A STUDY OF SMART DEVICE-BASED MOBILE IMAGING AND IMPLEMENTATION FOR ENGINEERING APPLICATIONS

A THESIS IN
Electrical Engineering

Presented to the Faculty of the University
of Missouri—Kansas City in partial fulfillment of
the requirements for the degree

MASTER OF SCIENCE

by
Jianfei Chen
B.E. Beijing University of Posts and Telecommunications, Beijing, China, 2010

Kansas City, Missouri
2013

A STUDY OF SMART DEVICE-BASED MOBILE IMAGING AND IMPLEMENTATION FOR ENGINEERING APPLICATIONS

Jianfei Chen, Candidate for the Master of Science Degree

University of Missouri—Kansas City, 2013

ABSTRACT

Mobile imaging has become a very active research topic in recent years thanks to the rapid development of computing and sensing capabilities of mobile devices. This area features multi-disciplinary studies of mobile hardware, imaging sensors, imaging and vision algorithms, wireless network and human-machine interface problems. Due to the limitation of computing capacity that early mobile devices have, researchers proposed client-server module, which push the data to more powerful computing platforms through wireless network, and let the cloud or standalone servers carry out all the computing and processing work.

This thesis reviewed the development of mobile hardware and software platform, and the related research done on mobile imaging for the past 20 years. There are several researches on mobile imaging, but few people aim at building a framework which helps engineers solving problems by using mobile imaging. With higher-resolution imaging and high-performance computing power built into smart mobile devices, more and more imaging processing tasks can be achieved on the device rather than the client-server module. Based on this fact, a framework of collaborative mobile imaging is introduced

for civil infrastructure condition assessment to help engineers solving technical challenges.

Another contribution in this thesis is applying mobile imaging application into home automation. E-SAVE is a research project focusing on extensive use of automation in conserving and using energy wisely in home automation. Mobile users can view critical information such as energy data of the appliances with the help of mobile imaging.

OpenCV is an image processing and computer vision library. The applications in this thesis use functions in OpenCV including camera calibration, template matching, image stitching and Canny edge detection. The application aims to help field engineers is interactive crack detection. The other one uses template matching to recognize appliances in the home automation system.

APPROVAL PAGE

The faculty listed below, appointed by the Dean of the School of Computing and Engineering have examined a thesis titled "A Study of Smart Device-based Mobile Imaging and Implementation for Engineering Applications", presented by Jianfei Chen, candidate for the Master of Science degree, and certify that in their opinion it is worthy of acceptance.

Supervisory Committee

ZhiQiang Chen, Ph.D.
Department of Civil and Mechanical Engineering

Yugyung Lee, Ph.D.
Department of Computer Science and Electrical Engineering

Yang Yi, Ph.D.
Department of Computer Science and Electrical Engineering

TABLE OF CONTENTS

ILLUSTRATIONS

TABLES

ACKNOWLEDGEMENTS

I would like to express my deepest gratitude to my adviser, Dr. ZhiQiang Chen, for his excellent guidance, patience and providing me with excellent atmosphere for doing research. I am sure it would have not been possible without his help. His research attitude and method is truly inspirational. He always encourages me to explore novel ways of doing research. He has invested a lot of time and energy in guiding through my thesis here at University of Missouri-Kansas City.

I would also like to thank my committee members, Dr. Yugyung Lee and Yang Yi for their support and interest in my thesis. Also, the team members of the Toyota project especially Dr. Vijay Kumar, Dr. Cory Beard, Dr. Praveen Rao and Dr. Walter D. Leon-Salas have motivated me in completing this thesis.

I would like thank Xuan Guo and Rahul Tripathi, who as friends were always willing to help and discuss problems with me. It would have been a lonely lab without them.

I also would like thank Feichen Shen, whom as a friend and colleague that I learned a lot stuff from. His computing and science's thinking method helps me view the problems in different ways.

Finally, and more importantly, I wish to thank my parents, for their care and encourage throughout the years.

CHAPTER 1

INTRODUCTION

In this chapter, a brief history of mobile imaging and the research motivation of this thesis are introduced. First the evolution of mobile imaging and hardware is reviewed. Then why mobile imaging is important and how it is going to affect our life and society will be discussed. Finally, the goals of this research and the organization of this thesis are stated.

1.1 Historical and Today's Mobile Imaging

The concept of modern mobile imaging has two core features: (i) portable generation of digitalized images; and (ii) deliverable products through mobile communication networks. Therefore, modern mobile imaging tightly relied on three technological advances that occurred during the last 20 years. The first was the development of small-form-factor computers since the late 90s (e.g., various brands of Palms or pocket PCs) [Forman and Zahorjan, 1994]. The second was the invention of digital imaging sensors. In the early 1990s, Dr. Eric Fossum and his coworkers developed the complementary metal–oxide–semiconductor (CMOS) active pixel sensor "camera-on-a-chip", which made the first step of producing mobile devices equipped with digital cameras [Fossum, 1997]. Last, the integration of mobile computers and digital imaging in a phone with mobile communication network. In 1997, Philippe Kahn shoot, sent, and shared with many people the first mobile phone-based digital image showing his newborn daughter (Figure 1.1) [Fullpower, 2013]. Although the quality of the image was quite primitive, it is considered as a remarkable event for mobile imaging.

Figure 1.1 The very first picture taken by and shared via a mobile phone.

Today's mobile devices tightly integrate multi-core CPUs, high-resolution cameras, flexible communication and wireless networks (e.g., 3G/4G cellular network, Wi-Fi including Wi-Fi Direct, Bluetooth, and near-field communication), interactive interface through sensitive touch-based screens, geographical position service, and embedded gyroscopic and acceleration sensors. Therefore, these devices are commonly regarded 'smart' and are represented by smartphones and smartphone-alike tablet devices (the latter usually lack cellular network support).

These smart mobile devices with embedded digital cameras have become truly ubiquitous. There are over one billion mobile devices around the world at the end of 2012 [Mobithinking, 2013], and many of them have embedded cameras. Millions of pictures are taken and saved in these devices every day for a variety of uses. Compared to traditional cameras, the advantage of mobile devices may include easy to carry, easy to share, and easy to modify due to the small form-factor hardware design, the communication capability, and the computing capability of mobile devices. As such,

today's mobile devices as everyday imaging utility are tend to be much more frequently used than single cameras.

1.2 Mobile Hardware Advances

Mobile imaging as a research area has been very active in recent years thanks to the rapid development of computing and sensing capabilities of mobile devices. This area features multi-disciplinary studies of mobile hardware, imaging sensors, imaging and vision algorithms, wireless network and human-machine interface problems. Due to the fast technological advance in very large scale integration (VLSI), highly integrated single chips can achieve a variety of functions for mobile devices. Multi-functional chips are implemented to make the modern mobile devices smaller, cheaper, more powerful, and more energy efficient. In the following, key advances in mobile CPUs, GPUs, imaging sensors and other embedded sensors are introduced.

Mobile CPU

Early mobile device has very limited computing capability. In 1984, British company Psion introduced the world first pocket computer named "Organizer", with a "6 by 6" keyboard arranged alphabetically. It has an 8-bit Hitachi 6301-family processor, running at 0.9MHz, with 4kB of ROM and 2kB of static RAM, and had a single-row monochrome LCD screen [Bioeddie, 2006]. The function of this device is very limited; it only provided a simple flat-file database, calculator and clock, with no operating system. The term PDA (Personal digital assistant) was used to describe such mobile devices. As the development of CMOS technology, more powerful computing units were introduced to mobile devices so that the device can have more functionality. In 1992, IBM debuted a prototype device combined a cell phone and PDA into one device, which was known as

Simon later in 1994 [Sager, 2012]. The CPU (Central Processing Unit) of Simon was called Vadem, a 16 MHz, 16-bit, x86-compatible processor, which was 14.4 times faster compared with the Organizer's processor. Thus, more tasks such as receiving telephone calls, facsimiles, emails and cellular pages can be done with Simon.

The development of mobile CPU rapidly advances during the last 10 years. Each year there will be new processor announced and the computing capability is growing rapidly. As Figure 1.2 shows some representative mobile CPUs with their computing speed in the last 10 years. Until early 2013, mobile CPU Tegra 4, which is announced by Nvidia has a quad-core CPU Up to 1.9 GHz designed for mobile devices [NVIDIA, 2013].

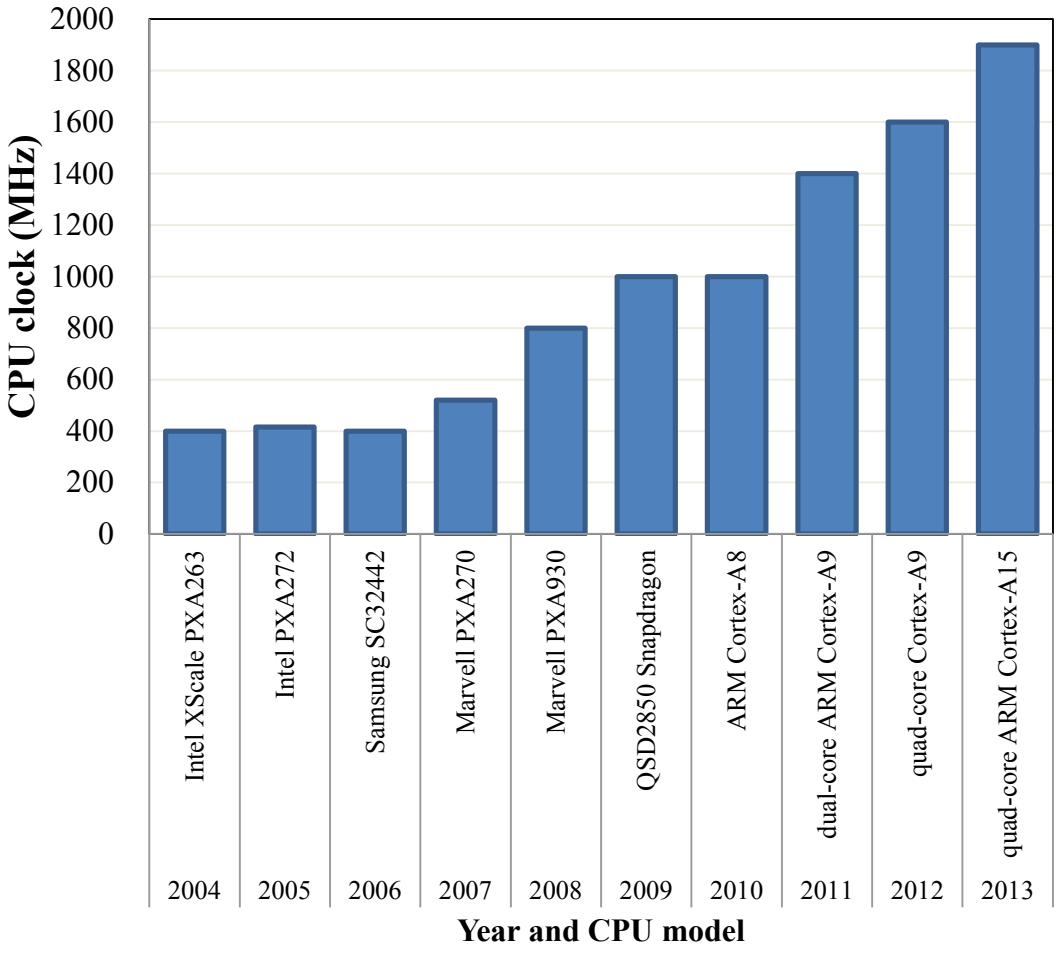


Figure 1.2 Representative mobile CPUs and their speed in recent years [Gsmarena, 2013].

Mobile GPU

Besides CPU, GPU (Graphics Processing Unit) is another important computing processor for mobile devices. GPU was originally used to accelerate image and graphic rendering for computers. GPU can improve the computing speed and accuracy for image and graphic applications. For example, in this article [Xue et al., 2006], the author indicated that using GPUs to process medical CT image could be 30-70 times faster than using CPUs only. Although the concept of GPU has existed since 1980s, it was brought

to mobile devices very recently. In 2007, PowerVR SGX GPU series chips were implemented in many mobile devices including Apple's iPhone [Klug, 2011]. Later, Nvidia's Tegra GPU series with 300-400 MHz core clock speed were introduced to mobile devices. With GPU units on mobile devices, significant improvement is expected to be gained for many applications, such as 3-D gaming, physics simulation for realistic effects and image processing [Krüger and Westermann, 2005; Owens et al., 2008; Cheng and Wang, 2011].

Imaging Sensors

Due to limitation such as power, form factors and cost, digital camera in a mobile device would require higher level of camera electronics integration to permit the miniaturization, which will cause a heavy loss on resolution. As mentioned earlier, the first image taken and shared on a mobile phone had a very low resolution of 320 × 240 pixels, and there are noise and grid on the image. Nowadays, mobile camera technology is fairly evolved, recent news reveals that the Nokia 808 smartphone has been equipped a camera with a resolution of 7152 × 5368 pixels, and is able to capture 1080p video stream data at 30fps (frame per second). Figure 1.3 shows some remarkable mobile devices' camera resolutions in the last decades. Notice that the resolution in pixels has increased more than 100 times when comparing the latest technology with the first commercial camera phone (J-SH04).

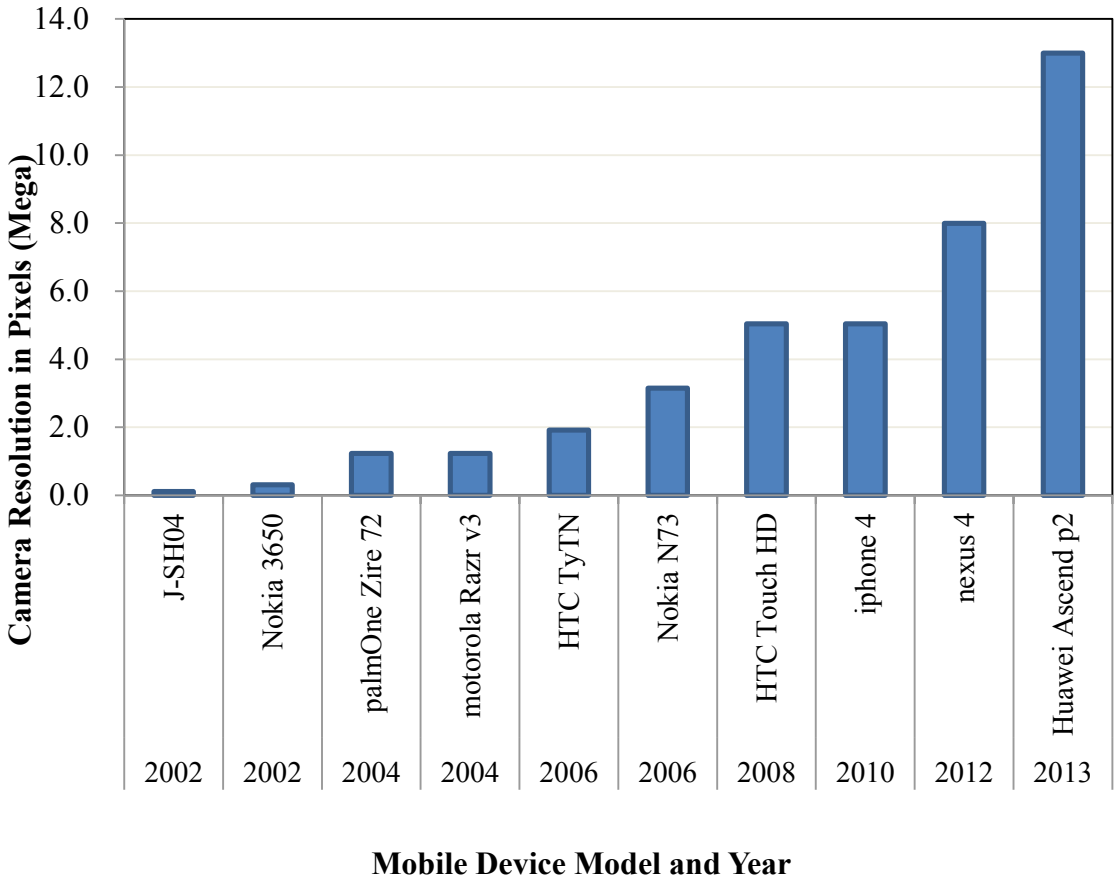


Figure 1.3 The evolution of mobile device camera resolution in the past decade.

Other Embedded Sensors

Besides the CPUs and GPUs enabled advanced computing and the high-resolution imaging capabilities, smart mobile devices tightly integrate many types of heterogeneous sensors, including accelerometers, gyroscopes, GPS, Wi-Fi/3G/4G interfaces, digital compass and so on. These sensors enhanced the novel use of mobile devices in many ways. For example, accelerometers are often used to automatically determine the orientation in which the user is holding the phone and to use that information to automatically re-orient the display between a landscape and portrait view or correctly

orient captured photos during viewing on the phone [Lane et al., 2010]. By using accelerometer, GPS and magnetometer sensors on smartphones, a non-intrusive method was introduced by Bhoraskar, et al. to detect city traffic and road condition [Bhoraskar et al., 2012]. Another study include the use of a smartphone's embedded accelerometer to recognize human activity [Khan et al., 2010]. Combining accelerometer, gyroscope, magnetometer, GPS and camera on mobile devices, researchers also developed a system to classify and recognize driving styles [Johnson and Trivedi, 2011].

1.3 Research Motivation and Objective

With higher-resolution imaging and high-performance computing power built into smart mobile devices, they have become a serious alternative choice to other professional imaging equipment. While mobile hardware and software systems are continually advancing, mobile imaging is applied in many areas, including but not limited to robotics, medical informatics, engineering, and automation control [Tao et al., 1998; Royer and Toh, 1999; Tu and Li, 2000; Kondo, 2002; Andrade et al., 2003; Yeh et al., 2005; Yu et al., 2007; Cho and Kim, 2011]. However, due to the ubiquitous nature of mobile devices, two technical challenges may adversely arise:

1) The first is the dilemma of large-scale data collection capability due to the massive use of mobile devices and the insufficient visual computing of individual devices;
2) The second is the coordination of multiple devices, when used in professional applications, to efficiently collect and share data in/for a spatially large area/object.

