Wave energy resource characterization and power prediction around Tanah Merah Ferry Terminal (TMFT)

Zhang, Xue

2016

Zhang, X. (2016). Wave energy resource characterization and power prediction around Tanah Merah Ferry Terminal (TMFT). Master's thesis, Nanyang Technological University, Singapore.

<https://hdl.handle.net/10356/68730>

<https://doi.org/10.32657/10356/68730>

**NANYANG
TECHNOLOGICAL
UNIVERSITY**

**Development of Android/Iphone Application for
Image Processing and Searching**

ZHANG JIAWEI

SCHOOL OF ELECTRICAL AND ELECTRONIC ENGINEERING

**A DISSERTATION SUBMITTED IN PARTIAL FULFILMENT OF
THE REQUIREMENTS FOR THE DEGREE OF
MASTER OF SCIENCE IN SIGNAL PROCESSING**

2015

Table of Contents

	Page
Abstract	iii
Acknowledgments	iv
List of Figures	v
Chapter 1 Introduction	1
1.1 Background and Motivation	1
1.2 Objective and Scope	3
1.3 Organizations of the Dissertation	5
Chapter 2 Literature Review	6
2.1 Android	6
2.2 WeChat Image Platform	9
2.3 Removing Light Reflection	12
2.3.1 Crossed Polarizers	12
2.3.2 Gradient Projection and Flash-exposure Sampling	14
2.3.3 Physical Constrained Optimization	16
2.4 Inpainting	17
Chapter 3 Pre-processing	21
3.1 Methods for Detect Glare Area	22
3.1.1 HSV Color Space Model	23
3.1.2 Thresholds for Gray Image	26
3.2 Light Reflection Removing and Recovery	26

3.2.1	Dilation	26
3.2.2	Inpainting	28
Chapter 4	Android Application	37
4.1	WeChat API	37
4.2	Main Application	41
Chapter 5	Results and Discussion	48
5.1	Pre-processing Results	48
5.1.1	Result	48
5.1.2	Discussions	51
5.2	Application Results	53
Chapter 6	Conclusions and Future work	62
6.1	Conclusions	62
6.2	Recommendation for Future Work	63
	References	64

Abstract

This project is mainly about designing an Android application related to image searching and recognition. With the WeChat API, the application realize the function of comparing the captured photo with the data library for the result of target MD5 code and use the code as an index obtain the specific name of the logo in the taken photo. There are two way to approach to the detected picture, one is taking from the camera right now; another is obtain from the album in the smartphone. Also the application gives a link at the result page so that users can easily get the access to the detailed information about the logo they are looking for. Besides, this project put forward an algorithm to decrease the glares in the images that need to be recognized.

Acknowledgments

This dissertation was a great learning experience thanks to the wonderful people who had helped the author to gain technical skills and valuable knowledge. The author is grateful to them, and would like to express her appreciation to the following:

1. Assoc. Prof. Tan Eng Leong – Dissertation Supervisor – For his guidance and support for the project. A/P Tan had guided the author in technical skills and also be generous to answer all the questions and doubt. Thanks for his kind advising on the process of this project, which made me be clearly to the objective of my experiment.
2. Ms. ZHAO YUQING – Master Student in NTU – For being thoughtful in cooperation the team works of building the user interface of the Android application and generously assistant related issues' collection.
3. Ms. CHEN QIAN – Master Student in NTU – For being very supportive and graciously in the collaboration of team work.
4. Every other research staffs and students in Communication Research Lab who had gave me hands during the period of my dissertation.

List of Figures

	Page
Figure 1.1 Previous Software.....	2
Figure 1.2 iCar Logo.....	4
Figure 1.3 Project Overview.....	4
Figure 2.1 the Structure of Android System	7
Figure 2.2 WeChat Services Framework.....	10
Figure 2.3 Test Image	11
Figure 2.5 ICA	14
Figure 2.6 Algorithm Overview.....	15
Figure 2.7 Basic Algorithm Overview.....	16
Figure 2.8 Inpainting Illustration	18
Figure 3.1 Comparison of with & without Flash	21
Figure 3.2 Overflow for Pre-processing	22
Figure 3.3 HSV Color Space Model.....	23
Figure 3.4 HSV Color Distribution.....	24
Figure 3.5 Code of Threshold for Gray Image	26
Figure 3.6 Code for Dilation.....	27
Figure 3.7 Sample of Function strel.....	27
Figure 3.8 Overflow of Inpainting Function.....	28
Figure 3.9 Code for Inpainting	29
Figure 3.10 Inpainting Code	32
Figure 4.1 Developer page.....	37
Figure 4.2 Register Icar App.....	38
Figure 4.3 Register Successfully	39
Figure 4.4 Application Authorization Code	39
Figure 4.5 Picture Library.....	40
Figure 4.6 Establishing Main UI.....	41
Figure 4.7 Permission Support.....	42
Figure 4.8 Main Activity	44
Figure 4.9 Searching Activity.....	45

Figure 4.10 Result Activity..... 45

Figure 4.11 Create Virtual Device..... 46

Figure 4.12 Debug and Run the Application 47

Figure 5.1 Original Image..... 48

Figure 5.2 Grayscale Image 49

Figure 5.3 High Brightness Section..... 49

Figure 5.4 Dilation of High Brightness Section 50

Figure 5.5 Damaged Image..... 50

Figure 5.6 Recovered Image 51

Figure 5.7 Threshold= 0.85×255 , Radius=5pixels..... 52

Figure 5.9 Threshold= 0.9×255 , Radius=5pixels..... 52

Figure 5.8 Threshold= 0.85×255 , Radius=10pixels..... 53

Figure 5.9 Application in Virtual Device 54

Figure 5.10 Main Page in Virtual Device 55

Figure 5.11 Application in Smartphone..... 56

Figure 5.12 Main Page 56

Figure 5.13 Camera Model 57

Figure 5.14 Taken Result..... 57

Figure 5.15 Searching Page 58

Figure 5.16 Result Page for Image from Camera 58

Figure 5.17 Homepage Link for Image from Camera 59

Figure 5.18 Main Page for Album Model..... 59

Figure 5.19 Album Model..... 60

Figure 5.20 Select Image 60

Figure 5.21 Result for Image from Album 61

Figure 5.22 Homepage Link for Image from Album..... 61

Chapter 1

Introduction

This chapter would introduce the background, motivation, objectives, scope and organization of this dissertation.

1.1 Background and Motivation

Car is one of the popular means of transportation nowadays. Distinguishing different automobile logos is important especially for motor lovers or someone who intend to acquaint this field. But for most people, especially female, it is a hard work to finish without any help. That is the motivation this project conducting for.

Customs show the needs and the application follow their requirements to fulfill the pursuing. According to the motivation, the team investigates the market and find out problems like: what the previous work have done before; what is the gap between the demand of customers and product existing now. The result shows that, in the application market, there were already some products introducing the automobile logos, but most of them are stiff. Those applications just give a list of car logo and the name corresponding to the image (shown at Figure 1.1 Previous Software). Customers need to compare them one after another by themselves which is not only wasting of time but confusing. It is also not convenient when they are outside and encounter with a car in the street or a peripheral products of a certain type of brand. They need to capture the main characteristic of the logo at a short time and remember it in minds until they open up the app and complete the comparison with a long list of illustrated book uncertainly. Usually they just cannot tell which the best matching logo is because the short-term memory of human brain can only last for a few seconds.

