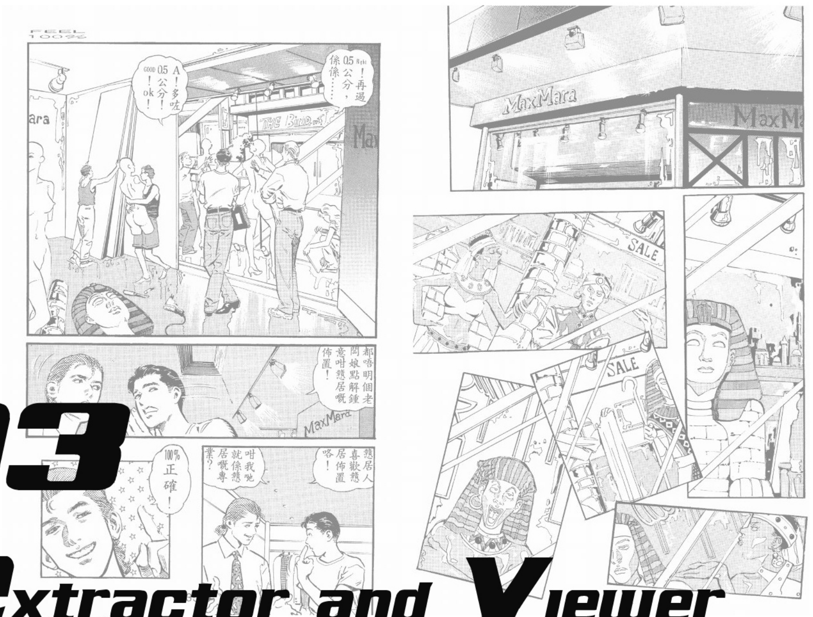
Supervised by
Prof. Michael R. Lyu



LYU1003

Comic Panel Extractor and Viewer for Enhancing Accessibility on iOS4

他故意在已吹得順直的頭髮上用手撥亂一下。
100000分貝！噪音超過了國際認可的標準。
亦是能令許樂起床的指定標準。
33

SUNG Siu Hang Aaron
1008610481
CHEUNG Kam Shun
1008619982

CUHK CSE

Department of Computer Science and Engineering
The Chinese University of Hong Kong

2010-2011
Final Year Project Report (1st Term)

LYU1003

Comic Viewer for iPhone

CHEUNG Kam Shun
1008619982

kscheun8@cse.cuhk.edu.hk

SUNG Siu Hang Aaron
1008610481

shsung8@cse.cuhk.edu.hk

Supervised by

Prof. Michael R. Lyu

Abstract

This report covers our study and progress in image processing for comic panel extractor in this semester.

It begins with an introduction about our project. We will give some background information about comic reading on mobile device, the motivation, the objective of our final year project as well as development environment related information.

It is then followed by a part for background researches. We will first introduce several approaches that are proposed for similar system. Then it is followed by the analysis of some common or essential algorithms for image processing that might need, e.g. Watershed Segmentation, etc.

For the product design part, we will give an overview of our system architecture. We will go through every assumption that we have made for those specific parts, as well as briefly introducing each component.

After this, we will be focus on our development progress as well as the experimental results, difficulties we faced and current limitations. For the progress part, we will talk about what we have done before as well as what we can demonstrate for the first half year.

Last but not least, we will introduce our future plan, including our next task and our ultimate goal.

Table of Contents

Chapter 1 Introduction

In this chapter, we will give some background information of this topic as well as the project motivation and objective. Some development environment related information will be given in this chapter also.

This chapter is constructed as following list:

- Background Information
- Motivation
- Objective
- Development Environment

1.1 Background Information

Reading comic is a hobby to lots of Hong Kong teenagers, especially those who play computer games. However, the most common way in Hong Kong that people get their comics is not to buy them, but to view a softcopy of the comic.

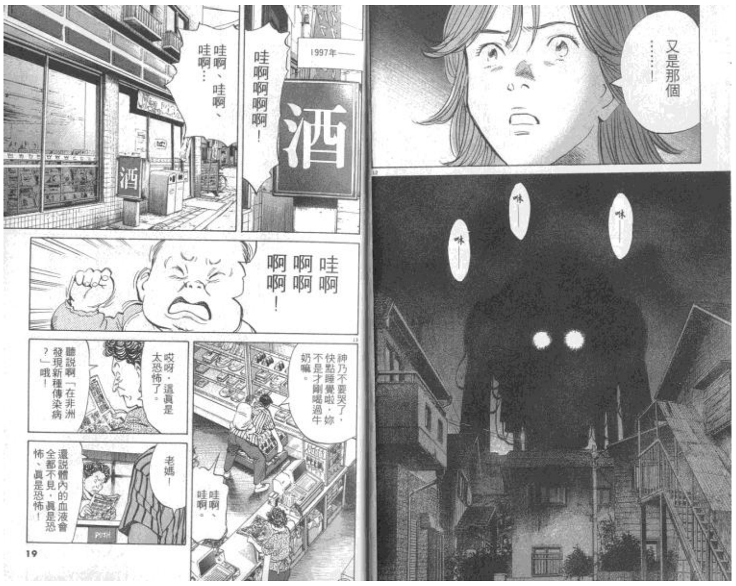

又是那個……！
咻——
咻——
咻——
1997年——
哇啊啊啊啊！
酒
哇啊、哇啊、哇啊…
酒
哇啊啊啊啊啊！
神乃不要哭了，快點睡覺啦，妳不是才剛喝過牛奶嘛。
哎呀，這真是太恐怖了。
聽說啊「在非洲發現新種傳染病？」哦！
老媽！
哇啊、哇啊。
還說體內的血液會全都不見，真是恐怖、真是恐怖！
19

Figure 1.1 - A sample page of a famous Japanese comic

There are usually two ways to view the softcopy of a comic. The first way is to visit an online comic website. There are plenty of online comic services provided by local websites or websites of the Mainland, e.g. Dm5, Manmee, kukudm, 8comic, etc. Another way is through online forums. There are also lots of online forums that provide free comic download, e.g. 2000FUN, KTXP, HKORZ, etc. In this project as well as the rest of this report, when we talk about comics, we mean downloaded comics, which are scanned pictures.

1.2 Motivation

Most people read comics on their PCs in the past few years. However, with the prevalence of mobile devices nowadays, more and more people read comics on these devices. Tablet PCs, especially iPad, successfully open a market for mobile comics.

The success of iPad comic raises a question – can we have smart phone comic? Smart phone comic then became a hot issue to the comic industry as well as many Apps developers. Due to the dimensions of the iPad, the page size is roughly 75%-80% of an actual comic book page, this is acceptable to most comic reader. However, when it comes to smart phone comic, screen size becomes a problem. There exist some comic Apps on the market. In general, there are two kinds of these Apps.

The first kind is usually produced by the comic industry. It has finely tuned the user interface, the comic panel size as well as panel sequence for comic reading on a specific device. The screen size problem has then been specifically handled. Hence, it provides a nice experience to the user. However, the main problem of this kind of Apps is that, the comic reader usually bundled with a specific comic series and cannot support comics from the third party.

The other kind of comic readers usually target on scanned pictures. However, most of them act as a normal image viewer only. Due to the screen limitation of the smart phones, reading traditional page-sized comic images through these Apps would be troublesome because of the frequent manual scrolling and zooming.

In the following table, we will compare the comic reading platforms in detail.

Platform	Advantage	Disadvantage
Traditional PC	● Provide easiest way to get the comic resource	● Default image viewer is not powerful enough ● Lack of mobility
Tablet PC	● Higher mobility comparing to PC ● Similar page size to original comic	● Need extra support to get and manage the comic resource
Smart phone	● Highest mobility ● Highest market share	● Screen size too small ● Need extra support to get and manage the comic resource

Table 1.1 – Comparison among comic reading platforms

An existing comic reader either acts as no more than a picture viewer or bundles with limited comic resource. Considering the considerable market share and the high mobility of smart phone, we believe it has the potential to share the comic market with tablet PC.

In order to achieve this, the first question is how we can make use of the existing comic resource on the Net. Therefore, we will implement a comic panel extractor and viewer accepting scanned pictures of existing comic resource on the Net as input and generating suggested reading sequence automatically.