To address these two problems, this exploratory thesis project is to investigate into the use of common computer vision algorithms in mobile devices and to develop intelligent mobile imaging applications for several engineering problems. First, by adapting the technology of mobile device's both hardware and software platforms, this project aims to build a framework that is suitable for mobile imaging applications and data sharing. Second, it will implement several image processing algorithms and computing tasks based on one of the most popular mobile operating systems. Different from the traditional image processing applications running on workstations or desktops, which impose less restriction on computation power and memory spaces, this work needs to explore novel approaches, combining existing image and vision libraries and the operating system, to achieve an efficient and lightweight implementation.

The thesis will carry out research into an imaging framework based on mobile devices, combining the mobile hardware and the operating system to achieve the tasks. Two application domains are considered: one is mobile condition inspection for civil infrastructure and the second is mobile smart energy for home automation. For the two applications, a brief insight of human-centered concept of image sensing and computing framework implemented on Android-based mobile platforms will be presented.

1.4 Research Tasks and Scope

With the aforementioned research objective and the two application problems, the specific goal of this project is to develop a real-time imaging framework that helps professional users complete their field work based on mobile devices. To achieve this, the basic research tasks are presented as follows:

1) Complete camera calibration that is necessary since image distortion was a significant effect on subsequent image analysis;
2) Investigate the development environments that have certain basic image and vision library, if not, it is necessary to build one;
3) Construct local wireless data network for collaborative imaging;
4) Implement real-time imaging applications for particular engineering applications;
5) Perform system evaluation and compare the computing capability of mobile platforms with workstations.

It is worth noting that mobile imaging in this thesis is beyond "mobile image sensing" only; rather, mobile imaging, relying on the modern smart mobile devices includes the following components, which define the phases of mobile imaging.

- Image acquisition
- Image processing and understanding
- Visual analytics

Android based mobile systems will be used in this project. There are a lot of image processing and computer vision libraries in the literature and practice. OpenCV (Open Computer Vision), which aims at real-time image processing, is selected in this thesis. Accordingly, the two engineering applications will be implemented for Android mobile devices. One application aims at civil infrastructure condition assessment, wherein image stitching and edge detection algorithms are used to conduct real-time and interactive crack detection. The other smart energy related application will mainly adopt the template matching algorithm for object recognition.

1.5 Thesis Organization

The organization of the rest of the thesis is presented as follows:

- A background and literature view of mobile platform and image and vision library will be covered in Chapter 2; it also contains introduction of current available technology of mobile wireless network.
- Chapter 3 will describe the details of how general imaging processing methods such as camera calibration, edge detection and object matching, are implemented on mobile phones.
- Contents in Chapter 4 show the implementation of collaborative mobile imaging framework and crack detection application for civil infrastructure condition assessment. Then system computing capability evaluation is performed.
- Chapter 5 will discuss another imaging processing framework and application for smart energy monitoring.
- Conclusions drawn from this thesis research and future research recommendations will be stated in Chapter 6.

CHAPTER 2

BACKGROUND AND RELATED WORK

In this Chapter, the research background with regard to mobile software platforms for research defined in this project will be discussed. First, the popular mobile operating systems are introduced. Then a general view of existing mobile image processing and vision libraries will be reviewed. Subsequently mobile wireless network technology and how this would help to build a local mobile imaging network is presented. The chapter will also make an extensive literature view of similar research efforts and the existing approaches to mobile imaging problems.

2.1 Mobile Operating Systems

Mobile imaging involves multiple technological areas, including computing, image sensing, network, and other embedded sensing. Mobile operating system provides the vital connections between these components.

General View

Early mobile devices used embedded simple operating systems to control their operations. The very first mobile device with limited computational capability can be traced back to “Organizer”, a pocket computer developed by Psion in 1984 [Bioeddie, 2006]. Although the idea of combining communication and computing on the same device was implemented in 1970’s, the first mobile device that achieved this was IBM’s personal communicator, IBM Simon in 1994, which was considered to be the first smartphone in the world. Simon used the file system from Datalight ROM-DOS along with file compression from Stacker. IBM created a unique touch-screen user interface for

Simon. Since then, the notion of mobile operating system becomes the fundamental structure for designing mobile devices.

During the last twenty years, various mobile operating systems have been developed for different mobile devices. From 1996, Palm OS and Windows CE were initially introduced for PDAs and handheld PC devices. Multiple mobile OS's were introduced in the next decades, such as Nokia S40, Symbian and Blackberry OS[Nokia, 2010; RIM, 2012]. Among these operating systems, Symbian has been reported as one of the most widely used mobile OS's worldwide. Reports from 2005 indicated that Symbian had 51% market share of the mobile OS [ZDnet, 2005]. In 2007, Apple Inc. unveiled iPhone with its unique operating system iOS, which was derived from OS X. The next year, Open Handset Alliance (OHA) led by Google Inc., released Android 1.0 with the HTC Dream as the first Android phone. Android platform grows rapidly both in system evolution and market share since then. A penetration trend of mobile operating systems for smart devices is shown in Figure 2.1 based a recent market report [Gartner, 2013]. Clearly, Android system has evolved quickly and leads the market share significantly.

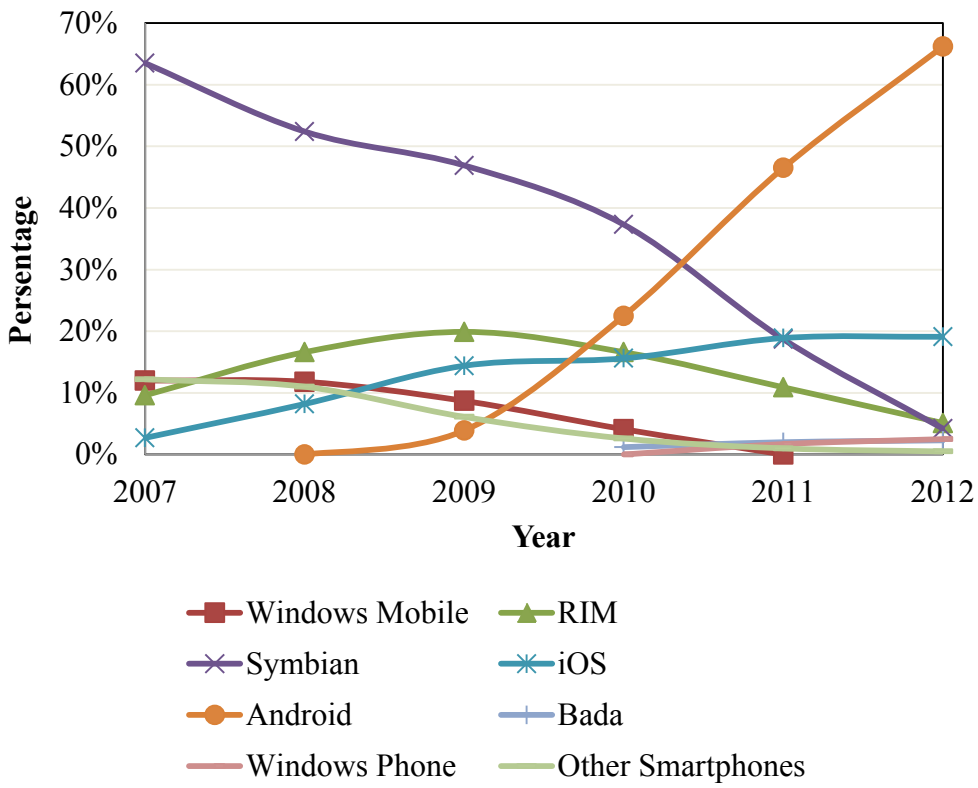


Figure 2.1 Mobile operating system market share from 2007 to 2012.

Modern mobile operating system combines the features of personal computer operating system with user interface, computation, personal entertainment, communication network, and sensing network. Earlier Mobile OSs, such as, Palm and Nokia S40 mainly focus on personal communication management. As mobile hardware rapidly evolved, multimedia entertainment becomes a rather important aspect for mobile devices. Photo capturing and video recording are two of the major multimedia applications.

Moreover, all recent mobile operating systems offer the capabilities for installing and running third-party software. Most manufacturers provide APIs support for various programming languages, such as C, C++, Java and Python. When elementary image

processing such as resize, rotation, compressing are added as a basic function in mobile OS's, third-party software applications are contributing to basic image editing, such as contrast adjustment, histogram equalization, de-noising, red-eye effect removal and so on.

Android System

In light of the share of Android system (the market share approaches to 70% worldwide at the end of 2012), the Android system currently is the leading mobile system. Android is widely supported by a considerable number of different mobile device manufactories, due to its open source design; Google Inc. encourages developers to implement applications by providing rich sets of APIs and tools [Google, 2013].

Android is a software architecture and platform for mobile devices that includes the basic operating system, middleware and key applications. The basic operating system is derived from Linux, and is designed primarily for touch-screen mobile devices such as smartphones and tablet. Earlier Android consists of a kernel based on Linux kernel version 2.6; from Android 4.0 Ice Cream Sandwich onwards, it is based on version 3.x. Other system components include middleware, libraries and APIs written in C, and other system application based on Apache Harmony.

Android is open-source and Google releases the code under the Apache License. It includes a set of C/C++ libraries used by various components of the Android system. It also includes a set of core libraries that provides most of the functionality using the Java programming language. Every Android application runs in its own process, with its own instance [Ehringer, 2010].

For application developers, Google Inc. has provided Android software development kit (SDK). The SDK includes a comprehensive set of development tools,

including a debugger, software libraries, a handset emulator, documentation, sample code, and tutorials. For user development, the common environment is Eclipse as an integrated development environment (IDE). This IDE has Android Development Tools (ADT) plugin, which makes developing applications simple and straightforward.

Most applications on Android platform are written in Java programming language, which renders the development cross platform and easy to port to different devices. However, it is known that image processing using Java language is slower when compared with C or C++ programming language. Thus, more efficient image and vision libraries are sought to enhance image processing performance in this project.

2.2 Mobile Imaging and Vision Libraries

Decades ago, image processing research was mainly focused on the pixel level, such as to acquire, enhance, restore or compress image [Picard, 1995]. As digital data have taken place of non-digital ones, those problems became low-level image processing problems for researchers. Nowadays, researchers are more focused on dealing with the content of images. For example, to implement a matching function, efficient extraction of key features of an image and computing similarity between these features and templates may be the key steps. However, these high-level image analysis problems are usually computationally demanding.

Modern imaging and vision libraries have been developed for solving these vision-level problems. These libraries are written in different languages for various application fields. Based on different operation platforms, there are various libraries available for researchers.

Common Image Processing libraries

There are a lot of computer vision and image processing libraries aiming at different applications. Most of them are written in C-language. VLFeat is a cross-platform open source collection of vision algorithms specially focusing on visual feature extraction and classification. Light-weight libraries may include OpenSURF, which is an open-source library focusing on feature extraction. CImg, a small C++ toolkit used to load/save images, access pixel values, display, transform, filter images and compute statistics of the image. Other language-based image and vision libraries exist, such as Java language-based libraries ImageJ and BoofCV.

Matlab is a high-level language and interactive environment for numerical computation, visualization, and programming [Mathworks, 2013]. Matlab has a powerful Image Processing Toolbox that provides a comprehensive set of standard algorithms and graphical tools for image processing, analysis, visualization, and algorithm development. With this toolbox, researchers can perform image enhancement, image de-blurring, feature detection, noise reduction, image segmentation, geometric transformations, image registration and so on. However, Matlab is an image processing toolbox other than a library, since all the processing procedure must be completed in this environment. To date, there is no primary or third-party porting effort for running native Matlab programs in a mobile device.

OpenCV (Open Source Computer Vision Library)

OpenCV is a library of programming functions mainly aimed at real-time image processing and computer vision (basic machine learning algorithms are implemented too). It has C++, C, Python and Java interfaces, and supports multiple platforms like Windows,

Linux, Mac OS, Android and iOS. It contains a rich set of optimized algorithms that can be used in many applications, spanning from factory product inspection, medical imaging, security, user interface, camera calibration, stereo vision, to robotics [Bradski and Kaehler, 2008].

OpenCV has been continuingly developed and maintained. Its alpha version was released in 1999, and the current OpenCV version is 2.4.3. One of the OpenCV's goals is to provide simple-to-use computer vision interfaces. In this way, researchers can build sophisticated applications easier and faster. Since its first release, OpenCV is widely used in many practical applications. Such efforts includes stitching satellite images, object tracking, security and intrusion detection systems, camera calibration, and unmanned aerial, ground, and underwater vehicles [Bradski and Kaehler, 2008]. Starting from 2011, version 2.3.0 supports Android platform with Java interface. Since then, OpenCV became a well-tailored vision library for mobile devices.

Figure 2.2 shows the OpenCV architecture design. Among the components shown in Figure 2.2, HighGUI is a component in OpenCV that allows users interact with the operating system, the file system, and hardware such as cameras and touch screens. It allows users to open windows, display images, read and write graphics-related files (both images and video), and to handle mouse, pointer, keyboard and touch events. HighGUI in OpenCV is designed to contain three parts: the hardware part, the file system part, and the GUI part. Hardware part mainly deals with the operation of cameras. It allows a simple way to query a camera and retrieve the latest image from the camera. The file system focuses on loading and saving images. HighGUI use an abstract method to load and save images and videos without concerning the particular device. This makes

regardless of the underlying OS, which can be Windows, Linux, iOS or Android. Hence, reading and saving images will only be one line of code. GUI allows researchers interactively operate with the OS like creating window and push the image into that window, or handle user activities such as mouse, keyboard and touch events on that window.

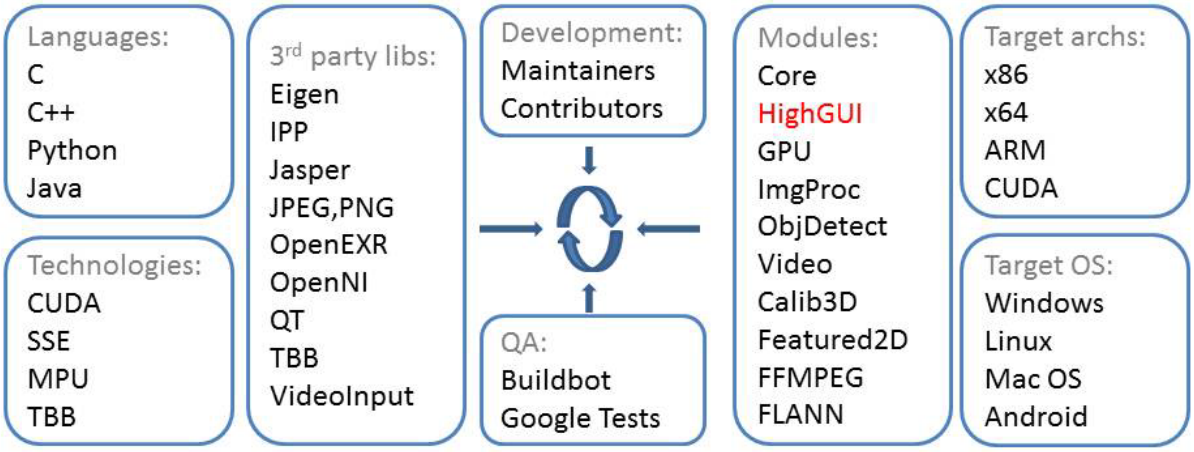


Figure 2.2 OpenCV architecture.

Besides the interactive component (HighGUI), OpenCV library provides over 500 functions that operate on images to accomplish specific tasks defined by users. These functions target at different aspects of image processing. Some are used to enhance, modify, transforms images; others focus on complex vision algorithm, such as matching, segmentation, motion tracking, and camera calibration.

2.3 Mobile Wireless Network

Data transmission plays an important role in mobile imaging. For example, when the computing resource power for mobile device limits the performance, a client-server based system is needed to push the image data to the server side and wait for the

feedback. Another instance would be multiple mobile devices that demand sharing the data to create a networked working environment. Thus wireless network is critical for mobile imaging if building diverse and efficient applications are desired.

Wireless networks have become increasingly popular since 1970s, and it has been adapted to enable mobility during the 1990s. Since the mobile cellular network started in 1980, it evolved rapidly from the first-generation system such as AMPs, TACs, and NMT, to the latest cellular technology such as Long Term Evolution (LTE) standard. For the meantime, the communication capability increases dramatically from voice service only to a combination of voice service and high-speed data service [Ojanpera and Prasad, 1998; Royer and Toh, 1999]. The LTE technology can have peak download rates up to 299.6 Mbit/s and upload rates up to 75.4 Mbit/s, which is even faster than some of the wired communication system. Such high-speed data rate and bandwidth will make mobile device more powerful on intensive imaging tasks wherein high-resolution images can be transferred easily.

Considering realistic applications and environments, however, mobile users may fall into situation with no cellular signal coverage or bad coverage. The solution for this would be wireless local area network (WLAN) based on IEEE 802.11 standards. Similar research has been done on combining cellular system with WLAN [Clark et al., 2002; Leung et al., 2002; Buddhikot et al., 2003; Shi et al., 2004]. There are also other standards than IEEE 802.11, such as Bluetooth and Zigbee, which help mobile devices to contribute local wireless networks but with different aims. These existing WLAN technologies and how they help with mobile imaging are reviewed in the following.

Wi-Fi Direct

Wi-Fi Direct is a standard that allows Wi-Fi devices to connect to each other without the need for a wireless access point. It allows Wi-Fi devices make a one-to-one connection, or a group of several devices connected simultaneously.