However it may takes minims for user to take out the smartphone and doing the comparison from the massive list. What if we can scan the logo and the result can be searched by the app automatically. What’s furthermore, its details just pop out by clicking on the button easily, such as the history, the latest product models, prices and so on.

Figure 1.1 Previous Software

Today internet is everywhere for us to connect to all the information we want and the camera on smartphones is powerful enough to capture the targeting car logo. It would be the right time for us to develop a smartphone application to help the users recognizing car logo all, even all the logo automatically instead of artificially

self-checking. In this project, an application for car brand recognition and searching is designed to meet the requirement. In the meanwhile, WeChat release its image and audio recognition platform which provide an excellent method to simplify the work for this idea becoming reality. So the project is developed base on WeChat image recognition platform.

Referring to the commercial prospect of this project, the customer searching history data, adverting inside the app and peripheral products market assemble the main part. Firstly, if the server gathering the searching history from all the users, the analyzed result gives a direct aspect of which motor company needs to put more effort on propaganda to their brand. It can be predicted that with the raising of their awareness, more purchaser are bring to their product and the sales would enhance tremendously. Second is about the product placement. According to the result of searching, the app can play some adverting related to the brand. For example, introduce new listing vehicle or best seller among the product. Last one is peripheral product. This is related to the branding effect. Give the link to commodities that have the logo on it, which can be extended like car tire, key chain, T-shirt even water bottle. All this above will bring up a lot of value in reality.

1.2 Objective and Scope

In this project, an Android application with the name of iCar (Figure1.2 iCar Logo shows the detail) is developed to achieve the objective of recognition car logos automatically by the smartphone.

Figure 1.2 iCar Logo

The function is realized through the following steps (also can seeing from the Figure 1.3 Project Overview). Firstly, picture the real car logos by system camera on custom devices. Then after some pre-processing, send the image to the server for image searching and recognition within the established picture library (the database storing the potential images that might be detected or similar to images that might be using for recognition). Finally, get the result of the logo and give detailed information such as its homepage link. Customers can follow the link for further information of the specific type of car. In this application, WeChat image recognition platform is used to assistant the work and all the jobs are conducted under java environment, programmed with eclipse.

Figure 1.3 Project Overview

1.3 Organizations of the Dissertation

This report has six chapters, starting from background, then introducing the basic theories using or related to this project, next giving description of the algorithm and procedure for conducting this dissertation, finally going to the result and analysis. From all the process above, at the last chapter the author draw a conclusion for this project and talk about the future work and possibility.

Chapter 1 is the introduction of the background, motivation, objective and scope of the project. It is a brief description of what have this project done.

Chapter 2 presents some basic concepts of this project and also some related work.

Chapter 3 introduces the previous processing algorithm used to reduce the light reflection on over-exposure photo.

Chapter 4 exhibits the architecture of this application.

Chapter 5 presents the result of this project and some discussion base on it.

Chapter 6 is the conclusion and recommendations about the project.

Chapter 2

Literature Review

In this chapter, the basic concepts and practical application of the android architecture, WeChat image recognition platform and image searching and recognition are described.

2.1 Android

Android, developed by Google and the Open Handset Alliance, is a free, open-source and Linux-based operating system which is mainly used in mobile devices such as smart phones and tablet PCs. It was originally developed by Andy Rubin, who makes this system focus more on hand phone. But later, in August 2005, Google acquired it and made it become the operation system that has the most users. November 2007, Google as well as 84 hardware manufacturers, software developers and telecommunication operators formed the Open Handset Alliance, working together to improve the development of Android system. Followed by Google with open source Apache License permits released the Android source code. The first Android smartphone launched in October 2008, then it gradually extended its field to other aspects, such as TVs, digital cameras, game and so on. Until the fourth quarter 2013, the global market share of Android platform for mobile phones has reached 78.1%. On September 24, 2013, when Google Android ushered in its fifth birthday, the number of devices using this system worldwide has reached 1 billion units.

The system architecture of Android uses a layered structure which is similar to the one for operating system. From Figure 1.1 the Structure of Android System you can see that Android is divided into four layers respectively, from the top to the bottom is

the application layer, application framework layer, the system runtime layer and the Linux kernel layer [1].

Figure 2.1 the Structure of Android System

Android operation system is released in conjunction with a series of core application package. The application package includes the client, SMS short message program, calendar, map, browser, contact management procedure and so on. All applications are written using JAVA. Developers can also have full access to the API (Application Programming Interface) which core application framework are using. The application architecture is designed to simplify the reuse of components; any application can publish its function blocks and any other applications can use the released function blocks while following the security rule.

What hidden behind each application is a series of services and systems, including: rich and scalable view (can be used to build applications), content providers (allows

applications to access data from another application or share their own data with others), resource manager (provide access to non-code resources such as localized strings, graphics, and layout), notification manager (allows applications to display a custom message in the status bar and activity manager (manage the application life cycle and provides common navigation rollback feature).

There are four components of Android developers: Activity, used to represent functions. Service, running in the background and does not provide interface presentation. Broadcast receiver, for receiving a broadcast. Content providers, supports multiple applications to store and retrieve data, the equivalent of a database.

The advantage of this Platform can be described at four main aspects. Firstly, Android is an open system. It allows any mobile terminal manufacturers takes part in the Open Handset Alliance of Android which supply a huge opportunity to a lot of talent programmers to develop the platform together. In the meanwhile, with the copiousness of user and application, Android blossoms quickly with a new appearance of its platform. Secondly, it shows that this operation system has been widely applied to abundant hardware. This point is also related to the one mention before. Since Android is an open system, the vendors offer a variety of products, each with a characteristic feature. The differences and features have no effect on the data synchronization, even compatibility of software. Another strong point is that Android platform provides an extremely broad and free environment to third-party developers. It is not hard for people to imagine how many novel and unique software will be born. But the problem that how to control the objective of the game and software avoiding bloody, violent and erotic is left for Android to solve. The last one is absolutely belongs to Google application. Going through 10 years of history, from the search giant to fully penetrating the Internet, Google services, such as maps, e-mail, search,

and etc., have become an important link between the users and the Internet. Now, the Android platform seamlessly combines these outstanding Google services with mobile phones.

2.2 WeChat Image Platform

This platform is an open system facing to majority of third-party developers with the ability to realize image recognition capability. Through this platform, third-party developers can implode the ability of image recognition into their own applications. By subjoining SDK (Software Development Kit) of this platform, the target application would be equipped with the functions of identification and searching mobile visual objects. The image recognition application can send a request to the cloud server and get the identification result. Though this strong service, users can identify posters, advertising, magazines and picture content on newspapers. Besides, after searching, developers can achieve extra functions correlated to the searching results, such as providing extended readings, guiding the audience making a purchase of certain commodity, participating associated activities, giving comments and other interactive.

WeChat public platform differs from previous projects. As the customers and developers can see its implementation is no longer based on computer underlying implementation. From the application level, the system needs to package upper level business the front display layer, logic layer and communication layer. The lower levels are the real service system. From the overall consideration, the system supports all the business through WeChat universal service framework design (details in Figure 2.2 WeChat Services Framework). So changes within the services framework only affect and make adjustment to the related WeChat interface. Multiple business systems using the same public service framework just need to update their application interface the service will work as normal. Through the

service interface, the WeChat service framework is separated from business logic and the registration register the business service to the framework. Meanwhile, the calling of business applications is through service dispatcher. On this basis, business applications can be developed brand new and can also package related services, provide corresponding provider and support for WeChat interface based on the existing business logic. The specific framework is shown below:

Figure 2.2 WeChat Services Framework

How to start your journey with WeChat image open platform at android application programming world?

