by full-custom designed, operate up to 400 MHz and 1.75 GHz, respectively. Because of the high-bandwidth of the FEE, it can be adopted for vision acceleration even in high-end GPU and commercial mobile CPU, which have the operation frequency as 1.5 and 1.2 GHz, respectively.

## 8.6 Conclusions

Interest point detection and matching is one of the most important algorithm in vision based DAS. A number of state-of-the-art works are introduced in order to increase robustness and performance. Until now, SIFT and SURF are treated as de facto standard in interest point and many hardware accelerators are introduced to accelerate them. In this chapter, we mentioned that FAST-BRIEF is good

alternative of SIFT and SURF because it can achieve similar robustness with much smaller overhead than SIFT and SURF. But, FAST-BRIEF also should be accelerate in hardware to support real-time DAS.

It is impossible that any algorithm can be implemented in hardware efficiently because of the difference between PC and silicon environment. Joint algorithm-architecture optimization should be considered to enhance performance, area, and power efficiency of hardware accelerator. So, a new segment test for interest point detection based on a string searching algorithm is proposed. It is logically equivalent to the original segment test, but it performs the segment test more efficiently in terms of area, performance, and power than the original by implementing a dedicated accelerator.

Independent but functionally sequential processes such as interest point detection and matching cause two critical problems; (1) area-efficiency loss and (2) unbalanced workload. The unified datapath can resolve these problems but it is difficult to implement two different algorithm in a datapath. In point of view of overall architecture, we shows that FAST and BRIEF can be merged into the same data-path, because the accelerator based on the proposed algorithm has a similar operation flow to BRIEF. The proposed scalable/reconfigurable architecture based on unified datapath can increases its performance linearly according to the number of cores.

Since the proposed hardware consists of 78.3 k logic gates without descriptor buffer, it is one of the smallest interest point detection and matching hardware. Nevertheless, overall hardware achieve 94.3 fps in HD ( $1920 \times 1080$ ) resolution at 200 MHz operating frequency. So, the proposed hardware gives a practical solution to detect corners from high resolution video stream processing on mobile and embedded environment.

## References

1. G. Klein, D. Murray, Simulating low-cost cameras for augmented reality composing. *IEEE Trans. Visual. Comput. Graphics* **16**(3), 369–380 (2010)
2. M. Ebrahimi et al., Adaptive sampling for feature detection, tracking, and recognition on mobile platforms. *IEEE Trans. Circuits Syst. Video Technol.* **21**(10), 1467–1475 (2011)
3. S. Kyo et al., in *A 100GOPS in-vehicle vision processor based on a ring connected 128 4-way VLIW processing elements*, Proc. IEEE Symp. VLSI Circuits(VLSIC '08), 28–29 June 2008
4. D. Geronimo et al., Survey of pedestrian detection for advanced driver assistance systems. *IEEE Trans. Pattern Anal. Mach. Intel.* **32**(7), 1239–1258 (2010)
5. D. Lowe, Distinctive image features from scale-invariant keypoints. *ACM Int. J. Comput. Vision* **60**(2), 91–110 (2004)
6. J.-Y. Kim et al., A 201.4GOPS 496 mW real-time multi-object recognition processor with bio-inspired neural perception engine. *ISSCC Dig. Tech. Papers*, 150–151 Feb 2009
7. S. Lee et al., A 345 mW heterogeneous many-core processor with an intelligent inference engine for robust object recognition. *ISSCC Dig. Tech. Papers*, 332–333 Feb 2010
8. E. Rosten et al., FASTER and better: a machine learning approach to corner detection. *IEEE Trans. Pattern Anal. Mach. Intell.* **32**(1), 105–119 (2010)
9. M. Calonder et al., BRIEF: binary robust independent elementary features. In *Computer Vision–European Conference on Computer Vision (ECCV) 2010* (pp. 778–792). Springer, Berlin Heidelberg (2010)

10. J.-S. Park, H.-E. Kim, L.-S. Kim. A 182 mW 94.3 f/s in Full HD Pattern-Matching Based Image Recognition Accelerator for an Embedded Vision System in 0.13-um CMOS Technology. *IEEE Trans. Circ. Sys. Video Tech.* **23**(5), 832–845 (2013)
11. H. Bay et al., Speeded up robust features (SURF). *Comput. Vis. Image Underst.* **110**(3), 346–359 (2008)
12. E. Mair et al., Adaptive and generic corner detection based on the accelerated segment test. *Lect. Notes Comput. Sci.* **6312**, 183–196 (2010)
13. Marek Kraft et al., System on Chip coprocessors for high speed image feature detection and matching, advanced concepts for intelligent vision systems. *Lect. Notes Comput. Sci.* **6915**(2011), 599–610 (2011)
14. K. Mikolajczyk et al., A performance evaluation of local descriptors. *IEEE Trans. Pattern Anal. Mach. Intell.* **27**(10), 1615–1630 (2005)
15. S. Winder et al., Picking the best daisy. in *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 178–185 (2009)
16. A. Torralba, Small codes and large image databases for recognition, in *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1–8 (2008)
17. M. Ozuysal et al., Fast keypoint recognition using random ferns. *IEEE Trans. Pattern Anal. Mach. Intell.* **32**, 448–461 (2010)
18. V. Chandrasekhar et al., Compressed histogram of gradients: a low bitrate descriptor. *Int. J. Comput. Vision* **94**(5), (2011)
19. B. Girod et al., Mobile visual search. *IEEE Signal Process. Mag.* **28**(4), (2011)
20. E. Rublee et al., ORB : an efficient alternative to SIFT or SURF. *IEEE Int. Conf. Comput. Vision (ICCV)*, 2564–2571 (2011)
21. Y. Moko et al., Implementation and evaluation of FAST corner detection on the massively parallel embedded processor MX-G. *IEEE computer society conference on computer vision and pattern recognition workshops (CVPRW)*, 157–162 (2011)
22. J. Clemons et al., EFFEX: an embedded processor for computer vision based feature extraction. *Design automation conference (DAC)*, 1020–1025, (2011)
23. J. Svab et al., FPGA-based speeded up robust features. *IEEE international conference on technologies for practical robot applications*, 35–41 (2009)
24. K. Dohi et al., Pattern compression of FAST corner detection for efficient hardware implementation. *Int. Conf. Field Programmable Logic Appl. (FPL)* **5**(7), 478–481 (2011)
25. R.N. Horspool, Practical fast searching in strings. *Softw. Pract. Experience* **10**(6), 501–506 (1980)
26. R. S. Boyer, J. S. Moore, A fast string searching algorithm. *Commun. ACM* **20**(10), (1977)
27. R. Baeza-Yates et al., A new approach to text searching. *Commun. ACM* **35**(10), 74–82 (1992)
28. M. Crochemore et al., Deux méthodes pour accélérer l’algorithme de Boyer-Moore. *Théorie des Automates et Applications, Actes des 2e Journées Franco-Belges*, 45–63 (1991)
29. R.M. Karp et al., Efficient randomized pattern-matching algorithms. *IBM J. Res. Dev.* **31**(2), 249–260 (1987)
30. R. Baeza-Yates et al., Average running time of the BMH heuristic. *Theo. comput. Sci.* **92**, 19–31 (1992)
31. E. Sall, M. Vesterbacka, Comparison of two thermometer-to-binary decoders for high-performance flash ADCs. *NORCHIP conference*, 253–256 (2005)
32. E. Rosten et al., Fusing points and lines for high performance tracking. *IEEE Int. Conf. Comput. Vision* **2**, 1508–1515 (2005)
33. D. Nister, H. Stewenius, Scalable recognition with a vocabulary tree. *IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recogn.* **2**, 2161–2168 (2006)
34. H. Kim et al., A 1 mJ/Frame unified media application processor with dynamic analog-digital mode reconfiguration for embedded 3D-Media contents processing. *IEEE J. Solid-State Circuits* **48**(8), (2013)

# Chapter 9

## Software Development Environment for Automotive SoC

Jeonghun Cho

**Abstract** Model-based development (MBD) uses models to describe various development processes and products. The model can provide development processes, requirement traceability, verification, validation and documentation. MBD can reduce the errors and increase the maturity level of the development process. This system is being used increasingly for the development of automotive software systems. AUTOSAR is a type of model-based development method for an automotive electronic system, which was developed by the automotive industry. AUTOSAR provides several models for model-based system development, e.g. the software component model, platform model and system model. The basic benefits of AUTOSAR in the development process are the relocation of the software components regardless of the hardware and the reuse of software components already developed. The case study of AUTOSAR development for automotive SoC is described. The automotive SoC consists of multiple processors, a customized vision process engine for an automotive vision system and peripheral. Each component of the SoC is mapped to the AUTOSAR layered architecture, such as MCAL, ECU abstraction and CCD.

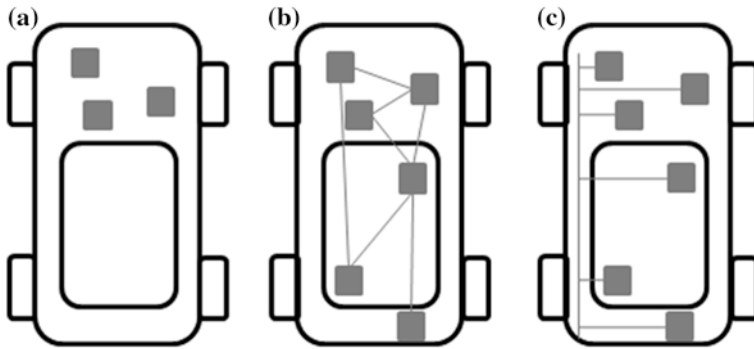
### 9.1 Introduction

In 1960, the Volkswagen Beetle was first equipped with electric circuits. In 1978, the Mercedes S-series was equipped with embedded systems for the Anti-lock Brake System (ABS). The early automotive electronic system consisted of an electronic control Unit (ECU), sensor and actuator. These systems operate autonomously and rarely share data with other systems. Currently, in the 2010s,

---

J. Cho (✉)

School of Electronics Engineering, Kyungpook National University, Daegu, South Korea  
e-mail: jcho@ee.knu.ac.kr



**Fig. 9.1** ECU topology. **a** Single ECUs, **b** connected ECUs, **c** networked ECUs

luxury cars are equipped and connected with almost a hundred ECUs. As the functionalities of vehicles increase, automotive electronic systems are becoming exponentially complicated for the entire life cycle, such as development, testing, production and maintenance.

### 9.1.1 Overview

In the early days, ECUs for a specific service were not connected to the others and worked independently, as shown in Fig. 9.1a. With time, various and complex functionalities from the customer has required complicated control mechanisms of automotive electronic systems. Consequently, ECUs are connected to others as a distributed control system via a network. The modern complicated engine control mechanism is based on a distributed control system. Currently, complex functions, such as Adaptive Cruise Control (ACC) and Adaptive Front Lamp System (AFLS), demands a sophisticated distributed control strategy with entire automotive domains.

To share some data from other ECUs, Fig. 9.1b shows the direct connections. This network topology however, causes an increase in the wire cost, which limits the flexibility because the network topology depends on the ECU structure. Therefore, most distributed control systems use a bus-based network to provide flexibility and scalability, as shown in Fig. 9.1c. These fully networked ECUs consist of smart sensors and smart actuators, which have their own network interfaces that they can connect to a network independently and ensure high flexibility and scalability. On the other hand, the greater the increase in the number of ECUs, smart sensors, and actuators, the more complicated the design, verification and validation of distributed control systems will be. Moreover, safety and dependability issues are critical in an automotive environment because increasing the network traffic can cause congestion, delays and omission of networks. To

solve this problem, the classic ‘Divide and conquer’ approach is used widely in the automotive industries. That is, automotive systems separate a few subsystems like the powertrain, chassis, body, and multimedia. Each subsystem has an independent network: FlexRay [1] for chassis, CAN [2] for body, and MOST [3] for multimedia. The gateway can complement the problems with integrating several heterogeneous networks but the bottleneck may not be good for the performance.

Model-Based Development (MBD) is a useful method in engineering for solving these problems in automotive software development. The benefits of model-based engineering include the ease to represent, simulate and generate auto code. AUTomotive Open Software Architecture (AUTOSAR) [4] developed by the automotive industry is a MBD method and is an open standard specification for the development of automotive electronic systems. AUTOSAR provides a software model that consists of a software component and interface, platform model, which contains an OS, communication and basic software modules, and a system model, which represents system topology and hardware descriptions for MBD.

### ***9.1.2 Importance of Software Environment***

Laws and regulations, such as mandatory safety equipment and exhaust emissions, can only be achieved with electronic systems. Customers require for more functionality and safety, and OEM wants to launch new vehicles with novel innovative features. Software technology of automotive electronic systems can become a good trade-off between cost and development. To apply new functionality to an automotive electronic system, identification of the requirements of new functionality is needed. The pressure of the requirements affects the architecture of automotive electronic systems, which consists of hardware components and software components. In addition, the economic constraints bring about an efficient development process of an automotive system, such as standard of hardware/software architecture for independence and reuse. The growing complexity of the software in automotive systems requires a well-defined development process to verify and validate the software.

Several issues need to be considered in automotive electronic systems as follows:

- (1) Requirements of regulation, law and customers, such as safety, economy, comfort, and services.
- (2) Complexity of ECUs and ECU networks as the functionality increases.
- (3) Use of Heterogeneous ECUs and ECU networks (CAN, LIN [5], FlexRay, MOST, etc.).

To fulfill these issues, a complex distributed automotive system is required. Automotive functions is a type of distributed system that consists of ECUs and ECU networks but still not structured with new technologies [6]. For some function, the ECUs are grouped into several subdomains, for example, power train,



body comfort, chassis and infotainment. The interfaces of ECU networks are not standardized between these domains, even within the same domain. In addition, the interrelation and the interaction are not considered between the nodes of the network. Software development processes have progressed according to the record of each subdomain for different ECUs and separated for a long time. The functional requirements of an automotive system were assigned to the ECUs one by one based on one unique ECU per function and very small interconnections.

The increasing interconnection of ECUs result in high complexity and high costs for the whole processes, e.g. development, verification and validation, etc. An informal development processes with unsuitable components of a distributed automotive system make these problems more complicated. The collaboration between several companies, OEM, supplier tier 1 and supplier tier 2, has made automotive systems more complex. Until the 2000s, the appropriate abstractions of automotive software architecture and integration concepts were not proposed. The automotive software architecture was developed with a single development strategy for a specific company but not a standardized solution for several automotive companies. Currently, many advanced automotive functionalities are distributed over several ECUs on the entire subdomains. For example, the indicators of functionalities on the dashboard were controlled by Electronic Stability Control (ESC) ECU, Electronic Power Steering (EPS) ECUs and Transmission Control Unit (TCU) in luxury vehicles. In addition, advanced automotive functionalities, as known as the Advanced Driver Assistance System (ADAS), cannot be implemented with a simple interconnection of ECUs. For example, the Lane Keeping Assistance System (LKAS) requires complex and safe interconnections of ECUs across a range of subdomains, EPS, vision system and powertrain domain. The traditional development strategy of automotive functionalities will require more interconnection for new advanced functionality.

To provide the standard automotive system architecture, OEMs and several suppliers constructed a partnership in 2003. AUTOSAR consists of 10 core partners to establish standard automotive industry de-facto for an automotive system architecture. The core partners are the BMW Group, Bosch, Continental, Daimler, Ford, GM, PSA Peugeot Citroën, Siemens VDO, Toyota and Volkswagen.

### ***9.1.3 Software Environments for Automotive System***

Offene Systeme und deren Schnittstellen für die Elektronik in Kraftfahrzeugen (OSEK) [7], in English “Open Systems and their Interfaces for the Electronics in Motor Vehicles”, is an open standard for an embedded real-time OS with a communications stack (COM) and network management (NM) of automotive electronic systems. OSEK was developed to provide standard real-time OS and software architecture for various automotive ECUs. German OEMs (BMW, Robert Bosch GmbH, Opel, Siemens VDO, DaimlerChrysler and Volkswagen Group) and the University of Karlsruhe developed OSEK in 1993. The French automotive

OEMs, which had a similar project called VDX (Vehicle Distributed eXecutive), Renault and PSA Peugeot Citroën, joined the OSEK consortium in 1994. At that time, OSEK/VDX was started. OSEK is standardized in ISO 17356 [8] and AUTOSAR accept the OSEK specifications as AUTOSAR OS. The AUTOSAR OS provides many features that include OSEK OS, OSEKtime and OSEK COM.

EAST-ADL is an ADL (Architecture Description Language) for automotive electronic systems, developed in Advancing Traffic Efficiency and Safety through Software Technology 2 (ATESST2) [9]. The system was designed to complement AUTOSAR with descriptions at a higher level of abstractions. Aspects covered by EAST-ADL include vehicle features, functions, requirements, variability, software components, hardware components and communication. Currently, it is maintained by the EAST-ADL Association in cooperation with the European FP7 MAENAD project.

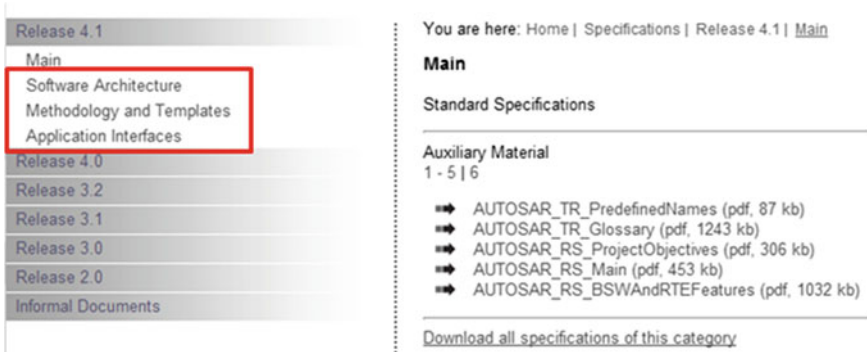
The first phase of AUTOSAR started in 2003, and at the end of 2006, the first AUTOSAR products were available on the market. For 3 years, 52 premium members joined the AUTOSAR consortium and made major specifications. The premium members were composed mainly of companies of suppliers, software and semiconductor industries. For example, the Hyundai-Kia Motor Company (HKMC) and Electronics and Telecommunications Research Institute (ETRI) are the premium member of AUTOSAR. After that, associate members, development members and attendees were added. Daesung Electronics and MANDO Corporation are currently associate partners.

The members of AUTOSAR agree that AUTOSAR can control the complexity of the automotive electronic system and increase the quality and earnings ratio of products. In 2008, the First AUTOSAR open conference & Eighth premium member conference was held in the USA. Recently, the fourth AUTOSAR open conference and fifth AUTOSAR open conference were held in 2012. In 2013, which is the 10th anniversary of AUTOSAR, AUTOSAR will hold the world tour.

### ***9.1.4 The Objectives of AUTOSAR***

The concept of AUTOSAR is the relocation of software components regardless of the hardware and reuse of software component already developed [10]. The AUTOSAR project objectives are used as the basis of refinement into the AUTOSAR main requirements, whether technical or non-technical. The AUTOSAR project objectives are as follows [11]:

- (1) Transferability of software
- (2) Scalability to different vehicles and platform variants
- (3) Different functional domains
- (4) Definition of an open architecture
- (5) Dependable systems
- (6) Sustainable utilization of natural resources



**Fig. 9.2** Main areas of AUTOSAR

- (7) Collaboration between various partners
- (8) Standardization of basic software functionality of automotive ECUs
- (9) Applicable automotive international standards and state-of-the-art technologies.