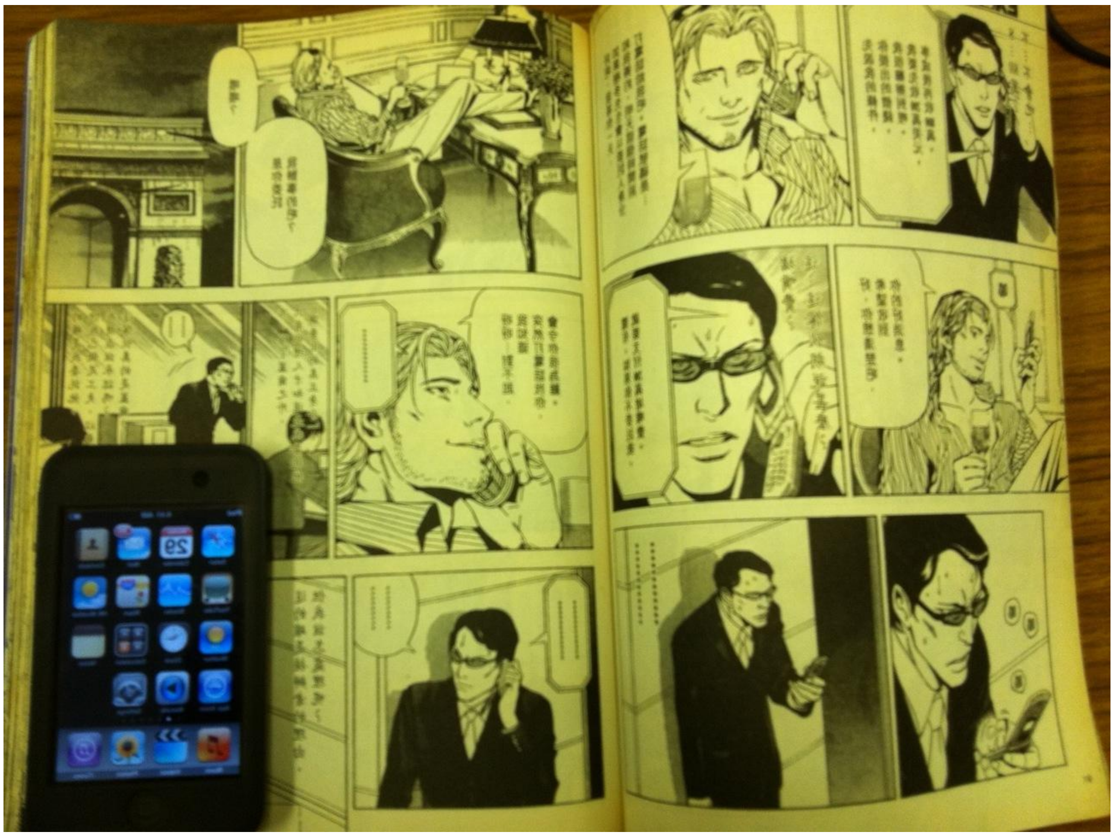


Figure 1.2 - Size Comparison. The screen size of a cell phone is only about ¼ of a traditional comic page

1.3 Objective

The objective of our project is to:

- Analyze the existing image processing algorithms for comic panel extraction,
- Implement comic panel extraction and generate suggested reading sequence and,
- Implement a simple comic viewer for reading the extracted panel in suggested reading sequence.

1.4 Development Environment

Content	Environment & Tool
Algorithm analysis	● Visual Studio 2008, Microsoft Windows ● C with OpenCV
Viewer implementation	● Xcode, Mac OS, iOS4.1 ● Objective C

Table 1.2 – Development Environment

Most of the algorithm investigation are carried out on the PC environment first as this platform has less restriction and the debugging is much more convenient.

For the implementation of the Viewer App, most of the works are done on a Mac and the testing part is carried out on the iOS.

1.5 Runtime Environment

The iOS developed by Apple Inc. is chose as the target platform of the project. iOS is the operating system of iPhone, iPad and iPod touch. The development on iPad is a bit different from others so the development on iPad will not be focused. Moreover, the screen size of iPad is much larger, so a specific viewer on iPad is not so necessary.

The first version of the system released on 2007. Originally it is called the "iPhone uses mac OS X". It runs on ARM family processor and is build on top of the Unix-like kernel Darwin. There are four abstraction layers: the Core OS layer, the Core Services layer, the Media layer, and the Cocoa Touch layer.

Figure 1.3 - Home Screen Capture of iOS

The main reason to choose iOS is that iOS is mature in terms of both amount of users and development environment.

Multi-tasking is a very important new feature to iOS that is beneficial to every developer. Computation power also makes progress. With multi-tasking as well as the more advanced computation power, more and more complicated calculations can be done on iPhone. For instance, iPhone might be treated as an I/O device and it might need a PC to be the server for complicated calculations. This situation will apparently change after iOS4.0 came to the world.

iOS is a popular mobile operating system nowadays. By a survey from Nielsen, iOS owns the second largest market share on the smart phone market, following the RIM developed by Blackberry. On top of it, there is an increasing trend on the number of user of iPhone. Besides, not only iPhone uses iOS system, the iPod touch is also able to run the iOS. Therefore the total number of iOS distribution is much higher than the statistic.

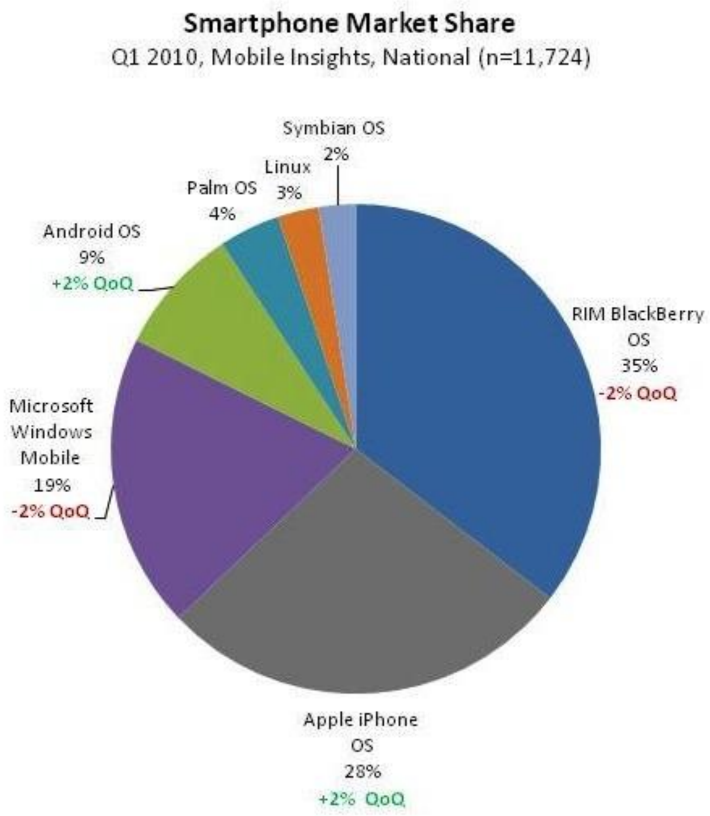


Figure 1.4 - Market Share Statistics of Smartphone from Nielsen[1]