Wi-Fi Direct device is capable of a peer-to-peer connection, and can support either an infrastructure or a P2P connection. Wi-Fi Direct devices have the ability to join infrastructure networks as typical stations, and must support Wi-Fi Protected Setup enrollee functionality. Devices are connected by forming Groups (in a one-to-one or one-to-many topology) that function in a manner similar to an infrastructure BSS (Basic Service Set). A single Wi-Fi Direct device is in charge of the group, including controlling which devices are allowed to join and when the Group is started and terminated [Alliance, 2010]. This technology allows engineers setting up a field local Wireless LAN without additional network devices.

Bluetooth

Bluetooth is a standard for wireless communications based on a radio system designed for short-range communications devices. The goal is to replace the cabling of devices such as headset, printer, joystick, mouse, keyboard, and so on. Bluetooth can also be used for communications between mobile devices, act as bridges between other networks, or serve as nodes of ad hoc networks [Ferro and Potorti, 2005] .

The protocol defines a whole communication stack that allows devices to discover each other and advertise the services they offer. The key features of Bluetooth are robustness, low power, and low cost. It operates in the unlicensed industrial, scientific and medical (ISM) band at 2.4 to 2.485 GHz, using a spread spectrum, frequency

hopping, full-duplex signal at a nominal rate of 1600 hops/sec. Bluetooth predefines several types of connection, each with a different combination of available bandwidth, error protection, and quality of service. Once a connection is established, the devices can optionally authenticate each other and then communicate.

ZigBee

Zigbee is a short range, low power, and low rate wireless networking protocol. This standard operates at two bands, the 2.4 GHz band with a maximum rate of 250 kbps and the 868-928 MHz band with data rates between 20 and 40 kbps. Zigbee is based on Direct Sequence Spread Spectrum (DSSS) and it uses binary phase shift keying (BPSK) in the 868/928 MHz bands and offset quadrature phase shift keying (O-QPSK) at the 2.4 GHz band.

The Zigbee standards define two types of devices, a full-function device (FFD) and a reduced function device (RFD). The FFD can operate in three different modes, a personal area network (PAN) coordinator, a router or a device. The RFD is intended for very simple applications that does not require the transfer of large amount of data and needs minimal resources. [Shuaib et al., 2006] Depending on the application requirements, the network can either work in a star topology or a peer-to-peer topology. One network may contains one coordinator, multiple routers and end-devices The coordinator device is responsible for starting the ZigBee network. The coordinator scans the RF environment for existing networks; then it chooses a channel and a network identifier or PAN ID. Once the setting up of the network is done, the coordinator will behave like a router. Both the router and end device can be viewed as peers. The data

transfer mechanisms can be from coordinator to a device, from device to coordinator and between peer devices.

A comparison table between the three WLAN technologies is listed blow (Table 2.1). Among these three methods, Zigbee is better at wireless control and sensing network that requires long battery life; Bluetooth is aiming voice application due to its fast frequency hopping system technology; Wi-Fi is good for high bandwidth requirement communication connections.

Table 2.1 The key characteristics of Zigbee, Bluetooth and Wi-Fi [Sena, 2010]

	ZigBee	Bluetooth	Wi-Fi
Application Focus	Monitoring & Control, sensor networks, home automation	Wireless connectivity between devices such as phones, laptops, headsets.	Wireless LAN connectivity, broadband Internet access
Networking Topology	Ad-hoc, p2p, star, mesh	Ad-hoc, p2p	Point to hub, p2p
Operating Frequency	900-928 MHz, 2.4GHz	2.4 GHz	2.4 and 5 GHz
Longest Range	10 - 100 meters	10 - 100 meters	50 – 100 meters
Power Consumption	Very low	Medium	High
Complexity	Low	High	High

2.4 Related Work in Mobile Imaging

Mobile imaging has become an active research topic in recent years. Due to the limitation of computing capacity for most mobile devices, researchers have proposed a variety of solutions. One solution is to push the data to more powerful computing platforms, which is called a client-server module, or cloud-based mobile imaging system. The user captures images with mobile devices, and sends the data to the cloud or

standalone servers, which will carry out all the computing and processing work. Then the servers will push the results and data to the mobile devices via available networks. In this effort [Jing et al., 1999], the authors provided a framework and categorization of the various ways of supporting mobile client-server computing for information access. Another simple image sharing system also used the client-server concept to make the best approach with mobile image processing [Sarvas et al., 2004]. A very recent research work on object recognition indicated that this client-server based approach is quite useful when large amount of data is processed by the server side [Gammeter et al., 2010] .

There are also various commercial products providing similar methods to mobile imaging. For example, in 2009, Nokia launched their "Point & Find" software [Kobie, 2009], which is the first system that does not restrict the use to a certain type of objects. When mobile device capture the image, the image is sent to a server database and search for a matching object. Then additional content and information of the object is sent back to the mobile device. The major advantage is that mobile devices do not need to handle heavy computing tasks locally beyond their capacity. The down side would be wireless connection that is needed but may be not possible in all environments.

When there is no wireless connection, or the mobile device has strong computing capacity which does not require server assistance, or the task is time sensitive, in-situ mobile image processing becomes the best solution. Hull et al. argues that if mobile imaging application runs on a client-server architecture will not be as desirable if there is delay due to the data connection is significant [Hull et al., 2010]. The authors proposed a method for mobile recognition of paper documents and an application to newspapers that let readers retrieve electronic data linked to articles, photos, and advertisements only

using mobile devices. They achieved a run time of 1.2 seconds per image with a collection of 140 newspaper pages on an HTC-8282 Windows Mobile phone. Another interesting research approach on mobile imaging is recognizing 2-dimensional bar code or QR (quick response) code. Much research has been done on this topic [Ohbuchi et al., 2004; Sun et al., 2007; Liu et al., 2008]. In these efforts, mobile devices capture the images of barcodes/QR-codes; then processing is conducted immediately leading to a URL. This URL further leads to a website accessed via the phone's browser. Other works have been done on using mobile phones to do real-time detection and tracking and focused on 3-D augmented reality [Feiner et al., 1997; Wagner et al., 2010].

2.5 Chapter Summary

This chapter discusses the background of mobile operating systems and introduces the most suitable mobile system for developers: Android. Also an insight of mobile image processing and vision libraries and mobile wireless network that will augment mobile imaging applications are presented. At last the chapter reviews the related research work of mobile imaging processing.

As shown in this chapter, a wide range of mobile image researches including cloud based and stand-alone based have been done with different strategies and approaches. However, to the author's knowledge, there is less research on building framework for engineering applications with interactive real-time mobile imaging.

CHAPTER 3

BASIC IMAGING PROCESSING METHODS FOR MOBILE APPLICATIONS

In this chapter, the general image processing methods that are essential for the development in the following chapters will be discussed. These methods include camera calibration, edge detection, image stitching and template matching, which are keys to build many real-time mobile imaging solutions.

3.1 Camera Calibration

Camera calibration in the context of computer vision is the process of determining the internal camera geometric and optical characteristics (intrinsic parameters) and/or the 3D position and orientation of the camera frame relative to a certain world coordinate system (extrinsic parameters). It is a very important step in order to extract metric information from 2D images, especially in applications that aim at solving quantitative measurements such as depth from the stereoscopy and motion from images [Tsai, 1987; Weng et al., 1992].

Over the past decades, much work has been done for camera calibration, Zhang [Zhang, 2000] classified those techniques roughly into two categories: photogrammetric calibration and self-calibration. The photogrammetric calibration is also considered as three-dimensional reference object-based calibration. Camera calibration is performed by observing a calibration object whose geometry in 3D space is known with very good precision. Self-calibration is performed by calculating the correspondences by moving a camera in a static scene, and then both the internal and external parameters are recovered through taking multiple images. This process can help reconstruct 3D structures.

However, there are many parameters to estimate by using self-calibration which may lead to obtaining unreliable results. Thus, in this thesis, photogrammetric calibration will be used.

Calibration and remapping (remapping is a process of taking pixels from one place in the image and locating them in another position in a new image) can correct the length distortion, and determinate the relation between the camera's natural units (pixels) and the real world units. Projective transformation maps the points in the physical world with coordinates (X, Y, Z) to the points on the projection plan with coordinates (x, y), with this simple form [Heikkila and Silven, 1997]:

$$\begin{bmatrix} x \\ y \\ w \end{bmatrix} = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} \tag{3.1}$$

Herein, w is added to (x, y), the homogeneous coordinates associated with point in the projective space. Then $w = Z$ can be calculated. Parameters f_x and f_y are camera focal lengths, and c_x and c_x are the optical centers expressed in pixels coordinates. If for both axes, a common focal length is used with a given α aspect ratio (usually 1), then $f_y = f_x \times \alpha$ and in the upper formula, there is a single f focal length. The matrix containing these four parameters is referred to as the camera matrix.

Camera distortion is a non-linear optical lens-induced modification to images. Camera distortion is mainly caused by radial and tangential factors. The presence of radial distortion manifests in form of the "barrel" or "fish-eye" effect. In practice, radial distortions is small and can be characterized by the first few terms of a Taylor series expansion around $r = 0$. Thus the following formula will be used:

$$x_{corrected} = x\left(1 + k_1 r^2 + k_2 r^4 + k_3 r^6\right) \tag{3.2}$$

$$y_{corrected} = y\left(1 + k_1 r^2 + k_2 r^4 + k_3 r^6\right) \tag{3.3}$$

where (x, y) is the original location (on the imager) of the distorted point and $(x_{corrected}, y_{corrected})$ is the new location as a result of the correction.

Tangential distortion is due to manufacturing defects resulting from the lens not being exactly parallel to the imaging plane. Tangential distortion is minimally characterized by two additional parameters, p_1 and p_2 as following:

$$x_{corrected} = x + \left(2p_1 y + p_2(r^2 + 2x^2)\right) \tag{3.4}$$

$$y_{corrected} = y + \left(p_1(r^2 + 2y^2) + 2p_2 x\right) \tag{3.5}$$

Thus in total there are five distortion parameters, which are bundled into one distortion vector:

$$Distortion_{parameters} = (k_1, k_2, p_1, p_2, k_3) \tag{3.6}$$

In this thesis, OpenCV is used to conduct the necessary calibration and to compute these intrinsic parameters in Equation. 3.6. The calibration procedure aims at the camera on a known structure (such as chessboard) that contains many individual and identifiable points. By taking multiple views from a variety of angles, then it is possible to compute each image's location and orientation of the camera. In addition, the intrinsic parameters of the camera can be computed.

Chessboard is used as the known structure in this thesis. About fifteen images are taken from multiple angles, each pattern equals in a new equation [Bradski and Kaehler, 2008]. Using these images as the input of calibration function, an output of camera intrinsic matrix, distortion coefficients, rotation vectors and translation vectors will be

calculated. The correction of the distorted images with the parameters is needed. The undistortion process is to compute a distortion map with the camera intrinsic matrix and the distortion coefficients, and apply this map to original images. The steps of calibration can be listed as follows:

1) Program will load calibration parameters such as where to get the input image;
2) Get input images of the chessboard either from real-time camera or from stored images;
3) Find patterns from the input images and position of the corner of each pattern on the chessboard
4) Calibration will be applied with parameters such as the set of corner points and size of the input image, and calculate the distortion coefficient matrix and camera matrix.
5) With the result from last step, undistortion is conducted for input images.
6) Save the calibration result and show the undistorted images

Table 3.1 shows how to implement these steps in pseudo codes using functions from OpenCV. Since the calibration process for a certain camera only need to do once, the calibrationData saved in the code can be applied to any image processing that based on this camera. Thus, one mobile device only need go through this process only once, and save calibrationData for future use. All the images that will be processed later in this thesis are taken from calibrated cameras.

Table 3.1 Camera calibration Pseudocode using functions in OpenCV

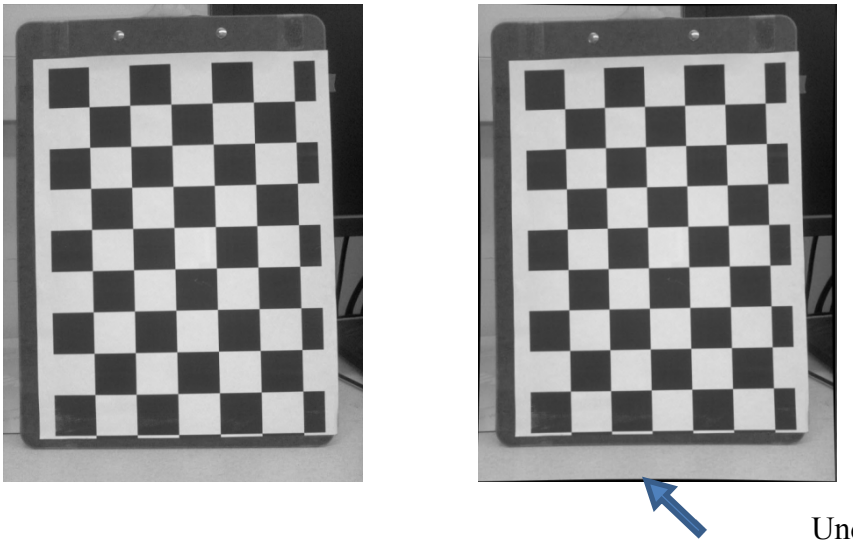
```
function camera_calibration
{
     Read settings file;
          if failed
                print "Invalid setting";
                return false;
          else
                return true;

     for (i = 0; i++; i <= end of the input image list)
         {
          ImageSet [i] = Read input image [i];
          CC [i] = run findChessboardCorners();
          if (i = end || false)
               break;
         }

     Camera calibration:
     NewImageSet = undistortion(ImageSet, calibrationData);
     Show NewImageSet;
}
```

Figure 3.1 shows the chessboard-based calibration results. Note that depending on the camera hardware, the calibration result might or might not be obvious. Here is slight difference between original and undistorted image, which means that this camera has high quality with little distortion. Heavy distortion may occur on low end and chip cameras.

(a)



(b) **(c)**

Figure 3.1 Calibration chessboard (a) a set of original chessboard images taken from different angle and distance; (b) original image; (c) a calibrated image, note comparing with (b), its edge is undistorted (where the arrow pointing at).

3.2 Edge Detection

Edges are salient image features that are useful for image analysis and classification in a wide range of applications. The study of edge detection has been for a long time and many algorithms exist [Davis, 1975; Peli and Malah, 1982]. Edge detection is an important step for many other high-level image processing and complex image understanding problems. Among many edge detection methods, one of the most widely used algorithms is the Canny edge detector [Canny, 1986].

Canny edge detector was developed by J. F. Canny in 1986, also known as the optimal detector. The original algorithm has the following advantages:

1) Low error rate: meaning a good detection of only existent edges.
2) Good localization: the distance between edge pixels detected and real edge pixels have to be minimized.
3) Minimal response: only one detector response per edge.

OpenCV uses Canny edge detector to find edges in the image. To implement the algorithm, the first step is to smooth image noise using a Gaussian filter. The Gaussian filter is defined by the kernel size. One adverse effect is that this filter will slightly blur the original image. For example, Gaussian kernel of $size = 5$ is often used as showing in Equation 3.7.

$$K = \frac{1}{159}\begin{bmatrix} 2 & 4 & 5 & 4 & 2 \\ 4 & 9 & 12 & 9 & 4 \\ 5 & 12 & 15 & 12 & 5 \\ 4 & 9 & 12 & 9 & 4 \\ 2 & 4 & 5 & 4 & 2 \end{bmatrix} \tag{3.7}$$

Canny algorithm uses four filters to detect horizontal, vertical and diagonal edges in the smoothed image. It applies a pair of convolution masks in both x and y directions. Then edge detection operator returns a value for the first derivative in the directions. The two masks in x and y are G_x and G_y, respectively,

$$G_x = \begin{bmatrix} -1 & 0 & +1 \\ -2 & 0 & +2 \\ -1 & 0 & +1 \end{bmatrix} \tag{3.8}$$

$$G_y = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ +1 & +2 & +1 \end{bmatrix} \tag{3.9}$$

The gradient strength and direction can be computed by

$$G = \sqrt{G_x^2 + G_y^2} \tag{3.10}$$

$$\theta = \tan^{-1} \frac{G_x}{G_y} \tag{3.11}$$

The edge direction angle is rounded to one of four angles representing vertical, horizontal and the two diagonals in OpenCV.

The next step is called non-maximum suppression. It searches the image with the estimation of the image gradients to determine if the gradient magnitude assumes a local maximum in the gradient direction. This removes pixels that are not considered to be part of an edge and only thin lines will remain.

Canny algorithm tries to assemble the individual edge candidate pixels into connected contours. These contours are formed by applying a hysteresis threshold to the pixels. There are two thresholds in Canny, upper and lower. If a pixel has a gradient larger than the upper threshold, then it is accepted as an edge pixel; if a pixel is below the

lower threshold, it is rejected. If the pixel's gradient is between the thresholds, then it will be accepted only if it is connected to a pixel that is above the high threshold. Canny recommended a ratio of high:low threshold between 2:1 and 3:1.

In OpenCV library, all the image process algorithms are packaged under Imgproc.class. This class contains a variety of image processing methods, including image filtering, geometric image transformations, miscellaneous image transformations, histograms, structural analysis and shape descriptors and feature detection. Canny edge detection belongs to image filtering. It is packed into one function called Canny with five parameters, including input image, output image, two thresholds and aperture size for the directional operators (Equation 3.8/3.9).

When implementing this edge detector into Android application, some of the steps mentioned in previous description are modified. The first step is converting input image into gray grayscale, which will speed up the computing. Then Gaussian filter will be applied to the image since controlling of the parameter sigma in Gaussian filter is needed to get rid of some noisy edges. With the necessary parameters in the original Canny function, the edge detection result is controllable by changing these parameters.