The results of every modeling of AUTOSAR are the exported AUTOSAR Extensible Markup Language (AR-XML), which is specified in UML 2.0. According to the top-down approach, AUTOSAR begins with description language. AUTOSAR descriptions need to be specified step-by-step via the meta-model level to a concrete model as a concrete XML file. Figure 9.2 presents the main specifications of AUTOSAR.

The basic benefits of AUTOSAR in the development process are the reuse and relocation of application software, as known as Software Component, regardless of hardware and lower layers, which is known as the basic Software Module, as shown in Fig. 9.3. According to this component-based approach, AUTOSAR Runtime Environment (RTE), which implements the concept of a Virtual Function Bus (VFB) is needed for the interaction between the application layers and BSW. The main concept is that the applications do not need to know the specific implementations and communication methods under RTE when they communicate with each other. The application developer requires no further knowledge about the system services and resources, even if there is knowledge about the interfaces.

AUTOSAR was established as a partnership to specify the automotive industry de-facto [12–15]. The AUTOSAR consortium attempts to accept as many existing solutions as possible that are needed and comply with the AUTOSAR concept. For example, OSEK for AUTOSAR OS and CAN for a communication stack are adopted. In addition, the AUTOSAR consortium works together with new and other standardization groups to try to accept their standards for the diversity of selection, e.g. TTCAN, LIN, SAE J1939, FlexRay, MOST, even Ethernet. In cases of a communication stack, one standard is a good solution for a specific case but not for all. In addition, a range of communication standards can be used for

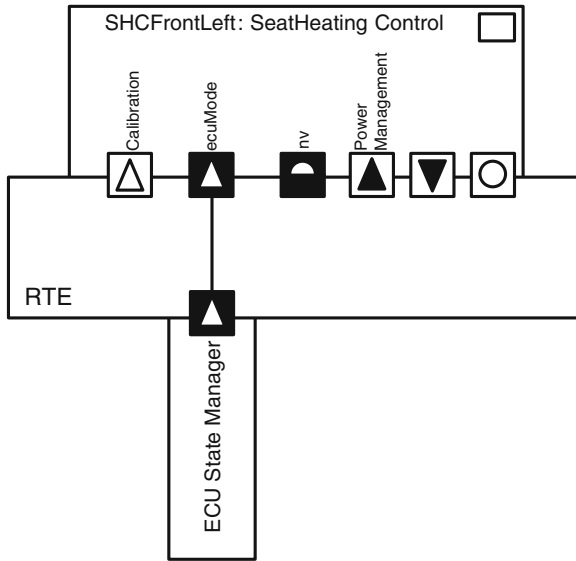


Fig. 9.3 AUTOSAR concept and layer



Fig. 9.4 European Projects

automotive electronic systems simultaneously as the requirements and characteristics of the system. AUTOSAR is a project of software technology for an automotive embedded system and safety. Several research projects for embedded software architecture have been performed, as shown in Fig. 9.4. For example, Information Technology for European Advancement—Embedded Architecture Software Technology/Embedded Electronic Architecture (ITEA-EAST/EEA) [16], Architecture Electronique Embarquée (AEE) [17], in English Embedded Electronic Architecture, and ATESST were finished at this time.

## 9.2 AUTOSAR Architecture

To overcome the drawbacks of traditional development processes and fulfill the variety of requirements from various customers, AUTOSAR presents a model-based development process for automotive open system architecture [18–25]. The

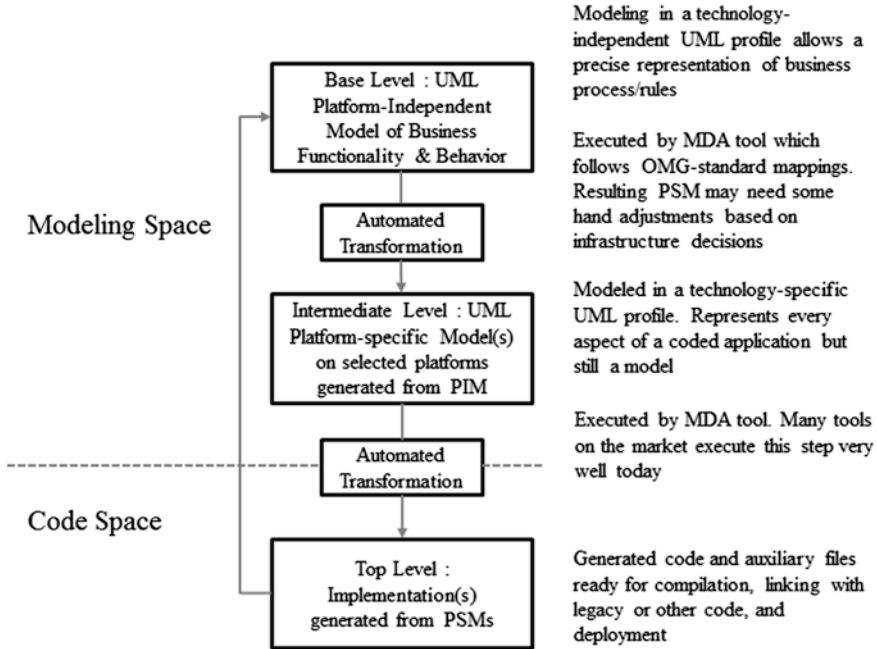


Fig. 9.5 Model-driven architecture of OMG

software architecture, ECU hardware and network topology are designed in software components and interfaces, which are described in AR-XML, which supports the AUTOSAR development process from the design to the integration [26, 27]. The modeling elements of AUTOSAR are defined based on the rule of the OMG Meta Object Facility, as shown in Fig. 9.5 [28].

### 9.2.1 AUTOSAR Concept

The first phase of the AUTOSAR development process is a design of the software component architecture. Figure 9.6 presents the example of AUTOSAR software component architecture with the AUTOSAR Tool.

AUTOSAR software architecture is composed of software components and interfaces. The boxes in Fig. 9.6 show the software components, and the inter-connections between the boxes mean the interfaces. The communication ports of the software components are placed at the border of the boxes. A communication port is either a provided port or a required port depending on the direction of data flow. The provided ports send the data only within communication, required ports receive the data only. AUTOSAR provides two types of interfaces, which are used widely in an elementary communication pattern:

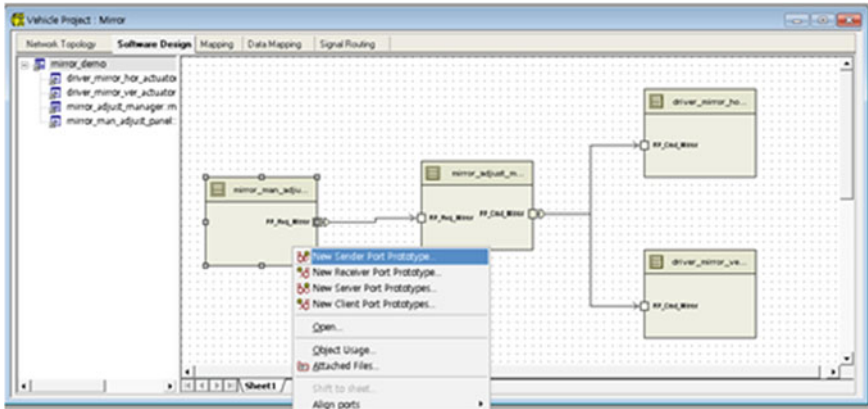


Fig. 9.6 Software architecture

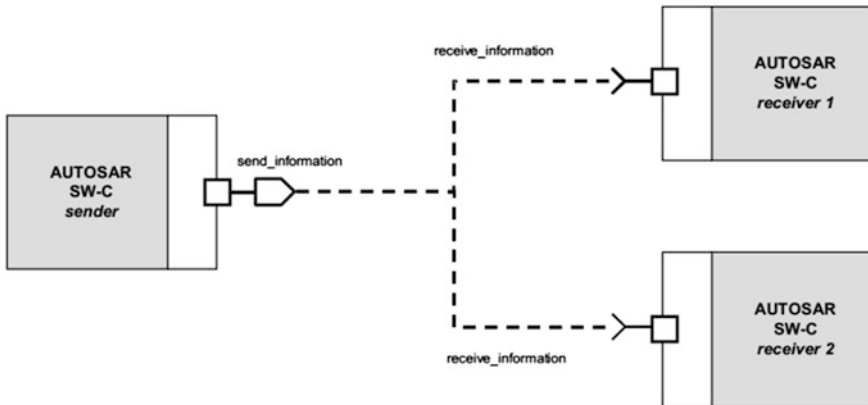


Fig. 9.7 AUTOSAR sender–receiver interface

- (1) Sender/receiver interface—A sender distributes information to one or several receivers, or one receiver obtains information from several senders, as shown in Fig. 9.7. This interface provides a solution to the asynchronous distribution of data where a sender transmits data to one and more receivers. The sender is not blocked and does not expect a response from receivers. The sender just provides the data and the receivers decides autonomously when and how to use the data. The sender does not know the id and number of receivers of a transmission. As an example, the sender can be a road speed sensor to transfer the vehicle speed value, and the receivers can be an instrumental panel to display the vehicle’s velocity and a door lock module to lock the doors at a specified speed.

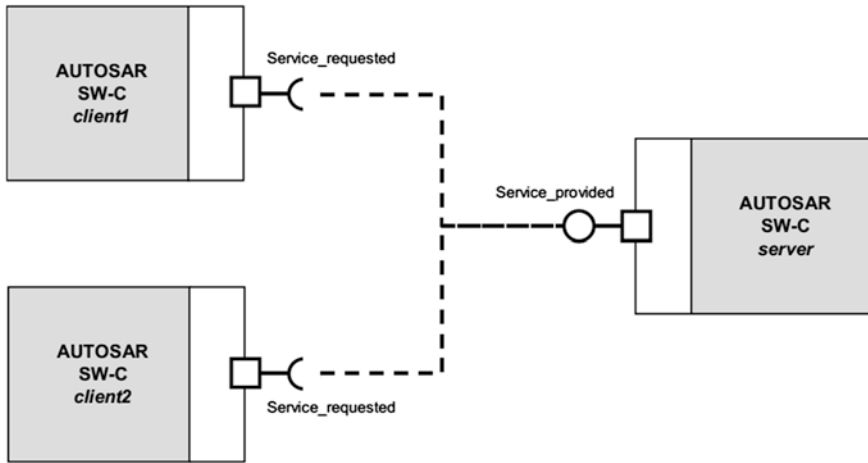


Fig. 9.8 AUTOSAR client–server interface

(2) Client/server interface—The server provides the operations and several clients can invoke those operations, as shown in Fig. 9.8. Each operation has a signature that consists of a name and list of parameters. Each parameter has a name, type and direction, which is either in, out and in–out. The client–server is a widely used communication interface in distributed systems that provide services and the client uses a service. The client invokes the request with a set of parameters. The server waits for incoming requests from a client, performs the requested service and returns a response to the request of the client. An AUTOSAR software component can be a client or a server depending on the direction of invocation. A single software component can be both a client and server depending on the software implementation. The client can be blocked with synchronous communication and non-blocked with asynchronous communication after the service request is invoked to the server until the response is received from the server. For example, the server can be a door lock module providing lock/unlock operations, and the client can be a remote key or central door lock module.

The software component architecture is designed regardless of the technical architecture, such as ECU hardware and network topology, that the software components will run on [29, 30]. Some software components can run on the same ECU and others may be able to run on different ECUs that are connected to the network. The communication between the software components is either an intra-ECU communication or an inter-ECU communication. The AUTOSAR RTE acts as a communication layer for inter-ECU communication and intra-ECU communication. The RTE provides communication abstraction of the AUTOSAR software components by the same interface and services through either inter-ECU communication, such as CAN, LIN, FlexRay and MOST, or intra-ECU communication, such as shared memory and message queue. As the communication

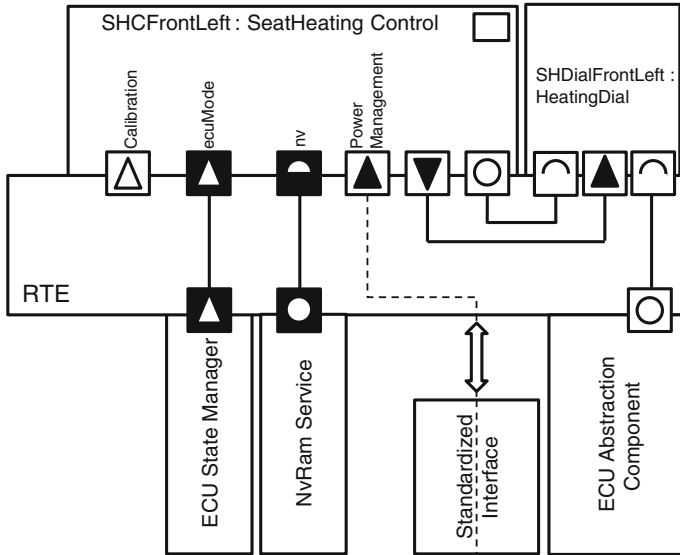


Fig. 9.9 RTE and communication

requirements of the software components are application dependent, the RTE needs to be customized for appropriate communication, as shown in Fig. 9.9. Most parts of the RTE will be generated to provide suitable communication services, whereas the communication resources, such as the network topology, are available. Some parts of the RTE might be customized via configuration. Therefore, the generated RTE will be suitable for a specific ECU, not for all the ECUs in the AUTOSAR system, and the software component will be separate from the hardware implementation details.

The hardware architecture was designed separately to the development of software component architecture. AUTOSAR aims to design the topology of a network of ECUs. Figure 9.10 gives an example of the hardware architecture with AUTOSAR tool. The example contains two ECUs connected to a CAN network.

Based on the defined software architecture and network topology, the software components will be mapped to the specific ECUs. In AUTOSAR standards, Software Component Template [31] describes the requirements that are required to support the interoperability and compliance between the AUTOSAR development tools.

### 9.2.2 Basic Software (BSW)

The AUTOSAR standard describes the list of Basic Software Modules (BSWs) [32]. These modules are structured in a layered software architecture. The BSW modules



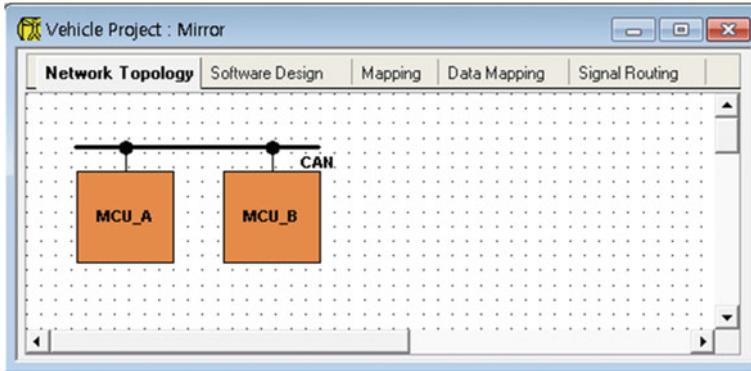


Fig. 9.10 Network topology

System Services	Memory Services	Communication Services	I/O Hardware Abstraction	Complex Drivers
Onboard Device Abstraction	Memory Hardware Abstraction	Communication Hardware Abstraction		
Microcontroller Drivers	Memory Drivers	Communication Drivers	I/O Drivers	

Fig. 9.11 Simple BSW modules

consist of device drivers, communication and I/O drivers, AUTOSAR services and AUTOSAR operating system, as shown in Fig. 9.11. The AUTOSAR standard describes more than 70 BSW modules. As an update of the AUTOSAR standard, the number of BSW modules can be changed. For example, the BSW modules that support the multicore microcontroller in AUTOSAR standard 4.0 need to describe more services than the BSW modules, which support a single core only in the former version.

Each of these BSW modules has a well-defined interface for communication with other modules and the RTE, shown as Fig. 9.12. Software components that are on the RTE cannot access any of the BSW interfaces directly but through the RTE only. The interface consists of Application Programming Interface (API) functions, data types and return codes from the APIs. Three types of Interfaces are used for communication between BSW modules, AUTOSAR interface, Standard AUTOSAR interface, and Standard interface. In addition, AUTOSAR defines the configuration parameters for BSW modules. An AUTOSAR interface defines the communication data between software components. The AUTOSAR interfaces are independent of specific programming languages, ECUs and networks. These are used to define the

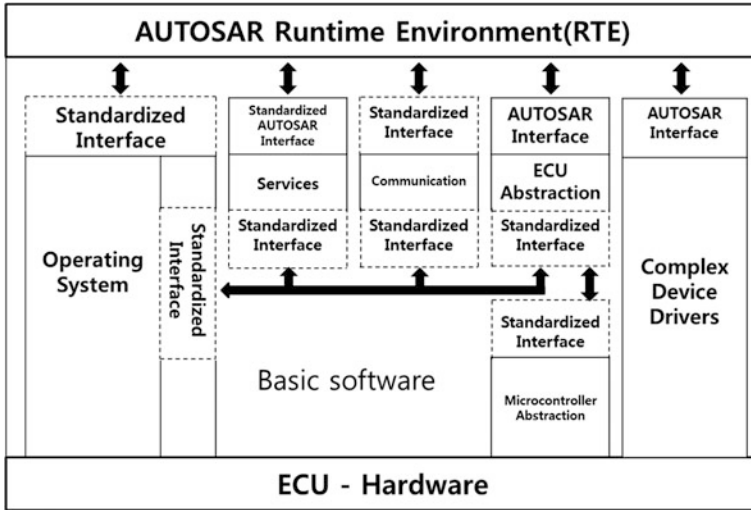


Fig. 9.12 Interfaces of the BSW modules

ports of the software components. Software components can communicate with each other through these ports, regardless of whether they are a send–receiver port or client–server port. Communication between software components is implemented by the RTE locally or via network. In most cases, the AUTOSAR interface should be able to transmit the data through the interface not only locally but also via a network. A Standard AUTOSAR interface is a type of AUTOSAR interface, in which the syntax and semantics are standardized in AUTOSAR. Standardized AUTOSAR interfaces are used typically to define the AUTOSAR services that are standardized services in BSW modules. The AUTOSAR standardized services are provided to software component via the Standardized AUTOSAR interface. A Standardized interface is an API, which is the standardized without an AUTOSAR interface technique in AUTOSAR. In most cases, standardized interfaces are defined for a specific programming language, such as C. Therefore, standardized interfaces are used between software component modules that are placed on a single ECU to ensure they are implemented with the same programming language. The standardized interface cannot ensure that the communication data between the software components can be transmitted via a network.