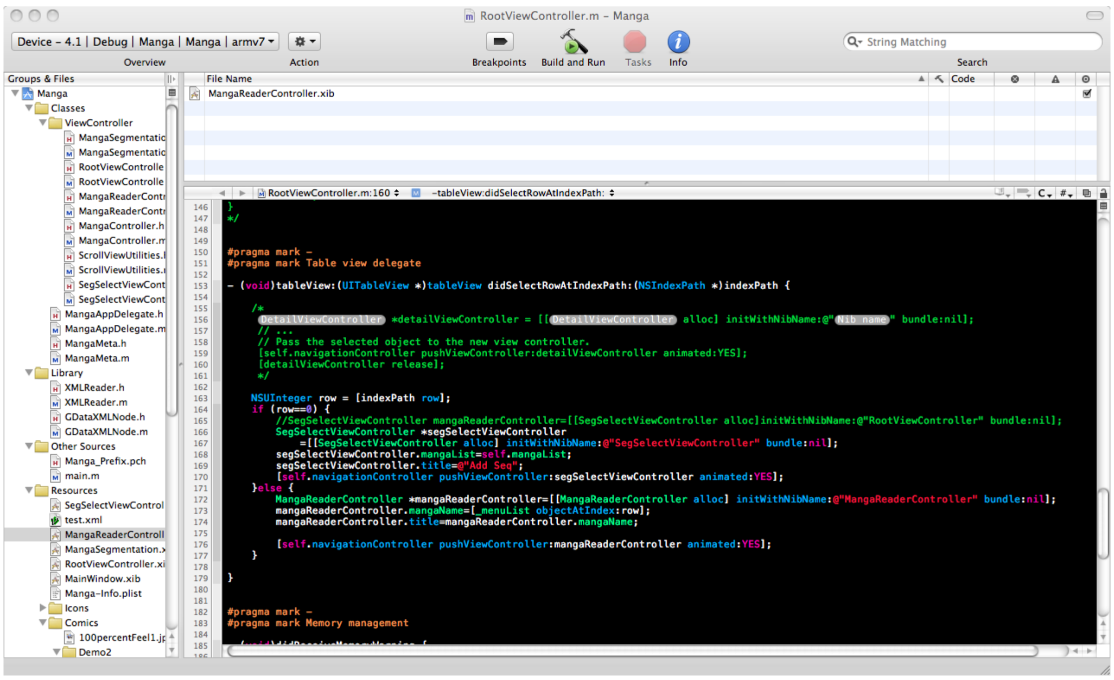
Apart from it, the development environment of iOS is also very mature. The IDE, XCode has been developed for a long time since the selling of Mac OS X. The debugger is the infamous GDB. The base framework, Cocca Touch, provided many useful functions and APIs. Moreover, a lot of 3rd party library framework has been developed. One important example is the OpenCV ported on the iOS. This facilitates the development process a lot.

Also, the documentation supported by Apple Inc. is sufficient. All the documents can be found on the Apple Development Connection

[1]http://big5.xinhuanet.com/gate/big5/news.xinhuanet.com/eworld/2010-06/07/c_12189124.htm

website and it is free of charge.

Last but not least, the hardware platform of iOS devices are unified and standardized. It is especially important for the resolution of the screen. The resolution of non-iPad iOS device is either 480x320 or 960x640. 960x640 is a double of 480x320, that means the layout can remain the same in all kind of non-iPad iOS devices. This can let the developer to focus more on the functionality instead of the compatibility.


RootViewController.m – Manga
Device – 4.1 | Debug | Manga | Manga | armv7
Overview
Action
Breakpoints
Build and Run
Tasks
Info
String Matching
Search
Groups & Files
File Name
Code
MangaReaderController.xib
RootViewController.m:160
–tableView:didSelectRowAtIndexPath:

```
}
*/


#pragma mark -
#pragma mark Table view delegate

- (void)tableView:(UITableView *)tableView didSelectRowAtIndexPath:(NSIndexPath *)indexPath {

    /*
     DetailViewController *detailViewController = [[DetailViewController alloc] initWithNibName:@"Nib name" bundle:nil];
     // ...
     // Pass the selected object to the new view controller.
     [self.navigationController pushViewController:detailViewController animated:YES];
     [detailViewController release];
     */

    NSUInteger row = [indexPath row];
    if (row==0) {
        //SegSelectViewController mangaReaderController=[[SegSelectViewController alloc]initWithNibName:@"RootViewController" bundle:nil];
        SegSelectViewController *segSelectViewController
            =[[SegSelectViewController alloc] initWithNibName:@"SegSelectViewController" bundle:nil];
        segSelectViewController.mangaList=self.mangaList;
        segSelectViewController.title=@"Add Seq";
        [self.navigationController pushViewController:segSelectViewController animated:YES];
    }else {
        MangaReaderController *mangaReaderController=[[MangaReaderController alloc] initWithNibName:@"MangaReaderController" bundle:nil];
        mangaReaderController.mangaName=[_menuList objectAtIndex:row];
        mangaReaderController.title=mangaReaderController.mangaName;

        [self.navigationController pushViewController:mangaReaderController animated:YES];
    }

}


#pragma mark -
#pragma mark Memory management
```

Figure 1.5 - Screen Capture of the XCode IDE

Chapter 2 Background Researches

In this chapter, we will describe our background researches as the following list:

- Different Existing Approaches
  - Multi-pass Approach
  - Recursion Approach
- Commonly Used Functions
  - Contour Finding
  - Hough Line Finding
  - Watershed Segmentation
  - Flood Fill

2.1 Different Existing Approaches

Due to the popularity of comic reading on mobile devices, the problem has been investigated by others before. There are two main approaches proposed by current researches. The first is multi-pass approach while the other one is recursion method.

In the following paragraphs, we will talk about these two approaches.

2.1.1 Multi-pass Approach

Multi-pass approach involves segmentation algorithm with domain specific pre-processing and post-processing[2]. The following list shows the main steps of one example of panel extraction:

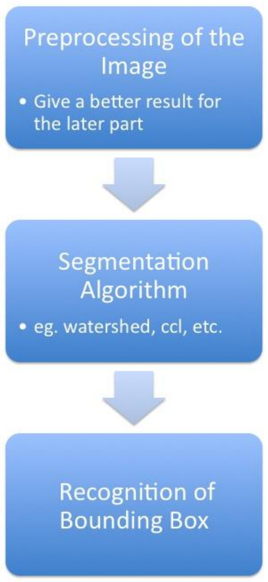

Preprocessing of the Image
• Give a better result for the later part
Segmentation Algorithm
• eg. watershed, ccl, etc.
Recognition of Bounding Box

Figure 2.1 – Multi-pass Approach Flow Chart

i. Convert image to grayscale,
ii. Threshold on white for capturing the whole background,
iii. Fill the background to black,
iv. Compute the different with the previous image,
v. Perform Watershed Segmentation on the binary image,
vi. Compute the bounding boxes covering each detected region,
vii. Remove small region and merge highly overlapping regions.

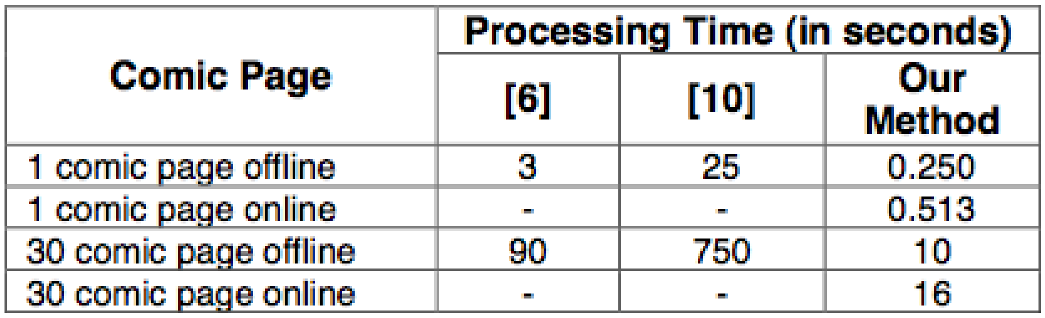
The first 4 steps are standard image processing primitives that ensure the image is under best condition for segmentation. Step vii is usually needed to reduce error. Note that the above algorithm assumes the panels are disjoint and the background color is white.

Besides the use of watershed as the segmentation, a modified connected component labeling algorithm[3,4] is also proposed as a segmentation algorithm.

In general, the multi-pass approaches are faster in speed. It is due to the heuristic division line detection in recursion

[2] C. Ponsard, V. Fries - *Enhancing the Accessibility for All of Digital Comic Books*
[3] F. Chang, C-J. Chen, and C-J. Lu. —*A Linear-Time Component-Labeling Algorithm Using Contour Tracing Technique",* Computer Vision and Image Understanding, 93(2):pp. 206-220, 2004.
[4] K. Arai & H. Tolle - *Automatic E-Comic Content Adaptation*

approaches that will be discussed later. One of the support evidences is the experiment taken by K. Arai & H. Tolle:



Comic Page	Processing Time (in seconds)		
	[6]	[10]	Our Method
1 comic page offline	3	25	0.250
1 comic page online	-	-	0.513
30 comic page offline	90	750	10
30 comic page online	-	-	16

Figure 2.2 – Processing time of the recursion approach

Where the "Our method" means the modified CCL and "[10]" is the method mentioned in the paper of H.C. Chung *et el.*[4]

For mobile devices, the different in performance is significant.

2.1.2 Recursion Approach

In this approach, uniform color stripes are first identified and used as separators to segment the color comic.