Step one: You need to enter the download page at WeChat open platform website and download Android SDK; a compressed package including demo, SDK, developing documentation and test application authorization code. Inside the demo examples use API in SDK with different functions.

Step two: Install the SDK demo on your smartphone and test applications using the test application authorization code. The role of test application authorization code is to allow you to undergo your pre-development work before your application to be approved by the authority. The users need to replace the code as soon as possible with a formal application authorization code post approved.

Step three: Open the SDK demo on your smartphone and select “Picturing”. Then picture and recognize the images below for testing (Figure 2.3 Test Image).

Figure 2.3 Test Image

Step Four: After successful recognition, the recognition results page shows the name and corresponding MD5 value of the picture matched. MD5 value is the unique identifier of the picture and can be used to develop extension services of picture identification.

2.3 Removing Light Reflection

Almost every picture taking by camera is done under light source; sun light, artificial light or flash light. The source from the camera side align the object or scenery from the front. However, if the target object is mental glass or something with mental-like surface and the photographer is close to the target, it is high probability that an aliasing will appear on the picture which is the shadow of the background as well as the photographer. So how to remove the aliasing is the main topic we are talking about. Within this project, we pursuing the way that take the picture with over-exposure and extract the highlight flare points then recovery the image for we want to reduce the light reflection only by one taking picture for the object. Below are some related works that others have done before.

2.3.1 Crossed Polarizers

The usual way is to use crossed polarizers. This is the physics way to reduce light reflection (details in Figure 2.4 Physical Model of Light for Camera) [2].

Figure 2.4 Physical Model of Light for Camera

Put one polarizer in front of the light source which means flash for the case of this project, and another rotated one in front of the lens. Adjust the angle the lens polarizer until the reflections is minimized. Using this method the image can be taking with a high quality instead of repair it.

This is also mentioned in the article by Farid, Hany, and Edward H. Adelson. They said that there is no correlation ship between target image and reflected image, so separating the reflection from the desired image can be conduct by taking a series of pictures through a linear polarizer arranged at certain distinct polarization angles which is so-call independent component analysis (ICA). The detail shows in the Equation (2.1) and Figure 2.5 ICA below [3].

$$\tilde{X} = (\tilde{R}_2 \tilde{S} \tilde{R}_1) Y \tag{2.1}$$

Figure 2.5 ICA

But things do come out for the flash is a sudden thing which means we do not have that much time to adjust the polarizer for the most suitable angle and the target camera is on smartphone which is not that professional to be equipped with an external polarizer.

2.3.2 Gradient Projection and Flash-exposure Sampling

Agrawal A, Raskar R, Nayar S K, and Li Y used a flash and no-flash (ambient) image pair to produce better flash images in 2005 [4]. They proposed two algorithms working together for reproduce an enhanced flash image with the highlight removed (show in Figure 2.6 Algorithm Overview). One is based on the gradient coherence model, a neotype gradient methods is applied to cancel the glare

points and bright areas in over-exposure photos. Another one is a luminance-ratio based method that can use the image pair to countervail the fading of luminance caused by depth.

Figure 2.6 Algorithm Overview

Flash gradient $\nabla \phi$ is corrupted by noise $\nabla \eta_F$ due to the reflected checkerboard. By removing the component of the noisy flash gradient vector $\nabla \phi'$ perpendicular to the ambient gradient vector $\nabla \alpha$, the visual effect of noise can be significantly reduced. The computed reflection layer corresponding to the removed component $\nabla \tilde{\eta}_F$ is also shown.

However, this algorithm can obviously reduce such effect, it need an image pair to do the processing work, so it does not fit for this project. The application users would hardly like to take two pictures of the objects with and without flash; it is too much burdensome for them. In the meanwhile, the objects should strictly locate in the same place in image which would be difficult for taking photo with smartphone. But the idea can be used for reference anyway.

2.3.3 Physical Constrained Optimization

In the article of Kong N, Tai Y W, Shin J S., based on physical constrained optimization, they present an access to separate reflection with three multiple polarized images which is taken behind glass [5]. In this algorithm the input three images are captured with a 45 degree polarizer angle among each other from the same point of view, however the outputs are the reflection image and the separated objective image from each input layer. In the paper they applied a multiscale scheme to automatically locating the optimum dividing model for reflection and objective image, for the mixing coefficient of the reflection and objective layers depends on the angle of incidence and the orientation of the plane of incidence. The following are the steps used inside their processing (show in Figure 2.7 Basic Algorithm Overview).

Figure 2.7 Basic Algorithm Overview

The first step: orthogonal image extract. In this procedure two orthogonal images are obtained from the three polarized images with a polarizer angle of 45 degrees differed from each other.

The second step: image separation. In this procedure, through approximating the dimensional changing of incidence angle, the processing can divide reflection and objective image.

The third step: reflection refinement. This procedure perfects these reflection or objective images by proposing constrained optimization.

The forth step: weak edge suppression. In this procedure optimizes the results with edge suppression to remove remaining weak edges.

2.4 Inpainting

Inpainting originally refers to the procedure that reconstructs the lost or destroyed part in images. For example, in a museum a senior professional artist repairs a painting with a long history. However, it now expands the meaning to digital world. Image inpainting, also known as image restoration, image interpolation or video interpolation, refers to the use of complex algorithms to replace lost, damaged image data, primarily to replace some small areas and blemishes. Image inpainting involves filling in part of an image or video using information from the surrounding area. The goal is to produce a revised image in which the inpainted region is seamlessly merged into the image in a way that is not detectable by a typical viewer [6]. The algorithm attempts to imitate basic approaches used by professional restorators. The algorithm also introduces the importance of propagating both the gradient direction (geometry) and gray-values (photometry) of the image in a band surrounding the hole to be filled-in.

It becomes popular this year for its application in a wide range. For example, remove text and logo information from an image or video record; reconstruct scans of deteriorated images by removing scratches or stains. In photography and cinema

industry, it is used for reversing the deteriorating film. This technique can also be applied to remove red-eye and water print in photographs.

With the development of inpainting technology, novel algorithm is introduced to this area. In the following, two kinds of algorithms are presented.

First is image inpainting technique based on fast marching method. In Telea A's article, a novel method of inpainting is proposed based on a reckoning of image gradient to make image smooth [7]. This approach regards the damaged area as level sets and recovery using the fast marching method (FMM). In Figure 2.8 Inpainting Illustration, equation (2.2), (2.3) and (2.4) give the mathematical model used for computing. As figure 2.8 shows, the dark area Ω is the region needs to be inpainted and point p is a must inpainting point on the boundary $\partial\Omega$. $B_\varepsilon(p)$ is a small neighborhood known image with size ε near point p . So the approximation of value of p is determined by the known neighborhood point q inside region $B(\varepsilon)$. As equation (2.2) shows, $I_q(p)$ the approximation of point p is obtained by value $I(q)$ and gradient $\nabla I(q)$ of point q , like the right side of figure.