### 9.2.3 Run-Time Environment (RTE)

The RTE is an important part of the AUTOSAR architecture. The RTE is the implementation of the interfaces of the AUTOSAR VFB for a specific ECU [33]. The RTE provides communication services between the AUTOSAR software components and basic software modules, which include the OS and communication

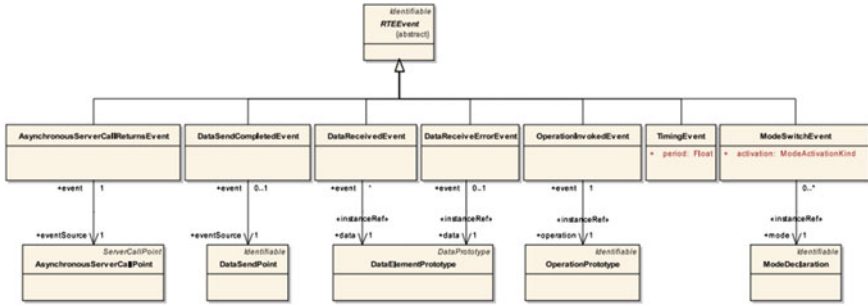


Fig. 9.13 Different types of RTEEvent

service. The RTE covers the variable elements of the system services that originate from the various mapping of software components to the ECUs and standardized RTE services. The RTE is generated for each ECU for code optimization of the RTE. In the part under the RTE, the BSW modules can access other modules via the appropriate interfaces directly, such as AUTOSAR interface and AUTOSAR standardized interface. In the part on the RTE, the AUTOSAR software components can communicate with each other via ports only and cannot access the BSW modules directly. The internal behavior of the AUTOSAR software components is implemented as runnable entities that mean the sequences of instruction. A runnable entity is a scheduling unit of an AUTOSAR software component and a basic control unit that is triggered by RTE. Runnable entities need to be mapped to tasks that are the basic schedulable unit of OS. The mapping of runnable entities needs to be described in the ECU configuration description. All runnable entities are triggered by RTEEvent. The possible RTEEvents are described in the meta-model, as shown in Fig. 9.13. The RTE supports a range of events, e.g. timer expired, data arrived and server call returns.

No RTEEvent is specified for the runnable entity, and the runnable entity cannot be activated by RTE. The runnable entities are categorized as follows:

- (1) Category 1: runnable entities do not support WaitPoints and need to terminate in a specific time. Because of these constraints, the category 1 runnable entities can be mapped to the basic tasks of AUTOSAR OS. The VFB specification distinguishes between category 1A runnable entities and category 1b.
- (2) Category 2: runnable entities have more than one WaitPoints or always wait for the response to return. Typically, category 2 runnable entities need to be mapped to the extended tasks of AUTOSAR OS, because the extended tasks support the waiting state only. Any runnable entities with WaitPoints or wait for the response will be classified as category 2 runnable entities.
- (3) Category 3: currently out of scope of the RTE specification.

The RTE generation needs to fit a specific ECU and system configuration. An RTE implementation needs to provide the precise functionality in a given specific configuration. The RTE generator shall produce the same RTE API and RTE code

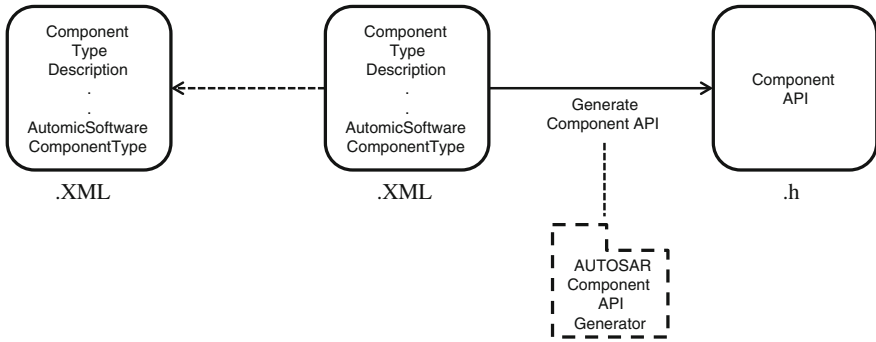


Fig. 9.14 RTE contract phase

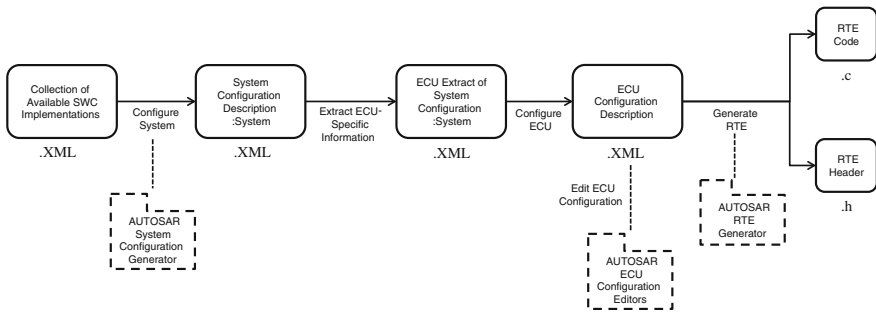
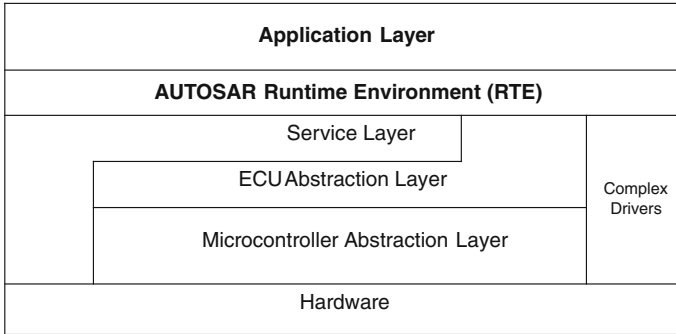


Fig. 9.15 RTE generation phase

when the input description and configuration are the same. The generation process is divided into two phases, contract phase and generation phase, as shown in Figs. 9.14 and 9.15.

- (1) Contract phase: The limited information on the AUTOSAR interface definitions of a component is used to generate application header files. The application header files define the contract between the component and RTE. For example, an AUTOSAR software component has a send port P with a data element D, and the contract phase will generate the RTE API function Rte\_Send\_P\_D. The software component uses this RTE API function to send data element D via send port P.
- (2) Generation phase: all relevant information about the components, such as mapping to ECUs and communication topology, is used to generate the RTE and the loc configuration optionally. One RTE is generated for each ECU in the AUTOSAR system. The application header files and all necessary BSW code need to be used for an executable file for an RTE.

The development of AUTOSAR software component with RTE specific APIs, which are known as application header files that are generated from the software



**Fig. 9.16** AUTOSAR layered software architecture

component internal behavior description in the contract phase, is unrelated to the ECU hardware. Because the generated application header files contain information on ports access and send/receive data, the source code of the software component can be provided regardless of the communication method via BSW. When the software component with application header files are compiled successfully, the object files will be analyzed, such as information on the actual memory needs for ROM and RAM. The delivery package of software components are as follows:

- (1) Software component type description
- (2) Software component internal behavior description
- (3) The actual software component implementation and compiled software component
- (4) Software component implementation description.

### ***9.2.4 AUTOSAR Layered Architecture***

AUTOSAR defines the layered software architecture for automotive electronic system [34]. BSW is the standardized software layer that provides resources and services to the AUTOSAR software components, as shown in Fig. 9.16.

The BSW is necessary to run the function of the software component but does not perform any function itself. The BSW is placed under the AUTOSAR RTE and cannot interact with the AUTOSAR software component directly. The BSW contains standardized software modules and ECU specific software modules. The standardized software modules include services, communication, operating system and microcontroller abstraction. The ECU specific software modules include the ECU abstraction and complex device driver. The Microcontroller Abstraction Layer (MCAL) is the lowest software layer of the basic software. MCAL makes higher software layers independent of the microcontroller and is implemented as microcontroller-dependent software modules. A higher level software component can access the microcontroller registers through MCAL but not directly. MCAL is

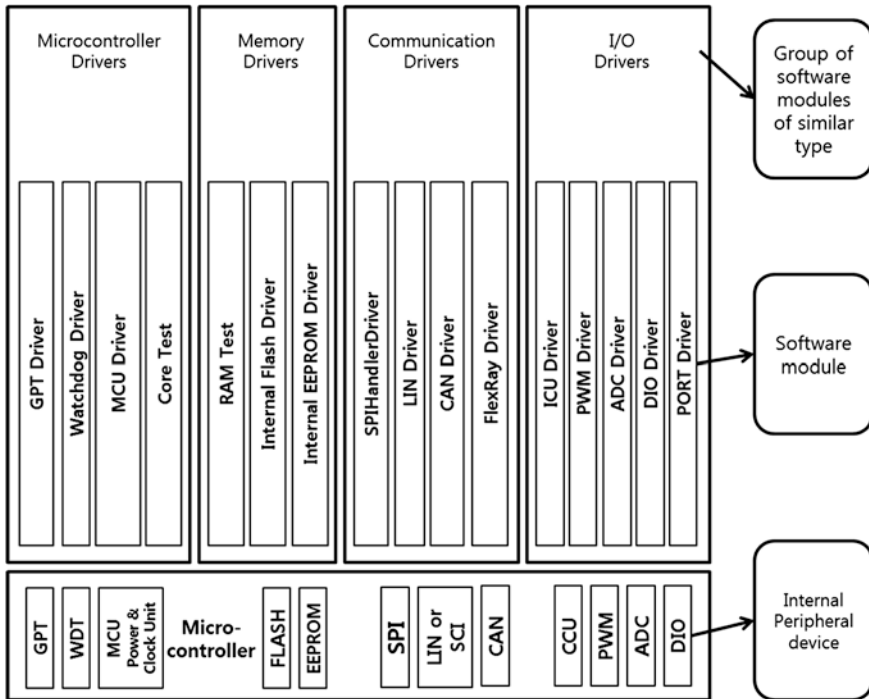
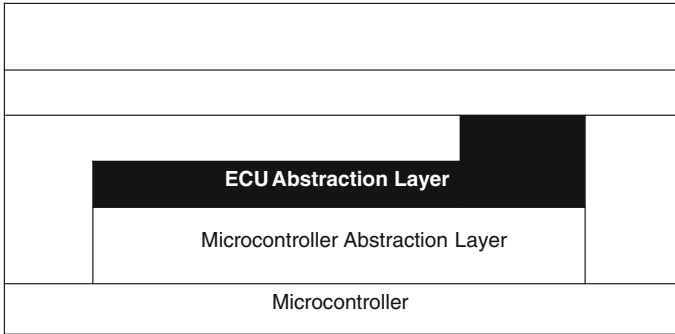


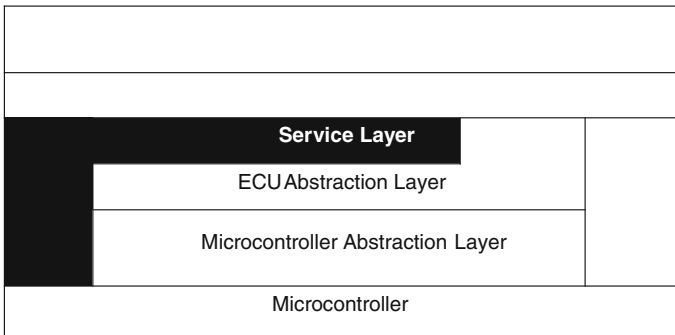
Fig. 9.17 Microcontroller abstraction layer (MCAL)

an ECU specific layer that provides standard interfaces to the higher level software components of BSW. The microcontroller peripherals are managed by MCAL, e.g. Digital I/O, analog/digital converter, and serial peripheral interface. MCAL implements the notification mechanisms to support the distribution of operation and information to different higher level software components. The MCAL modules can be considered microcontroller device drivers and are divided into four groups: microcontroller drivers, memory drivers, communication drivers and I/O drivers, as shown in Fig. 9.17. The communication drivers contain the CAN, LIN and FlexRay driver. The I/O drivers include Pulse Width Modulation (PWM), Analog-Digital converter (ADC) and digital I/O driver.

The ECU abstraction layer is on the MCAL, as shown in Fig. 9.18. The ECU abstraction layer provides a software interface to the electrical values of the specific ECU and external devices on the ECU board. The ECU abstraction layer decouples the higher level software components from the underlying ECU hardware layout and is implemented as microcontroller independent and ECU hardware dependent software modules. In addition, the ECU abstraction layer provides an interface for the peripherals and devices in the ECU regardless of the internal or external parts of the microcontroller and connection to the microcontroller.



**Fig. 9.18** ECU abstraction layer



**Fig. 9.19** System services

On the ECU abstraction layer, the service layer provides additional system services, such as the Nonvolatile Random Access Memory (NVRAM) manager, Diagnostic Event Manager (DEM), flash and memory management, as shown in Fig. 9.19. The service layer is the highest layer of the basic software that can access hardware directly and provide the basic services for application and BSW. In most cases, the service layer is implemented as microcontroller- and ECU-independent software modules.

The AUTOSAR operating system that is compatible with OSEK is a part of the service layer, as shown in Fig. 9.20. The requirements for the AUTOSAR operating system can be specified as AUTOSAR aims at a common architecture for all automotive domains. The following gives an example:

- (1) The OS is configured and scaled statically.
- (2) The OS can predict real-time performance.
- (3) The OS provides a priority-based scheduling.
- (4) The OS provides protective mechanism at runtime.
- (5) The OS can run on low-end controllers without external resources.

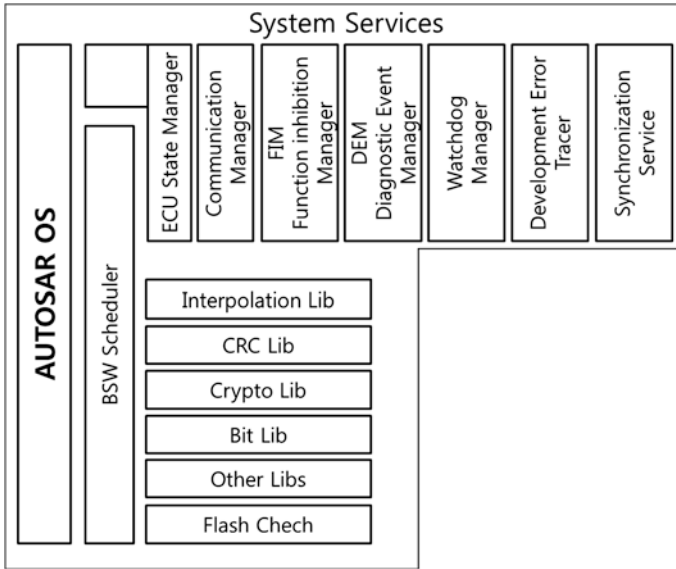


Fig. 9.20 Detailed view of the system services

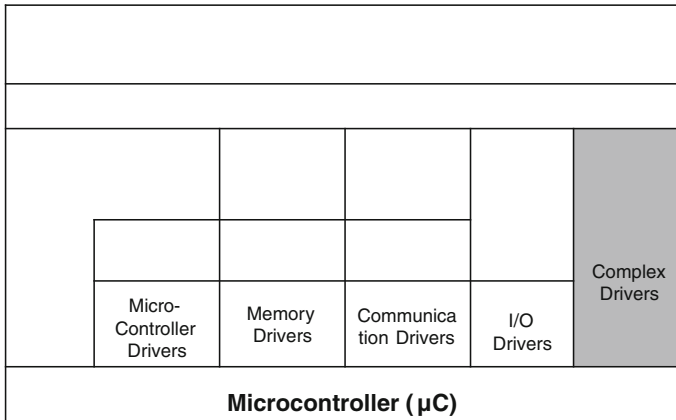


Fig. 9.21 Complex device driver

The AUTOSAR OS has direct access to the hardware resource for some specific information, e.g. the timer in the microcontroller for scheduling. This is the reason for some system services allowing the service layer to access the hardware resource directly. The standard OSEK OS, which is also known as ISO 17356-3, was used for the AUTOSAR OS.

The Complex Device Driver (CCD) can access the hardware resource directly, as shown in Fig. 9.21. The purpose of the complex device drivers is to support and



resource the critical applications and new device drivers that have not been standardized with the standardized interfaces of the AUTOSAR software architecture. In addition, CCD can provide special purpose functionality for high timing constrains and migration. CCD may be implemented as application-, microcontroller- and ECU-dependent software modules.

The BSW and Software Component (SWC) are separated strictly, and the RTE is placed between them. The RTE covers the BSW from the SWC, and allows the SWC to access the BSW services with standardized interfaces, such as microcontroller peripherals and memory services. Another important responsibility of RTE is the provision of communication services between the SWCs, regardless of whether they are within an ECU or not. The RTE can be considered middleware on a specific ECU that is an implementation of the VFB concept.

## **9.3 Demonstration of AUTOSAR ECUs**

### ***9.3.1 Migration to AUTOSAR ECU***

The AUTOSAR concept and benefits have been proved by several OEMs. AUTOSAR will provide simple develop environments to an automotive electronic system. Nevertheless, the initial high costs of model-based development and the uncertainty for the future make it difficult for OEMs to adopt the AUTOSAR system. The modification and optimization of the existing system can be a good solution for OEMs to develop an AUTOSAR conformant system. Migration can provide an opportunity of transition to AUTOSAR system. The following lists the required activities to transit:

- (1) Training for the AUTOSAR development process and tools
- (2) Inspection of the existing system and AUTOSAR system
- (3) Documentation of the AUTOSAR development process
- (4) Conversion of the existing system to the AUTOSAR system
- (5) Conformance between the existing system and AUTOSAR system.

For migration from an existing system to an AUTOSAR system, the existing software architecture is classified into AUTOSAR BSW, RTE and AUTOSAR software components. After classification, the classified software needs to be arranged according to the AUTOSAR layered architecture. Based on the AUTOSAR architecture, the existing platform needs to be mapped to AUTOSAR BSW, such as the OS, communication stack, device drivers. To validate the migration from an existing system to an AUTOSAR system, some comparisons between the existing system and AUTOSAR system are required. The items of comparison are listed as follows:

- (1) Conformance between the existing system and AUTOSAR system
- (2) Performance comparison between the existing system and AUTOSAR system



**Fig. 9.22** Components of the LKAS

- (3) Reusability between the existing system and AUTOSAR system
- (4) Maturity of the development process between the existing system and AUTOSAR system.

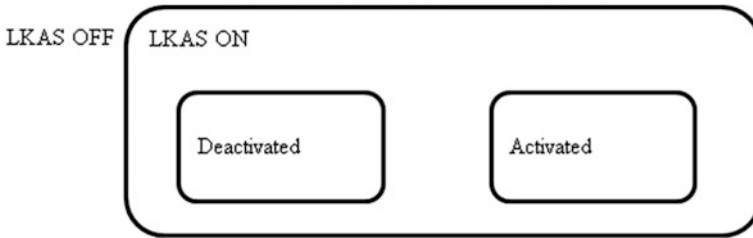
For example, the AUTOSAR migration workflow can be as follows:

- (1) Start the migration with the legacy system
- (2) Develop a simple AUTOSAR SWC from the legacy application and RTE via the AUTOSAR interfaces
- (3) Develop AUTOSAR interfaces of legacy middleware to the RTE
- (4) Develop AUTOSAR BSW and replace the legacy middleware
- (5) Develop more AUTOSAR SWCs on the RTE from legacy application
- (6) Replace the remainder of the legacy application and legacy middleware.

### 9.3.2 LKAS System in AUTOSAR

The main function of the LKAS is to support the driver in keeping the vehicle within the current lane [35]. The LKAS acquires the position of the vehicle within the lane and controls the lateral movement of the vehicle when required. The LKAS consists of sensor/actuator component, processing unit and HMI component. Figure 9.22 presents the components of the LKAS.