The pseudo code for implementing basic Canny edge detection shown in Table 3.2. It is noticed that if the thresholds and sigma in the code are set as dynamic parameters, a real-time interaction-based imaging and edge detection can be implemented. Since the mobile device is able to print the resulting image on the screen, the user of mobile device can change these parameters through a sliding bar in real-time.

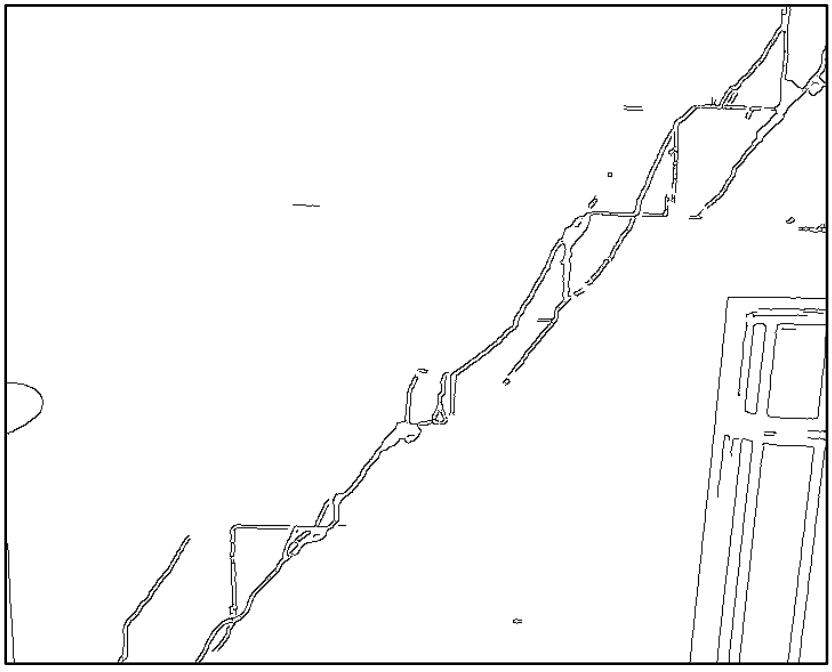
Table 3.2 Edge detection Pseudocode using functions in OpenCV

```
function Edge_detection
{
        if (switch = static)
           inputImage = Read image from file;
        else
           inputImage = Read frames from camera in real
time;

        Color converting;
        BlurredImage = Run GaussianBlur (inputImage, Sigma);
        EdgeImage = Run Canny(BlurredImage,Parameters);
        Show EdgeImage;
}
```

A wall crack image and the result of edge detection image are shown in Figure 3.2. The parameters have been tuned to get the best visual result, this means only the crack of user's interest are remained in the image, all other noisy information is filter out.

(a)



(b)

Figure 3.2 Canny edge detection: (a). original wall crack image (b). Canny edge detection is applied to the image, with optimal parameter value, only crack edge is shown

3.3 Image Stitching

Stitching is a fundamental task in image processing used to match two or more pictures taken at different times, by cameras or from different viewpoints. OpenCV has implemented an automatic stitching pipeline. The pipeline can divide into image registration and compositing.

The registration entails a feature detection step. There are multiple feature detection algorithms in OpenCV, such as Orb feature detection and SIFT feature detection. Once features have been extracted from all n images, they can then be matched. Each feature is matched to its k nearest neighbors in the feature space using a k-d tree to find approximate nearest neighbors. A k-d tree is an axis aligned binary space partition, which recursively partitions the feature space at the mean in the dimension with highest variance [Lowe, 2004; Brown and Lowe, 2007]. Then it takes features of all images and conduct pairwise matches between all images and estimates rotations of all cameras. To find out all the matching images, OpenCV uses Random Sample Consensus (RANSAC) to select a set of inliers that are compatible with a homograph between the images and applies a probabilistic model to verify the match. Once pairwise matches have been established between images, the sequence of the matching images can be found. Finally, with wave correction and final panorama estimation, the registered image set is obtained.

In the compositing part, OpenCV uses the registration data to straighten the images. It assumes that camera's horizontal axis lays in a plane, so the plane that contains the camera center and the horizon can be computed. Applying a global rotation removes the wavy effect from output.

For a better vision result of the stitching, OpenCV also corrects the compensation exposure error. An error function is defined over all images. The function is the sum of compensation gain normalized intensity errors for all the overlapping pixels.

The last step is blending the images. There are two types of blendings in OpenCV, Feather Blender and Multi-band Blender [Burt and Adelson, 1983]. Feather Blender is a linear blending method; it may cause mis-registration errors due to mis-modeling of the camera, radial distortion and so on. Multiband Blender blends low frequencies over a large spatial range and high frequencies over a short range. This algorithm can keep the smooth transitions between images, and preserving high frequency details.

In this thesis, considering the specialty of mobile imaging, a simplified pipeline is implemented in a work flow as shown in Figure 3.3. By implementing this pipeline, a test of a set of images showing long crack damages on structure is applied. This scenario is quite often when field engineers want to record the damages. They tend to take many images on parts of the crack and then hope to stitch them together. Figure 3.4 shows the result.

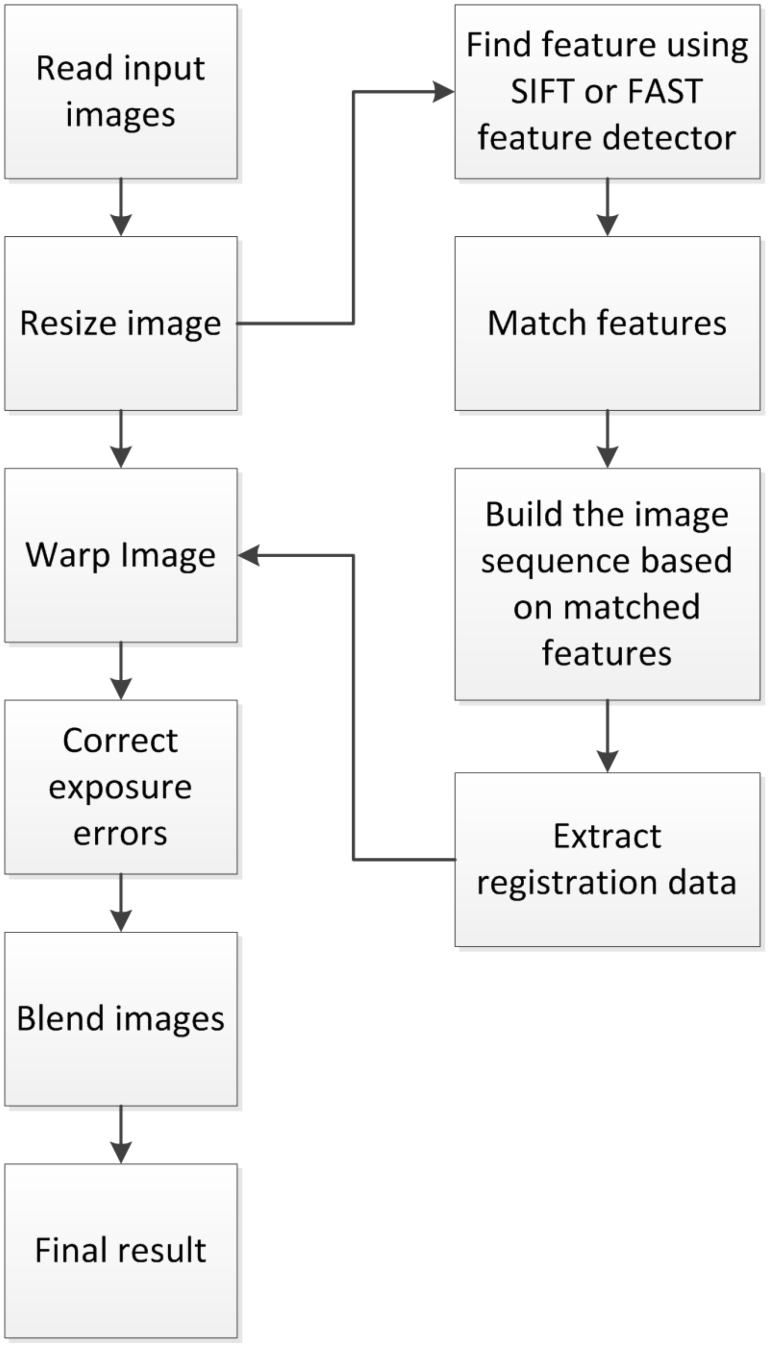
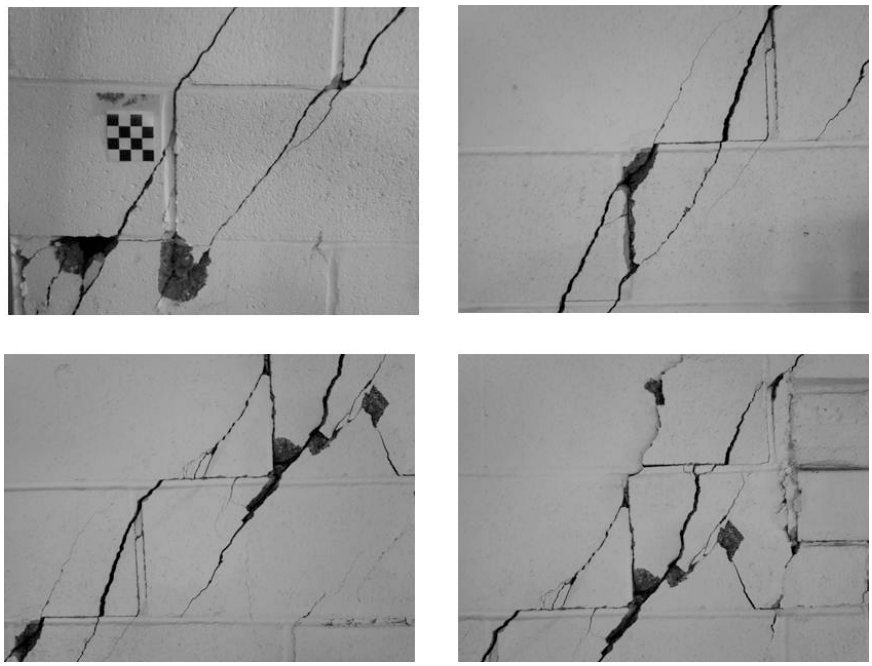
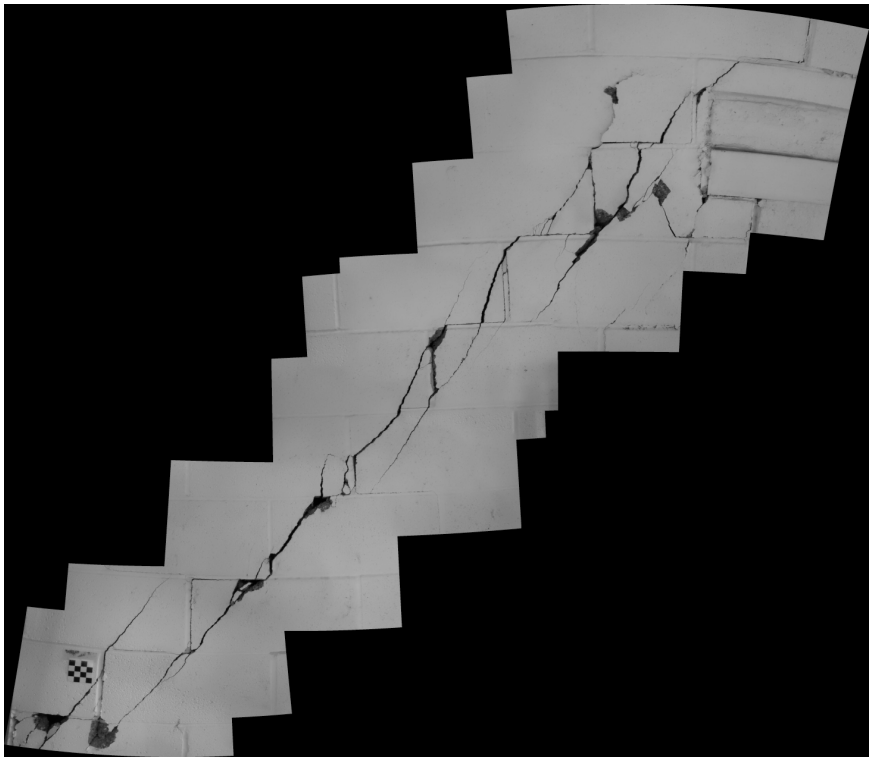


Figure 3.3. Image stitching pipeline on Android platform



(a)



(b)

Figure 3.4 Stitching illustrator: (a) local images of the crack (b) stitched image from 10 images showing a panoramic view without any detail losses.

3.4 Template Matching

Template matching is a classical approach to the problem of locating an object in an image. The image will be searched for an object by applying a template at each location in the image. The template usually contains a standard object of interest and is usually smaller than the original image. The decision regarding the existence of the object at the given location is made by comparing this template with a predetermined threshold.

The typical template matching process involves cross-correlating the template with the scene image and computing a measure of similarity between them to determine the distance. Since the evaluation of the correlation is computationally expensive, there has been a need for low-cost correlation algorithms for real-time processing. A large number of correlation-type algorithms have been proposed [Vanderbrug and Rosenfeld, 1977; Secilla et al., 1988; Choi and Kim, 2002]. One of the methods is to use an image pyramid for both the template and the scene image, and to perform the registration by a top-down search. Other fast matching techniques use two-pass algorithms; use a sub-template at a coarsely spaced grid in the first pass, and search for a better match in the neighborhood of the previously found positions in the second pass [Prokop and Reeves, 1992].

In OpenCV, a top-down search algorithm in space domain, which is also known as pixel level, is used to complete the matching. The image in which a match to the template is expected to find is called source image I, while the patch image that will be compared to the source image is called the template image T. To identify the matching area, it is necessary to compare the template image against the source image by sliding it. That is moving the template one pixel at a time (left to right, up to down). At each

location, a distance metric is calculated so it represents how similar the template is to that particular area of the source image. And for every location the template image T over source image I, it will store the metric in a result matrix R. Each location in R contains the match metric, the highest the value in R means the best possible match location.

There are six template matching methods implemented in OpenCV. The corresponding relations in terms of R and T, I are described as follows:

- Square difference matching methods: CV_TM_SQDIFF

  This method matches the squared difference, so a perfect match will be 0 and bad matches will be a large value.

$$R(x,y) = \sum_{x',y'} (T(x',y') - I(x+x',y+y'))^2 \tag{3.12}$$

- Correlation matching methods: CV_TM_CCORR

  This method multiplicatively matches the template against the image. A perfect match will be large and bad matches will be small or 0.

$$R(x,y) = \sum_{x',y'} (T(x',y') \times I(x+x',y+y')) \tag{3.13}$$

- Correlation coefficient matching methods: CV_TM_CCOEFF

  This method matches a template relative to its mean against the image relative to its mean. Therefore a perfect match will be 1 and a perfect mismatch will be –1; a value of 0 simply means that there is no correlation (random alignments).

$$R(x,y) = \sum_{x',y'} (T'(x',y') \times I'(x+x',y+y'))$$

$$T'(x',y') = T(x',y') - 1/(w \times h) \times \sum_{x'',y''} T(x'',y'') \tag{3.14}$$

$$I'(x+x',y+y') = I(x+x',y+y') - 1/(w \times h) \times \sum_{x'',y''} I(x+x'',y+y'')$$

For the three methods described above, there are also normalized versions [Rodgers and Nicewander, 1988]. The normalized methods can help reduce the effects of lighting differences between the template and the image. In each case, the normalization coefficient is the same. Thus, these normalized methods are shown below:

$$R(x,y) = \frac{\sum_{x',y'} (T(x',y') - I(x+x',y+y'))^2}{\sqrt{\sum_{x',y'} T(x',y')^2 \times \sum_{x',y'} I(x+x',y+y')^2}} \tag{3.15}$$

$$R(x,y) = \frac{\sum_{x',y'} (T(x',y') \times I(x+x',y+y'))}{\sqrt{\sum_{x',y'} T(x',y')^2 \times \sum_{x',y'} I(x+x',y+y')^2}} \tag{3.16}$$

$$R(x,y) = \frac{\sum_{x',y'} (T'(x',y') \times I'(x+x',y+y'))}{\sqrt{\sum_{x',y'} T'(x',y')^2 \times \sum_{x',y'} I'(x+x',y+y')^2}} \tag{3.17}$$

Tradeoff usually exists between accuracy and computation speed for the above methods. In general, square difference matching is simpler but less accurate while correlation coefficient matching is most sophisticated and costs more computing power.

To implement template matching, a pre-store template image in disk or memory is needed, and read camera frame as the source image. Then slide the template through the source image. After the program calculates the matching result, minMaxLoc function will be used to locate the highest or lowest (depending on matching methods) value in the R matrix, which is the best match. Furthermore, an area of good match around that best

matching point must be ensured in order to avoid random template alignments that just happen to work well. Since in real matching problems, slight misalignments of the template should not cause the result vary too much, which means that a good matching point contains good matching points nearby. Therefore, the programs will slightly smooth the result image before using minMaxLoc to seek the maximum or minimum value. Finally, a rectangle is rendered on the source image to show the result to users of the application. The following pseudo code describes this procedure.

Table 3.3 Template Matching Pseudocode using functions in OpenCV

```
function Template Matching
{
    templateImage = Read image from file;
    inputImage = Read frames from camera in real time;
method = Choose Matching Methods;
    Result = Run matchTemplate(templateImage,
inputImage,method);
Smooth Result;
Find matching point in Result;
Plot Matching point position;
}
```

3.5 Chapter Summary

This Chapter discussed three basic image processing methods in OpenCV: camera calibration, Canny edge detection, image stitching and template matching. First the mathematic theories of these image processing algorithms are introduced, and the implementation of these algorithms on mobile devices with OpenCV pseudo codes are presented. Finally some examples of using real images are shown to prove that the codes

are working on Android platforms. The following chapter will discuss a framework based these mobile image processing methods that help people deal with real-world problems.