In this approach, uniform color stripes are first identified and used as separators to segment the color comic. A line can be represented as follow:

$$\rho = x\cos\theta + y\sin\theta$$

or

$$ax+by+c=0$$

Then different value of {ρ, θ} or {a, b, c} are tired to get a probability of being a stripe. If the value is higher than the threshold, then it will be treated as a stripe.

After that, a page will be segmented into sub-regions in a recursive manner. Panels are recognized as the sub-regions that cannot be further segmented. The structure of the panels can be determined through the panel extraction process which turns out to give us a suggested reading sequence.

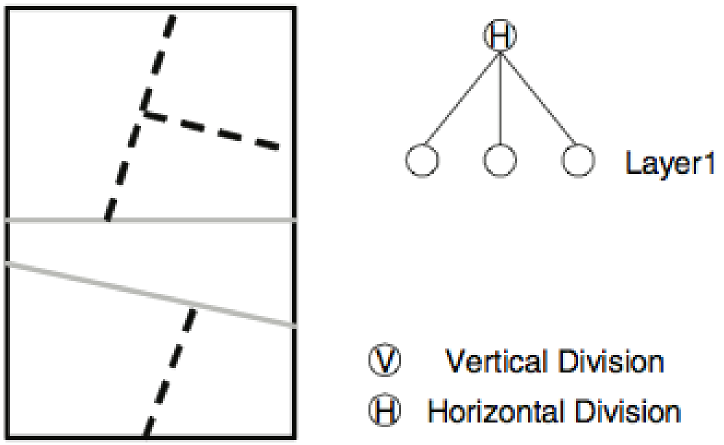


Figure 3: The structure of a comics page at first step (left) and the tree structure of it (right).

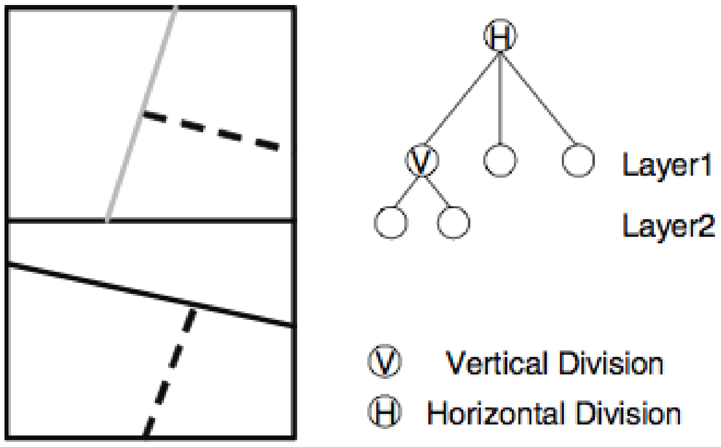


Figure 4: The structure of a comics page at second step (left) and the tree structure of it (right).

Figure 2.3 - Generation of the reading sequence with extracted panels simultaneously.[5]

The following flow chart illustrates this approach in a clearer way:

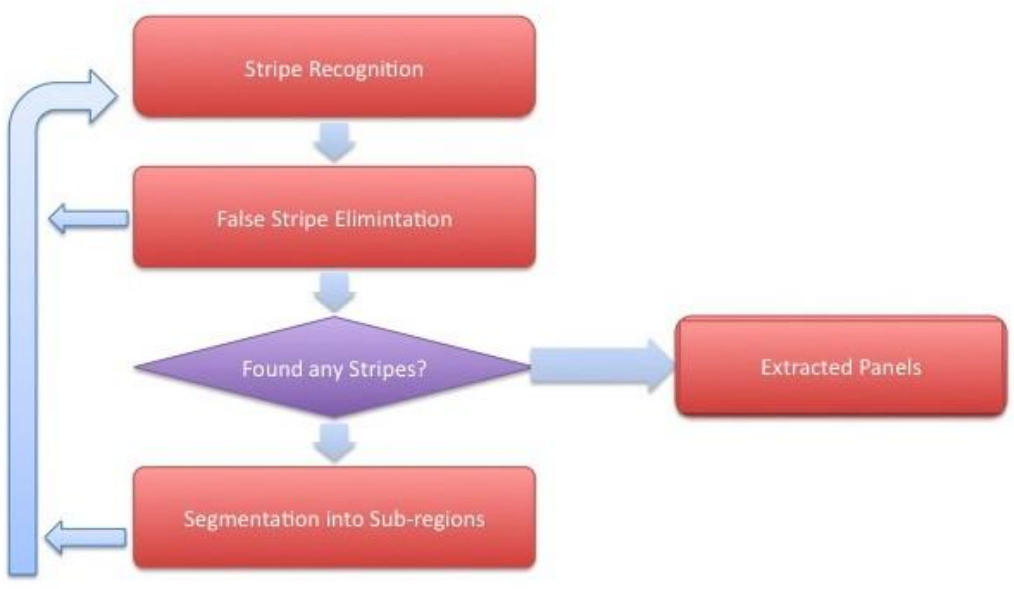
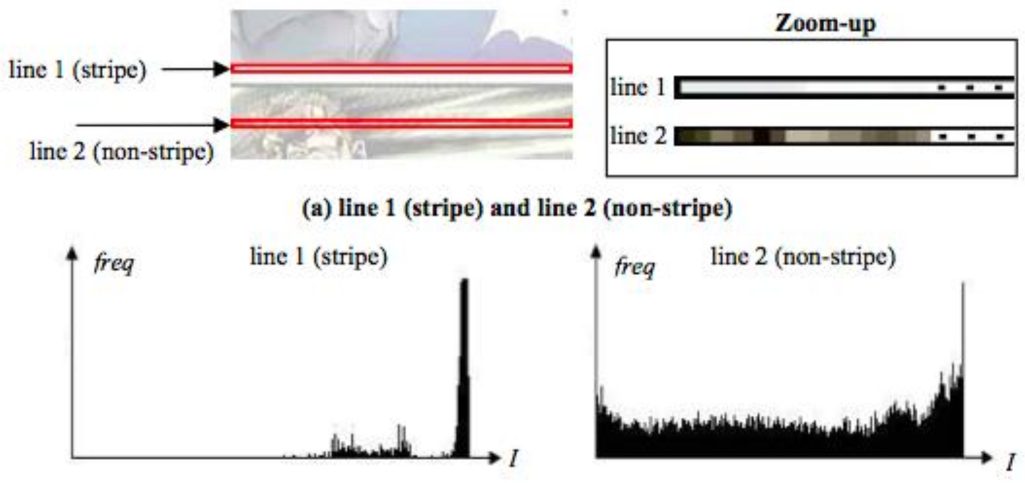


Figure 2.4 – Flow chart of the recursion approach

Two methods of stripe detections are proposed. The first one is by calculating the histogram[5] while the second one is calculate the density gradient[6].

From the research of Chung[5], the histogram of a stripe will have a sharp peak comparing to non-stripe.

[5] T. Tanaka et el., *Layout Analysis of Tree-Structured Scene Frames in Comic Images*, IJCAI'07 Proceedings of the 20th international joint conference on Artifical intelligence
[6] H.C. Chung, H. Leung & T.Komura, "*Automatic Panel Extraction of Color Comic Images*," Advances Multimedia Inform. Processing – Pcm 2007, vol. 4810/2007, pp. 775–784, 2007.



(a) line 1 (stripe) and line 2 (non-stripe)

(b) Intensity histograms of line 1 (stripe) and line 2 (non-stripe)

Fig. 4. Intensity histograms of a stripe line and a non-stripe line

Figure 2.5 – A sharp peak occur on the right side for a stripe on white background comic (captured from the research paper of Chung [6])

A threshold of 90% of pixels around the peak in the intensity histogram is being suggested. Then it will be determined whether it is a false stripe. Two conditions of false stripe occurrence are being suggested.

- False stripes often appear inside small panel regions.
- A real stripe has a large contrast with the surrounding pixels. [6]

If the line stripe passes the two conditions above, then it will go through the segmentation part.