For the development of LKAS, the states and transitions of LKAS need to be defined for the first time. As a draft of the international standard of LKAS, Fig. 9.23 shows the basic state of LAKS. The menu of HMI or the switch of LKAS allows the transition between LKAS off and LKAS on. Because the LKAS is a type of assistant system but not an autonomous system, the driver assumes all the responsibility. The important requirements of the LKAS are control of the LKAS any time when the driver wishes to and the indication of the current state of the LKAS. When the LKAS is switched on, it has to assist the lane keeping within 5 s and indicate the current state



**Fig. 9.23** Basic states of LKAS

if the activation condition of LAKS is fulfilled. When the activation condition is not fulfilled, the LKAS has to indicate the deactivation state of the LKAS. When the LKAS is switched off, it should not assist in lane keeping gracefully for safety.

Basically, the components are distributed on the vehicle and connected via the IVN (In-Vehicle Network). The detection sensor, such as vision sensor, processes the objects and lane information. With this information, the LKAS determines the vehicle's position within the lane and controls the EPS module. In addition, HMI, such as a cluster, indicates the states of the LKAS and switches on the LKAS functionality, as shown in Fig. 9.24. Each component of the LKAS can be an AUTOSAR ECU.

For the development of AUOTSAR LKAS, the AUTOSAR software components and interfaces need to be defined. As the structure of the LKAS, the AUTOSAR software components and interfaces can be defined, as shown in Fig. 9.25. For HMI of LKAS, the HMI control and HMI indicator software component are defined.

The interface between the HMI Control and LKAS Control is a simple client/server port because the HMI Control SWC just switches the LKAS on and off. When the HMI Control SWC requests the LKAS on/off service, the LKAS Control SWC performs the LKAS on/off service and returns the result of the requested service. An AUTOSAR port is either a PPort or an RPort. A PPort provides an AUTOSAR interface and an RPort requires an AUTOSAR interface. In the case of the Client/Server Interface, a PPort in the AUTOSAR software component provides an implementation of the services in the Client/Server Interface and an RPort in the AUTOSAR software component can invoke the services in the Client/Server Interface. The CS\_RP\_OnOff port in the HMI Control SWC means that the RPort can invoke the OnOff service in the Client/Server interface. The CS\_PP\_OnOff port in the LKAS Control SWC means that is a PPort, which provides an OnOff service in the Client/Server interface. Depending on the state of LKAS, i.e. either on or off, the LKAS Control SWC request indicates a service to the HMI Interface SWC via the CS\_RP\_indicate port. When the HMI Indicate SWC receives the Indicate service, the HMI Indicate SWC attempts to indicate the state of the LKAS. The HMI Control and HMI Indicate SWC can be mapped to some ECUs, such as cluster ECU and Body Comfort Module (BCM), as shown in Figs. 9.26 and 9.27.

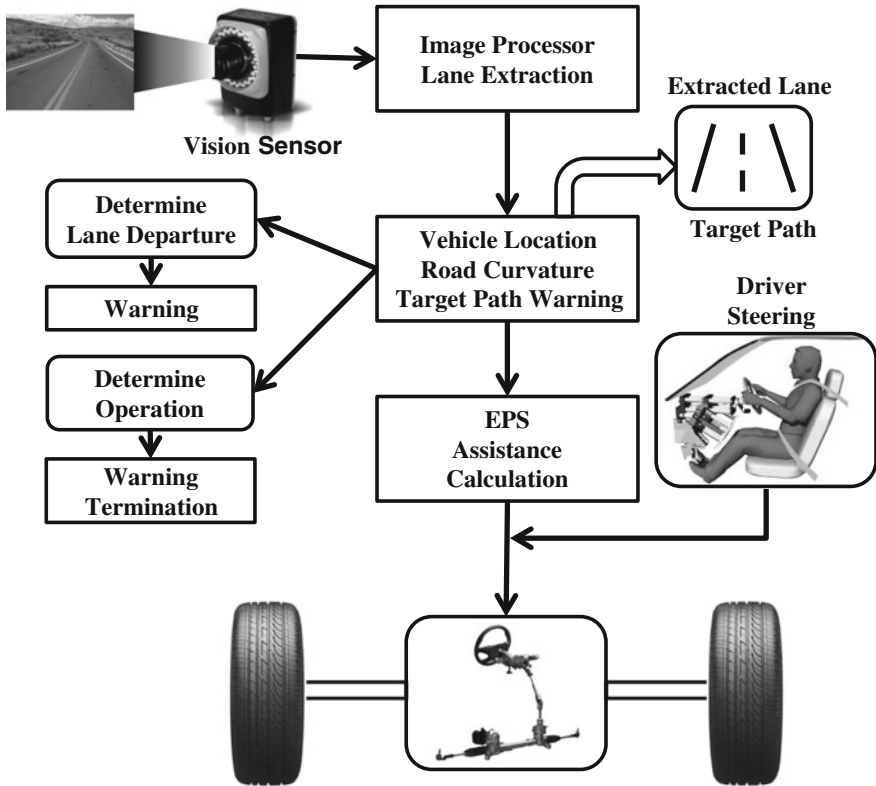


Fig. 9.24 Sub-systems of the LKAS

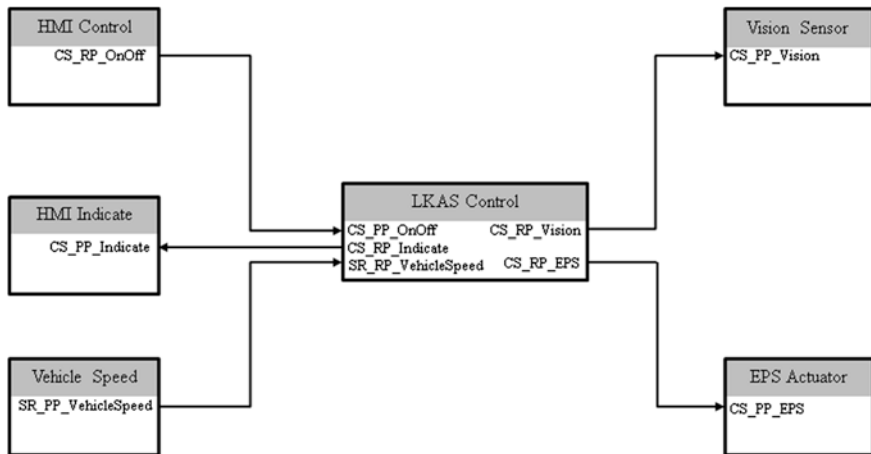


Fig. 9.25 AUTOSAR SWC of LKAS



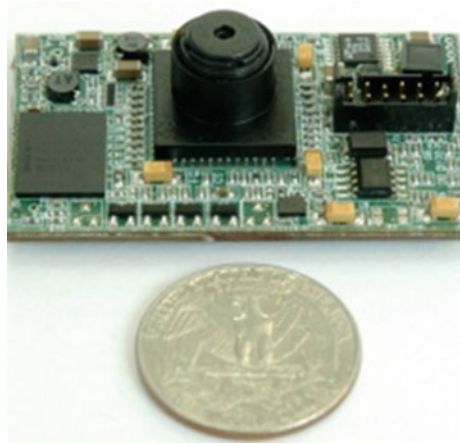
Fig. 9.26 Implementation of the HMI Indicator (VW and Honda)



Fig. 9.27 Implementation of the HMI Controller (VW and Honda)

According to the National Highway Traffic Safety Administration (NHSTA), approximately 70 % of single vehicle highway accidents in United State occur in lane and road departures. To assist lane keeping in the highway, the LKAS will be activated over the reference vehicle speed that fulfills the laws and regulation of each country. To detect the vehicle speed, the Vehicle Speed SWC sends the current vehicle speed data to the LKAS Control SWC periodically. In the case of a Sender/Receiver Interface, a PPort in the AUTOSAR software component generates the data defined in the Sender/Receiver Interface and an RPort in the AUTOSAR software component reads the data in the Sender/Receiver Interface. The SR\_PP\_Vehicle Speed port in the Vehicle Speed SWC means that a PPport generates the vehicle speed data in the Sender/Receiver interface. The SR\_RP\_Vehicle Speed port in the LKAS Control SWC means there is an RPort that reads the Vehicle Speed data in the Sender/Receiver interface. In contrast to the Client/Server interface, the Sender/Receiver Interface does not expect any result and reply of data transmission from the receiver port. When the vehicle speed exceeds the reference speed, the LKAS Control SWC requests vision

**Fig. 9.28** Mobileye mono vision sensor



services to the Vision Sensor SWC. Because of the data complexity and processing time, the entire vision related-services need to be implemented in a Vision Sensor SWC. When the Vision Sensor SWC receives the vision service, it performs vision processing, such as pre-processing, tracker and classification lane detection, and returns the result of the service, such as position of the vehicle within the lane, rate of departure and service errors within few milliseconds. This can be depend on the internal behavior of the SWC and ECU hardware and is not fixed.

In the case of a mono vision sensor, System-on-Chip (SoC), which was designed for low power consumption, as a inexpensive computing platform and to meet automotive qualification requirements, was used for the vision-based solution, such as lane detection, vehicle detection, radar vision fusion, traffic sign detection, camera application, and head lamp control. Figure 9.28 presents the vision sensor of Mobileye [36].

The Mobileye SoC architecture consists of two 32 bit RISC processors, four Vision Computing Engines and several peripherals, as shown in Fig. 9.29. As the migration steps described before, the conformance class of BSW needs to be decided. For easy implementation of the BSW modules on SoC, the ICC1 can be chosen at the first step. Most of the memory services, communication services and I/O will be implemented as the AUTOSAR standard. On the other hand, the scheduler of the OS and communication stack may not be easy to implement. The ECU resource template can be described as the specification of SoC, such as the ECU type and resources. One processor in the SoC is used to execute the software and the other processor is used to control the Vision Computing Engines. The components run on a logic processor can be AUTOSAR software components because the components do not depend on the hardware. Nevertheless, the components that run on the control processor cannot be AUTOSAR software components because these depend on the hardware. The hardware-dependent vision processing components may be implemented as a CDD. To develop the AUTOSAR software components run on a logic processor, communication via

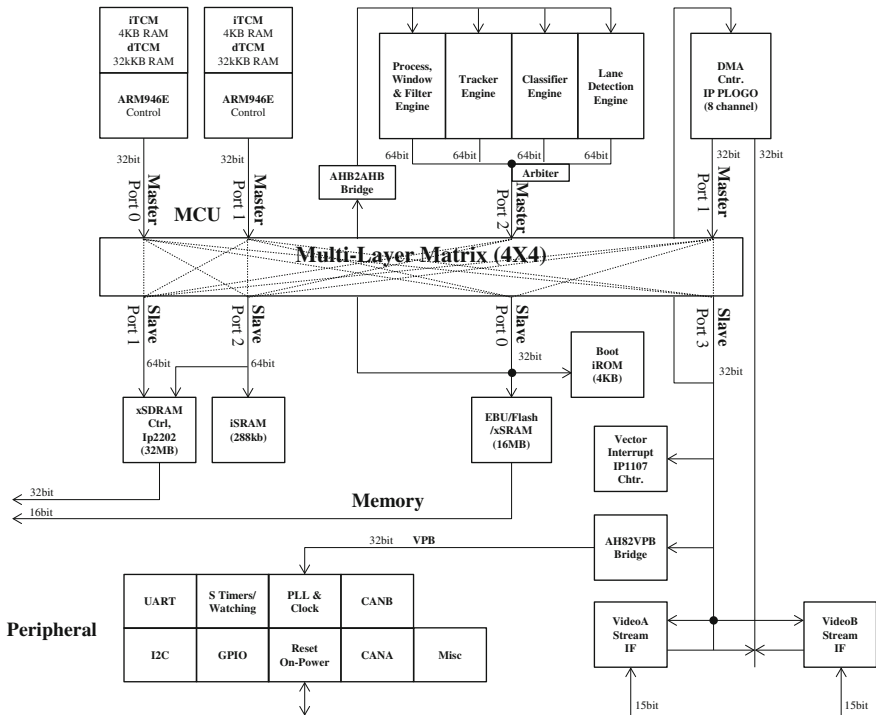


Fig. 9.29 Architecture of a mono vision sensor

peripherals and the interfaces between the RTE and AUTOSAR software component need to be analyzed. AUTOSAR OS-application will be used to support the multi-processor architecture in SoC [37]. Figure 9.30 presents the Mobileye SoC in the AUTOSAR architecture.

When the LKAS Control SWC receives the result of the Vision service, the LKAS Control SWC can determine the steering torque and steering angle to help the vehicle remain within the lane. As the algorithm of the LKAS Control SWC, the vehicle is just placed within the lane or placed in the center of the lane. During activation of the LKAS, the LKAS Control SWC requests Vision service to Vision Sensor SWC periodically.

Development of the internal behavior of the SWCs is required when the design of the AUTOSAR SWCs and interfaces is complete. The internal behavior of SWC can be modeled using a variety of MBD tools via AR-XML, such as a MATLAB/Simulink [38] and dSpace TargetLink [39], as shown in Fig. 9.31.

The development process of the internal behavior of SWC with the MBD tools is as follows:

- (1) Import AUTOSAR SWC from the AUTOSAR tool into MBD tool via AR-XML file.

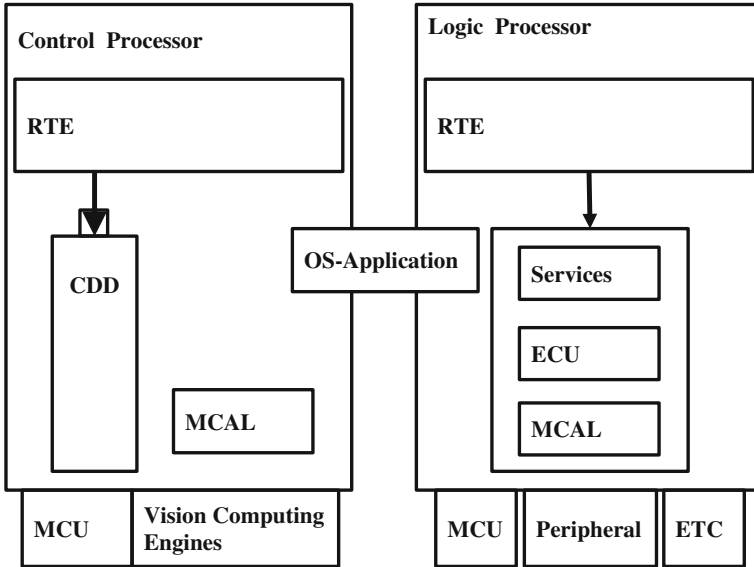


Fig. 9.30 Mobileye SoC in AUTOSAR

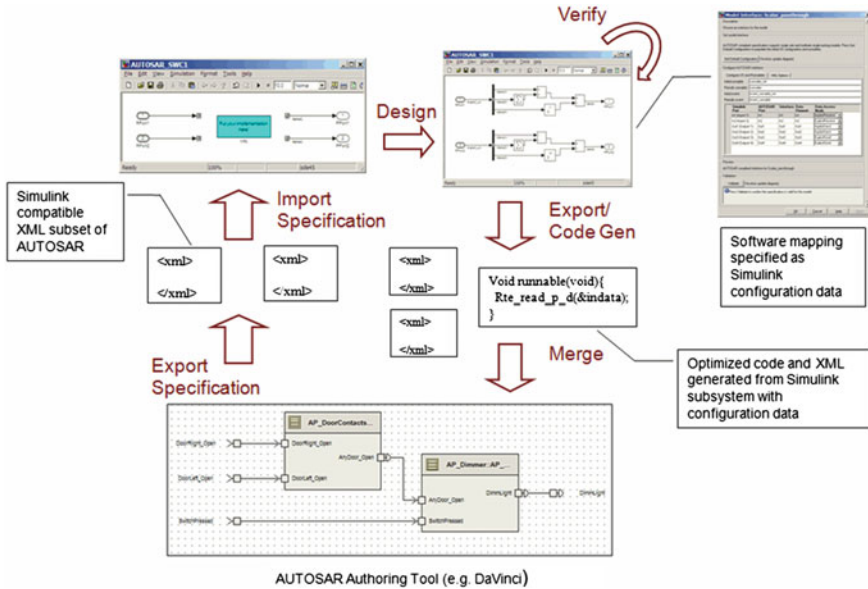
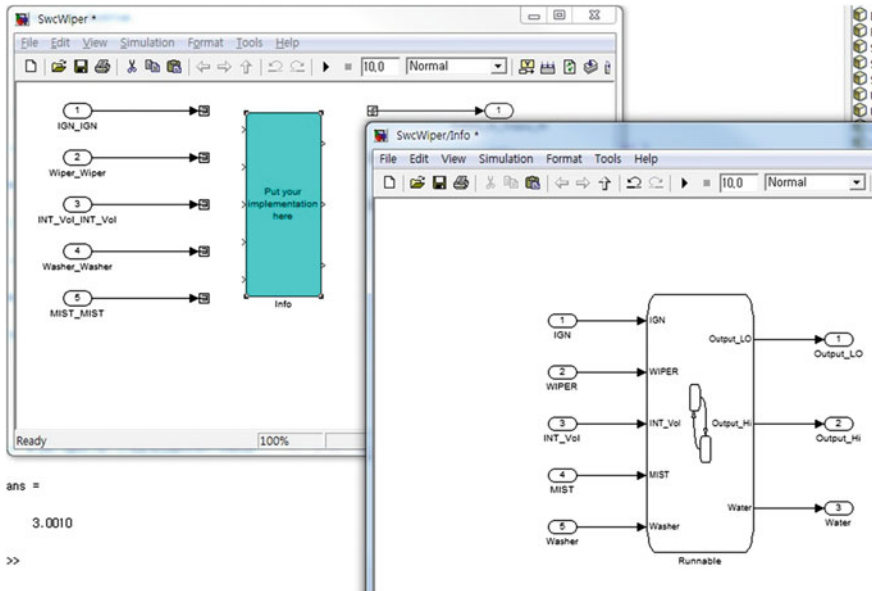


Fig. 9.31 Round-trip workflow with MATLAB





**Fig. 9.32** Import AR-XML and load model

- (2) Generate the model from the imported AUTOSAR SWC.
- (3) Load the behavioral model of the SWC that is already developed with the MBD tool, such as Simulink models and Stateflow models.
- (4) Validate the behavioral model of SWC and the configuration as the AUTOSAR SWC.
- (5) Generate the source code and AR-XML file for the AUTOSAR SWC.
- (6) Import the generated AR-XML file that contains the internal behavior of the SWC from the MBD tool.
- (7) Simulate the AUTOSAR SWC using the AUTOSAR tool.

In the case of MATLAB/Simulink, the `arxml.importer` class is used to parse an AUTOSAR SWC description file that is generated using the AUTOSAR tool. After that, the behavioral model, such as Simulink model and stateflow model, can be used to import AUTOSAR SWC, as shown in Fig. 9.32. When the application of the behavioral model is finished, the AUTOSAR interfaces need to be configured.