Figure 2.8 Inpainting Illustration

$$I_q(p) = I(q) + \nabla I(q)(p - q) \quad (2.2)$$

$$I(p) = \frac{\sum_{q \in B_\varepsilon(p)} \omega(p, q) [I(q) + \nabla I(q)(p - q)]}{\sum_{q \in B_\varepsilon(p)} \omega(p, q)} \quad (2.3)$$

$$|\nabla T| = 1 \text{ on } \Omega, \text{ with } T=0 \text{ on } \partial\Omega \quad (2.4)$$

Equation (2.3) demonstrates the inpainting algorithm to approximate point p by adding up the weighted estimation of all the points q in $B_\varepsilon(p)$ with a normalized weighting function $w(p, q)$.

For inpainting all the Ω region, the algorithm is iteratively using equation (2.3) to all the points of boundary $\partial\Omega$, then advance the boundary inside Ω with the raising of distance from original $\partial\Omega$ to the whole area. In equation (2.4) T is the distance map of Ω to $\partial\Omega$ which are exactly successive on isoclines.

Second is a convolution based image inpainting. This kind of algorithm is mainly about using an adaptive kernel to smooth the boundary of damage area within the image. It applies a gradient approach with the known pixels values around the damaged area to compute weights in convolving mask. In the paper of Noori H, Saryazdi S, Nezamabadi-Pour H, they propose a method with five steps to complete the processing of all damaged pixels: choosing a neighborhood pixel, computing central gradient of known pixels with function (2.5), calculating weights with equation (2.6) and (2.7), estimating damaged pixel with function (2.8) [8].

$$X_k = f(P_k) - f(P) \quad (2.5)$$

$$F(x) = \begin{cases} 1 - \left(\frac{x}{\alpha}\right)^2 & \text{if } |x| \leq \frac{\alpha}{2} \\ \left(\frac{x}{\alpha} - 1\right)^2 & \text{if } \frac{\alpha}{2} \leq |x| \leq \alpha \\ 0 & \text{if } |x| \geq \alpha \end{cases} \quad (2.6)$$

$$w(k) = \frac{1}{n} F(X_k) \quad (2.7)$$

$$f'(p) = (1 - \sum_{k=1}^n w(k)) f(p) + \sum_{k=1}^n w(k) f(k) \quad (2.8)$$

For all the functions above: $F(x)$ is the function to compute weights from the image gradient, x is gradient value of the current pixel in the image, α is a parameter giving an estimation of the missed pixel gradient, k presents the pixel position and w is the kernel weight at k , n is the number of known pixel in the current neighborhood, $f(p)$ is estimated value, $f(k)$ is value of a known pixel in the current neighborhood [8].

Chapter 3

Pre-processing

In this chapter, the main purpose is interpreting the algorithm applied during the pre-processing procedure. At this project, removing the effect of glare point in the picture constitutes the main function of previous processing. The following description will be divided into two parts, one is the introduction of the methods for detect the bright sunlight sections and another is the steps using in this project for removing light reflection area.

Here in this project, the goal is reducing the light reflection only by one taking picture for the object, so the author appeals to the way that take the picture with over-exposure which can decrease the light reflection of the mental-like surface but also introduce some glare points into the result (show an example in Figure 3.1 Comparison of with & without Flash). Then find out and extract the highlight flare points, finally recovery the image (see Figure 3.2 Overflow of Pre-processing).

Figure 3.1 Comparison of with & without Flash

Figure 3.2 Overflow for Pre-processing

3.1 Methods for Detect Glare Area

After taking a picture of the target object with flash, the image is obtained by the application. The first step is detecting the aliasing part which means the glare area in the image.

As we all know, the glare part is the area with high luminance value. So detecting glare area also can be referred to detecting the area whose luminance value beyond the threshold. Here are two ways to do the detection.

3.1.1 HSV Color Space Model

HSV (Hue, Saturation, and Value) color space is based on an intuitive nature of color created by AR Smith in 1978, which also known as hexagonal Hexcone Model. It is corresponding to a cylindrical coordinate system in a conical subset (Figure 3.3 HSV Color Space Model and Figure 3.4 HSV Color Distribution) [13].

Figure 3.3 HSV Color Space Model

The color parameters in the model are: Hue (H) is a fundamental attribute of color, is usually referred to color names, such as red, yellow and so on. With angle measurement, hue ranges from $0^\circ \sim 360^\circ$. Calculated in the counterclockwise direction, red is 0° , the green is 120° , blue for 240° . They are complementary colors are yellow at 60° , cyan at 180° , magenta at 300° . Saturation (S) refers to the purity of color, the higher the value, the more saturated the colors are. Range from 0 to 1. Value (V) means the luminance (L) of target color, taking 0 (black)-255 (white).

Figure 3.4 HSV Color Distribution

The algorithm of RGB transforming to HSV color space is shown in equation (3.1), (3.2) and (3.3) [14]. In the equations, 'max' and 'min' mean the maximum and minimum value of r, g and b.

$$h = \begin{cases} 0^\circ & \text{if } \max = \min \\ 60^\circ \times \frac{g-b}{\max-\min} + 0^\circ, & \text{if } \max = r \text{ and } g \geq b \\ 60^\circ \times \frac{g-b}{\max-\min} + 360^\circ, & \text{if } \max = r \text{ and } g < b \\ 60^\circ \times \frac{b-r}{\max-\min} + 120^\circ, & \text{if } \max = g \\ 60^\circ \times \frac{r-g}{\max-\min} + 240^\circ, & \text{if } \max = b \end{cases} \quad (3.1)$$

$$s = \begin{cases} 0, & \text{if } \max = 0 \\ \frac{\max-\min}{\max} = 1 - \frac{\min}{\max}, & \text{otherwise} \end{cases} \quad (3.2)$$

$$v = \max \quad (3.3)$$

The algorithm of HSV transforming to RGB color space is shown equation (3.4), (3.5), (3.6), (3.7), (3.8) and (3.9).

$$h_i \equiv \left\lfloor \frac{h}{60} \right\rfloor \pmod{6} \quad (3.4)$$

$$f = \frac{h}{60} - h_i \quad (3.5)$$

$$p = v \times (1 - s) \quad (3.6)$$

$$q = v \times (1 - f \times s) \quad (3.7)$$

$$t = v \times (1 - (1 - f) \times s) \quad (3.8)$$

$$(r, g, b) = \begin{cases} (v, t, p), & \text{if } h_i = 0 \\ (q, v, p), & \text{if } h_i = 1 \\ (p, v, t), & \text{if } h_i = 2 \\ (p, q, v), & \text{if } h_i = 3 \\ (t, p, v), & \text{if } h_i = 4 \\ (v, p, q), & \text{if } h_i = 5 \end{cases} \quad (3.9)$$

Use the function `rgb2hsv`, the hue, saturation, and value information will be get and with the luminance value in V component, the program can be easily designed to find out the area that is too bright.