COLLABORATIVE AND INTERACTIVE MOBILE IMAGING

A collaborative and interactive mobile imaging framework is introduced in this chapter, which helps engineers to solve real engineering problems. First, the motivation for constructing such a framework and how it helps engineers is discussed. Then the framework is introduced and implemented using smart mobile devices. Also the problem of sharing large amount of data is considered in this framework.

4.1 Motivation

Built infrastructure supports modern civilization. However, civil infrastructure is threatened by natural hazards, technological accidents and terrorist attacks. The impact of these extreme events on civil infrastructure often leads to tremendous life and property losses in a short period of time. Consequently, rapid post-disaster response and routine inspection technologies concern stakeholders of all sectors. In the engineering community, learning from disasters has been a tradition [GREENE et al., 2004]. It is a typical practice that field engineers are usually sent to the disaster afflicted communities to perform rapid damage assessment of engineered structures, lifeline systems and geological structures, wherein imaging of damaged objects is a ubiquitous activity. As seen in many disaster reports written by civil engineers, photo pictures among other sensed data usually comprise the major contents of the resulting engineering reports.

During these field activities, besides imaging devices, GPS, and other sensing, computing and networking devices are being pervasively used in field, which enable rapid collection of civil infrastructure data. However, this practice has adversely led to two technical challenges that pose to field engineers. The first is the dilemma of massive

data collection capability and insufficient visual computing and knowledge discovery utilities specially developed for civil infrastructure engineers. The second is how to efficiently collect and share data for a spatially large area/object, and discover engineering knowledge collaboratively considering a multi-person engineer team in the field. As a consequence, field engineers usually take a few days for conducting ground investigation then take weeks to organize and prepare technical reports. For serious quantification and verification of collected field data, the process usually takes much longer.

Realizing this, a framework based on the human-centered concept of mobile sensing and computing is proposed. This concept attempts to realize a collaborative image sensing and computing network with the popular Android mobile devices. A scenario where multiple field engineers working collaboratively but distributed in a geographical area is considered. One engineer forms a unit of sensing and computing. It assumes that all units are equipped with Android mobile devices. Nonetheless, heterogeneous sensing devices besides Android devices are present between different units. Typical equipment includes digital cameras, accelerometers, and other common structural and geotechnical sensing devices and loggers. Network connections within and between the units are assumed to be generic wireless networks. Currently available wireless network options include Bluetooth, direct Wi-Fi, and other P2P protocols.

In this thesis, the configuration and capability of this proposed collaborative mobile sensing and computing network for civil infrastructure condition assessment is presented. Also, the challenges in designing such a system are demonstrated. With actual imaging examples, the proposed framework will enable seamless integration of sensing,

imaging, real-time processing and knowledge discovery in future engineers-centered field reconnaissance and civil infrastructure.

4.2 Collaborative Mobile Imaging Framework and Design

In this framework, every engineer with his or her equipment in the work zone is considered a Field Engineer Unit (FEU), as shown in Figure 4.1. Each unit has two basic elements, which are the human (the engineer) and the devices. A unit can do video capturing, imaging, GPS locating, and other structural or geological data sensing and processing. Its topology is shown Figure 4.1 as well.

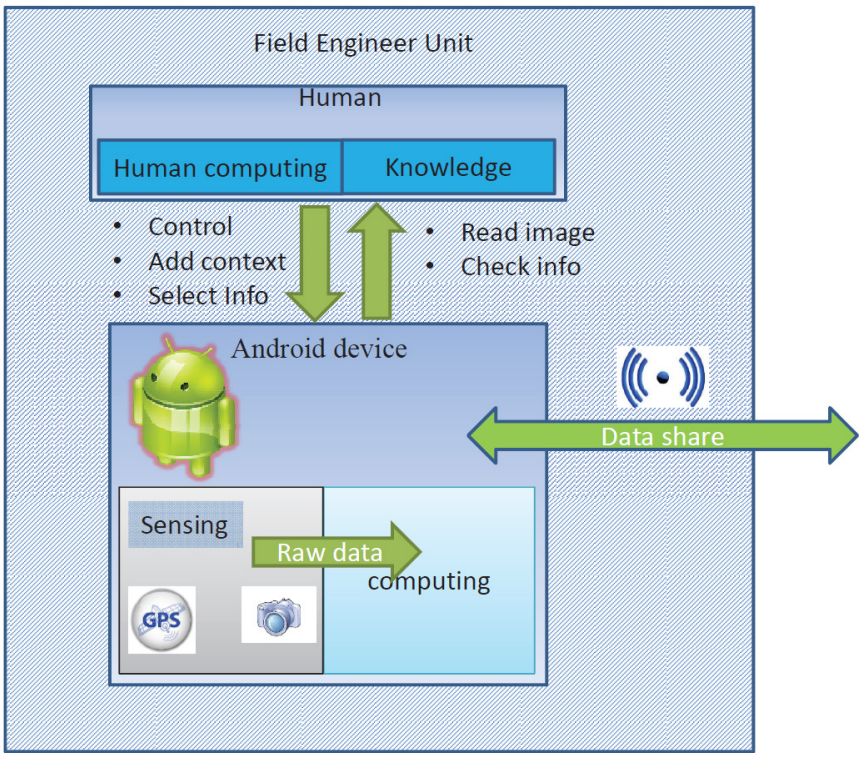


Figure 4.1 The structure of Field Engineer Unit

With the configuration of this basic unit, a collaborative mobile sensing and computing network is formed. The topology of this collaborative framework is shown in

Figure 4.2. The term of 'collaborative' first comes from the basis that different FEUs are connected via available generic P2P networks; hence, basic meta data, complex meta data, large data sets and knowledge can be shared within the FEUs. For example, a unit who is equipped with a professional digital camera can do detailed imaging; then the high-resolution imagery data can be immediately sent through Wi-Fi Direct to another unit who has powerful computing capability (Figure 4.3).

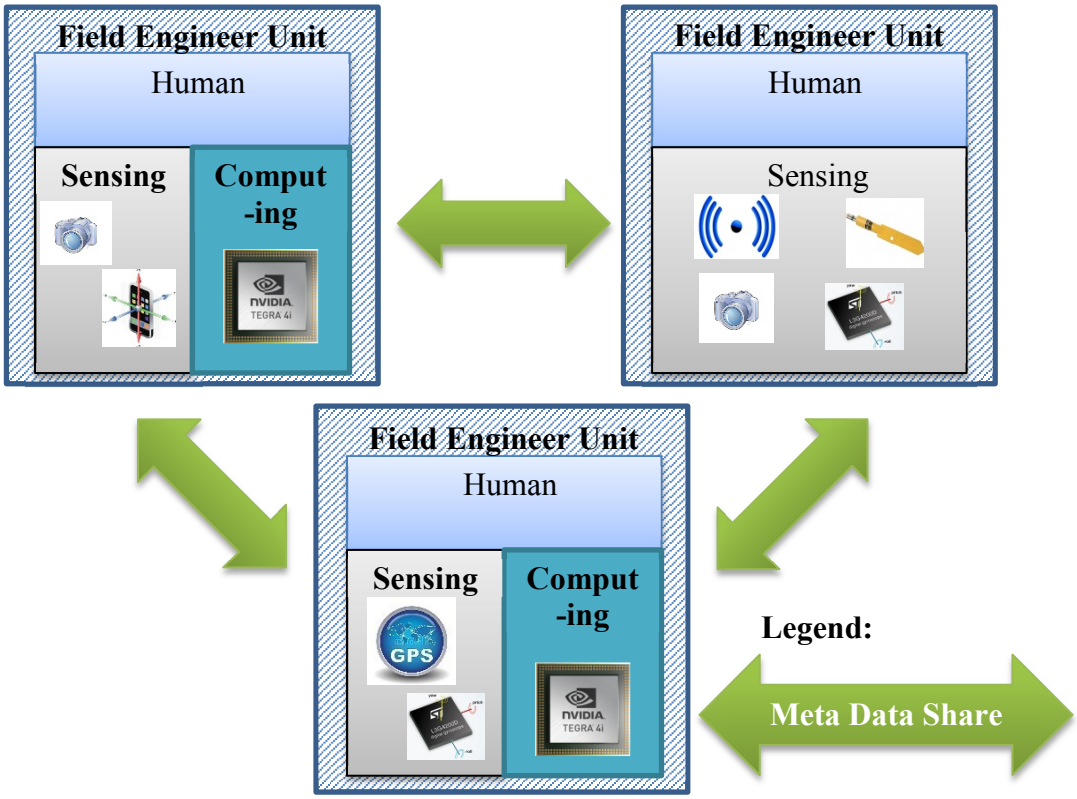


Figure 4.2 Collaborative mobile sensing framework

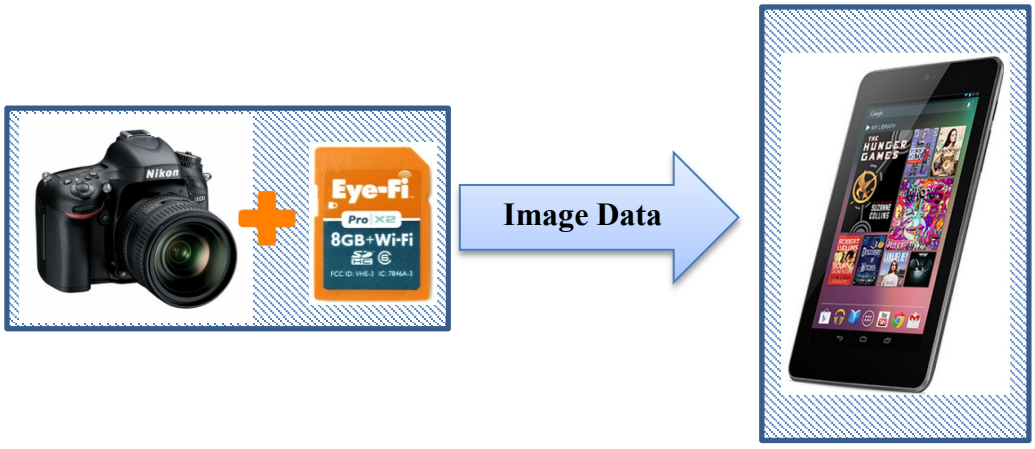


Figure 4.3 Wi-Fi Direct that connects digital camera and smartphone with the help of Eye-Fi® card [EyeFi, 2013].

When multiple units focus on different tasks or the same task, this collaborative network between units can make the procedure of data processing more efficient. It is expected that this sort of transmission should occur rarely due to possible limits on wireless network bandwidth. Nonetheless, it prescribes that basic meta data of each FEU, including current location of FEUs, current tasking, current sensing/computing load, will be pushed to the remaining FEUs constantly. Cellular connection is an option in this proposed framework, due to a possible consequence of disasters wherein relay communication towers may be damaged or the communication traffic is too congested. If cellular connection is available, it provides an external connection of the proposed collaborative network to a virtually centralized cloud server.

Complex Meta Data Sharing

Contextual information along with raw or processed data engineering is considered as Complex Meta Data (CMD), as shown in Figure 4.4. In this work, engineering CMD may contain different data type including raw imagery data, processed data and human-added contextual description. Different FEUs can share the metadata in

the field. These metadata sets, once visualized in a dynamic Geographical Information System portal, can be simultaneously shared with other engineers worldwide who need such information.

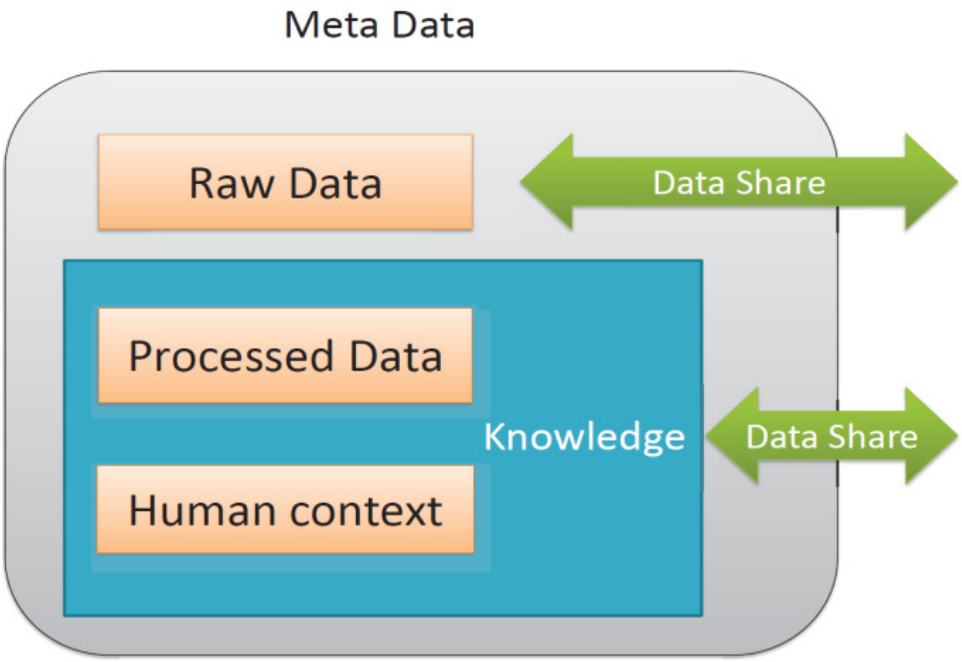


Figure 4.4 An illustration of the Complex Meta Data and Sharing

On the other hand, if the field engineer need information stored in the cloud server, they can download the metadata together with local information they obtained to get a best result they need. For example, it is possible that the engineers need satellite images in an earthquake disaster scene, which is too big for a mobile device to process. They can choose part of the image that has location and date information as a metadata set to download to their field device. By combining with the local sensors and computing units, engineer can conduct 3D-reconstruction for the built objects of interest.

Advanced Collaborative Knowledge Discovery

The term of 'collaborative' also indicates the potential interaction between the human and the devices within a FEU. It is still difficult to use artificial intelligence on modern computing units to completely replace human. In our design, a collaborative relationship between the engineer and his/her sensing devices is a good solution. The Field Engineer Unit shown in Figure 4.1 has two elements. The first element is human, who can practice human computing and is equipped with scientific and engineering knowledge. The other element is computing unit that has attached sensors. When the sensors obtain field image data, it transmits the raw data to the computing unit of the device. However, large amount of raw data will reduce the meaningful engineering information for further use. Thus, before the device decide to store or upload the raw data, the engineer should add professional information to turn the raw data into knowledge.

In detail, the engineer can check and review the raw data captured from the sensors, then modify or process the data. For example, if the data is a crack image taken by the camera, engineer first should call the camera calibration program to receive the correct image parameters. Then engineer will determine the crack parameters with his professional knowledge which can be called human computing, and adds this information to the raw data. Meanwhile the engineer can give further instruction to the device to help the device's data processing. In this process, the modern human-machine interface of any mobile devices can be fully exploited, including touch screen and motion-based gesture inputs.

For example, in a crack detection practice, the engineer can lively control the color window radius and spatial window radius of mean-shift filtering and the sigma

value of Gaussian filter. As a result, adding human computing in this process will make the smart device smarter. Gesture-based control that selects where the crack is located in the image domain will make the device tackle only certain part of the image; therefore, the computationally hard segmentation issue is partially avoided.

4.3 Implementation

General Features

Android-based development is generally applied in the proposed software for acquiring, processing and adding human knowledge. The procedure made the unit with android based device become an information center which combines human knowledge, raw data and processing unit. Further, if the information center can connect to Internet through local P2P network and cellular network, all engineer units in that field will be able to share information and access the data. More complex tasks can be done via such collaborative mini-scale crowd intelligence. For example, very large-scale satellite images can only be saved on server side. The FEUs in the network can individually download different small portions of the large satellite image, combine them with ground photos. Subsequently, the FEUs can collectively build a detailed building information model with 3D-reconstruction techniques augmented by the sensed field damage data.

The application is aiming at realizing the collaborative mobile computing system. Digital imaging can be done by a calibrated digital camera or by the Android device itself. In the former case, the digital camera sends the captured images (within a FEU) to the FEU's Android computing device. The Eye-Fi method in Figure 4.3 is used in this effort.

The application workflow built on the android platform is shown in Figure 4.5. In this figure, users will see the device's camera real-time view on the screen. When users

decide to begin run some algorithms, the program will call the corresponding function. All images are considered as matrixes during the processing procedure. For example, when the user wants to do a mean shift filter to the video stream, the program will save the stream frame in the stack as a matrix, and apply the mean shift filter function to the matrix in the stack. Then the program will read the recalculated matrix and convert it to bitmap and print the frame back on the screen. Users can modify the parameter of the function until the process get the best result for certain situation.