Validation of the configuration is needed to generate the AUTOSAR SWC. If no errors occur, the AUTOSAR SWC can be generated using the MDB tool, such as the real-time workshop and embedded coder in MATLAB/Simulink, shown as Fig. 9.33. Currently, the generated AUTOSAR SWC can be merged using the AUTOSAR tool via AR-XML for development and simulation. In the case of the Vector DaVinci Developer [40], the Import XML file in the menu is used to import the generated AR-XML. Simulation of the SWC on the VFB is possible when the

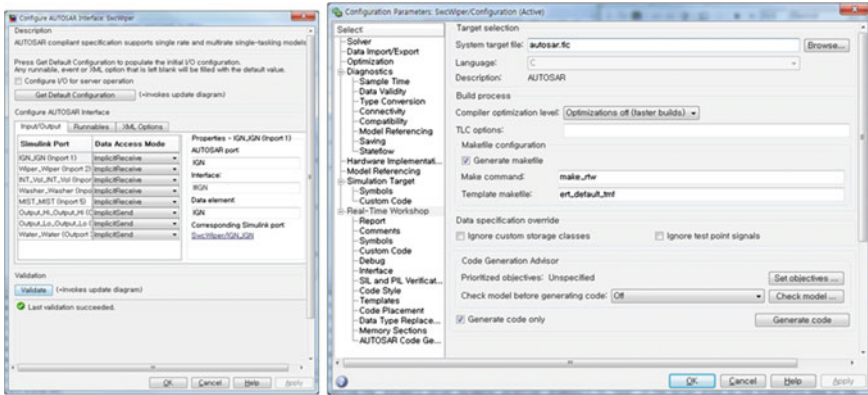


Fig. 9.33 Validation and code generation

internal behavior of the SWC is merged into the SWC and interfaces. Before AUTOSAR system integration and further development, the simulation of the SWC on the VFB is useful for validating the functionality of the AUTOSAR SWCs, which are the application software components of the AUTOSAR system.

### 9.4 Conclusions

Automotive software developers have been faced with many challenges in development methodology according to increasing the automotive functions that have an effect on the complexity of the code as well as the reliability and safety issues. This was caused by standardization of the automotive software development environment, where many leading companies related to the automotive industry proposed the AUTOSAR concept, which provides the consistency and methodology of the basic software and interfaces to support the reusability and scalability for the development phases. The members of the AUTOSAR partnership are in cooperation with important studies that can improve the automotive software maturity and compete with each other to improve the implementation of the AUTOSAR standard.

The complexity of the automotive functions required a different approach compared to traditional microcontroller-based software. Vision processing applied to LDWS and LKAS is one of the features that require high computation power, and it is difficult to satisfy processing power with even the state-of-the-art 32-bit microcontrollers. SoC can be an excellent alternative that is already used widely in IT industries, such as smart phones. In the AUTOSAR environment, the hardware accelerator can be implemented in a complex device driver instead of any layer of the layered architecture because access through a layered architecture can suffer

from performance degradation despite the reusability and scalability from a well-defined architecture, and the computation power can be a more important issue in vision processing than reusability and scalability.

## References

1. FlexRay Consortium, FlexRay, <http://www.flexray.com/>
2. ISO, ISO 11898-1:2003 road vehicles—controller area network (CAN)—Part 1: data link layer and physical signalling, <http://www.iso.org/>
3. MOST Cooperation, MOST, <http://www.mostcooperation.com/>
4. AUTOSAR, “AUTomotive Open System Architecture, <http://autosar.org>
5. LIN Consortium, Local interconnect network, <http://www.lin-subbus.org>
6. F.S.-L.N. Navet, *Automotive Embedded Systems Handbook*, (CRC Press, New York, 2008)
7. OSEK/VDX, OSEK VDX portal, <http://www.osek-idx.org>
8. ISO, “ISO 17356-3:2005 road vehicles—open interface for embedded automotive applications—Part 3: OSEK/VDX operating system (OS), <http://www.iso.org>
9. ATESSST, EAST-ADL, <http://www.atesst.org>
10. AUTOSAR, AUTOSAR\_TechnicalOverview, <http://autosar.org>
11. AUTOSAR, AUTOSAR\_RS\_ProjectObjectives, <http://autosar.org>
12. S.B.H.H.J.B.S.F.K.-P.S.W.G.N.M.T.W.F.W.J.R.L.L.T.S.P.H.R.R.J.L.A. Helmut Fennel, Achievements and exploitation of the AUTOSAR development partnership, in *Convergence Transportation Electronics Association and SAE International*, Detroit (2006)
13. H.S.R.S. Marc Graniou, Advantages and challenges of introducing AUTOSAR for safety-related systems, in *SAE*, Detroit (2009)
14. M.Z. Juergen Cordes, AUTOSAR gets on the road—more and more, in *SAE*, Detroit (2012)
15. D.K.M.K. Dominik Reinhardt, Achieving a scalable E/E-architecture using AUTOSAR and virtualization, in *SAE*, Detroit (2013)
16. ITEA, ITEA2, <http://www.itea2.org>
17. AEE, Architecture Electronique Embarquée, <http://www.cetim.fr/>
18. G.L.E.W. Anila Mjeda, The AUTOSAR standard—the experience of applying simulink according to its requirements, in *SAE*, Detroit (2007)
19. K.K. Kenji Nishikawa, TOYOTA electronic architecture and AUTOSAR pilot, in *SAE*, Detroit (2007)
20. J.S. Oliver Niggemann, Models for model’s sake: why explicit system models are also an end to themselves. in *International Conference on Software Engineering*, Leipzig, (2008)
21. T.K.J.L.V. Devendra Rai, Model-based development of AUTOSAR—compliant applications: exterior lights module case study, in *SAE*, Detroit (2008)
22. R.T. Guido Sandmann, Development of AUTOSAR software components within model-based design, in *SAE*, Detroit (2008)
23. M.S. Guido Sandmann, AUTOSAR—compliant development workflows: from architecture to implementation—tool interoperability for round-trip engineering and verification and validation, in *SAE*, Detroit (2012)
24. M.S. Richard, E. Lotoczky, New methods of debugging and testing improve the software quality of AUTOSAR ECUs, in *SAE*, Detroit (2013)
25. B. Chown, Test generation technology for an AUTOSAR simulation platform, in *SAE*, Detroit (2013)
26. V.J.J.L. Ulrich Freund, Multi-level system integration of automotive ECUs based on AUTOSAR, in *International Conference on Software Engineering*, Leipzig (2008)
27. G. Wang, Integration of hardware specific features of microcontroller into the AUTOSAR standard, in *SAE*, Detroit (2009)

28. OMG, OMG meta object facility, <http://www.omg.org/mof/>
29. H.L.M.Y.Z.S. Wei Peng, Deployment optimization for AUTOSAR system configuration, in *International Conference on Computer Engineering and Technology*, Chengdu (2010)
30. K.C.S.S.M.M.S. Ahmed Daghsen, Software function allocation and configuration of an AUTOSAR—compliant system, in *SAE*, Detroit (2012)
31. AUTOSAR, AUTOSAR\_RS\_SoftwareComponentTemplate, <http://autosar.org>
32. AUTOSAR, AUTOSAR\_BasicSoftwareModules, <http://autosar.org>
33. AUTOSAR, AUTOSAR\_SWS\_RTE, <http://autosar.org>
34. AUTOSAR, AUTOSAR\_EXP\_LayeredSoftwareArchitecture, <http://autosar.org>
35. ISO, ISO 11270 Intelligent transport systems—lane keeping assistance systems (LKAS), <http://www.iso.org/>
36. Mobileye, Mobileye processing platform, <http://www.mobileye.com>
37. AUTOSAR, AUTOSAR\_SWS\_OS, <http://autosar.org>
38. MathWorks, Simulation and model-based design, [www.mathworks.com](http://www.mathworks.com)
39. dSPACE, Automatic production code generator, [www.dspace.com](http://www.dspace.com)
40. Vector, DaVinci Developer 3.3, [www.vector.com](http://www.vector.com)

# Chapter 10

## Reliability Issues for Automobile SoCs

Sungju Park

**Abstract** Current vehicles are built with complex electronic systems embedded with more than a hundred microprocessors through complicated automotive networks. In the de facto ISO 26262 standard in the automotive industry, Automotive Safety Integrity Level (ASIL) is classified into four different levels. In this chapter, the ISO 26262 hardware ASIL is described in detail. Then, we introduce fault diagnosis architectures that use various design for testability techniques such as scan design, built-in self-test, IEEE boundary scan design, and error correcting codes for increasing hardware reliability.

### 10.1 Introduction

Auto manufacturers admit that only two mechanisms can cause a vehicle to accelerate; the driver's foot on the accelerator and the cruise control. Prior to the advent of the electronic fuel injected vehicle with cruise control, the only reported events, similar to Sudden Acceleration, were due to throttle valves, which were stuck open due to a mechanical bind while moving. These were actually failures to decelerate caused by mechanical binding of the throttle valve or its linkage. These accidents were rare and received little or no attention. Drive-by-wire mainly relies on complex electronic systems and with the lack of adequate EMI protection, in addition to sudden accelerations numerous other electronic anomalies will occur. This has already manifested the air bag, ATC (automatic traction control) and ABS (anti-lock braking system) malfunctions, and roll over prevention control malfunction. It is admitted by the auto industry that these EMI fault aspects will leave no trace of their occurrence [1]. This is the same finding that was made by the

---

S. Park (✉)  
Hanyang University ERICA Campus, Ansan, South Korea  
e-mail: paksj@hanyang.ac.kr

aircraft industry, medical electronics industry, radio operated devices industry such as cranes, and even the wheelchair industry. Nowadays, very complex hardware components are largely used in safety-critical domains such as automotive, aerospace, medical, industrial, and railway. Besides systematic failures and misuse, the root causes of HW functional failures are defects, flaws (defects whose presence does not interfere with normal operation during manufacturing test, but which cause catastrophic failure) and random faults (permanent, intermittent and transient faults caused by random events during operation) [1–3].

To handle this complexity, functional safety standards like IEC 61508 [4] and ISO 26262 [5] are giving requirements, evaluation metrics and methodologies to define “how much safe” or “how much available” shall be a given component or system. Consequently, the design and test of reliable and available integrated circuits require a specific approach affecting both digital and analogue HW; moreover, safety oriented methodologies shall be available to drive the specifications and verify the implementation. This chapter introduces, stress test driven qualification requirements of AEC-Q100 (automotive electronics councils), H/W fault models of ISO 26262 (the international norm ruling functional safety for automotive) and various design techniques to improve reliability of the electronic components embedded in automobiles.

### ***10.1.1 AEC-Q100 Electrical Component Qualification Requirement***

The AEC (Automotive Electronics Councils) Component Technical Committee defines common electrical component qualification requirements containing detailed qualification and requalification requirements and including unique test methods and guidelines for the use of generic data. Components meeting these specifications are suitable for use in the harsh automotive environment without additional component level qualification testing. Suppliers may advertise components meeting these specifications without restrictions, but the specifications cannot be changed without the approval of the Sustaining Members (currently Autoliv, Continental, Delphi, Gentex, Harman, Johnson Controls, TRW Automotive, and Visteon).

AEC-Q100 contains a set of failure mechanism based stress tests and defines the minimum stress test driven qualification requirements, references, and test conditions for qualification of integrated circuits (ICs). These tests are capable of stimulating and precipitating semiconductor device and package failures. The objective is to precipitate failures in an accelerated manner compared to use conditions. This set of tests can be classified as the following sub items.

AEC-Q100-001: WIRE BOND SHEAR TEST  
 AEC-Q100-002: HUMAN BODY MODEL (HBM) ELECTROSTATIC DISCHARGE (ESD) TEST  
 AEC-Q100-003: MACHINE MODEL (MM) ELECTROSTATIC DISCHARGE (ESD) TEST  
 AEC-Q100-004: IC LATCH-UP TEST  
 AEC-Q100-005: NONVOLATILE MEMORY WRITE/ERASE ENDURANCE, DATA RETENTION, AND OPERATIONAL LIFE TEST  
 AEC-Q100-006: ELECTRO-THERMALLY INDUCED PARASITIC GATE LEAKAGE (GL) TEST  
 AEC-Q100-007: FAULT SIMULATION AND TEST GRADING  
 AEC-Q100-008: EARLY LIFE FAILURE RATE (ELFR)  
 AEC-Q100-009: ELECTRICAL DISTRIBUTION ASSESSMENT  
 AEC-Q100-010: SOLDER BALL SHEAR TEST  
 AEC-Q100-011: CHARGED DEVICE MODEL (CDM) ELECTROSTATIC DISCHARGE (ESD) TEST  
 AEC-Q100-012: SHORT CIRCUIT RELIABILITY CHARACTERIZATION OF SMART POWER DEVICES FOR 12 V SYSTEMS

Under the test conditions specified by AEC-Q100 standards, the semiconductor companies, manufacturing vehicular chips, are supposed to notify the qualification test results for each test category. A summary of the test result is as follows:

Endurance Cycling with HTOL: 0/231 pcs  
 Endurance Cycling with Data Retention: 0/231 pcs  
 Dynamic Early Fail Study: 0/2400 pcs  
 Pre-Condition Test: 0/924 pcs  
 High Temp. Storage Life Test: 0/231 pcs  
 Pressure Cooker Test: 0/231 pcs  
 Temperature Cycle Test: 0/231 pcs  
 Highly Accelerated Stress Test: 0/231 pcs  
 ESD-HBM: 0/36 pcs  
 ESD-MM: 0/36 pcs  
 ESD-CDM: 0/9 pcs  
 Latch-Up Test: 0/18 pc

### ***10.1.2 ISO 26262 Road Vehicle Functional Safety Requirements***

ISO 26262 is a functional safety standard for automotive industry. For a competitive product, the automotive manufacturers need to conduct the standardized development process and they require their suppliers to comply those standards

too. The scope of ISO 26262 includes all safety related electrical/electronic (E/E) systems for automotive application [6–8].

Like its parent standard IEC 61508, ISO 26262 is a risk based safety standard. It assesses the risk of hazardous operational situations qualitatively; moreover, it defines the safety measures to avoid or control the systematic failures and detects or controls the random hardware failures or it mitigates their effects.

- Provides an automotive safety lifecycle (management, development, production, operation, service, decommissioning) and supports tailoring the necessary activities during these lifecycle phases.
- Covers functional safety aspects of the entire development process (including such activities as requirements specification, design, implementation, integration, verification, validation, and configuration).
- Provides an automotive-specific risk-based approach for determining risk classes ASILs (Automotive Safety Integrity Levels).
- Uses ASILs for specifying the item's necessary safety requirements for achieving an acceptable residual risk.
- Provides requirements for validation and confirmation measures to ensure a sufficient and acceptable level of safety is being achieved.

ISO 26262 is an adaptation of functional safety standard IEC 61508 for Automotive Electric/Electronic Systems. A significant difference between these two standards is that ISO 26262 considers controllability while IEC 61508 does not. Controllability is the ability to avoid the hazardous events on the action taken by a driver. Considering this, ISO 26262 classifies ASIL into four different levels, depending on its severity, probability of exposure and the controllability. After determining ASIL, the product is developed in the process according to the method and measure of the corresponding ASIL. As a result, the main purpose of implementing this standard keeps all the records of safety-related activities from the development process to ensure the functional safety.

ISO 26262 is composed of 10 parts: 1. Vocabulary, 2. Management of functional safety, 3. Concept phase, 4. Product development: system level, 5. Product development: hardware level, 6. Product development: software level, 7. Production and operation, 8. Supporting processes, 9. ASIL oriented and safety-oriented analyses, and 10. Guideline. The development process of item level starts from part 3 to 7, and all the other parts are supporting process. The development process generally follows V-style process model and can be depicted as Fig. 10.1.

As in Fig. 10.1, functional safety concept of the item is derived in the concept phase, and product development at system level will be started. In the product development phase, the system design is specified and each development processes at hardware and software levels are initiated individually. Although the ISO 26262 standard lists all the requirements to be complied, it is difficult to catch the steps of each activities what to do and where to start. This section explains how to develop a product to achieve the functional safety according to the standard, especially for the product development at hardware level.



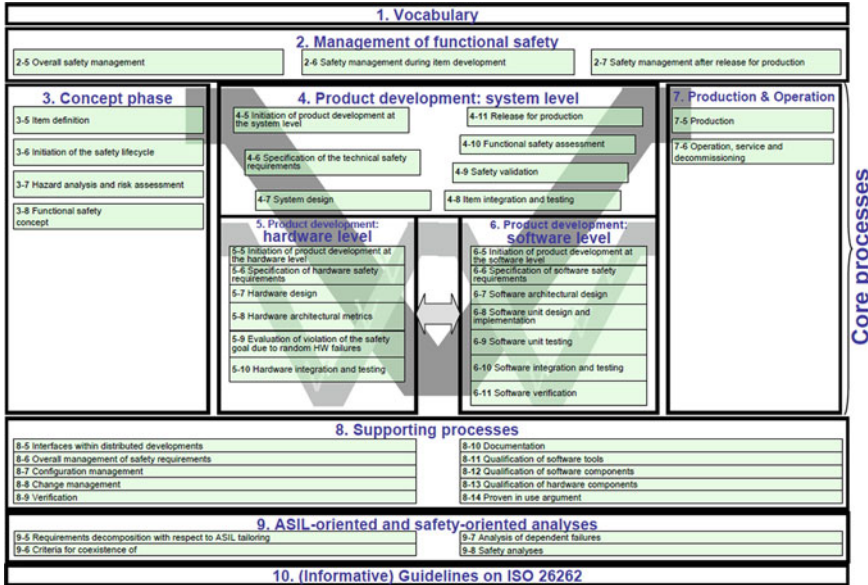
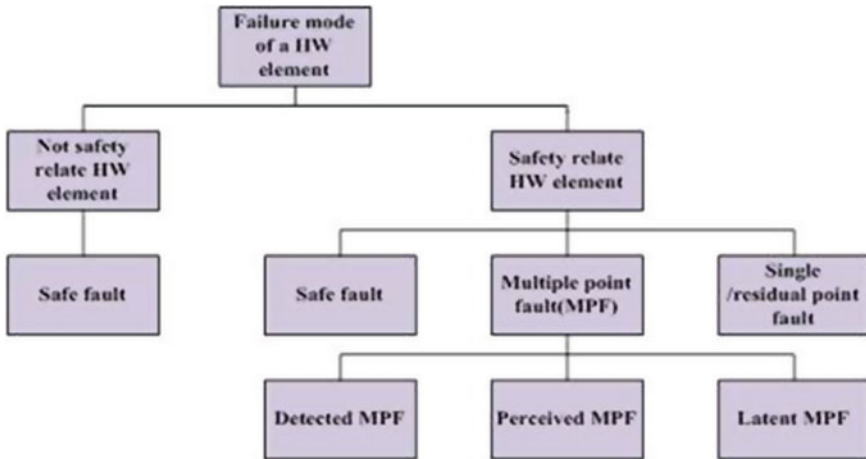


Fig. 10.1 Overview of ISO 26262

The hardware development process, according to ISO 26262, starts from planning the hardware development. The plan includes specifying the methods and measures to be used during hardware design. Next, the hardware safety requirement is specified. This will be derived from several sources such as technical safety concept, software safety requirements and system design specification. Hardware safety requirement specification should be consistent to these documents and the hardware is designed according to the hardware safety requirements specification. In the hardware design process, the hardware architectural metric with respect to the random hardware failures needs to be evaluated and verified for ASIL C and D.