3.1.2 Thresholds for Gray Image

As the Figure 3.5 Code of Threshold for Gray Image shows, the function `rgb2gray` converts the true color image from RGB color space to grayscale image. It removes the color information: hue and saturation, but remains the luminance value ranging from 0 to 255, dark to bright. So with a threshold of 85% of original luminance, the program extracts the partition below the standard and made them into black which means the glare part is kept in gray image BW.

```
gimg = rgb2gray(img); %255=brightest,0=darkest
figure
imshow(gimg);
title 'grey image'

threshold=255*0.85;
BW=gimg;
indices = find(BW<threshold);
BW(indices) = 0;
figure
imshow(BW);
title 'the part that are too bright'
```

Figure 3.5 Code of Threshold for Gray Image

3.2 Light Reflection Removing and Recovery

3.2.1 Dilation

After detecting the glare area of the image, then it should be processed with dilation (as the code shown below Figure 3.6 Code for Dilation). The dilation here means expanding the areas which are detected to be the glare point in the image. This process is to ensure when recovery conduct on the target image, the repair part would not be influenced by the bright place.

```
Matrix = strel('disk',10);
BW=imdilate(BW, Matrix);
figure
imshow(BW);
title 'processed bright part'
```

Figure 3.6 Code for Dilation

In the code, the function `Matrix = strel ( 'disk', 10)` is to create a disk-shaped element with a radius ten. In figure 3.7 Sample of Function `strel` gives an example of `strel` function with radius equal to three. `Matrix` is the region; within who is all value one (white).

Figure 3.7 Sample of Function `strel`

The function `BW = imdilate (BW, Matrix)` dilates the grayscale, binary, or packed binary image `BW` with the structuring element object `Matrix`, which means expanding the boundary area with the pattern of disk-shaped region whose radius is ten pixels. `BW` is logical and `Matrix` is flat element, so `imdilate` performs binary dilation; otherwise, it performs grayscale dilation. Besides `Matrix` here is an array of structuring element objects, `imdilate` performs multiple dilations of the input image,

using each structuring element in succession. After this step the BW becomes the final image with all the part that need to be cancelled from the original image.

3.2.2 Inpainting

Inpainting is the technology that can reconstruct small damaged portions inside images. After obtaining the glare area, the next step is applying inpainting function for image recovery. The following flow chat shows the detail (See in Figure3.8 Overflow of Inpainting Function). Then the corresponding code is shows in Figure 3.9 Code for Inpainting.

Figure 3.8 Overflow of Inpainting Function

```
G = double(img);
BW1(:,:,1)=im2double(BW);
BW1(:,:,2)=im2double(BW);
BW1(:,:,3)=im2double(BW);
G(BW1>0.50) = NaN;
figure
image(uint8(G))
title 'need to recover'

G1=G;
G1(:,:,1) = inpaint_nans(G(:,:,1),2);
G1(:,:,2) = inpaint_nans(G(:,:,2),2);
G1(:,:,3) = inpaint_nans(G(:,:,3),2);
figure
image(uint8(G1))
title 'recover'
```

Figure 3.9 Code for Inpainting

As you can see, the input image is color image, however the detected glare area is in greyscale format, so when clear the glare area, the process need to be conducted from three channels. The method for clearing is setting the glare area in original image to NaN. Then we use a function called `inpaint_nans` to do the neighborhoods based isotropic diffusion to the damaged image. The parameters 2 in the function means uses Del² algorithm, but solving a direct linear system of equations for NaN elements. This method will be the fastest possible for large systems since it uses the sparsest possible system of equations. Not a least squares approach, so it may be least robust to noise on the boundaries of any holes. This method will also be least able to interpolate accurately for smooth surfaces. Extrapolation behavior is linear [10].

Here we give some details about the algorithm used in computing the data values of damaged area from the neighborhood information. All the missing data inside the holes are fitting this situation. It creates a sparse matrix with replication and tiling array of the set parameters. Then use the known neighborhood information to

assimilate the value in the sparse matrix and smooth the boundary. The `inpaint_nans` function here uses a PDE based algorithm. Partial differential equations (PDE) are expressions involving an unknown function in many independent variables and their partial derivatives up to a certain order [12].

In the program, the `inpaint_nans` function firstly finds the NaN components inside the damaged image and put them into `nan_list`. As for `nan_list`, it is a matrix with unrolled index and rolled indexes (row and column information of its position) of every NaN component. Then use Δ^2 method to calculate the value of every NaN component. Here we give a simple example to explain the method [10].

Assume we have a simple gray image $A = [1, 2, 2, 1; 3, \text{NaN}, \text{NaN}, 3; 1, 2, 2, 1]$, which contains two NaN elements need to be recovered from its neighborhood. This is shown in equation (3.10).

$$A = \begin{bmatrix} 1 & 2 & 2 & 1 \\ 3 & \text{NaN} & \text{NaN} & 3 \\ 1 & 2 & 2 & 1 \end{bmatrix} \quad (3.10)$$

For an equally spaced grid, the Laplacian equation (or Poisson's equation of heat conduction at steady state, or for the fickle, Ficke's law of diffusion) would apply. All of these result in the PDE (shown in equation (3.11)):

$$u_{xx} + u_{yy} = 0 \text{ or } \nabla^2 u = 0 \quad (3.11)$$

where u_{xx} is the second partial derivative of u with respect to x , u_{yy} is the second partial with respect to y , and u is damaged point.

Approximating this PDE using finite differences for the partial derivatives, implies that at any node in the grid, we could replace it by the average of its four neighbors.

Thus the two NaN elements generate two linear equations (shown in equation (3.12) and equation (3.13)):

$$A(2, 2) = (A(1, 2) + A(3, 2) + A(2, 1) + A(2, 3)) / 4 \quad (3.12)$$

$$A(2, 3) = (A(1, 3) + A(3, 3) + A(2, 2) + A(2, 4)) / 4 \quad (3.13)$$

Since we know all the parameters but $A(2, 2)$ and $A(2, 3)$, substitute their known values. So the linear equation becomes to (shown in equation (3.14) and equation (3.15)):

$$A(2, 2) = (2 + 2 + 3 + A(2, 3)) / 4 \quad (3.14)$$

$$A(2, 3) = (2 + 2 + A(2, 2) + 3) / 4 \quad (3.15)$$

which equal to equation (3.16) and equation (3.17):

$$4 * A(2, 2) - A(2, 3) = 7 \quad (3.16)$$

$$-A(2, 2) + 4 * A(2, 3) = 7 \quad (3.17)$$

We can solve for the unknowns using equation (3.18):

$$u = [4 \ -1; \ -1 \ 4] \setminus [7; 7] \quad (3.18)$$

and get the result in equation (3.19) and equation (3.20)::

$$A(2, 2) = u(1) = 2.3333; \quad (3.19)$$

$$A(2, 3) = u(2) = 2.3333; \quad (3.20)$$

For an arbitrary number of NaN elements in an array, the above scheme is the whole process for Del² method with a very slick application of sparse linear algebra in Matlab. The sparse matrix is used for solving the calculation problem. The process can be simulating as following code (shown in Figure 3.10 Inpainting Code).

```

L = find((nan_list(:,2) > 1) & (nan_list(:,2) < n));
nl=length(L);
if nl>0
    fda=sparse(repmat(nan_list(L,1),1,3), ...
               repmat(nan_list(L,1),1,3)+repmat([-1 0 1],nl,1), ...
               repmat([1 -2 1],nl,1),n*m,n*m);
else
    fda=spalloc(n*m,n*m,size(nan_list,1)*5);
end

% 2nd partials on column index
L = find((nan_list(:,3) > 1) & (nan_list(:,3) < m));
nl=length(L);
if nl>0
    fda=fda+sparse(repmat(nan_list(L,1),1,3), ...
                   repmat(nan_list(L,1),1,3)+repmat([-n 0 n],nl,1), ...
                   repmat([1 -2 1],nl,1),n*m,n*m);
end

% eliminate knowns
c=fda(:,known_list);
rhs=-fda(:,known_list)*A(known_list);

% and solve...
B=A;
k=nan_list(:,1);
B(k)=fda(k,k)\rhs(k);