For the gradient density approaches suggested by T. Takana *et al.* , the density gradient gθ (x, y) of the θ direction at the position (x, y) on an image is given as follows:

$$g_\theta(x,y) = g_x(x,y)\cos\theta + g_y(x,y)\sin\theta$$

where $g_x(x, y)$ is the density gradient of the X-axis direction, and g_y (x, y) is the density gradient of the Y-axis direction. The accumulation value A(ρ,θ) of the density gradient value of the θ direction is obtained along the straight line L(ρ, θ). The density gradient value A(ρ, θ) of the straight line L(ρ, θ) on an image is given as follows:

$$A(\rho, \theta) = \sum_{(x,y) \in S(\rho,\theta)} g_\theta(x, y)$$

In order to reduce the false detection of division lines, the value must undergo some process first. Normally, the central part of the image is expected that there is the high possibility a division line is located. Therefore, in order to raise the weight of the central part of the image, a weighted accumulation value of gradient AW (ρ, θ) is obtained by multiplying A(ρ, θ) by a Gaussian function G(ρ) on ρ :

$$AW(\rho, \theta) = G(\rho)A(\rho, \theta)$$

where G(ρ) = exp(−ρ2/σ2), and σ = [Image height]/4.

After that, the evaluation value H_W (ρ, θ) of segment without white pixels of some extent length is lowered by multiplying A_W (ρ, θ) by the ratio of the length of the longest white pixel column to that of the segment. In order to avoid duplicate detection, the straight line judged as division line is stored in a cache list. The straight lines which are similar to the straight lines stored in the cache list are not chosen as a division line in the further division line detection. Under the constraint, the candidate of division line is obtained as

$$(\rho_p, \theta_p) = \arg\max_{\rho,\theta} H_W(\rho, \theta).$$

Whether $L(\rho_p,\theta_p)$ is division line or not is decided by whether $H_W(\rho p, \theta p) \geq h_{th}$ or not. As h_{th} is raised, division lines detected are limited more. We decided h_{th} heuristically. In the implementation, θ is changed from -90° to 90° at 1° step, and ρ is changed within the image at 1 pixel step.[5]

Despite advantages of the recursion algorithm, it is worth to note that the recursion approach has two drawbacks.

- It may not deal with comics with just one panel on a page correctly.
- The division lines on the page which are not in uniform color will fail.

To conclude, the outline of method, advantages and disadvantage as list as follows:

		Pros	Cons
Multi Pass Approaches	The whole image undergo a few stages of processing	Relatively less computation power required	Fail if there are overlapping panels
Recursion Approaches	The image goes through a process-divide recursion	Can generate the reading sequence at the same time	• Relative slower • Page with one panel only will fail

Table 2.1 – Pros & Cons of the two approaches

2.2 Commonly Used Functions

In this section, we will highlight some functions that might be useful for our project. We will describe them in a theoretical manner, including how they work and what parameters they require.

2.2.1 Contour Finding

OpenCV has a series of functions for contour finding. We have studied the basic one, cvFindContour().

This function retrieves contours from the binary image. The contours are a useful tool for shape analysis, object detection and recognition. Therefore we can get some useful information from the contours for comic panel extraction.

The function will return the number of retrieved contours and fill up the first contour pointer as well. This pointer points to the first contour founded or simply NULL if the image is totally in black and no contour is found. Other contours can be reached from first contour using the h next and v next links.[7]

There are some other functions that support cvFindContour(), e.g. cvCreateContourTree(), cvDrawContours(), etc. Contours found belong to different connected components, which can be used to generate bounding boxes to give a rough sight of the panel structure of the whole image.

[7] OpenCV Reference Manual, v2.1, March 18,2010

2.2.2 Hough Line Finding

Hough line finding is another important technique that might help us to gather useful information for panel extraction. We have examined cvHoughLines2() which finds lines in a binary image using a Hough transform.

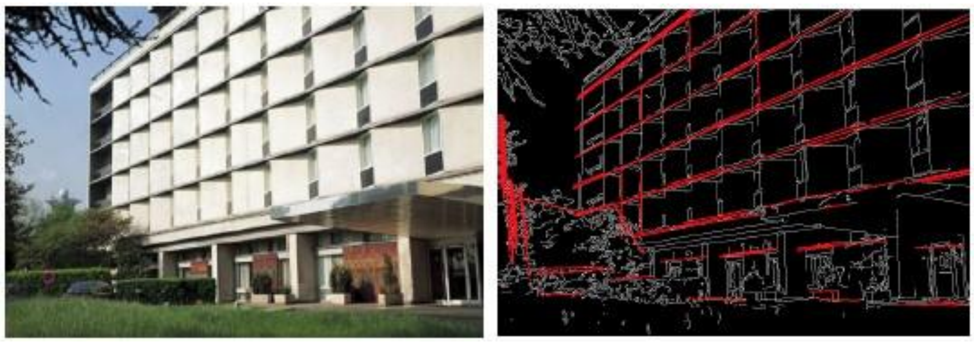


Figure 2.6 - Demonstration for Hough line detection

This function accepts 3 methods as transform variant:

- CV_HOUGH_STANDARD
- CV_HOUGH_PROBABILISTIC
- CV_HOUGH_MULTI_SCALE

CV_HOUGH_STANDARD is classical Hough transform. Under this method, every line is represented by two floating-point numbers (p, q), where p is the distance between the origin and the line, and q is the angle between the normal of the line and x-axis.

CV_HOUGH_PROBABILISTIC is the probabilistic Hough transform. It will be more efficient if the target image contains a few long linear segments. This method returns segments, represented by starting point and end point, instead of the whole line.

CV_HOUGH_MULTI_SCALE is the multi-scale variant of the classical Hough transform. The lines are thus encoded in the same way as in CV_HOUGH_STANDARD.[7]

Noted that Hough lines do not equal to the contours of the target image, they are the supplementary information to the contours.

2.2.3 Watershed Segmentation

Watershed segmentation is also a very important part to our study.

"A drop of water falling on a topographic relief flows along a path to finally reach a local minimum. Intuitively, the watershed of a relief corresponds to the limits of the adjacent catchment basins of the drops of water."[8] This illustrates a brief idea of watershed in image processing.

In image processing, when it comes to watershed, we are usually talking about segmentation. Different watershed lines can be computed for segmentation purpose. Watershed segmentation requires contour information first. We can imagine that once we drop a drop of water onto the image, it spreads out until reaching the contour.

In OpenCV, cvWatershed() implements one of the variants of

[8] The watershed transformation for multiresolution image segmentation, S. Wegner, T. Harms, J. H. Builtjes, H. Oswald and E. Fleck

the Watershed algorithms, non-parametric marker-based segmentation.

Users are required to manually input some markers roughly indicating different regions of the target image. These markers are treated as seeds of the future image regions while the other pixels, which are not marked, should be defined by the algorithm and set to 0's. Each pixel in markers will then be set to one of values of the seed component, or -1 if it is at boundaries between the regions.[7]

In other words, all pixels having a value greater than 0 are supposed to belong to a region, they are segmented according to the value assigned.

2.2.4 Flood Fill

The traditional flood fill algorithm takes three parameters:

- A start node,
- A target color and,
- A replacement color.

The algorithm looks for all nodes in the array which are connected to the start node by a path of the target color, and changes them to the replacement color. There are many ways to implement this algorithm, but they all make use of a queue or stack data structure, explicitly or implicitly. Here shows one implicitly stack-based recursive flood-fill implementation example for a two-dimensional array:

```
Flood-fill (node, target-color, replacement-color):
  1. If the color of node is not equal to target-color, return.
  2. Set the color of node to replacement-color.
  3. Perform Flood-fill (one step to the west of node, target-color, replacement-color).
     Perform Flood-fill (one step to the east of node, target-color, replacement-color).
     Perform Flood-fill (one step to the north of node, target-color, replacement-color).
     Perform Flood-fill (one step to the south of node, target-color, replacement-color).
  4. Return.
```

In OpenCV, Flood-fill is also implemented and it requires user to input a point in target image as seed. cvFloodFill() then fills a connected component with a given color.

Chapter 3 Product Design

In this chapter, we will focus on product design and anything related to it.

The structure of this chapter is shown as below:

- Convention
- Assumption
- System Architecture Design

3.1 Convention

We will talk about convention in this part clarifying some terms on a comic page.