Random hardware failure normally occurs arbitrarily in a hardware element; however, its failure rate can be predicted with reasonable accuracy. The random hardware failure data can be obtained from several sources. To calculate the hardware architectural metrics, the failure mode needs to be defined. Each faults occurring in a safety-related hardware element can be classified as shown in Fig. 10.2 and are defined in the standard as follows:

- Safe fault: The fault whose occurrence will not significantly increase the probability of violation of a safety goal
- Multiple point fault: One fault of several independent faults that in combination, leads to a multiple point failure (either perceived, detected or latent).
- Perceived: This fault is deduced by the driver, without detection by a safety mechanism within a prescribed time.



**Fig. 10.2** Failure modes classification of a hardware element

- **Detected:** This fault is detected by a safety mechanism to prevent the fault from being latent within a prescribed time.
- **Latent:** This fault is neither detected by a safety mechanism nor perceived by the driver.
- **Single point fault:** The fault in an element which is not covered by a safety mechanism and where the fault leads directly to the violation of a safety goal.
- **Residual fault:** Portion of a fault which by itself leads to the violation of a safety goal, occurring in a hardware element, where that portion of the fault is not covered by existing safety mechanisms.

### 10.1.2.1 Hardware Architectural Metric Calculation

To obtain objective evidence for the hardware design, the hardware architectural metric needs to be calculated for ASIL C and D according to the following procedure:

- Estimate failure rate of single point fault and latent multiple fault.
- Estimate diagnostic coverage of safety mechanism.
- Calculate “single point faults metric” and “latent faults metrics.”
- Compare the metrics with target values.

The detailed explanation for the procedure is given as follows:

- Estimate failure rate of single point fault and latent multiple fault.
- The failure rate of a hardware part can be estimated either of the following sources.

**Table 10.1** Example of required faults or failures to derive diagnostic coverage

Components	Recommendations for diagnostic coverage		
	Low (60 %)	Medium (90 %)	High (99 %)
Relay	Does not energize or de-energize Welded contacts	Does not energize or de-energize Individual contacts welded	Does not energize or de-energize Individual contacts welded
Invariable memory range	Struck-at for data and addresses	d.c. fault model for data and addresses	All faults which affect data in the memory
Variable memory range	Struck-at for data and addresses	d.c. fault model for data and addresses	d.c. fault model for data and addresses dynamic cross-over for memory cells no, wrong or multiple addressing

- Using a recognized industry sources (i.e., IEC 62380, IEC 61709, MIL HDBK 217 F notice 2, RAC HDBK 217 Plus, NPRD95, EN50129 Annex C, EN 62061 Annex D, RAC FMD97, MIL HDBK 338, etc.)
- Using statistics based on field returns or tests.
- Using expert judgment based on engineering approach.

(c) Estimate diagnostic coverage of safety mechanism.

Diagnostic coverage can be calculated using every element or part used in the hardware architecture given with their achievable diagnostic coverage.

Table 10.1 shows the example components, consisting hardware architecture and their faults or failures that need to be analyzed to derive diagnostic coverage. For relays, to achieve 99 % of diagnostic coverage, two factors need to be detected: (i) does not energize or de-energize, (ii) individual contacts welded. This table is only a part of the quotation of the standards.

Techniques to derive the diagnostic coverage and the maximum achievable diagnostic coverage for the selected component are given in Table 10.2. This is an example of a variable memory range from Table 10.1. Block replication in Table 10.2, for instance, aims to detect all bit failures of the memory by duplicating the address space in two memories. The mechanism is as follows: the first memory is operated in the normal manner, the second memory stores the same information inversely and it is inverted again when accessed in parallel to the first, and the outputs difference leads to a failure message. This technique can achieve high diagnostic coverage.

(c) Calculate “single point faults metric” and “latent faults metrics.”

When all the failure rates are estimated, a single point faults metric and latent faults metrics are calculated through the equations [8].

(d) Compare the metrics with target values.

Numerical target values for “single point faults metric” and “latent faults metrics” are given in Table 10.3. Appropriate target values w.r.t. their ASILs are

**Table 10.2** Diagnostic technique/measure and their coverage for variable memory

Diagnostic technique/measure	Maximum diagnostic coverage considered achievable
RAM test “checkerboard” or “march”	Low
One bit redundancy	Low
Detection of RAM data failures with error-detection-correction codes (EDC)	High
Block replication	High

**Table 10.3** Target values for single point faults metric and latent faults metric

	ASIL B (%)	ASIL C (%)	ASIL D (%)
Single point faults metric	>90	>97	>99
Latent faults metric	>60	>80	>90

chosen for the hardware architectural metric of the product. Finally, by comparing the result of calculated metrics with the target values, the product can be claimed the standard compliant.

According to ISO 26262, the hardware development process consists of planning, hardware-safety requirement specification, hardware design and integration and testing. The calculation of hardware architectural metrics, during hardware design process for ASIL C and D, is explained in detail by four steps. Evaluation of hardware architectural metrics with respect to random hardware failures and final claiming evidence are included in the steps. By obtaining hardware architectural metrics, the architectural detailed design can have ASIL dependent pass/fail criteria and can be objectively assessable.

However, it is very hard to estimate the failure rates and diagnostic coverage for vehicular SoCs (system-on-chips) embedding digital/analog circuits with various memory components. SoC defects are modeled as stuck-at, bridge, delay, cross-talk, retention, etc. and the fault coverage for each fault, thus failure rate cannot be simply estimated. Diagnosis at an SoC becomes even harder problem than the estimation of the failure rate. Ad hoc and structured design for testability techniques are adopted in designing SoCs to alleviate the difficulties in failure analysis and improve the reliability. State of art design for reliability techniques are required by vehicular SoCs to achieve ASIL D level of ISO 26262 hardware functional safety.

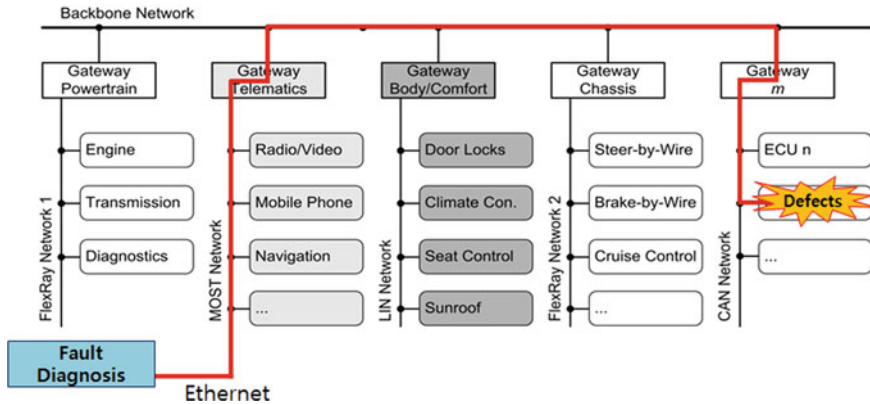
Although, developing an ISO 26262 compliant product is not a simple task, automotive companies should implement the standard in advance to their development process; which can be evident of the organizational capability of performing functional safety in near future.

### 10.1.3 Vehicular Networks, CAN/LIN, FlexRay, and MOST

Controller area network (CAN) is a new type of serial bus, which is developed by Bosch Corporation to solve the data exchange problem among numerous electronic devices of future automobiles, in 1980s. CAN bus has the merits of high intelligence, fault-tolerant and reliability, which can support distributing real-time control, therefore, it was popular right away, especially in the automobile industry. As specified by the CAN standard, an off-the-shelf controller includes an error detection mechanism based on a cyclic redundancy check and automatic retransmission of messages, thus due to its error robustness, CAN is frequently used in safety critical application such as active-steering or for controlling heavy industrial machinery [9].

FlexRay is a new network communication standard that provides a high-speed serial communication, time triggered bus and fault tolerant communication between electronic devices for future automotive applications. FlexRay supports a time-triggered scheme and an optional event triggered scheme. The upper bound of the data rate is 10 Mbps and it provides two channels for redundancy. FlexRay was developed for next generation automobiles by a consortium founded by BMW, Bosch, Daimler Chrysler and Philips in 2000, which is extended to several automotive and semiconductor industry members including Freescale Semiconductors, Bosch, General Motors, and Hyundai-Kia Motors. In 2006, FlexRay protocol was first applied to the electronically controlled dampers of BMW X5 series.

MOST—Media Oriented Systems Transport—is the de-facto standard for multimedia and infotainment networking in the automotive industry. The technology was designed from the ground up to provide an efficient and cost-effective fabric to transmit audio, video, data and control information between any devices attached even to the harsh environment of an automobile. Its synchronous nature allows simple devices to be able to provide content and others to render that content with the minimum hardware. At the same time, it provides unique quality of service for transmission of audio and video services. Although its roots are in the automotive industry, MOST can be used for applications in other areas such as other transportation applications, A/V networking, security and industrial applications. MOST150 provides a physical layer to implement Ethernet in automobiles with 150 Mbps bandwidth. It also integrates an Ethernet channel with adjustable bandwidth in addition to the three established channels (control message channel, streaming data channel and packet data channel) of the other grades of MOST. MOST150 also permits isochronous transfer on the synchronous channel. Although the transfer of synchronous data requires a frequency other than the one specified by the MOST frame rate, it is also possible with MOST150. MOST150's advanced functions and enhanced bandwidth will enable a multiplex network infrastructure capable of transmitting all forms of infotainment data, including video, throughout an automobile. Figure 10.3 shows the hierarchical automobile network connected through MOST, FlexRay, CAN, and LIN. The red line



**Fig. 10.3** Diagnosis of failures through hierarchical automobile networks

indicates that it is imperative to diagnose any failure in ECUs and network components at real time.

Semiconductor chips for automobile application adapt many reliability features complying with ISO26262. For example, as shown in the Fig. 10.4, Renesas supports functional safety activities compliant with ISO26262 by providing ECU with lock-step, memory components with Error Correcting Codes, CRC for data transmission, watch dog timer to monitor abnormal status, self test for logic/memory/analog components.

### 10.1.4 Fault Models, Test Pattern Generation, and Fault Simulation

#### 10.1.4.1 Fault Models in Digital Logic Circuits

- Stuck-at fault: A signal or the gate output is stuck at a 0 or 1 value, independent of the inputs to the circuit.
- Bridging fault: Two signals are connected together when they should not be. Depending on the logic circuitry employed, this may result in a wired-OR or wired-AND logic function. Since there are  $O(n^2)$  potential bridging faults, they are normally restricted to signals that are physically adjacent in the design.
- Open fault: Here a wire is assumed broken, and one or more inputs are disconnected from the output that should drive them. As with bridging faults, the resulting behavior depends on the circuit implementation.
- Delay fault model: The signal eventually assumes the correct value, but more slowly (or rarely, more quickly) than normal.

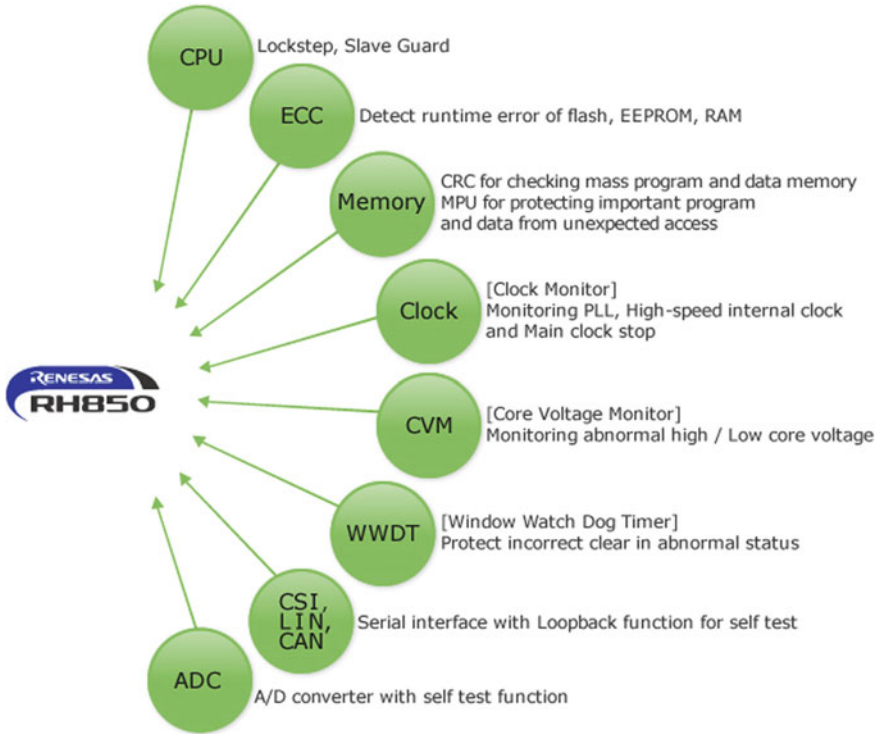


Fig. 10.4 Reliability features of an automobile ECU

- Intermittent fault: It is a malfunction of a device or a system that occurs at intervals, usually irregular, in a device or system that functions normally at other times.
- Soft error: These errors occur when the radioactive atoms in the chip’s material decay and release alpha particles into the chip. Because an alpha particle contains a positive charge and kinetic energy, the particle can hit a memory cell and cause the cell to change state to a different value. The atomic reaction is so tiny that it does not damage the actual structure of the chip.
- Aging fault: Circuit aging effects caused by, for example, negative bias temperature instability (NBTI) further affect the timing of the circuit during its lifetime. As a consequence, some paths, which were not expected to fail based on pre-silicon estimates, may fail after fabrication. The initial impact of NBTI is on the temporal increase of the threshold voltage of PMOS transistors in a design, which will reduce the drive current and slow down the switching of the circuit, and the operational frequency of the circuit will reduce over time.

### 10.1.4.2 Fault Coverage

Fault coverage refers to the percentage of some type of fault that can be detected during the test of any engineered system. High fault coverage is particularly valuable during manufacturing test, and techniques such as design for test (DFT) and automatic test pattern generation are used to increase it. In electronics for example, stuck-at fault coverage is measured by sticking each pin of the hardware model at logic '0' and logic '1', respectively, and running the test vectors. If at least one of the outputs differs from what is to be expected, the fault is said to be detected. Conceptually, the total number of simulation runs is twice the number of pins (since each pin is stuck in one of two ways, and both faults should be detected). However, there are many optimizations, which can reduce the needed computation. In particular, often many non-interacting faults can be simulated in one run, and each simulation can be terminated as soon as a fault is detected [10].

### 10.1.4.3 Basics of Automatic Test Pattern Generation

A defect is an error introduced into a device during the manufacturing process. A fault model is a mathematical description of how a defect alters design behavior. The logic values observed at the device's primary outputs, while applying a test pattern to some device under test (DUT), are called the output of that test pattern. The output of a test pattern, when testing a fault-free device that works exactly as designed, is called the expected output of that test pattern. A fault is said to be detected by a test pattern if the output of that test pattern, when testing a device that has only that one fault, is different than the expected output. The ATPG process for a targeted fault consists of two phases: fault activation and fault propagation. Fault activation establishes a signal value at the fault model site that is opposite of the value produced by the fault model. Fault propagation moves the resulting signal value, or fault effect, forward by sensitizing a path from the fault site to a primary output.

ATPG can fail to find a test for a particular fault in at least two cases. First, the fault may be intrinsically undetectable, such that no patterns exist that can detect that particular fault. The classic example of this is a redundant circuit, designed so that no single fault causes the output to change. In such a circuit, any single fault will be inherently undetectable.

Second, it is possible that a pattern(s) exist, but the algorithm cannot find it. Since the ATPG problem is NP-complete there will be cases where patterns exist, but ATPG gives up since it will take an incredibly long time to find them.

### 10.1.4.4 Fault Simulation

A fault simulator emulates the target faults in a circuit in order to determine which faults are detected by a given set of test vectors. Because there are many faults to emulate for fault detection analysis, fault simulation time is much greater than that

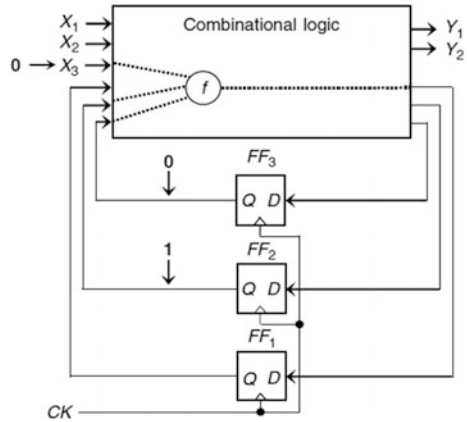


required for design verification. To accelerate the fault simulation process, improved approaches have been developed in the following order. Parallel fault simulation uses bit-parallelism of logical operations in a digital computer. Thus, for a 32-bit machine, 31 faults are simulated simultaneously. Deductive fault simulation deduces all signal values in each faulty circuit from the fault-free circuit values and the circuit structure in a single pass of true-value simulation augmented with the deductive procedure. Concurrent fault simulation is essentially an event driven simulation to emulate faults in a circuit in the most efficient way. Hardware fault simulation accelerators based on parallel processing are also available to provide a substantial speed-up over purely software-based fault simulators.

### ***10.1.5 Design for Testability Techniques***

The testability of combinational logic decreases as the level of the combinational logic increases. A more serious issue is that good testability for sequential circuits is difficult to achieve. Because many internal states exist, setting a sequential circuit to a required internal state can require a very large number of input events. Furthermore, identifying the exact internal state of a sequential circuit from the primary outputs might require a very long checking experiment. Hence, a more structured approach for testing designs that contain a large amount of sequential logic is required as part of a methodical design for testability (DFT) approach. Initially, many ad hoc techniques were proposed for improving testability. These techniques relied on making local modifications to a circuit in a manner that was considered to result in testability improvement. While ad hoc DFT techniques do result in some tangible testability improvement, their effects are local and not systematic. Furthermore, these techniques are not methodical, in the sense that they have to be repeated differently on new designs, often with unpredictable results. Due to the ad hoc nature, it is also difficult to predict how long it would take to implement the required DFT features. The structured approach for testability improvement was introduced to allow DFT engineers to follow a methodical process for improving the testability of a design. A structured DFT technique can be easily incorporated and budgeted for as part of the design flow and can yield the desired results. Furthermore, structured DFT techniques are much easier to automate. To date, electronic design automation (EDA) vendors have been able to provide sophisticated DFT tools to simplify and speed up DFT tasks. Scan design has been found to be one of the most effective structured DFT methodologies for testability improvement. Not only can scan design achieve the targeted fault coverage goal, but it also makes DFT implementation in scan design manageable.