```

Figure 3.10 Inpainting Code

Here we give an example of how the inpainting code works. Firstly, we have a 5*3 (n*m, 'n' represents the row and 'm' represents the column) input image A, shown in equation (3.21).

$$A = \begin{bmatrix} 11 & 12 & 13 \\ 21 & NaN & 23 \\ NaN & NaN & NaN \\ 41 & NaN & 43 \\ 51 & 52 & 53 \end{bmatrix} \quad (3.21)$$

So the nan_list and known_list are (shown in equation (3.22) and (3.23)):

$$\text{nan_list} = \begin{bmatrix} 3 & 3 & 1 \\ 7 & 2 & 2 \\ 8 & 3 & 2 \\ 9 & 4 & 2 \\ 13 & 3 & 3 \end{bmatrix} \quad (3.22)$$

$$\text{known_list} = [1, 2, 4, 5, 6, 10, 11, 12, 14, 15]' \quad (3.23)$$

As for the sparse matrix, for first partial nan_list (:,2), the component of sparse (i, j, v, x, y) are repmat (nan_list (L, 1), 1, 3), repmat (nan_list (L, 1), 1, 3) + repmat ([-1 0 1], nl, 1), repmat ([1 -2 1], nl, 1), n*m and n*m. L=5, nl=5, m*n=n*m=15. So each component can be represented as equation (3.24), (3.25), (3.26) and (3.27). Component i and j represent the index of none zero elements in sparse matrix and v indicates the related value of each index. Component x and y describe the size of the matrix.

$$\text{repmat}(\text{nan_list}(L, 1), 1, 3) = \begin{bmatrix} 3 & 3 & 3 \\ 7 & 7 & 7 \\ 8 & 8 & 8 \\ 9 & 9 & 9 \\ 13 & 13 & 13 \end{bmatrix} \quad (3.24)$$

$$\text{repmat}([-1 \ 0 \ 1], \text{nl}, 1) = \begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix} \quad (3.25)$$

$$\text{repmat}(\text{nan_list}(L, 1), 1, 3) + \text{repmat}([-1 \ 0 \ 1], \text{nl}, 1) = \begin{bmatrix} 2 & 3 & 4 \\ 6 & 7 & 8 \\ 7 & 8 & 9 \\ 8 & 9 & 10 \\ 12 & 13 & 14 \end{bmatrix} \quad (3.26)$$

$$\text{repmat}([1 \ -2 \ 1], \text{nl}, 1) = \begin{bmatrix} 1 & -2 & 1 \\ 1 & -2 & 1 \\ 1 & -2 & 1 \\ 1 & -2 & 1 \\ 1 & -2 & 1 \end{bmatrix} \quad (3.27)$$

As shown before, we get the fda of first partial. We do the same thing for the second partial and obtain components $\text{repmat}(\text{nan_list}(L, 1), 1, 3)$, $\text{repmat}(\text{nan_list}(L, 1), 1, 3) + \text{repmat}([-n \ 0 \ n], \text{nl}, 1)$, and $\text{repmat}([1 \ -2 \ 1], \text{nl}, 1)$ as equation (3.28), (3.29), (3.30) and (3.31). Add the fda with that obtained from the first one and get the final fda, shown in equation (3.32).

$$\text{repmat}(\text{nan_list}(L, 1), 1, 3) = \begin{bmatrix} 7 & 8 & 9 \\ 7 & 8 & 9 \\ 7 & 8 & 9 \end{bmatrix} \quad (3.28)$$

$$\text{repmat}([-n \ 0 \ n], \text{nl}, 1) = \begin{bmatrix} -5 & 0 & 5 \\ -5 & 0 & 5 \\ -5 & 0 & 5 \end{bmatrix} \quad (3.29)$$

$$\text{repmat}(\text{nan_list}(L, 1), 1, 3) + \text{repmat}([-n \ 0 \ n], \text{nl}, 1) = \begin{bmatrix} 2 & 7 & 12 \\ 3 & 8 & 13 \\ 4 & 9 & 14 \end{bmatrix} \quad (3.30)$$

$$\text{repmat}([1 \ -2 \ 1], \text{nl}, 1) = \begin{bmatrix} 1 & -2 & 1 \\ 1 & -2 & 1 \\ 1 & -2 & 1 \end{bmatrix} \quad (3.31)$$

$$\text{fda} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & -2 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 & -4 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 1 & -4 & 1 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & -4 & 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & -2 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \quad (3.32)$$

As for the instruction $\text{rhs} = -\text{fda}(:, \text{known_list}) * \text{A}(\text{known_list})$, $\text{fda}(:, \text{known_list})$ means only keeping the columns that exist in the known_list , $\text{A}(\text{known_list})$ is a matrix of the value of known elements of A (shown in equation (3.33)) and the result rhs is shown in equation (3.34).

$$\text{A}(\text{known_list}) = [11, 21, 41, 51, 12, 52, 13, 23, 43, 53]' \quad (3.33)$$

$$\text{rhs} = [0, 0, -62, 0, 0, 0, -56, 0, -136, 0, 0, 0, -66, 0, 0]' \quad (3.34)$$

After these, command $\text{B}(\text{k}) = \text{fda}(\text{k}, \text{k}) \setminus \text{rhs}(\text{k})$ calculates the final result of the value of each NaN element. So we get the recovered image B , shown in equation (3.35).

$$\mathbf{B} = \begin{bmatrix} 11 & 12 & 13 \\ 21 & 22 & 23 \\ 31 & 32 & 33 \\ 41 & 42 & 43 \\ 51 & 52 & 53 \end{bmatrix} \tag{3.35}$$

Chapter 4

Android Application

In this chapter, two topics are explained. One is how to perform WeChat API in your application and another is about the main UI used in this project.

4.1 WeChat API

Before starting establishing an application workplace, the WeChat API is needed first. This is the significant point for this application, because all the realization of functions is based on this service. The question comes: how to get the API? Firstly, go to the website: <http://pr.weixin.qq.com> and register for an account. Then at the developer options (shown in Figure 4.1 Developer page), register for a new App (shown in Figure 4.2 Register Icar App).

Figure 4.1 Developer page

Register my App

We will give you the auditing results in 3 workdays after the Apps being submitted. During the time, you can modify your Apps' information.

App Name

Icar

App Classification

Entertainment

App Description

It is an app designed for all the clients to recognize car logo directly by taking a photo when they meet with the car they want to get more information about.

Description can not be empty.

Website

Estimated Volume of Service

1000

Submit

Back

Figure 4.2 Register Icar App

After successfully register the App (as the Figure 4.3 Register Successfully presents) wait for three to five working days as the application goes through the authority check. Then go back to the Picture Management page, click on my app; at the detail page, collect the application authorization code as Figure 4.4 Application Authorization Code showed. This code is extremely significant, because when the API installed in the application, the image searching and recognition service will need this code to active. In the application, an activity called preInitImg is used for calling the code to do the searching and recognition work. The following gives more details for this action.

```
private void preInitImg () {  
    ImgSearcher.shareInstance().setListener(this);  
    mInitSucc=ImgSearcher.shareInstance().init(this,  
screKey);}
```

Figure 4.3 Register Successfully

Figure 4.4 Application Authorization Code

Besides all the things above, there is also a picture library option in the picture management page (seeing in Figure 4.5 Picture Library). In the library, you need to upload all the pictures that custom might use for comparison; for example, the logo the all the automobile brand, the brand in different place the different background colors. This should be a very big database which in the future might have the ability to learn from the previous history, which means every time the user use the application to do the comparison, as long as the result is correctly matched, the taken picture can be added into the database to make the future comparison more accurate.