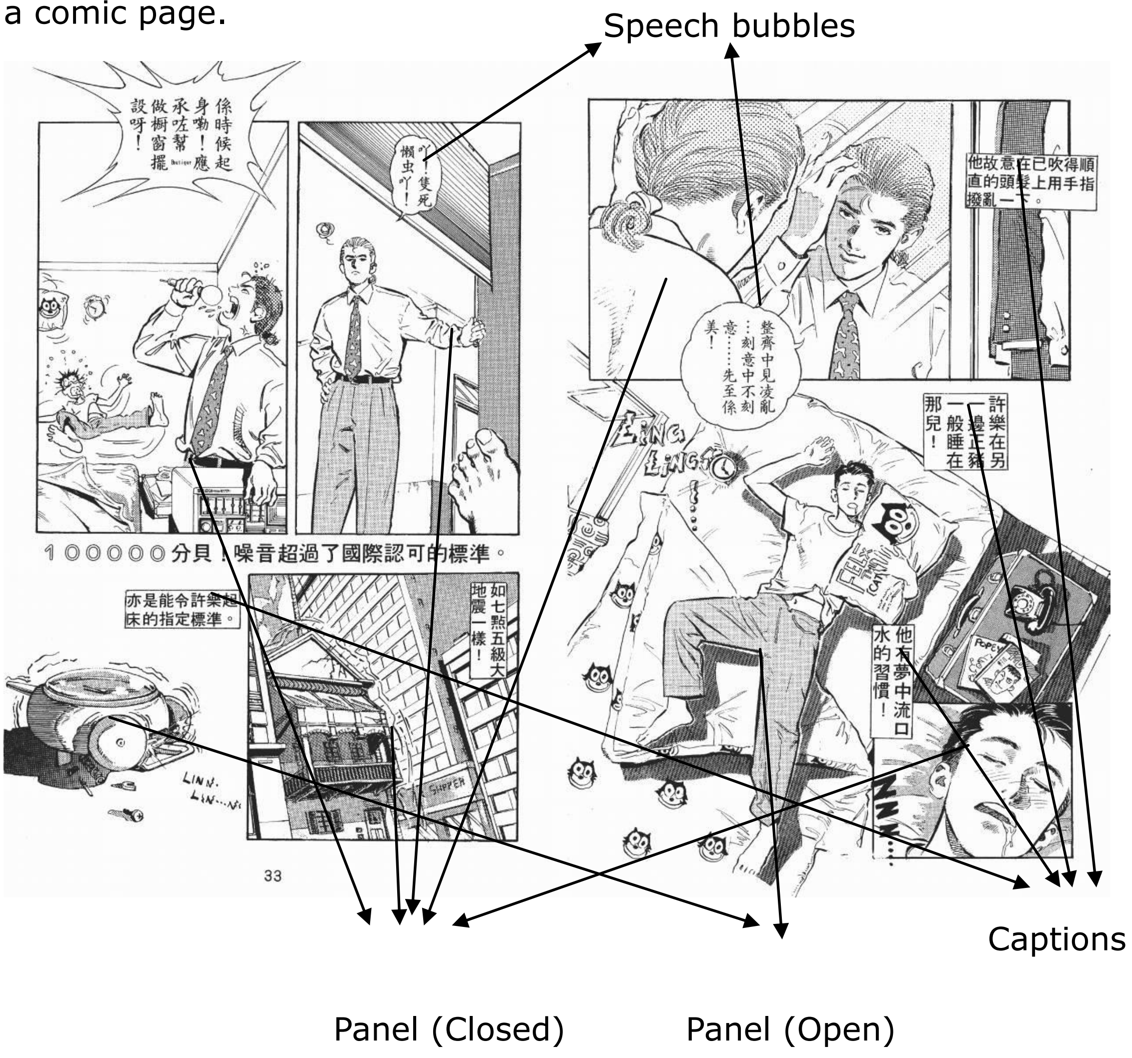

Speech bubbles
係時候起身嘞！應承咗幫做櫥窗擺設呀！
Boutique
吖！隻死懶虫吖！
100000分貝！噪音超過了國際認可的標準。
亦是能令許樂起床的指定標準。
如七點五級大地震一樣！
他故意在已吹得順直的頭髮上用手指撥亂一下。
整齊中見凌亂……刻意中不刻意……先至係美！
許樂在另一邊正豬一般睡在那兒！
他有夢中流口水的習慣！
33
Captions
Panel (Closed)
Panel (Open)

Figure 3.1 – Sample comic page

- *Panels* are individual frames that consist of a single drawing depicting a frozen moment.
- *Captions* are generally for narrative purpose. They are usually rectangles located near the edge of a panel.
- *Speech bubbles* indicate what a character says. In-panel speech bubbles are for characters that appear in the panel. Off-panel speech bubbles are for characters that cannot be seen in the panel.
- *Thought bubbles* are similar to speech bubbles but for thought expression only. Thought bubbles are not shown in the above picture.

3.2 Assumption

Due to the difficulties found on the research, some assumptions are made to tackle the problems. The project will start from some simpler cases, and investigate more complex cases later.

To simplify the complexity, we assume:

- No overlapping on panels of the comic panels is assumed.
- The background of the comic is assumed to be white.
- The comic must be in grayscale.

Comics with overlapping panels will be more difficult to perform traditional segmentation algorithm such as watershed. Modifications on algorithm must be done to deal with those comics.

The second and third assumption is also made to reduce the complexity of the problem.

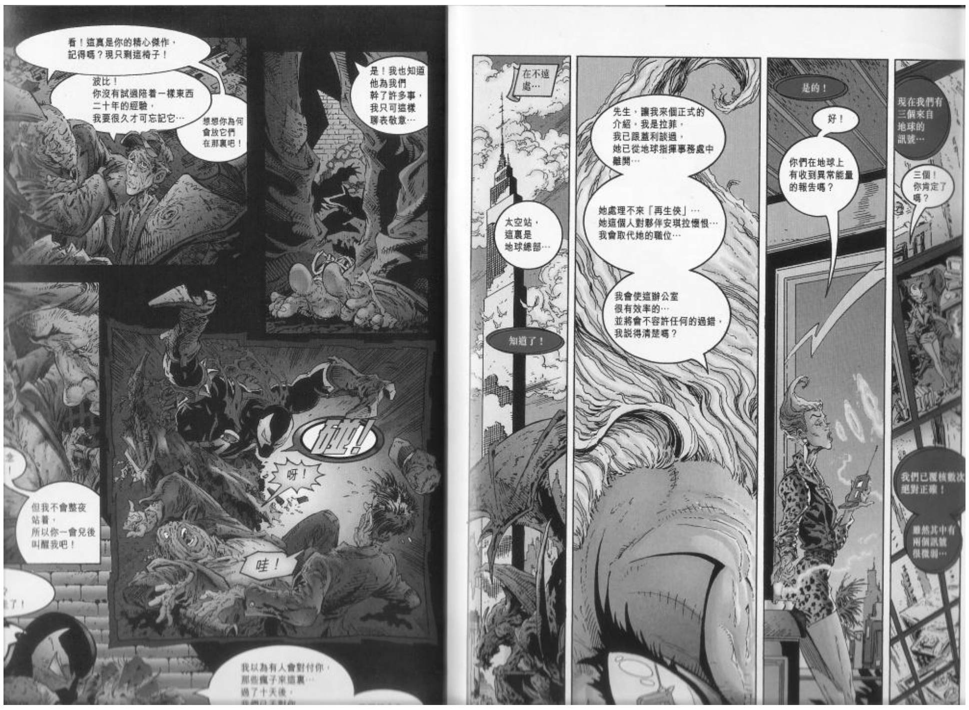

看！這真是你的精心傑作，記得嗎？現只剩這椅子！
波比！你沒有試過陪着一樣東西二十年的經驗，我要很久才可忘記它…
想想你為何會放它們在那裏吧！
是！我也知道他為我們幹了許多事，我只可這樣聊表敬意…
碰！
呀！
但我不會整夜站着，所以你一會兒後叫醒我吧！
哇！
我以為有人會對付你，那些瘋子來這裏…過了十天後，
在不遠處…
太空站，這裏是地球總部…
知道了！
先生，讓我來個正式的介紹，我是拉菲，我已跟蓋利談過，她已從地球指揮事務處中離開…
她處理不來「再生俠」…她這個人對夥伴安琪拉懷恨…我會取代她的職位…
我會使這辦公室很有效率的…並將會不容許任何的過錯，我說得清楚嗎？
是的！
好！
你們在地球上有收到異常能量的報告嗎？
現在我們有三個來自地球的訊號…
三個！你肯定了嗎？
我們已覆核數次絕對正確！
雖然其中有兩個訊號很微弱…