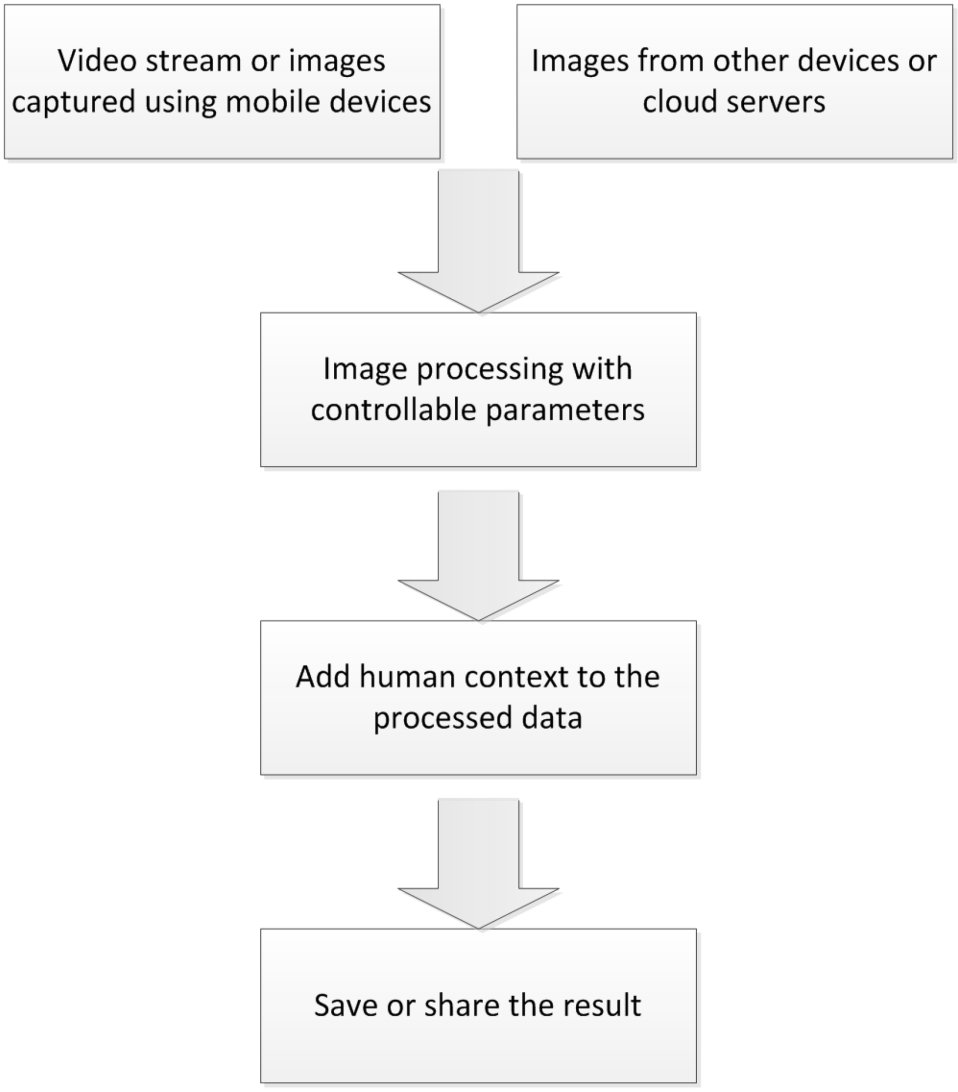


Figure 4.5 The Workflow for implementing imaging processing application within one Android device.

Interface Design

The application's user interfaces is shown in Figure 4.6. The left side is a window showing real time image from the camera. The right side of the window is an interactive panel where users can interact with the application. For example, the civil infrastructure condition knowledge information for users to choose includes structure type, material type and so on. When engineers take a picture with mobile device, she or he can add the information to the picture and becomes a metadata set, which not only contains raw image information but with human knowledge. Figure 4.6 shows an asphalt pavement crack image, and engineers can save this information just as the right side parts show the options.

Figure 4.6 Application user interface, with camera sense on the left, user input on the right.

Collaborative Crack Detection

Crack detection is one of the most common tasks in civil infrastructure condition assessment. The tradition usually follows a routine, in which first, images are captured using digital cameras; and then the cumbersome processing is performed after the field inspection. Utilizing the proposed approach, field-based real-time or near-real time crack detection and quantification can be realized. Herein, the basic implementation based on the Android Development Kit and the OpenCV library is illustrated.

The crack detection implemented in this thesis is based on stitching and Canny edge detection. By recording a video stream of crack scene, the application will capture frames from the video and stitch these frames with the algorithm stated in Chapter 3. Thus, a panoramic sense of the crack is obtained. Then Gaussian filter will be applied to reduce noise and the Canny algorithm to detect the crack. The function has multiple parameters that can help engineers to get the best processed data they need.

Figure 4.7 shows the application workflow of Canny crack detection. In this workflow, user will always have a view of the processed image (the edge image). By changing the parameters, which are sigma and thresholds, user may ultimately have the best visual result.

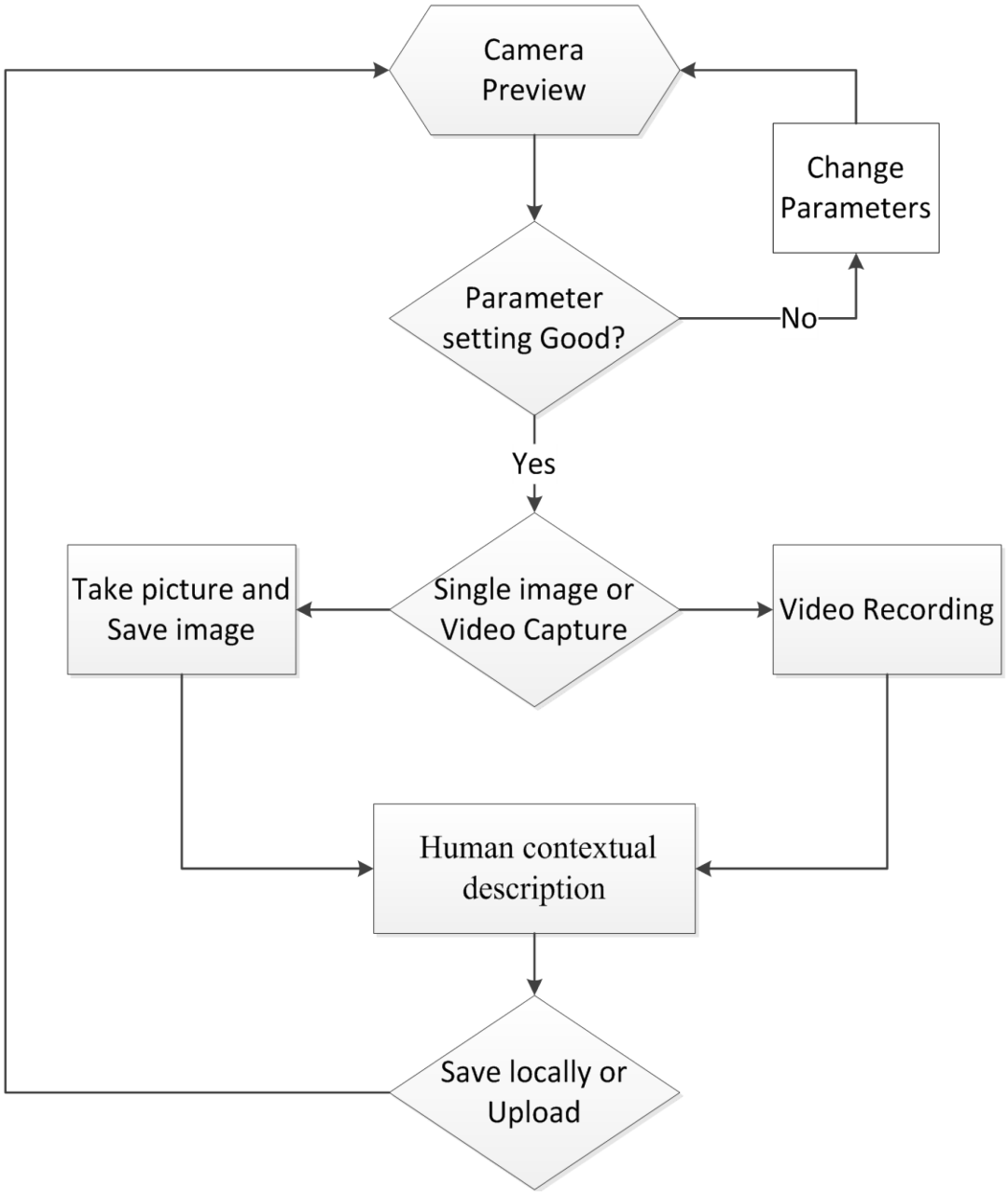


Figure 4.7 The crack detection programing workflow.

As descripted in chapter 3, a Gaussian filter is implemented before Canny edge detection, the kernel window of Gaussian procedure moves in the direction of the maximum increase in the joint density gradient, this smoothing procedure can wipe out the glitches of the image but details such as crack information are preserved. The parameters of the spatial window and the color window size resolution have different effects on smoothing the image. Only features with large spatial support are represented in the filtered image when the spatial window radius Sp increases. On the other hand, only features with high color contrast survive when the color window radius Sr is large. The effect of color window radius is shown in Figure 4.8.

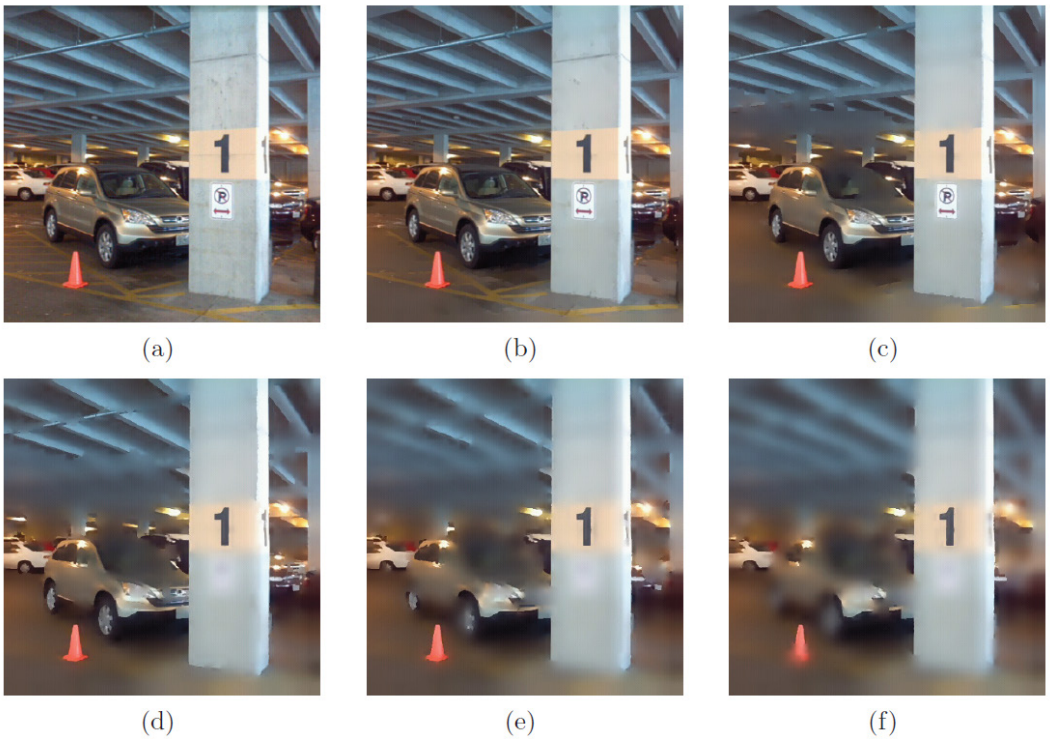
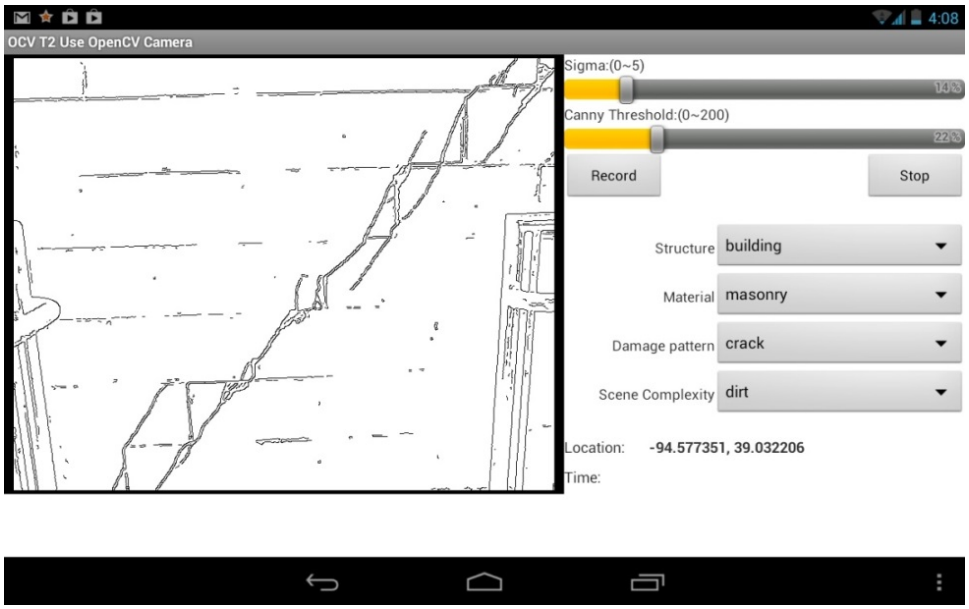


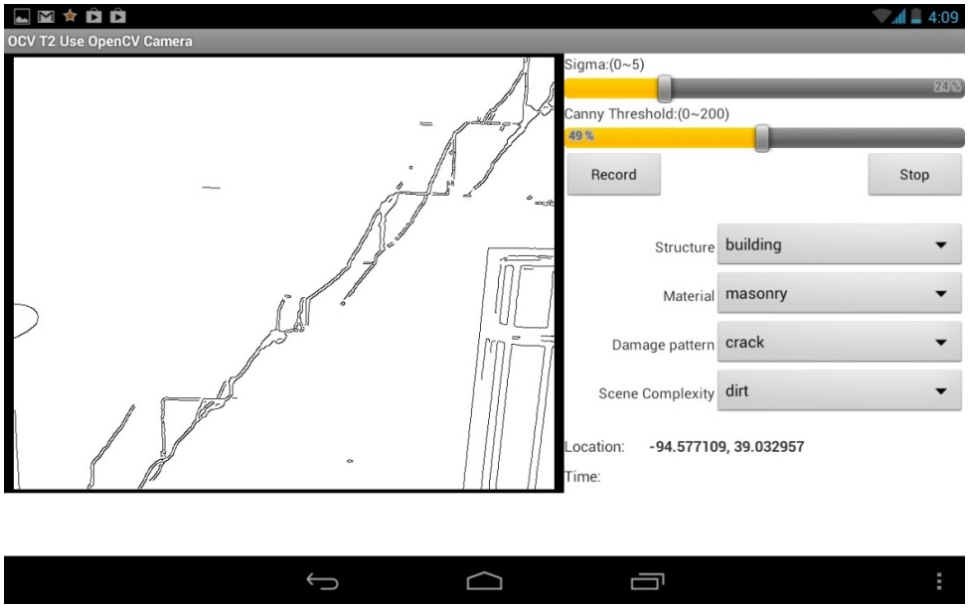
Figure 4.8 Parameter color window size radius Sr's effect on result image: (a) original image; (b) Sr = 10; (c) Sr = 30 ;(d) Sr = 50 ; (e) Sr = 70; (f) Sr = 90.

The second step is to use Canny algorithm to highlight the edge of the crack while smoothing out other noisy information. When using Gaussian filter smoothing the image, the larger the sigma value is, the more noisy information will be removed. Then Canny algorithm will get different result depending on the value of sigma. Figure 4.8 shows when parameter sigma increases, the more of the detail information is smoothed. In the program for Gaussian blur filtering, the spatial window radius is taken a small and constant value to maintain a high-speed computing. Effectively, the sigma in Gaussian blur filtering reveals the level of details that engineers want to reserve.

Figure 4.9 shows that by changing sigma and threshold value, user can expect a best result from canny detection. With higher sigma and threshold, the image show less edge information from the background which means less noise, and only high contrast objects are remain. In this way, the user can control these two parameters in real time by sliding the seeking bar, and then finding the best result according to specific situation.



(a)



(b)

Figure 4.9 The Android application that user can do real time parameter control and interactively add human contextual description. (a) Sigma=0.7, Threshold=44. (b) Sigma=1.2, Threshold=98.

Associated with content in Chapter 4, the application has a variety of user and mobile interactive functions deployed such as changing Canny parameters and add engineering knowledge to image data. As a result, this crack detection is a typical mobile image application that meets the goal of the proposed research objective.

4.4 Computational Capability Evaluation

Another problem with mobile imaging, as mentioned earlier, concerns the computing capability of mobile devices. The mobile devices hardware in 1990s and early 2000s were fairly slow. However, analysis in Chapter 1 reveals that modern mobile devices are equipped with powerful CPUs and GPUs. Herein whether these mobile devices can handle heavy duty image processing and how they will performer in real-time application are investigated.

Experiments reveal that most android device on the market can handle real-time when Gaussian blur, canny and some other functions are called. However, when the program call advanced feature detectors or analysis functions, e.g. the SIFT feature detector [Lowe, 1999], the processing rate will be lower than usual. User may notice the time lag when viewing the video stream. Thus, an evaluation on both Android platform and desktop Linux platform is performed to test the performance when calling different OpenCV functions.

In this thesis, an extensive amount of testing is conducted using many field images and all tests are written using the same version of the OpenCV (ver. 2.4.3). Three representative algorithms in OpenCV are tested, which include mean shift filtering for image enhancement, Canny edge detection, and the FAST feature extraction for advanced image object recognition [Rosten and Drummond, 2006]. Canny edge detection is

selected because it is used in one of the main applications and the results are visually straightforward. Mean-shift filtering is usually used as a pre-step for image segmentation, while FAST feature extraction is commonly used by many recognition processes.

Figure 4.10 shows one of the test images, which is a well-defined concrete crack. Therefore, no preprocessing is used prior to testing the algorithms. The main variable measured is the cost of CPU time for running these image algorithms. Let each function run 100 loops and the mean value is calculated.

Figure 4.10 One of the crack images used for image processing and analysis performance evaluation.

Although the time consumption values listed in Table 4.1 indicate the basic trend that among all the three image analysis methods, the performance of the mobile device (which is considered a high-end device as of 2012) lacks behind the workstation. However, consider the power consumption and the mobility, the android device still owns an outstanding performance index of the overall execution speed in image processing.