Figure 4.5 Picture Library

4.2 Main Application

The first step of main application is to establish a new workspace and application or import existing projects into the workspace (show in Figure 4.6 Establishing Main UI). At this step, the decision of which version of android operation system should be made as well as the name and logo of this application.

Figure 4.6 Establishing Main UI

Then after install all the android SDK in the eclipse with the corresponding android operation system version and building tools. Also some project configuration need to build for building the application. Firstly, a file named libWXPrNet.so need to be added in the content under libs\armeabi. Then, introduce Jar package named

wximg.jar: select "Add External JARs" under the engineering properties -> Java Build Path -> Libraries. After all the project configuration, there is also a AndroidManifest.xml setting. Some necessary permissions support needs to be added into the workspace (like Figure 4.7 Permission Support shows).

```
<uses-permission  
android:name="android.permission.ACCESS_NETWORK_STATE"></uses-permission>  
<uses-permission android:name="android.permission.INTERNET"></uses-permission>  
<uses-permission android:name="android.permission.ACCESS_WIFI_STATE"></uses-permission>  
<uses-permission android:name="android.permission.CHANGE_WIFI_STATE"></uses-permission>  
<uses-permission android:name="android.permission.READ_PHONE_STATE"></uses-permission>  
<uses-permission android:name="android.permission.RECORD_AUDIO"></uses-permission>
```

Figure 4.7 Permission Support

Inside the WeChat image recognition SDK there is a bunch of API that can be used in the application.

First is the initialization:

```
ImgSearcher.shareInstance ().setListener (this);  
if (ImgSearcher.shareInstance ().init (this, screKey) != 0) { //Failed to initialize}
```

Second is Starting to recognition:

```
ImgSearcher.shareInstance ().start (img);
```

Notice: img is the content of image byte[]. After starting all the stages in the application obtain and callback from sentence

```
public void onGetState (ImgSearcherState state)
```

The third one is cancel recognition:

```
ImgSearcher.shareInstance ().cancel();
```


If the function returns non-zero value, it means the processing is cancel, or else use the sentence in second API's notice to get the stage of this API.

The fourth is ImgListener, and rewrite the function:

```
@Override
public void onGetError(int arg0) { // TODO Auto-generated method stub }

@Override
public void onGetResult(ImgResult result) { // TODO Auto-generated method stub
    String mResUrl;
    String mResMD5;
    String mResPicDesc;
    if (result != null) {
        if (1 == result.ret && result.res != null) {
            int resSize = result.res.size();
            for (int i = 0; i < resSize; ++i) {
                ImgResult.Result res = (ImgResult.Result) result.res.get(i);
                if (res != null) {
                    mResMD5 = res.md5;
                    mResUrl = res.url;
                    mResPicDesc = res.picDesc; }
            }
        }
    }
}

@Override
public void onGetState(ImgSearcherState state) {
    // TODO Auto-generated method stub }
```

The fifth API is release system source:

`ImgSearcher.shareInstance ().destroy ();`

Notice: when apply this function; it should be made sure that the cancel function is finished.

The main programming user interfaces are shown in below figures; Figure 4.8 Main Activity, Figure 4.9 Searching Activity, Figure 4.10 Result Activity. Use almost all the functions above and also others for establishing the application.

Figure 4.8 Main Activity

After the programming part finished, the coder needs to test the application for its feasibility. Inside eclipse, there are two ways for running the program; first is by a virtual device; another is running on a real device, an android phone.

Figure 4.9 Searching Activity

Figure 4.10 Result Activity

When testing the complete application on a virtual device, you need firstly build one by yourself in the eclipse, like Figure 4.11 Create Virtual Device showing. Give a name and choose for the certain type of device as well as the version of CPU, skin and keyboard. What needs to pay attention to is the size of RAM and internal storage. If the RAM is too big, the computer might not be able to run the application for its hardware cannot tolerate such work; or else it is too small, it will take a long time for the virtual device to load the application.

Figure 4.11 Create Virtual Device

Chapter 5

Results and Discussion

In this chapter there are two parts of content going to be discussed. First one is the result of pre-processing for removing the glare area in the image. Second one is the result of the established application.

5.1 Pre-processing Results

5.1.1 Result

In this part, the results of removing the glare area are shown in steps with the processing approach. First is the over-exposure input image as Figure 5.1 Original Image shows. As you can see from the following, at the right side of this picture, there exists a highlight area that might affect the recognition result. Also in the central and left part, a lot of glare points introduce aliasing to the image which reduce the accuracy of image searching and recognition consequence.

Figure 5.1 Original Image

Then Figure 5.2 Grayscale Image and Figure 5.3 High Brightness Section give the result of detect the highlight areas. Change the color image to grayscale and keep all the parts whose luminance are exceed the threshold which is the target highlight area that wanted in this move. The threshold is 0.85×255 which means that inside the image all the points whose luminance is higher than 85 percent of the biggest value are select as the masking.

Figure 5.2 Grayscale Image

Figure 5.3 High Brightness Section

After the detection, dilation of high brightness section is conducted according to the procedure in chapter 3. Using disk shaped points with radius of 5 pixels to dilate the area so that the recovery is in a better quality. Figure 5.4 Dilation of High Brightness Section shows the consequence. Figure 5.5 Damaged Image illustrates the original image without the damaged parts.

Figure 5.4 Dilation of High Brightness Section

Figure 5.5 Damaged Image

Finally, Figure 5.6 Recovered Image is the result for inpainting. Use the neighborhood pixel values to approximate the similar values for damaged area so that after removing the glare part the whole image looks smooth.

Figure 5.6 Recovered Image

5.1.2 Discussions

In this part, the author proposes a comparison of consequences using different parameters for luminance detect threshold and dilation mask radius. Respectively, in Figure 5.7, Figure 5.8 and Figure 5.9 gives the recovery image with parameters setting as: threshold = $0.85 * 255$, radius = 5 pixels; threshold = $0.9 * 255$, radius = 5 pixels; threshold = $0.85 * 255$, radius = 10 pixels.

As you can see in the consequences, figure 5.7 with parameter setting as threshold = $0.85 * 255$, radius = 5 pixels get the best recovery result. The arrows in figure 5.8 point that the two glare points are not reduce when threshold value is too high. As well as the arrows in figure 5.9 illustrate that if the dilation radius is too large, it will

influence the recovery of inpainting. So the key point for this process is to choose a suitable set of parameters.

Figure 5.7 Threshold= 0.85×255 , Radius=5pixels

Figure 5.9 Threshold= 0.9×255 , Radius=5pixels

Figure 5.8 Threshold=0.85*255, Radius=10pixels

5.2 Application Results

In this sub-chapter, the result is going to be the exhibition of application. It is divided into two parts, showing on virtual device as well as android device.

Firstly, the Figure 5.9 Application in Virtual Device and Figure 5.10 Main Page in Virtual Device show the logo page and user interface on virtual device established in chapter 4 on eclipse. Because the virtual device on computer have no real random-access memory (RAM), the access through album is impossible. Also the camera model can never be used for missing system camera and memory. So this virtual device can only demonstrate the main page without accessing to any function.