Figure 3.2 - A Comic with non-white background

3.3 System Architecture Design

3.3.1 Viewer

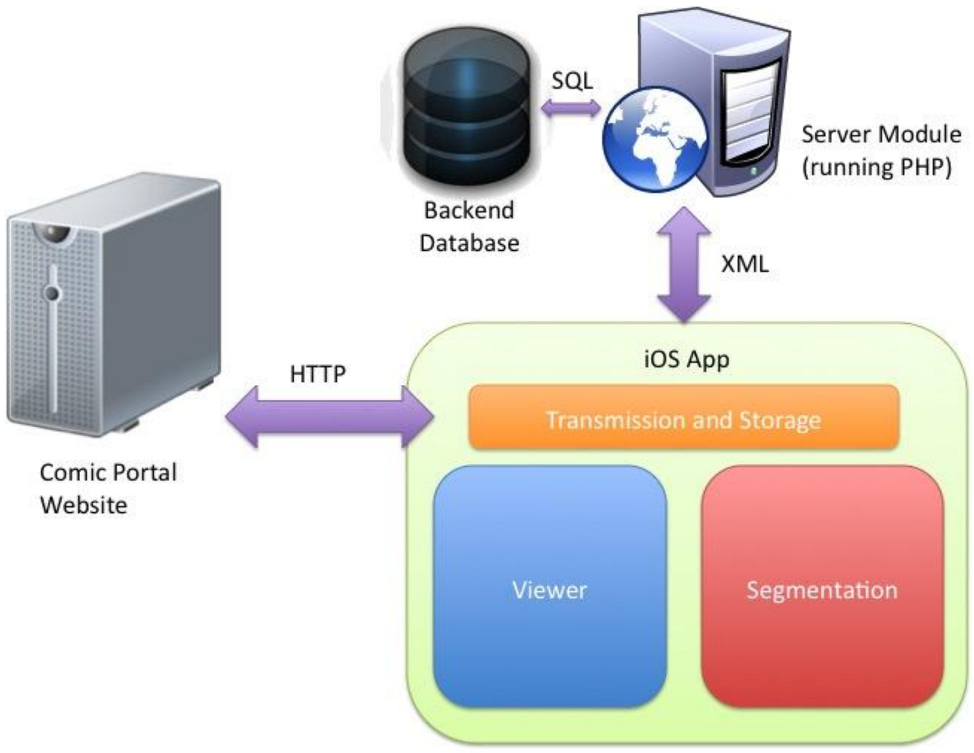


Figure 3.3 – System component

The architecture of the system consists of two major component, the server module and the Viewer app on iOS.

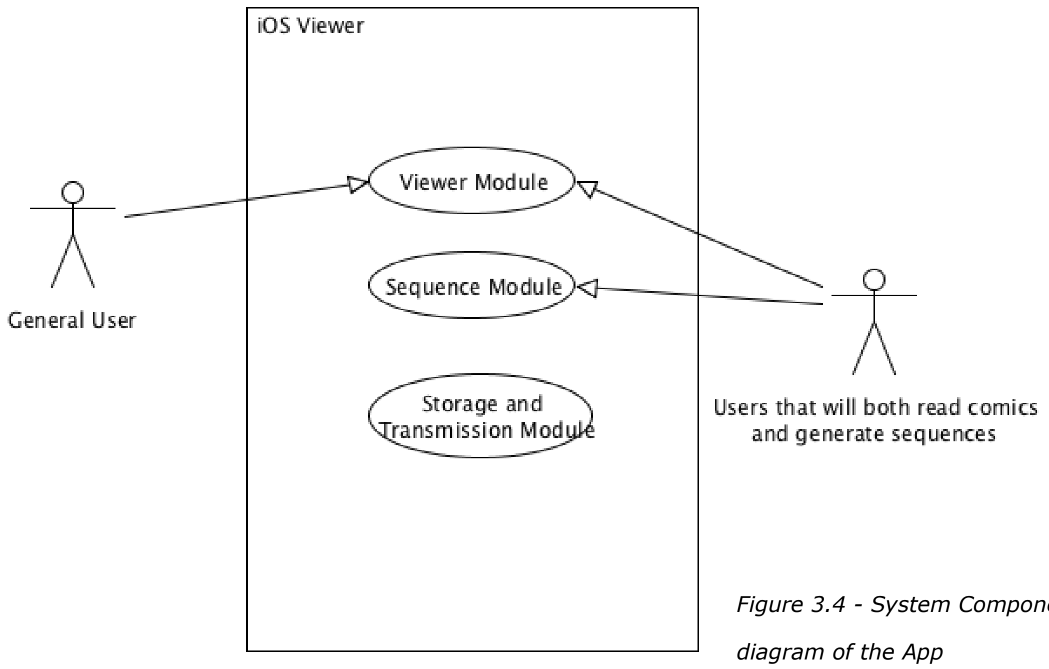

iOS Viewer
Viewer Module
Sequence Module
Storage and Transmission Module
General User
Users that will both read comics and generate sequences

Figure 3.4 - System Component diagram of the App

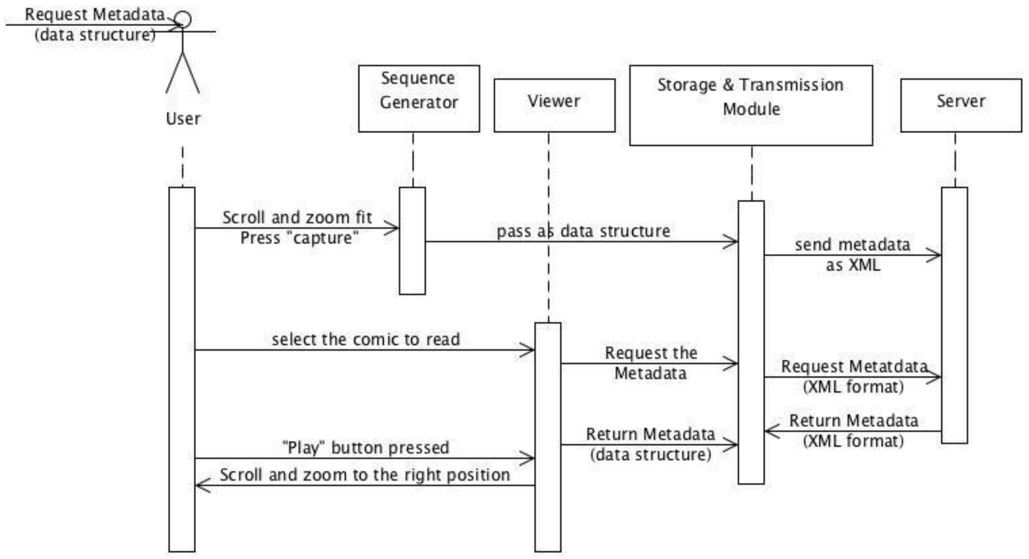
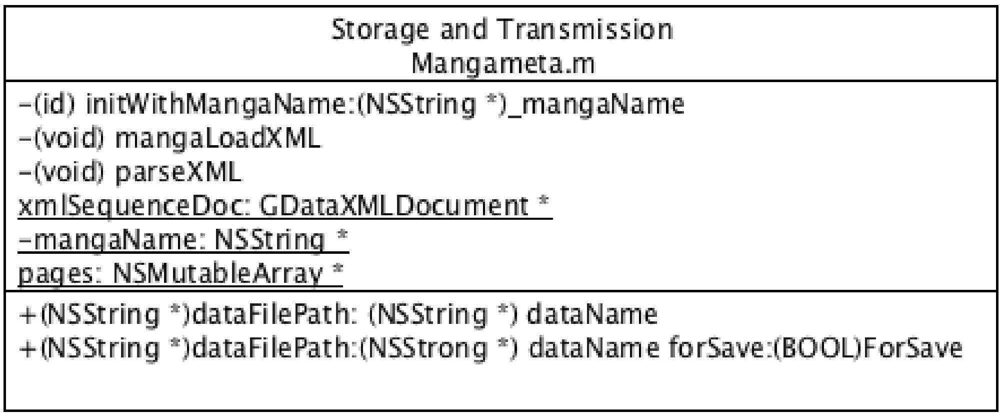

Request Metadata (data structure)
User
Sequence Generator
Viewer
Storage & Transmission Module
Server
Scroll and zoom fit Press "capture"
pass as data structure
send metadata as XML
select the comic to read
Request the Metadata
Request Metatdata (XML format)
Return Metadata (XML format)
Return Metadata (data structure)
"Play" button pressed
Scroll and zoom to the right position

Figure 3.5 - Sequence diagram of the App

The Viewer App can be divided into three major parts, including Metadata Transmission and Storage Module, Viewing Module and Segmentation Creation Module.