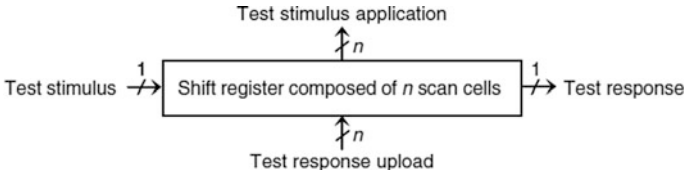
**Fig. 10.5** Difficulty in testing a sequential circuit



### 10.1.5.1 Scan Design

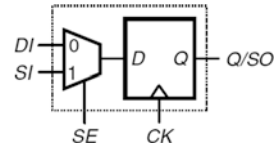
Scan design, the most widely used structured DFT methodology, attempts to improve testability of a circuit by improving the controllability and observability of storage elements in a sequential design. Typically, this is accomplished by converting the sequential design into a scan design with three modes of operation: normal mode, shift mode, and capture mode. Circuit operations with associated clock cycles conducted in these three modes are referred to as normal operation, shift operation, and capture operation, respectively. In normal mode, all test signals are turned off, and the scan design operates in the functional configuration. In both shift and capture modes, a test mode signal TM is often used to turn on all test-related fixes that are necessary to simplify the test, debug, and diagnosis tasks, improve fault coverage, and guarantee the safe operation of the circuit under test. These circuit modes and operations are distinguished using additional test signals or test clocks. In order to illustrate how scan design works, consider the sequential circuit shown in Fig. 10.5.

This circuit contains combinational logic and three D flip-flops. Assume that a stuck-at fault  $f$  in the combinational logic requires the primary input  $X_3$ , flip-flop FF2, and flip-flop FF3 to be set to 0, 1, and 0, respectively, to capture the fault effect into FF1. Because the values stored in FF2 and FF3 are not directly controllable from the primary inputs, a long sequence of operations may have to be applied in order to set FF2 and FF3 to the required values. Furthermore, in order to observe the fault effect on the captured value in flip-flop FF1, a long checking experiment may be required to propagate the value of FF1 to a primary output. From this example, it can be seen that the main difficulty in testing a sequential circuit stems from the fact that it is difficult to control and observe the internal state of the circuit. Scan design, whose concept is illustrated in Fig. 10.6, attempts to ease this difficulty by providing external access to selected storage elements in a design. This is accomplished by first converting selected storage elements in the design into scan cells and then stitching them together to form one or more shift

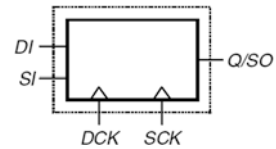


**Fig. 10.6** Scan design concept

**Fig. 10.7** Muxed-D scan cell

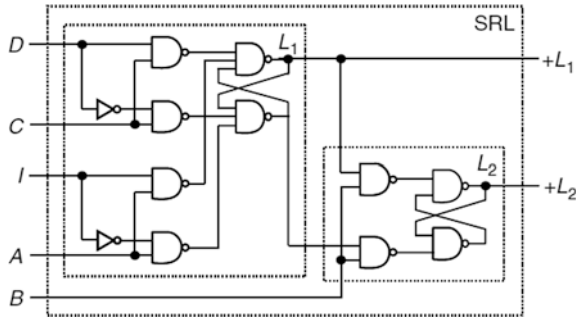


**Fig. 10.8** Clocked-scan

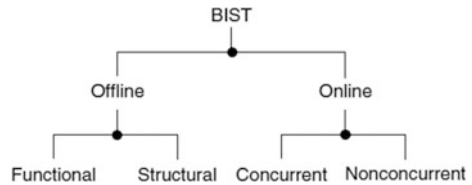


registers, called scan chains. In the scan design illustrated in Fig. 10.6, the  $n$  storage elements are now configured as a shift register in shift mode. Any test stimulus and test response can now be shifted into and out of the  $n$  scan cells in  $n$  clock cycles, respectively, without having to resort to applying an exponential number of clock cycles to force all storage elements to a desired internal state. Hence, the task of detecting fault  $f$ , in Fig. 10.4, becomes a simple matter of: (1) switching to shift mode and shifting in the desired test stimulus, 1 and 0, to FF2 and FF3, respectively; (2) driving a 0 onto primary input X3; (3) switching to capture mode and applying one clock pulse to capture the fault effect into FF1; and, finally (4) switching back to shift mode and shifting out the test response stored in FF1, FF2, and FF3 for comparison with the expected response. Because scan design provides access to internal storage elements, test generation complexity is reduced. Because there are two input sources in a scan cell, a selection mechanism must be provided to allow a scan cell to operate in two different modes: normal/capture mode and shift mode. In normal/capture mode, data input is selected to update the output. In shift mode, scan input is selected to update the output. This makes it possible to shift in an arbitrary test pattern to all scan cells from one or more primary inputs while shifting out the contents of all scan cells through one or more primary outputs. Internal structures of three widely used scan cell designs: muxed-D scan, clocked-scan, and level-sensitive scan design (LSSD) are to be shown in Figs. 10.7, 10.8 and 10.9, respectively.

**Fig. 10.9** Level-sensitive scan design (LSSD)



**Fig. 10.10** Logic BIST techniques

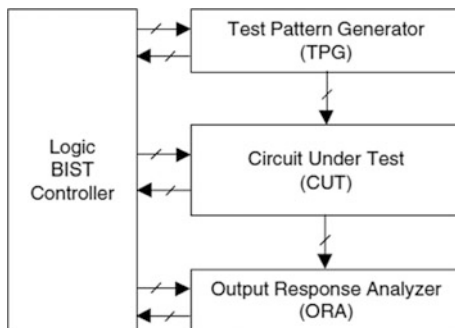


### 10.1.5.2 Built-In Self-Test

With recent advances in semiconductor manufacturing technology, the production and usage of very-large-scale integration (VLSI) circuits has run into a variety of testing challenges during wafer probe, wafer sort, pre-ship screening, incoming test of chips and boards, test of assembled boards, system test, periodic maintenance, repair test, etc. Traditional test techniques that use automatic test pattern generation (ATPG) software to target single faults for digital circuit testing have become quite expensive and can no longer provide sufficiently high fault coverage for deep submicron or nanometer designs from the chip level to the board and system levels. One approach to alleviate these testing problems is to incorporate built-in self test (BIST) features into a digital circuit at the design stage. With logic BIST, circuits that generate test patterns and analyze the output responses of the functional circuitry are embedded in the chip or elsewhere on the same board where the chip resides. There are two general categories of BIST techniques for testing random logic: (1) online BIST and (2) offline BIST. A general form of logic BIST techniques is shown in Fig. 10.10.

Online BIST is performed when the functional circuitry is in normal operational mode. It can be done either concurrently or nonconcurrently. In concurrent online BIST, testing is conducted simultaneously during normal functional operation. The functional circuitry is usually implemented with coding techniques or with duplication and comparison. When an intermittent or transient error is detected, the system will correct the error on the spot, rollback to its previously stored system states, and repeat the operation, or generate an interrupt signal for repeated failures. In nonconcurrent online BIST, testing is performed when the functional

**Fig. 10.11** A typical logic BIST



circuitry is in idle mode. The test process can be interrupted at any time so that normal operation can resume. Offline BIST is performed when the functional circuitry is not in normal mode. This technique does not detect any real-time errors but is widely used in the industry for testing the functional circuitry at the system, board, or chip level to ensure product quality. Functional offline BIST performs a test based on the functional specification of the functional circuitry and often employs a functional or high-level fault model. Normally such a test is implemented as diagnostic software or firmware. Structural offline BIST performs a test based on the structure of the functional circuitry. There are two general classes of structural offline BIST techniques: (1) external BIST, in which test pattern generation and output response analysis is done by circuitry that is separate from the functional circuitry being tested, and (2) internal BIST, in which the functional storage elements are converted into test pattern generators and output response analyzers. Some external BIST schemes test sequential logic directly by applying test patterns at the inputs and analyzing the responses at its outputs. Such techniques are often used for board-level and system level self-test. The BIST schemes discussed here all assume that the functional storage elements of the circuit are converted into a scan chain or multiple scan chains for combinational circuit testing. Such schemes are much more common than those that involve sequential circuit testing. Figure 10.11 shows a typical logic BIST system using the structural offline BIST technique.

The test pattern generator (TPG) automatically generates test patterns for application to the inputs of the circuit under test (CUT). The output response analyzer (ORA) automatically compacts the output responses of the CUT into a signature. Specific BIST timing control signals, including scan enable signals and clocks, are generated by the logic BIST controller for coordinating the BIST operation among the TPG, CUT, and ORA. The logic BIST controller provides a pass/fail indication once the BIST operation is complete. It includes comparison logic to compare the final signature with an embedded golden signature, and often comprises diagnostic logic for fault diagnosis. As compaction is commonly used for output response analysis, it is required that all storage elements in the TPG, CUT, and ORA be initialized to known states prior to self-test, and no unknown

(X) values be allowed to propagate from the CUT to the ORA. In other words, the CUT must comply with additional BIST-specific design rules.

There are a number of advantages to using the structural offline BIST technique rather than conventional scan: BIST can be made to effectively test and report the existence of errors on the board or system and provide diagnostic information as required; it is always available to run the test and does not require the presence of an external tester. Because BIST implements most of the tester functions on-chip, the origin of errors can be easily traced back to the chip; some defects are detected without being modeled by software. N-detect, a method for detecting a fault N times, is done automatically. At-speed testing, which is inherent in BIST, can be used to detect many delay faults. Test costs are reduced due to reduced test time, tester memory requirements, or tester investment costs, as most of the tester functions reside on-chip itself. However, there are also disadvantages associated with this approach. More stringent BIST-specific design rules are required to deal with unknown (X) sources originating from analog blocks, memories, non-scan storage elements, asynchronous set/reset signals, tristate buses, false paths, and multiple-cycle paths, to name a few. Also, because pseudo-random patterns are mostly used for BIST pattern generation, additional test points (including control points and observation points) may have to be added to improve the circuit's fault coverage. While BIST-specific design rules are required and the BIST fault coverage may be lower than that using scan, BIST does eliminate the expensive process of software test pattern generation and the huge test data volume necessary to store the output responses for comparison. More importantly, a circuit embedded with BIST circuitry can be easily tested after being integrated into a system. Periodic in-system self-test, even using test patterns with less than perfect fault coverage, can diagnose problems down to the level where the BIST circuitry is embedded. This allows system repair to become trivial and economical.

### 10.1.5.3 IEEE 1149.X Boundary Scan Design

Boundary scan design is a design for testability technique to simplify the application of test patterns for the detection and diagnosis of different faults at levels of packages (e.g. chips, modules, boards, backplanes). In circuit test based on the bed-of-nails probing technique makes it possible to test each chip and the interconnections among chips. However it requires the automatic test equipment to probe each chip pin and the increasing use of surface mounting techniques make it difficult to perform in-circuit test. Boundary scan is aiming to improve the card level testability by embedding a dedicated boundary scan register or making use of the part of the scan register in each chip. IBM boundary scan design has been developed in support of reduced pin count test and interconnect test where the boundary scan latches belong to the scan register. IEEE 1149.1 boundary scan design which uses an explicit test protocol is becoming a widely adopted industry standard. The conceivable defects on the interconnections among chips can be modeled as stuck-at, bridging, delay, and intermittent faults. A few test pattern

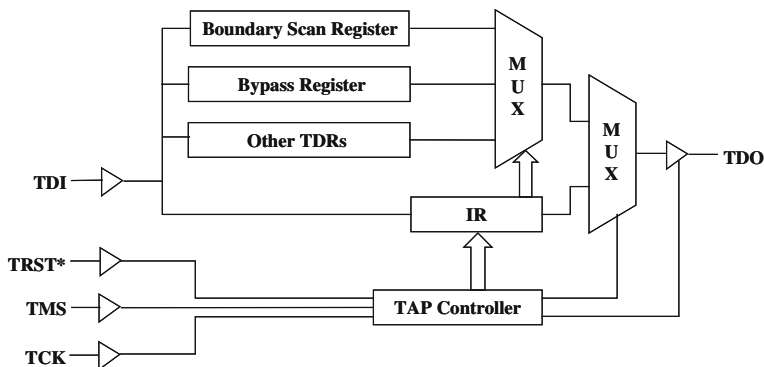


Fig. 10.12 IEEE 1149.1 boundary scan design

generation algorithms for static faults have been developed. IEEE boundary scan architecture consists of Test Access Ports (TAP), TAP controller, instruction and data registers. Test Data Input (TDI), Test Data Output (TDO), Test Clock (TCK), and Test Reset (TRST) constitute the TAP and TRST can be used optionally. Each input and output pin of a chip is connected to input and output boundary scan cells respectively. IEEE boundary scan instructions can be classified into compulsory ones such as BYPASS, EXTEST, and SAMPLE/PRELOAD and optional ones such as CLAMP, HIGHZ, and RUNBIST. TAP controller is a finite state machine with 16 states which mainly enable to apply patterns to data and instruction registers and to observe the test responses. Figure 10.12 shows a standard IEEE boundary scan design where the boundary scan, bypass or other test data register is connected to TDI-TDO path upon the instruction decoded.

The interconnect faults on a board can be summarized as follows:

1. S-at-1 and S-at-0: The conventional stuck at fault model.
2. S-open: The fault model for CMOS implementations which models any open net fault as either a pull-up or pull-down circuit. Initialization and transition patterns, that is, a two pattern test is required to detect a stuck-open fault.
3. Shorted Nets Faults: AND, OR, OPEN, DOMINATOR: The fault model for shorted nets fault scan be classified into AND, OR, OPEN and DOMINATOR type faults. Suppose two nets:(A, B) are shorted and let the logic values at each net be  $V(A)$  and  $V(B)$  respectively then:
  - (a) An AND type short results in logic 0 if either net is logic 0.
  - (b) Conversely an OR type short results in logic 1 if either net is logic 1.
  - (c) We call A DOMINATES B if  $V(A)$  appears at both nets regardless of  $V(B)$ . Similarly B DOMINATES A if  $V(B)$  always appears at both nets regardless of  $V(A)$ .
4. Delay fault: '0  $\rightarrow$  1' or '1  $\rightarrow$  0' transition cannot reach the receiver within a specified amount of time.

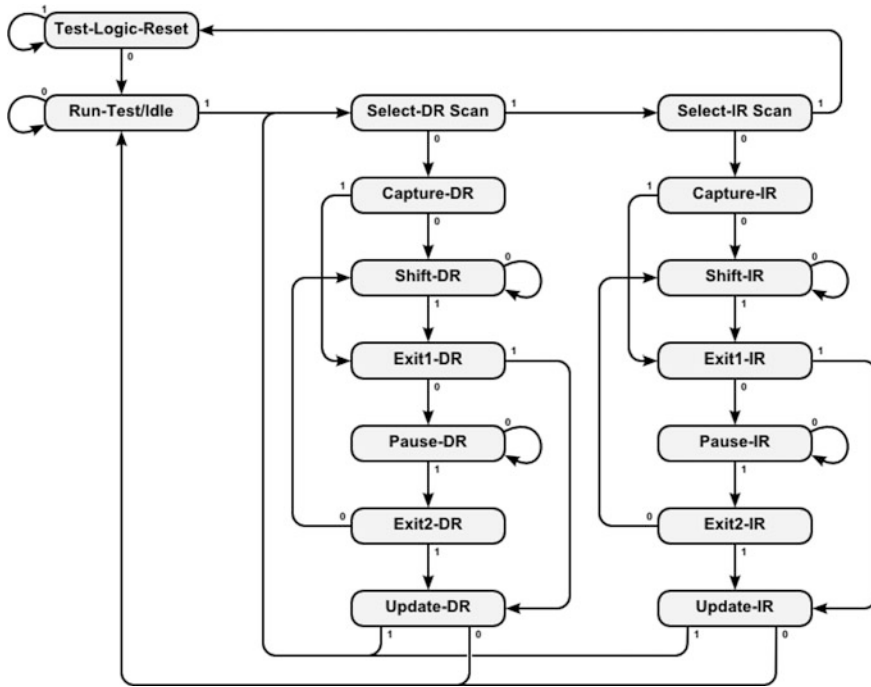


Fig. 10.13 16 states transition diagram of the TAP controller

Among 16 states of the TAP controller shown in Fig. 10.13, Update-DR and Update-IR states are active on the falling edge of the TCK while all the others are on the rising edge.

Although the IEEE boundary scan was initially focused to board and system level testing, recently the boundary is coming down to IP cores so that the SoC comprising of reusable cores can be tested and debugged through that boundary scan chains.

The method to apply and observe interconnect test patterns and state transitions of the Test Mode Selector can be summarized as follows:

1. EXTEST instruction is read and decoded. The state transition is :  
 RESET → IDLE → Scan-DR → Scan-IR → Capture-IR → Shift-IR → ... → EXIT1-IR →  
 Update-IR →
2. Interconnect test patterns are serially applied through the boundary scan register. The corresponding state transitions:  
 Scan-DR → Capture-DR → Shift-DR → ... → EXIT1-DR →
3. Test patterns read are applied to Update latch and the signals are propagated to input Boundary Scan Cells (BSC) in parallel. The corresponding state transitions:  
 Update-DR → Scan-DR → Capture-DR →



4. Test responses captured are shifted out through BSCs to TDO. The corresponding state transitions:  
Capture-DR → Shift-DR → ... → EXIT1-DR → Update-DR and Update-IR states are active on the falling edge of the TCK while all the others are on the rising edge.

#### 10.1.5.4 IEEE 1500 Wrapper

IEEE 1500 defines a standard for embedded core test interfaces, which includes a test wrapper, test ports, and wrapper control signals. In general, the core test access mechanism (TAM) and test method are supposed to be defined by SoC designers. The IEEE 1500 wrapper consists of Wrapper Instruction Register (WIR), Wrapper Bypass Register (WBY) and Wrapper Boundary Register (WBR). There are two test access terminals defined. One is mandatory Wrapper Serial Port (WSP) and the other is Wrapper Parallel Port (WPP). Unlike the IEEE 1149.1, wherein TAP control is defined as a standard, the test control logic of the IEEE 1500 is supposed to be customized by SoC integrators. However, in general, the IEEE 1149.1 TAP controller becomes the main control source for the embedded core test. IEEE 1500 instructions are classified into compulsory ones such as WS\_BYPASS, WS\_EXTEST, Wx\_INTEST, etc., and optional ones, such as WS\_INTEST\_SCAN, Wx\_EXTEST, WS\_SAFE, etc. Figure 10.14 shows an example of an IEEE 1500 wrapped core with a scan chain. It includes a parallel test domain through WPI and WPO as well as a serial test domain through WSI and WSO. According to different test modes, test paths are diversely reconfigurable using multiplexers. Depending upon the available scan input and output ports, internal scan chains can be directly connected to external WPI and WPO ports, or indirectly connected through internal input and output WBRs. Scan chains are controlled by se and clk signals, and Wrapper Serial Control (WSC) signals manipulate the IEEE 1500 wrapper instruction, bypass, and boundary registers.

Although the IEEE 1500 standard does not explicitly specify the access mechanism through the WSP, it presents the simple interface logic between the IEEE 1149.1 TAP controller and the IEEE 1500 WSP as shown in Fig. 10.15. The WRCK is directly connected to the TCK, and the SelectWIR signal is connected to Select from TAP via choice of either WIR or WBR. While Capture and Shift operations occur on the rising edge of the WRCK, Update operation occurs on the falling edge of the WRCK. The IEEE 1149.1 TAP is used to set the test mode of each core, configure test paths and generate control signals.