Figure 5.9 Application in Virtual Device

Figure 5.10 Main Page in Virtual Device

Although the virtual device cannot achieve the function of the application, a real android device is absolutely possible to accomplish this task. In the following twelve figures, the demonstration is illustrating completely.

Figure 5.11 Application in Smartphone is the logo page in smartphone. Figure 5.12 Main Page is the main UI of this application. The arrow in this figure shows that click on this camera button for using this function of taking pictures.

Figure 5.11
Application in Smartphone

Figure 5.12
Main Page

Then in Figure 5.13 Camera Model and Figure 5.14 Taken Result, it shows the processing of taking a photo of automobile logo from a wheel by system camera. After capturing, the picture will rotate automatically to fit the right direction (there is a limitation that only angle value with multiples of 90 degree can successfully

achieve the auto-rotating). When you confirm the photo in this page, click on the checkmark for next step.

Figure 5.13
Camera Model

Figure 5.14
Taken Result

After confirming the photo been captured, the application will jump to searching page (shown in Figure 5.15 Searching Page). The searching speed depends on the speed of your internet, the size of uploading image and the size of your library. There is also a cancel button if you want to give up current searching.

When searching is finished, it will self-link to the result page (shown in Figure 5.16 Result Page for Image from Camera) without any action by users. In this page, a compressed picture of the searching target is exhibiting at the top. Then follow by the result of logo's name (as the yellow arrow pointing to) and the homepage link button.

Figure 5.15

Searching Page

Figure 5.16

Result Page for Image from Camera

Finally, after clicking on the homepage button, the application will trigger an instruction to jump out to system browser for detail information of the homepage for the result logo (shown in Figure 5.17 Homepage Link for Image from Camera).

Besides the above procedure, the customer can also click on the album button (pointed by the yellow arrow) at the main page for choosing an exist picture in their smartphone for recognition (as Figure 5.18 Main Page for Album Model shows).

Figure 5.17

Homepage Link for Image from Camera

Figure 5.18

Main Page for Album Model

Figure 5.19 Album Model and Figure 5.20 Select Image show the approach to the system image. After choosing the image in your smartphone, the application will jump to searching page, similar as Figure 5.15 Searching Page showing, to upload the image to server and searching the best matched result.

Figure 5.19
Album Model

Figure 5.20
Select Image

Finally, after searching is the result page (shown in Figure 5.21 Result for Image from Album) and homepage linking of the result (shown in Figure 5.22 Homepage Link for Image from Album) just like the camera mode. Click on the return button at the android device it can go back to main page for other recognition.

Figure 5.21

Result for Image from Album

Figure 5.22

Homepage Link for Image from Album

Chapter 6

Conclusions and Future work

The description of this chapter is mainly about the summary of the whole project and some recommendations for future work that are related to this project and application.

6.1 Conclusions

This project completes the construction of android application named iCar by eclipse with WeChat image recognition API. The main function of this application is to search and recognize the automobile logo that are taken real time by system camera or obtained from the album of smartphone. In the server port, the uploaded image is compared with the data library for similarity. Once the picture is similar to the target one, the MD5 code of this library image will be return to the android phone. The MD5 codes are sorted by the logo type. So the result in the application is changed into the name of target logo and the homepage link about further information.

This project also provided an approach to reduce even remove the reflection in the image. Take the image with over exposure by the camera, most of the reflection will be covered but glare areas will be introduce to the image. So detect the glare area, and then cancel them and recovery with inpainting function are used to remove glare areas.

6.2 Recommendation for Future Work

This project aims to develop an android application that helps customers to recognize car brand automatically. Although the testing application successfully built and work well so far, some problem are still existing in and can be promoted in the future. Among the whole project and some investigator's advice, the main problem is in the following two aspects.

First is the abundance of automobile image library. The requirement of making this application works accurately and universal applicably leads to the result that establishing a full-scale image library which contains all kinds of auto brand logos with every possible background. So it is better that the application can link its data library to big searching engine database for resource. Also, the image library can be built with learning function which means every time with new input the database renew itself so that the result searching next is more accurate.

Second is the enhancement of commercial meaning of this application. Due to the limitation of functions of this application, in future the developer can add additional functionality with market value such as giving more detailed information of the searching result, setting up advertising bars at bottom with the concept of big data for the interest of user or built some link to the electricity supplier items that related to the searching result.

References

- [1] Baidu Encyclopedia, "Android", 2015, [Online Resources]
http://baike.baidu.com/link?url=P8UYpgTi2uQnmojA_OtKCnDWnmYBYZShAhlncStF_v7Dj8dYDum0-yuDiQoM9l_pDQcLlMbxNvg4LUpCNuabZzey9r-QaToflEzNqna9j4lbgalpgaRdN4RlTnCxxk5hqafyrAl4eybKxeDQElvVejD3S49-hEX3wxyVA5CqiOMG
- [2] Dusan Psotny, Removing lens flare from digital photographs. Charles University in Pargue Faculty of Mathematics and Physics, Department of Software and Computer Science Education. 2009, pp:7-8
- [3] Farid H, Adelson E H. Separating reflections from images by use of independent component analysis [J]. JOSA A, 1999, 16(9): 2136-2145.
- [4] Agrawal A, Raskar R, Nayar S K, et al. Removing photography artifacts using gradient projection and flash-exposure sampling [C]//ACM Transactions on Graphics (TOG). ACM, 2005, 24(3): 828-835.
- [5] Kong N, Tai Y W, Shin J S. A Physically-Based Approach to Reflection Separation: From Physical Modeling to Constrained Optimization [J]. Pattern Analysis and Machine Intelligence, IEEE Transactions on, 2014, 36(2): 209-221.
- [6] Al-Azzawi A K, Saripan M I, Jantan A, et al. A review of wave-net identical learning & filling-in in a decomposition space of (JPG-JPEG) sampled images [J]. Artificial Intelligence Review, 2010, 34(4): 309-342.
- [7] Telea A. An image inpainting technique based on the fast marching method [J]. Journal of graphics tools, 2004, 9(1): 23-34.
- [8] Noori H, Saryazdi S, Nezamabadi-Pour H. A bilateral image inpainting [J]. IJST Trans. Electr. Eng, 2011, 35(E2): 95-108.
- [9] MathWorks, R2015a Documentiom, "imdilate" 2015 [Online Resources]
<http://www.mathworks.com/help/images/ref/imdilate.html?searchHighlight=>

imdilate&nocookie=true.

- [10] Matlab Central, File Exchange, “inpaint_nans”, 2012 [Online Resources]
<http://www.mathworks.com/matlabcentral/fileexchange/4551-inpaint-nans>.
- [11] Schönlieb C B. Applying modern PDE techniques to digital image restoration [J]. Mathworks Newsletter. Available at, 2012.
- [12] Schönlieb C B. Modern PDE Techniques for Image Inpainting. Doctor thesis. University of Cambridge. 2009.
- [13] Blog of Zhi Xingzhi, “RGB and HSV color model”, 2012 [Online Resource]
http://blog.csdn.net/liuying_1001/article/details/7363353
- [14] Wikipedia, “HSL and HSV”, 2015 [Online Resource]
http://en.wikipedia.org/wiki/HSL_and_HSV