Table 4.1 A comparative performance evaluation showing processing speed for the test image shown in Figure 4.10 (the time unit is millisecond).

Platform	Specification	Image Analysis Methods		
		Mean-shift Filtering	Canny Edge Detection	FAST feature extraction
Google Nexus 7	Nvidia®1.6 GHz quad-core CPU; 1G RAM; Android 4.1	233.3 (4.2 fps)	337.8 (2.9 fps)	552.6 (1.8 fps)
Linux Workstation	Intel® Core™2 Duo CPU E8400 Frequency: 3GHz; 4G Ram; Ubuntu Linux 12	80.5 (12.4 fps)	32.0 (31 fps)	3.8 (263 fps)

If the time into frames per second (fps) is concerned, for the Canny detection as an example, it will performs at nearly 3 fps, which is relatively acceptable for a real-time image application running on mobile device. However, one should see that for real-time video-image processing of large-scale image as shown in Figure 4.10, the speed is not sufficient compared to a regular imaging frame rate (e.g. 24 fps).

4.5 Chapter Summary

In this chapter the collaborative and interactive mobile imaging framework is introduced. Android application is developed to realizing the framework. Experiments show that they work fine with regular Android-based mobile devices. The chapter also investigates the computing capability and performs an evaluation test. The conclusion is that mobile devices computing capability is less than a work-station but still can be considered as good for many real-time image processing tasks.

Last chapter discussed a novel framework that helps Civil Engineers (who have professional knowledge and sufficient technological knowledge for acceptance of mobile imaging) deal with field problems and implements the interactive mobile image application of crack detection. In Chapter 5, the framework will be extended to the domain of home automation, particularly the smart energy application for the regular public.

5.1 Home Automation Technologies

Overview

Home automation can be described as the introduction of technology within a home environment to improve the life quality of its occupants by providing services such as multimedia entertainment, tele-health and energy conservation. Home automation applications integrate comfort, energy saving, security and communications functions. The aim of a home automation system is to provide homes with a certain degree of 'intelligence' and to improve the quality of life of its inhabitants. The typical application for home automation may contain tasks like switching lights, automatic air-conditioning, controlling appliances or temperature from remotely such as mobile devices or Internet.

In recent years, much research work has been done in the general area of Home Automation. For example, the Georgia Institute of Technology developed an 'Aware Home', based on ubiquitous computing that senses and recognizes potential crises, to assist declining elderly memory and find behavioral trends [Kidd et al., 1999]. The Helal et al. developed Gator Tech Smart House at the University of Florida for the elderly and

the disabled, which was based on environmental sensors for comfort and energy efficiency, safety and security, activity/mobility monitoring, reminder/prompting technologies, fall detection systems, smart devices and appliances [Helal et al., 2005]. A Java-based automation system was developed that can monitor and control home appliances via the Internet [Al-Ali and Al-Rousan, 2004]. Based on a standalone embedded system board integrated into a PC-based server at home, this system connected physically all the home automation appliances. There are also a lot of standards and protocols adopted by the leading companies in the market Some notable examples are KNX (ISO/IEC14543-3-X and EN50090 standards), Lonworks (ISO/IEC 14908, EN14908 and EIA-709-1 standards) and X10 (a well-known international and open industry standard for communication among electronic devices) [Sánchez et al., 2011].

Smart energy is a key concept that belongs to home automation including intelligent energy monitoring and saving. Within a smart energy application, window shades, HVAC, central heating, and so on may be controlled depending on the information collected by several types of sensors that monitor parameters such as temperature, humidity, light, and presence. Unnecessary waste of energy can thus be avoided.

There are many commercial products that provide such solutions such as smart outlets and smart meters to monitor and control the usage of energy. ThinkEco® Modlet uses wireless communication to connect to a USB hub that plugs into a home or office computer [ThinkEco, 2012]. The system then monitors and records electricity usage from all Modlet's in the area to a platform that can be accessed online to manage power consumption from any location. The Smart Load Switch Plug by Freelux® allows access

to usage patterns and money spent to power various electronics via an online platform [International, 2012]. Tendril® provides a Zigbee-based Home Area Network (HAN) system that allows users easily monitor and estimate energy usage, and control any connected appliances [Tendrilinc, 2012]. All these solutions provide energy monitoring and various methods for intelligent or automatous energy saving.

UMKC E-SAVE Project

E-SAVE is a research project focusing on extensive use of automation in conserving and using energy wisely in home automation at the University of Missouri-Kansas City (UMKC). It started an energy lab called "Lab for E-Save Through Automation and Education" in the School of Computing and Engineering, UMKC with a gift fund from Toyota. An important goal of the project is to transform the behavior of individuals and communities towards a more efficient and greener use of energy.

The project has three main components for deploying automation (a) intelligent network, (b) smart apps, and (c) intelligent appliances and devices. Intelligence in appliances and devices will be introduced through smart apps and these devices and appliances will be connected through intelligent network.

The best method to make people aware of saving energy is to making energy visible through information feedback. This may include providing more informative bills [Wilhite and Ling, 1995]; putting energy labels on domestic appliances [Boardman, 2004]; providing in-depth energy advice via leaflets, websites and face-to-face, and most recently, through a range of in-home real time displays and monitors [Abrahamse et al., 2007; Anderson and White, 2009]. These researches reveal providing users the energy consumption feedback will result in energy saving [Darby, 2006].

Figure 5.1 shows the E-SAVE smart energy framework. The two key steps to implement smart energy in this smart framework are: energy information gathering and autonomous visualization. The energy information is collected by Zigbee enabled sensor network. One method for getting information is to let the mobile device access the sensing network directly, which needs additional hardware installed on the mobile side. The other way is to push the data from sensor network into the server accessible to the mobile devices. Autonomous visualization of the information on mobile devices is another important step, which is the main research work contributed by this thesis.

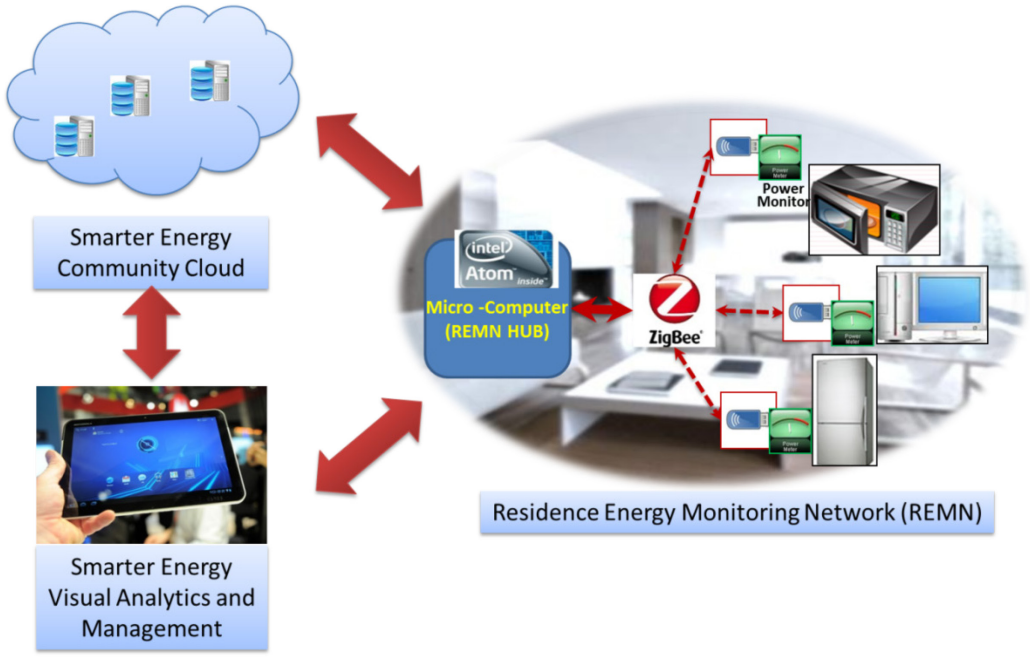


Figure 5.1 E-SAVE Smart Energy Framework.

5.2 Mobile Imaging for Smart Energy

Visualized information can be better accepted by people about how energy is used. Therefore, users may tend to save energy. Showing the picture of the appliance along with its energy information would be a good way. In this work, an Android application is implemented to realize this concept. The core of this application is to let a mobile camera capable of "seeing" the appliance.

In this thesis, template matching is used for recognizing the identity of an appliance. Subsequently, the application will generate graphs or charts based on the data to make users aware of the energy usage. The procedure is summarized as follows:

1) Target the appliance with a smartphone or tablet, show user what the camera have on the device screen;
2) Call template matching function, search what appliances are in the area;
3) Return template matching result;
4) If match found, request this appliance's energy data from server; If match failed, repeat step 2;
5) Show the appliance's name and its energy information on the mobile device.

In these steps, template matching, which is a time consuming process, is done locally on the mobile device. This requires that the application pre-stores the templates of the appliances of interest on the device. Also, limited number of templates may lead mobile device hard to recognize new coming appliances. But for a small application area, this approach works well.

Figure 5.2 shows the template matching example in the E-SAVE lab. The left side is the microwave template image; on the right is shown the real-time template matching

result using multiple algorithms mentioned in Chapter 3.4. Note from different angle other than the template image, the application still can recognize the microwave at the right location. Different colored rectangles in the figure indicate different template matching methods. The normalized correlation matching method mis-matched because of the lighting difference between the template and the object. However, it is noticed that its normalized matching method has reduced this lighting difference and works well.

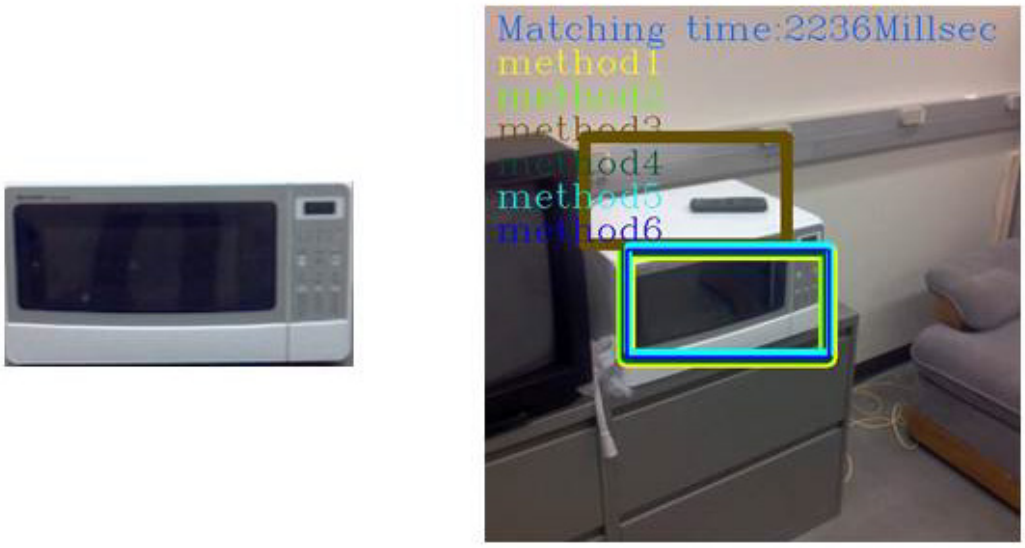
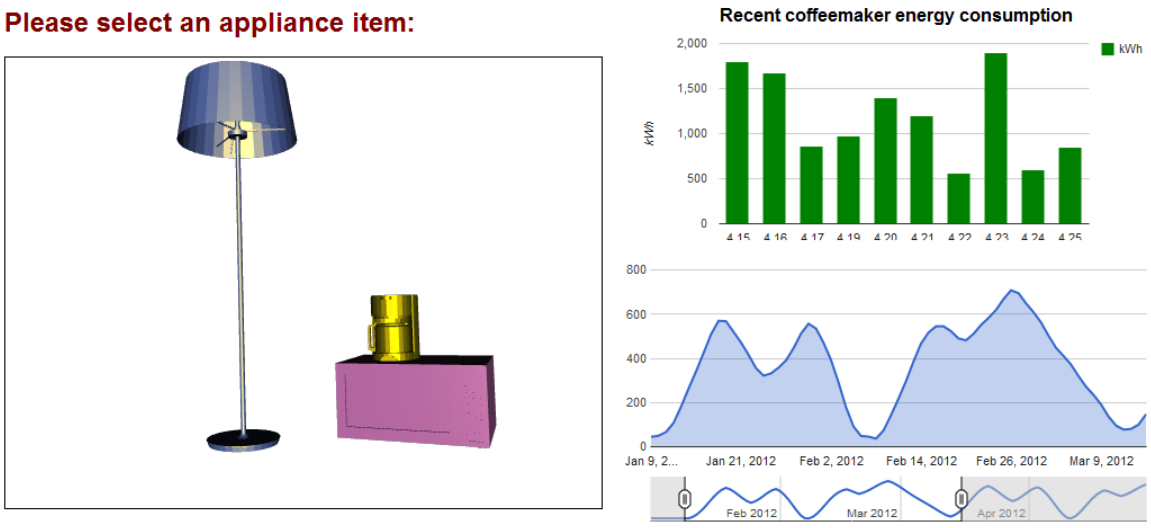


Figure 5.2 Template matching: left: microwave template image; right: matching result.

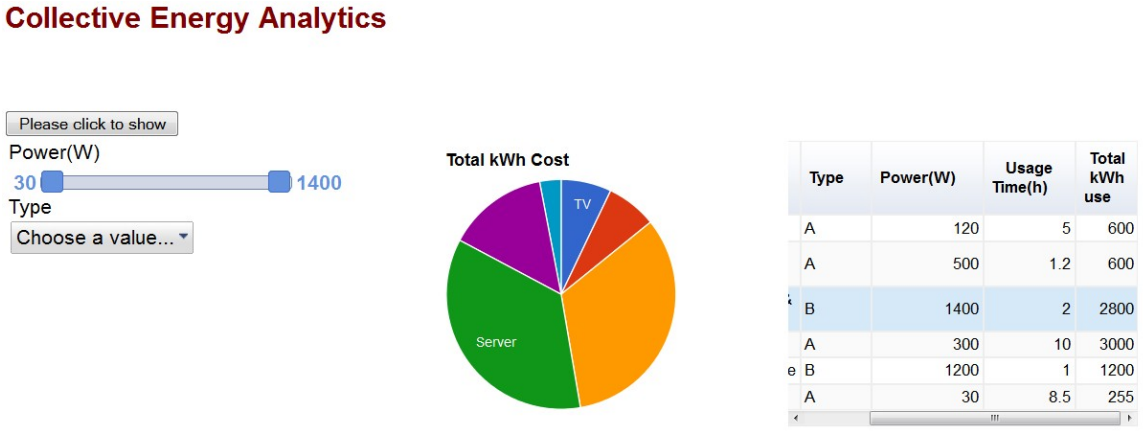
5.3 Platform-independent Energy Analytics

With the help of template matching, a mobile device is able to recognize ('see') what appliance the user is targeting. The energy information of that appliance is then returned by the E-SAVE project database server. To implement the energy analytics, a graph or chart based analytics is needed. Using Android platform's built-in user interface is a direct approach that would require much effort on coding and optimizing.

Consider that mobile devices are not the only device for users, a platform-independent energy analytics system is designed in this thesis. Herein, a novel visualized data analysis method is implemented by using web technology shown in Figure 5.3. This application is build based on WebGL, X3DOM and Google Chart [X3DOM.org, 2010; KHronos, 2011; Google, 2012].



(a)



(b)

Figure 5.3 A web-based implementation of energy analytics: (a) 3-D model and continuous data with column charts; (b) Summary of the energy data shown in a pie chart and a table.

As shown in Figure 5.3 (a), a set of 3-D models created by Google Sketchup are used to represent the appliances in the E-SAVE lab. X3DOM will transfer these 3-D models into HTML language and render them in 3-D so that user can view and rotate these models in a web browser such as Firefox and Google Chrome. When the server receives request for the target appliance information from mobile device, it will highlight the corresponding through template matching or user-input model and show the energy data on the web page just like shown in the right side part of Figure 5.3 (a). Users can check details information about that appliance by clicking and dragging into the chart and graph. These data will be dynamically changed when any update comes from the sensor network described in the above section. A different type of data analysis is shown in Figure 5.3 (b). Here, a pie chart is displayed to show the total energy consumption percentage of all the appliances in the lab, with a table right beside the chart listing all the detail data.

With this web-based analytics, users can instantly have access to the energy data over the past months, days, etc., and learn the energy consumption of each appliance through a mobile phone, tablet, or a computer from anywhere and at any time.

5.4 Chapter Summary

This Chapter mainly described the E-SAVE project and the implementation of mobile imaging in smart energy. A review of current smart energy and home automation is presented, which indicates that visualized feedbacks can effectively improve people's awareness of energy consumption and efforts towards saving energy. Then template matching is used to detect and identify appliances, which is the key step for designing the

smart application. Subsequently, an effective analytics web is designed to show the visualized energy data.

CONCLUSION AND RECOMMENDATION

6.1 General Summary

In this thesis, mobile imaging and computer vision library and their integration with current mobile devices are explored. The notion of mobile imaging is much extended and is beyond image sensing only, which actually includes three inter-connected components: image acquisition, Image processing and understanding, visual analytics.

With this novel notion of mobile imaging, a framework of realizing collaborative mobile imaging for field engineers conducting civil infrastructure condition assessment is presented. The proposed capabilities of this system include data sharing within different units (nodes) of the networked imaging, advanced interactive image processing, and knowledge discovery. The framework allows engineers work collaboratively when distributed in a geographical area. A long crack detection application is demonstrated, which verifies the framework towards seamless integration of sensing, imaging, real-time processing and knowledge discovery in future engineers-centered field reconnaissance and civil infrastructure.