#### 10.1.5.5 Memory Testing

It becomes highly important to test various kinds of defects rapidly and precisely to reduce the testing cost and to improve the memory quality especially under the

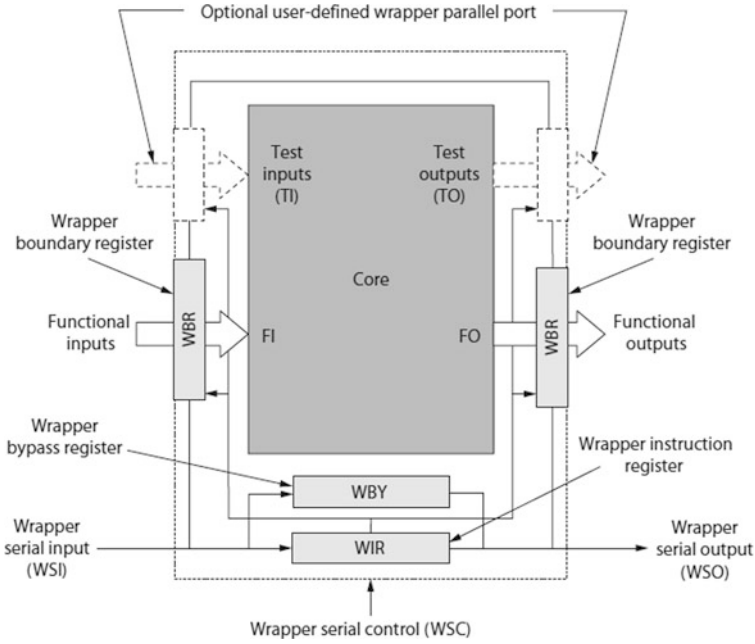


Fig. 10.14 Example core and its IEEE 1500 wrapper

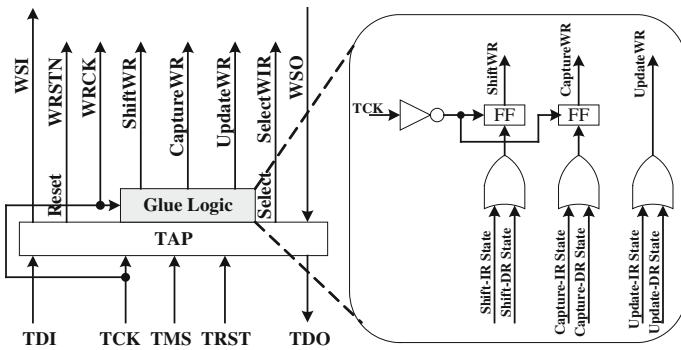


Fig. 10.15 IEEE 1149.1 TAP-to-IEEE 1500 WSP interface logic

SoC design environment. Memory defects can be modeled as stuck-at, coupling, transition, address decoder, and pattern-sensitive faults and it is known that the 80 % of the failures are due to the leakage defects. Among the different testing algorithms ranging from  $O(\sqrt{n})$  to  $O(n \log(n))$ , BIST(Built-In Self Test) techniques with  $O(\sqrt{n})$  to  $O(n)$  complexity algorithms are widely adopted for embedded memories. Memory test patterns can be generated deterministically or randomly through either a test equipment or a BIST circuitry. Test patterns

generated randomly can detect not only modeled defects but non-modeled and timing defects, nevertheless deterministic march patterns are widely adopted for BIST and off-chip testing for their simplicity.

### 10.1.5.6 Fault Models and Definitions

Faults modelled from the memory defects can be summarized as followings.

- (1) Stuck-at-Fault(SF) : Either a cell or a line is stuck to logical '0' or '1'.
- (2) Transition Fault(TF) :  $0 \rightarrow 1$  (or  $1 \rightarrow 0$ ) transition is impossible on a cell or a line.
- (3) Coupling Fault(CF) : When a cell is written to  $0 \rightarrow 1$  (or  $1 \rightarrow 0$ ), the content of the other cell is changed. CF is generalized to k-coupling fault when k-1 cells are changed and further more classified into Inversion or Idempotent coupling faults upon the content changed.
- (4) Address Decoder Fault(ADF) : No cell will be accessed with a certain address, or multiple cells are accessed simultaneously, or a certain cell can be accessed with multiple addresses.
- (5) Address Decoder Open Faults(ADOF) : CMOS address decoder open faults are caused by open defects in the CMOS logic gates of the memory address decoders and, due to their sequential behavior, cannot be mapped to faults of the memory array itself.
- (6) Retention Faults(RF) : A cell fails to retain its logic value after some time. This fault is caused by a broken pull-up resistor.
- (7) Neighborhood Pattern Sensitive Fault (NPSF) : a typical neighborhood pattern sensitive faults preventing the base cell from being transitioned to certain value is called as 'static' NPSF, and an NPSF is named as 'dynamic' when a transition on the neighborhood cells triggers the transition on the base cell.

### 10.1.5.7 Memory March Test

A bit-oriented March C– algorithm is given in Table 10.4 as a March test example [11]. In Table 10.4, there are six **March elements**, denoted as **M0**, **M1**, ..., **M5**.

In each March element, we first specify the address sequence:  $\uparrow$  means that the address sequence is in ascending order,  $\downarrow$  means that the address changes in descending order, and  $\&$  means that either  $\uparrow$  or  $\downarrow$  is acceptable. Consider M1, for example; the address sequence begins at the lowest address and changes in ascending order toward the highest address. For each address (memory cell), perform a read operation (with an expected 0 in the fault-free case) and write back the complemented bit immediately, then continue to the next address. The algorithm is also called the March 10 N algorithm as it requires 10 N read/write operations, where N is the number of memory cells (address locations).

**Table 10.4** The March C–algorithm

$\Downarrow(w0);$	$\Uparrow(r0w0);$	$\Uparrow(r1w0);$	$\Downarrow(r0w1);$	$\Downarrow(r1w0);$	$\Downarrow(r0);$
$M_0$	$M_1$	$M_2$	$M_3$	$M_4$	$M_5$

March C– is known to completely detect SAFs, unlinked AFs, unlinked TFs, and CFs (including CFins, CFids, and CFsts). It also detects SOFs if M1 is extended to  $r0w1r1$ , or M2 to  $r1w0r0$ . The resulting algorithm is called the extended March C– algorithm. In order to reduce the test cost, appropriate fault models and test algorithms should be chosen. Some other March tests are summarized below:

**Modified algorithmic test sequence (MATS)** –  $\{\Downarrow(w0); \Downarrow(r0,w1); \Downarrow(r1)\}$

**MATS<sup>+</sup>** –  $\{\Downarrow(w0); \Uparrow(r0,w1); \Downarrow(r1,w0)\}$

**Marching 1/0** –  $\{\Uparrow(w0); \Uparrow(r0,w1,r1); \Downarrow(r1,w0,r0); \Uparrow(w1); \Uparrow(r1,w0,r0); (r0,w1,r1)\}$

**MATS<sup>++</sup>** –  $\{\Downarrow(w0); \Uparrow(r0,w1); \Downarrow(r1,w0,r0)\}$

**March X** –  $\{\Downarrow(w0); \Uparrow(r0,w1); \Downarrow(r1,w0); \Downarrow(r0)\}$

**March C** –  $\{\Downarrow(w0); \Uparrow(r0,w1); \Uparrow(r1,w0); \Downarrow(r0); \Downarrow(r0,w1); \Downarrow(r1,w0); \Downarrow(r0)\}$

**March C<sup>–</sup>** –  $\{\Downarrow(w0); \Uparrow(r0,w1); \Uparrow(r1,w0); \Downarrow(r0,w1); \Downarrow(r1,w0); \Downarrow(r0)\}$

**March A** –  $\{\Downarrow(w0); \Uparrow(r0,w1,w0,w1); \Uparrow(r1,w0,w1); \Downarrow(r1,w0,w1,w0); \Downarrow(r0,w1,w0)\}$

**March Y** –  $\{\Downarrow(w0); \Uparrow(r0,w1,r1); \Downarrow(r1,w0,r0); \Downarrow(r0)\}$

**March B** –  $\{\Downarrow(w0); \Uparrow(r0,w1,r1,w0,r0,w1); \Uparrow(r1,w0,w1); \Downarrow(r1,w0,w1,w0); \Downarrow(r0,w1,w0)\}$

### 10.1.5.8 Memory BIST

Figure 10.16 diagrams a BIST design and the interface between the BIST logic and the embedded DRAM [12]. The BIST activation control (BAC) input activates the BIST logic; the embedded DRAM is in normal mode when BAC is 0 and in BIST mode when BAC is 1. The BIST controller is a finite-state machine; its state transition is controlled by the BIST control selection (BCS) input. The BIST controller also controls the scan chains, shifting in test patterns and commands from the BIST scan-in (BSI) input and shifting out results from the BIST scan-out (BSO) output.

As Fig. 10.16 shows, the controller contains multiple chains. The decode logic and test mode selection modules determine the proper data register to scan in the test commands and subsequently activate the sequencer. The sequencer generates the DRAM’s timing sequence, with the help of some built-in counters and the timing generator. The comparator compares and reports any discrepancy between the output data from the DRAM and the original input data generated by the sequence controller. The BIST logic has three additional I/O signals. The BIST ready flag (BRD\*) indicates when the BIST sequence is finished, so that the go/no-go indicator signal

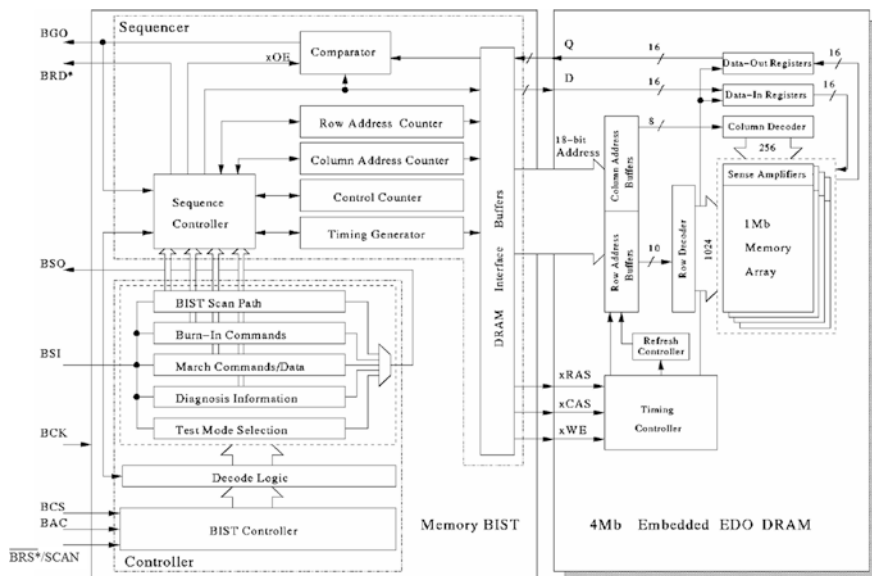


Fig. 10.16 Block diagram of a Memory BIST

(BGO) can be sampled to check that the embedded DRAM is functioning correctly. The BRS\*/SCAN signal acts as both the reset and scan test control. All registers in the BIST controller finite-state machine are scanned, and before we use the BIST logic to test the DRAM, the logic itself is scan tested. Finally, we need a BIST clock (BCK) input. BCK and BAC must be dedicated; others cannot share these two input pins (for example, by using multiplexers). But BRD\* is optional and may be removed if pin count is a concern. In that case, we can encode BGO to signal the completion of the BIST sequence and show the test result. The reset (BRS\*) also is optional, since a short synchronizing sequence for the BIST controller can be the reset sequence. However, the SCAN pin is still required in that case. Apart from BCK and BAC, all other BIST I/O signals can share pins with signals outside the DRAM core; thus, we can use multiplexed pins to reduce pin overhead.

The BIST supports the following test modes:

*Scan test*—testing the BIST logic, except the BIST controller finite-state machine. We execute scan test at the beginning of the BIST sequence to ensure the circuit’s correct functionality. In addition, we test all registers in the DRAM core in this mode.

*Memory BIST*—functional testing of the DRAM using march algorithms. This mode exercises various operation modes, such as non-page-mode test, page mode test, refresh test, and retention test. This mode also supports diagnosis. In that case, the BIST logic can shift out the address of any faulty cell, column, or row to the external tester via the scan mechanism. We can test for retention faults in this mode or in a separate test mode.

*Burn-in*—stress testing to screen out unreliable parts that may fail in infancy. This mode uses the BIST logic to exercise the entire memory cell array in a more efficient method than the normal read/write access. The default burn-in test is to use a march algorithm supported in the memory BIST mode.

*Timing-fault test*—testing for critical timing faults by running the BIST clock at an appropriate speed. Among these faults are incorrect setup time, hold time, and data arrival time of various control and data signals. We can simultaneously detect some timing faults, such as incorrect setup time and hold time, when we perform functional test (in memory BIST mode). We can test for others by using different BIST clock periods or an external.

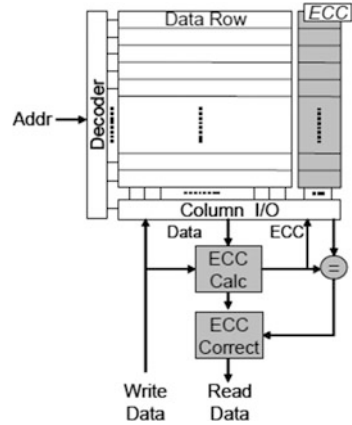
### 10.1.5.9 Memory Error Correcting Codes

With the scaling of process technologies into the nanometer regime, the reliability of embedded memory systems becomes an increasingly important concern for digital system designers. Nano-scale components themselves are increasingly likely to fail, and the growing amount of on-chip memory creates more possible points of failure. As designers integrate more of the memory hierarchy onto the processing die, the number and size of memory arrays on these system-on-chip (SoC) and microprocessors will increase. Therefore, errors occurring in the embedded memory systems are a growing threat to the overall SoC and processor reliability and yield [13].

To protect the integrity of the data in the memory, error correcting code (ECC) plays a vital role. The most common codes used are single error correction-double error detection (SEC-DED) codes. The most popular are Hamming, and Hsiao. These codes can correct single bit errors in a codeword and can detect double bit errors. These codes require storing additional check bits in the memory. Along with the columns needed to carry out ECC, the memory includes additional spare columns for repair. In some cases, check bits are used along with spare rows and columns to provide a combined fault-tolerance. Current memory designs contain redundant rows, columns, and sub-arrays to tolerate manufacture-time hard errors and thus improve yields. When faulty bits are detected during product testing, the faulty addresses are remapped to redundant spare rows or columns using built-in self repair (BISR) techniques. While in the worst-case most defective memories on the tail end of the statistical curve may use all of the spare resources, most memories will have unused spare resources after the repair.

We focus our attention on the conventional systematic linear block SEC-DED codes. ECC is typically applied to the data on a per-word basis (e.g., 64 bits) and allows error detection/correction at the cost of extra bits of code storage per word and shared calculation/checking/correction logic (Fig. 10.17).

**Fig. 10.17** Error detection and correction



The length of the code words, the number of information bits and the number of check bits are denoted by  $n$ ,  $k$  and  $r = (n - k)$ , respectively. The H-matrix is:

$$H = [A^T | I_{n-k}] \tag{10.1}$$

Where  $A$  is a  $k$ -by- $(n-k)$  parity check generator matrix and  $I_{n-k}$  is an  $(n-k)$ -by- $(n-k)$  identity matrix. The code generator matrix denoted as  $G$  is defined as follows:

$$G = [I_k | A] \tag{10.2}$$

If  $u$  is a  $1$ -by- $k$  data bit vector, then its corresponding  $n$  bit codeword vector  $x$  is formed as  $x = u \cdot G$ . In this paper, the codes are described by their  $(r$ -by- $n)$  parity check matrix (H-matrix).  $C$  is a codeword of the code if and only if:

$$H \cdot C^T = 0 \tag{10.3}$$

An error vector  $E$  is defined as an  $r$ -bit vector where the bits that are in error have a value  $1$  and all the other bits are  $0$ . An erroneous message  $W$  error can be represented as follows:

$$W_{error} = C \oplus E \tag{10.4}$$

The syndrome,  $S$ , is defined as follows:

$$S = H \cdot W_{error} = H \cdot (C \oplus E) = H \cdot E \tag{10.5}$$

The value of the syndrome is equal to zero if the transmitted codeword is not corrupted. If the received codeword contains detectable errors then the syndrome is non-zero. If the received codeword contains correctable errors, then the syndrome

identifies the error pattern corrupting the transmitted codeword, and these errors can then be corrected.

For single error correction (SEC) Hamming code, each column vector in the H-matrix is non-zero and distinct. This ensures that the syndrome for any single bit error will result in a unique syndrome. By decoding the syndrome, it is possible to determine which bit the error is in and flip the value of that bit to correct the error.

## 10.2 Conclusions

It is imperative to design and manufacture highly reliable automobile SoCs satisfying AEC-Q100 stress and ISO 26262 functional safety. After describing stress and safety conditions, different design for reliability techniques have been introduced. Advanced BIST/BISR, ECC and fault tolerant architectures have to be adopted to cope with various static and dynamic failures.

## References

1. S.J. Sero, Your car could take off by itself sudden acceleration is not a myth. Renaissance engineering, forensic engineers. <http://www.forensicfacts.com/sudden%20acceleration%20article.pdf>
2. H. Tahne, *Safe and Reliable Computer Control Systems Concepts and Methods* (Mech Lab Univ Stock, Stockholm, 1996)
3. R. Mariani, The impact of functional safety standards in the design and test of reliable and available integrated circuits, in *IEEE European Test Symposium (ETS)*, 2012
4. IEC 61508-1/7:2010, IEC
5. International Organization for Standardization (ISO): Road vehicles, Functional safety. ISO 26262-1:2011 [http://www.iso.org/iso/iso\\_catalogue/catalogue\\_tc/catalogue\\_detail.htm?csnumber=43464](http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=43464) and ISO 26262-10:2012 [http://www.iso.org/iso/catalogue\\_detail?csnumber=54591](http://www.iso.org/iso/catalogue_detail?csnumber=54591)
6. ISO 26262-5:2011, Road vehicles, Functional safety: Part 5: Product development at the hardware level
7. ISO 26262-10, Road vehicles, Functional safety: Part 10: Guideline on ISO 26262
8. S. Jeon, J. Cho, Y. Jung, S. Park, T. Han, Automotive hardware development according to ISO 26262. In *13th International Conference on Advanced Communication Technology (ICACT)* (2011) pp. 588–592
9. [en.wikipedia.org/wiki](http://en.wikipedia.org/wiki)
10. L.T. Wang, C.W. Wu, X. Wen (eds) *VLSI Test Principles and Architectures: Design for Testability*, Elsevier, 2006, ISBN 13:978-0-12-370597-6
11. A.J. van de Goor, Using March tests to test SRAMs. *IEEE Des. Test Comput.* **10**(1), 8–14 (1993)
12. C.-T. Huang, J.-R. Huang, C.-F. Wu, C.-W. Wu, T.-Y. Chang, A programmable BIST core for embedded DRAM. *IEEE Des. Test Comput.* **16**(1), 59–70 (1999)
13. J. Jung, U. Ishaq, J. Song, S. Park, Efficient use of unused spare columns for reducing memory miscorrections. *J. Semicond. Technol. Sci.* **12**(3), 331–340 (2012)