In addition, the application of mobile imaging has been extended for the smart energy project E-SAVE at UMKC. By applying template matching and a novel development of web-based energy analytics, the public can have an awareness of energy consumption and data analytics in real-time.

6.2 Future Work and Recommendation

There are several possible ways to improve the thesis work. Camera calibration plays an important role in mobile imaging. In this work, camera calibration is done by collection of chessboard images using mobile camera and place them on a Linux workstation to do calibration. The result in Chapter 3 shows only slight distortion. However, more efforts can be done by implementing this process in a more complex environment with imaging and computing in one mobile device. Another improvement is to expand the framework to a server-client based mobile imaging. It can extend mobile imaging by improving computing power and storage space.

Besides the server-client model, which relies on the remote server or cloud infrastructure, new concept for mobile computing called "Cloudlets" has emerged [Wolbach et al., 2008; Satyanarayanan et al., 2009]. Cloudlet represents the middle tier of a 3-tier hierarchy: mobile device - cloudlet - cloud. It aims at helping resource-intensive but latency-sensitive mobile applications by "bringing the cloud closer". It is believed that there will be more possible applications for mobile imaging by considering this architecture.

A highly interesting application would be integrating mobile imaging with augmented reality. It would be possible to develop a program that can read the image data from mobile camera and create virtual 3-D objects in real-time using mobile devices. This kind of applications would be useful in research areas such as medical, military and entertainment industries.

Last, as mobile hardware, including both imaging and computing hardware, is evolving faster and more advanced capability is expected in the future, it is desired to

continue exploring more imaging or computing intensive applications that can be further applied to a variety of research and commercial areas.

REFERENCE LIST

Abrahamse, W., L. Steg, C. Vlek and T. Rothengatter [2007] "The effect of tailored information, goal setting, and tailored feedback on household energy use, energy-related behaviors, and behavioral antecedents", Journal of Environmental Psychology 27(4): 265-276.

Al-Ali, A.-R. and M. Al-Rousan [2004] "Java-based home automation system", Consumer Electronics, IEEE Transactions on 50(2): 498-504.

Alliance, W.-F. [2010] "Wi-fi certified wi-fi direct", White Paper: http://www.wi-fi.org/news_articles. php.

Anderson, W. and V. White [2009] "Exploring consumer preferences for home energy display functionality", Centre for Sustainable Energy, Tech. Rep.

Andrade, R., A. V. Wangenheim and M. K. Bortoluzzi [2003] "Wireless and PDA: A novel strategy to access dicom-compliant medical data on mobile devices", International Journal of Medical Informatics 71(2): 157-164.

Bhoraskar, R., N. Vankadhara, B. Raman and P. Kulkarni [2012] "Wolverine: Traffic and road condition estimation using smartphone sensors. " Communication Systems and Networks (COMSNETS), 2012 Fourth International Conference on, IEEE. :1-6

Bioeddie [2006] "Psion organiser one the original psion the machine", http://www.bioeddie.co.uk/models/psion-organiser-1.htm.

Boardman, B. [2004] "New directions for household energy efficiency: Evidence from the UK", Energy Policy 32(17): 1921-1933.

Bradski, G. and A. Kaehler [2008] Learning opencv: Computer vision with the opencv library, O'Reilly Media, Incorporated.

Brown, M. and D. G. Lowe [2007] "Automatic panoramic image stitching using invariant features", International Journal of Computer Vision 74(1): 59-73.

Buddhikot, M., G. Chandranmenon, S. Han, Y.-W. Lee, S. Miller and L. Salgarelli [2003] "Integration of 802.11 and third-generation wireless data networks. " INFOCOM 2003. Twenty-Second Annual Joint Conference of the IEEE Computer and Communications. IEEE Societies, IEEE. 1: 503-512

Burt, P. J. and E. H. Adelson [1983] "A multiresolution spline with application to image mosaics", ACM Transactions on Graphics (TOG) 2(4): 217-236.

Canny, J. [1986] "A computational approach to edge detection", Pattern Analysis and Machine Intelligence, IEEE Transactions on(6): 679-698.

Cheng, K.-T. and Y.-C. Wang [2011] "Using mobile gpu for general-purpose computing–a case study of face recognition on smartphones. " VLSI Design, Automation and Test (VLSI-DAT), 2011 International Symposium on, IEEE. :1-4

Cho, W.-H. and T.-C. Kim [2011] "Image enhancement technique using color and edge features for mobile systems. " IS&T/SPIE Electronic Imaging, International Society for Optics and Photonics. 78760U-78760U-6

Choi, M.-S. and W.-Y. Kim [2002] "A novel two stage template matching method for rotation and illumination invariance", Pattern recognition 35(1): 119-129.

Clark, M. V., K. K. Leung, B. McNair and Z. Kostic [2002] "Outdoor ieee 802.11 cellular networks: Radio link performance. " Communications, 2002. ICC 2002. IEEE International Conference on, IEEE. 1:512-516

Darby, S. [2006] "The effectiveness of feedback on energy consumption", A Review for DEFRA of the Literature on Metering, Billing and Direct Displays :486.

Davis, L. S. [1975] "A survey of edge detection techniques", Computer graphics and image processing 4(3): 248-270.

Ehringer, D. [2010] "The dalvik virtual machine architecture", Techn. report (March 2010).

EyeFi, I. [2013] "Wifi sd cards, " from http://www.eye.fi/.

Feiner, S., B. MacIntyre, T. Höllerer and A. Webster [1997] "A touring machine: Prototyping 3d mobile augmented reality systems for exploring the urban environment", Personal and Ubiquitous Computing 1(4): 208-217.

Ferro, E. and F. Potorti [2005] "Bluetooth and wi-fi wireless protocols: A survey and a comparison", Wireless Communications, IEEE 12(1): 12-26.

Forman, G. H. and J. Zahorjan [1994] "The challenges of mobile computing", Computer 27(4): 38-47.

Fossum, E. R. [1997] "Cmos image sensors: Electronic camera-on-a-chip", Electron Devices, IEEE Transactions on 44(10): 1689-1698.

Fullpower, I. [2013] "First camera-phone image june 11, 1997, " 2013, from http://www.fullpower.com.

Gammeter, S., A. Gassmann, L. Bossard, T. Quack and L. Van Gool [2010] "Server-side object recognition and client-side object tracking for mobile augmented reality. " Computer Vision and Pattern Recognition Workshops (CVPRW), 2010 IEEE Computer Society Conference on, IEEE. 1-8

Gartner, I. [2013] "Gartner says worldwide sales of mobile phones, " from http://www.gartner.com/newsroom/id/2017015.

Google [2013] "Application fundamentals", http://developer.android.com/guide/components/fundamentals.html.

Google, I. [2012] "Introduction to using chart tools", https://developers.google.com/chart/interactive/docs/index.

GREENE, M., P. GROSSI, S. TUBBESING, N. BASOZ and R. LOVE [2004] "Learning from earthquakes: New directions and initiatives. " Proceedings of the 13th World Conference on Earthquake Engineering.

Gsmarena [2013] "A comprehensive look at the mobile industry", http://www.gsmarena.com.

Heikkila, J. and O. Silven [1997] "A four-step camera calibration procedure with implicit image correction. " Computer Vision and Pattern Recognition, 1997. Proceedings., 1997 IEEE Computer Society Conference on, IEEE. 1106-1112

Helal, S., W. Mann, H. El-Zabadani, J. King, Y. Kaddoura and E. Jansen [2005] "The gator tech smart house: A programmable pervasive space", Computer 38(3): 50-60.

Hull, J. J., X. Liu, B. Erol, J. Graham and J. Moraleda [2010] "Mobile image recognition: Architectures and tradeoffs. " Proceedings of the Eleventh Workshop on Mobile Computing Systems & Applications, ACM. 84-88

International, F. [2012] http://www.freelux.eu/.

Jing, J., A. S. Helal and A. Elmagarmid [1999] "Client-server computing in mobile environments", ACM computing surveys (CSUR) 31(2): 117-157.

Johnson, D. A. and M. M. Trivedi [2011] "Driving style recognition using a smartphone as a sensor platform. " Intelligent Transportation Systems (ITSC), 2011 14th International IEEE Conference on, IEEE. 1609-1615

Khan, A. M., Y.-K. Lee, S. Lee and T.-S. Kim [2010] "Human activity recognition via an accelerometer-enabled-smartphone using kernel discriminant analysis. " Future Information Technology (FutureTech), 2010 5th International Conference on, IEEE. 1-6

KHronos [2011] "Webgl specification", https://www.khronos.org/registry/webgl/specs/1.0/.

Kidd, C., R. Orr, G. Abowd, C. Atkeson, I. Essa, B. MacIntyre, E. Mynatt, T. Starner and W. Newstetter [1999] "The aware home: A living laboratory for ubiquitous computing research", Cooperative buildings. Integrating information, organizations, and architecture: 191-198.

Klug, B. [2011] "Ti announces omap4470 and specs: Powervr sgx544, 1.8 ghz dual core cortex-a9", http://www.anandtech.com/show/4413/ti-announces-omap-4470-and-specs-powervr-sgx544-18-ghz-dual-core-cortexa9.

Kobie, N. [2009] "Nokia's 'point & find' uses camera phone for search", http://www.itpro.co.uk/610402/nokias-point--find-uses-camera-phone-for-search.

Kondo, Y. [2002] "[medical image transfer for emergency care utilizing internet and mobile phone]", Nippon Hoshasen Gijutsu Gakkai Zasshi 58(10): 1393.

Krüger, J. and R. Westermann [2005] "Gpu simulation and rendering of volumetric effects for computer games and virtual environments. " Computer Graphics Forum, Wiley Online Library. 24: 685-693

Lane, N. D., E. Miluzzo, H. Lu, D. Peebles, T. Choudhury and A. T. Campbell [2010] "A survey of mobile phone sensing", Communications Magazine, IEEE 48(9): 140-150.

Leung, K. K., B. McNair, L. J. Cimini Jr and J. H. Winters [2002] "Outdoor ieee 802.11 cellular networks: Mac protocol design and performance. " Communications, 2002. ICC 2002. IEEE International Conference on, IEEE.1: 595-599

Liu, Y., J. Yang and M. Liu [2008] "Recognition of qr code with mobile phones. " Control and Decision Conference, 2008. CCDC 2008. Chinese, IEEE. 203-206

Lowe, D. G. [1999] "Object recognition from local scale-invariant features. " Computer vision, 1999. The proceedings of the seventh IEEE international conference on, IEEE.2: 1150-1157

Lowe, D. G. [2004] "Distinctive image features from scale-invariant keypoints", International Journal of Computer Vision 60(2): 91-110.

Mathworks, I. [2013] "Matlab and simulink for technical computing, " from http://www.mathworks.com/.

Mobithinking. [2013] "Global mobile statistics 2013 part a: Mobile subscribers; handset market share; mobile operators, " from http://mobithinking.com/mobile-marketing-tools/latest-mobile-stats/a.

Nokia [2010] "Your apps on the world's largest mobile platform", http://www.developer.nokia.com/Develop/Series_40/Platform/.

NVIDIA [2013] "Introducing nvidia® tegra® 4", http://www.nvidia.com/object/tegra-4-processor.html.

Ohbuchi, E., H. Hanaizumi and L. A. Hock [2004] "Barcode readers using the camera device in mobile phones. " Cyberworlds, 2004 International Conference on, IEEE. 260-265

Ojanpera, T. and R. Prasad [1998] "An overview of third-generation wireless personal communications: A european perspective", Personal Communications, IEEE 5(6): 59-65.

Owens, J. D., M. Houston, D. Luebke, S. Green, J. E. Stone and J. C. Phillips [2008] "Gpu computing", Proceedings of the IEEE 96(5): 879-899.

Peli, T. and D. Malah [1982] "A study of edge detection algorithms", Computer graphics and image processing 20(1): 1-21.

Picard, R. W. [1995] "Light-years from lena: Video and image libraries of the future. " Image Processing, 1995. Proceedings., International Conference on, IEEE.1: 310-313

Prokop, R. J. and A. P. Reeves [1992] "A survey of moment-based techniques for unoccluded object representation and recognition", CVGIP: Graphical Models and Image Processing 54(5): 438-460.

RIM [2012] "Research in motion is now black berry", http://ca.blackberry.com/company.html.

Rodgers, J. L. and W. A. Nicewander [1988] "Thirteen ways to look at the correlation coefficient", The American Statistician 42(1): 59-66.

Rosten, E. and T. Drummond [2006] "Machine learning for high-speed corner detection." Computer vision–eccv 2006, Springer: 430-443.

Royer, E. M. and C.-K. Toh [1999] "A review of current routing protocols for ad hoc mobile wireless networks", Personal Communications, IEEE 6(2): 46-55.

Sager, I. [2012] "Before iphone and android came simon, the first smartphone", http://www.businessweek.com/articles/2012-06-29/before-iphone-and-android-came-simon-the-first-smartphone.

Sánchez, P., M. Jiménez, F. Rosique, B. Álvarez and A. Iborra [2011] "A framework for developing home automation systems: From requirements to code", Journal of Systems and Software 84(6): 1008-1021.

Sarvas, R., M. Viikari, J. Pesonen and H. Nevanlinna [2004] "Mobshare: Controlled and immediate sharing of mobile images. " Proceedings of the 12th annual ACM international conference on Multimedia, ACM. 724-731

Satyanarayanan, M., P. Bahl, R. Caceres and N. Davies [2009] "The case for vm-based cloudlets in mobile computing", Pervasive Computing, IEEE 8(4): 14-23.

Secilla, J. P., N. Garcia and J. Carrascosa [1988] "Template location in noisy pictures", Signal Processing 14(4): 347-361.

Sena [2010] "The comparison of wi-fi, bluetooth and zigbee", http://www.sena.com/blog/?p=359.

Shi, M., X. Shen and J. W. Mark [2004] "Ieee 802.11 roaming and authentication in wireless lan/cellular mobile networks", Wireless Communications, IEEE 11(4): 66-75.

Shuaib, K., M. Boulmalf, F. Sallabi and A. Lakas [2006] "Co-existence of zigbee and wlan, a performance study. " Wireless Telecommunications Symposium, 2006. WTS'06, IEEE. 1-6

Sun, A., Y. Sun and C. Liu [2007] "The qr-code reorganization in illegible snapshots taken by mobile phones. " Computational Science and its Applications, 2007. ICCSA 2007. International Conference on, IEEE. 532-538

Tao, C., R. Li and M. A. Chapman [1998] "Automatic reconstruction of road centerlines from mobile mapping image sequences", Photogrammetric engineering and remote sensing 64(7): 709-716.

Tendrilinc [2012] http://www.tendrilinc.com/energy-providers/product/insight/.

ThinkEco [2012] "Energy waste at the plug", http://www.thinkecoinc.com/.

Tsai, R. [1987] "A versatile camera calibration technique for high-accuracy 3d machine vision metrology using off-the-shelf tv cameras and lenses", Robotics and Automation, IEEE Journal of 3(4): 323-344.

Tu, Z. and R. Li [2000] "Automatic recognition of civil infrastructure objects in mobile mapping imagery using a markov random field model. " Proceedings of the ISPRS 19 th Congress. 16-23

Vanderbrug, G. J. and A. Rosenfeld [1977] "Two-stage template matching", Computers, IEEE Transactions on 100(4): 384-393.

Wagner, D., G. Reitmayr, A. Mulloni, T. Drummond and D. Schmalstieg [2010] "Real-time detection and tracking for augmented reality on mobile phones", Visualization and Computer Graphics, IEEE Transactions on 16(3): 355-368.

Weng, J., P. Cohen and M. Herniou [1992] "Camera calibration with distortion models and accuracy evaluation", IEEE Transactions on pattern analysis and machine intelligence 14(10): 965-980.

Wilhite, H. and R. Ling [1995] "Measured energy savings from a more informative energy bill", Energy and buildings 22(2): 145-155.

Wolbach, A., J. Harkes, S. Chellappa and M. Satyanarayanan [2008] "Transient customization of mobile computing infrastructure. " Proceedings of the First Workshop on Virtualization in Mobile Computing, ACM. 37-41

X3DOM.org [2010] "Welcome to x3dom", http://x3dom.org/docs/dev/.

Xue, X., A. Cheryauka and D. Tubbs [2006] "Acceleration of fluoro-ct reconstruction for a mobile c-arm on gpu and fpga hardware: A simulation study. " Proc. SPIE.6142: 61424L

Yeh, T., K. Grauman, K. Tollmar and T. Darrell [2005] "A picture is worth a thousand keywords: Image-based object search on a mobile platform. " CHI'05 extended abstracts on Human factors in computing systems, ACM. 2025-2028

Yu, S.-N., J.-H. Jang and C.-S. Han [2007] "Auto inspection system using a mobile robot for detecting concrete cracks in a tunnel", Automation in Construction 16(3): 255-261.

ZDnet [2005] "Mobile os market shares in 2005", http://www.zdnet.com/blog/itfacts/mobile-os-market-shares-in-2005-symbian-51-linux-23-windows-17/10153.

Zhang, Z. [2000] "A flexible new technique for camera calibration", Pattern Analysis and Machine Intelligence, IEEE Transactions on 22(11): 1330-1334.

VITA

Jianfei Chen was born on April. 23th, 1987 in Chongqing, China. He was educated in Bashu High School and graduate in 2006. He attended Beijing University of Posts and Telecommunications and received his Bachelor degree in Electrical and Information Engineering in 2010, Beijing, China.

In 2010, he was admitted into the Master of Science in Electrical and Computer Engineering department at University of Missouri Kansas City, Kansas City, MO, USA. His area of interest is image processing.

He is expecting to graduate in May 2013 and continuing his Ph.D. program in Electrical and Computer Engineering department at University of Missouri Kansa