3.3.1.1 Metadata Transmission and Storage Module



```
Storage and Transmission
Mangameta.m
-(id) initWithMangaName:(NSString *)_mangaName
-(void) mangaLoadXML
-(void) parseXML
xmlSequenceDoc: GDataXMLDocument *
-mangaName: NSString *
pages: NSMutableArray *
+(NSString *)dataFilePath: (NSString *) dataName
+(NSString *)dataFilePath:(NSStrong *) dataName forSave:(BOOL)ForSave
```

Figure 3.6 - First draft of the module

Metadata, including the panel and sequence information, is managed by this module. The backbone of this part is XML. There are serval advantages of XML over other approaches.

- XML fully supported Unicode
- XML can be transferred through HTTP
- XML is strictly defined but flexible so very suitable for medium of machine communication

The first one is very important for local comics. For the second one, both stateless communication method and persistent method have been considered. Since one single file is needed to be transfer for a single comic, so a stateless approach will be enough. HTTP is one of the most prevalent stateless transfer approach so a lot of work can be saved if building on it.

MetaData is stored in XML format. When the viewer modules tries to retrieve the data, it decode the XML and converted it into corresponding data structure. On the other hand, if the segmentation module generates the panel and sequence information, it is converted by the module from data structure stored in ram to XML format.

Moreover, the module will also be responsible for obtaining the comi resources from the Internet.

The implementation of this part relied heavily on an external library by Google called GDataXML. It is written in Objective C and provide functions of XML read and write in DOM method. Moreover, it also support XPath which may be useful in the future.

The transmission part relies heavily on the iOS Cocca Touch framework, the NSURL class and the instance method *writeToURL*.

3.3.1.2 Viewing Module

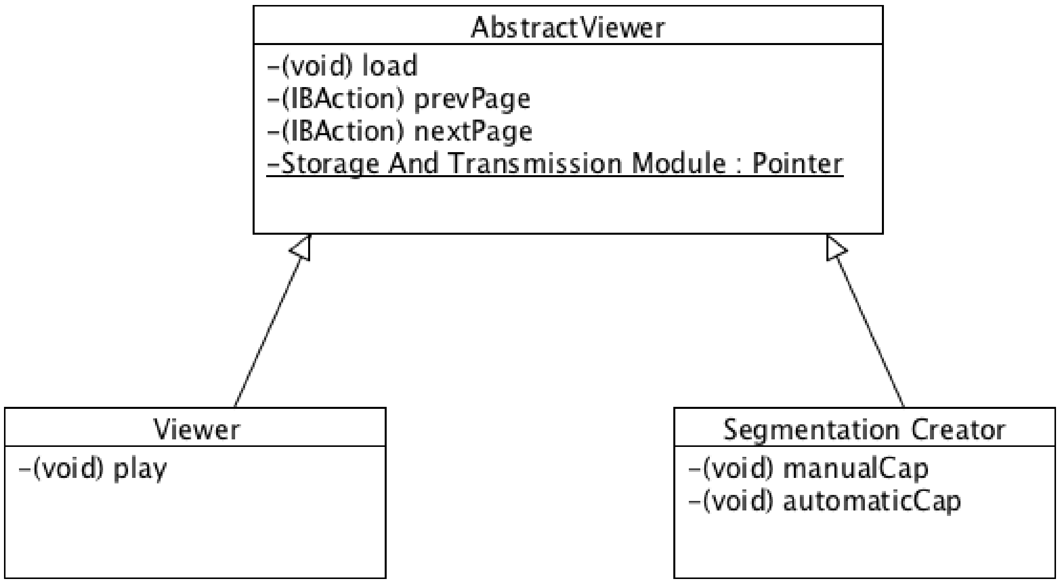


Figure 3.7 – Class diagram of the viewing module

Some of the navigation function is the same between Viewer module and Segmentation Creation Module, so a father class will be created first and the two module will inherited from this class.

The Viewing Module is responsible for the "playing" of the predefined sequence and zooming of the panel. It retrieves panel details and sequence data from the transmission and storage module. When the user pressed the play button on the tab bar, it will jump to the next panel.

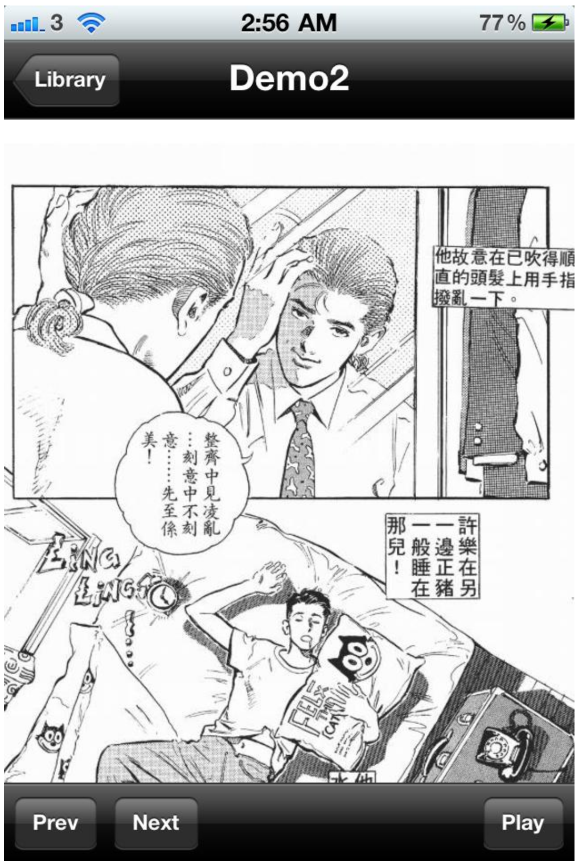

3
2:56 AM
77 %
Library
Demo2
他故意在已吹得順直的頭髮上用手指撥亂一下。
整齊中見凌亂……刻意中不刻意……先至係美！
許樂在另一邊正豬一般睡在那兒！
Prev
Next
Play

Figure 3.8 - Capture of Viewer Module, the page navigation buttons are on the left-bottom while the play button is on right-bottom

Moreover, the viewer module also makes use of the multi-touch function on the iOS. The user can zoom in or out by pitching if they feel the default zoom rate is not suited for them.

3.3.1.3 Segmentation Creation Module

This part is responsible for generating metadata of the comics. Current it only supports the manual mode, semi-automatic or automatic mode may be supported in the future. The design of this module will use the abstract factory design pattern.

Under the manual mode, the user scroll to the right position and zoom fit. Then a capture button can be pressed to record the information of the current frame. The process is repeated until the data of the whole comics is recorded. The viewing sequence will also be generated at the same time according to the capture sequence.

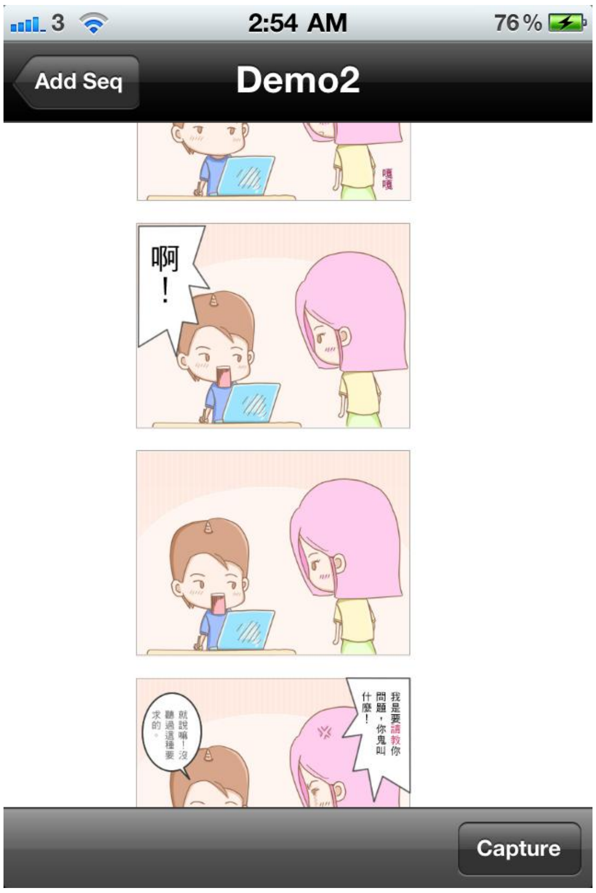


Figure 3.9 - Capture of the Segmentation module, the "Capture" button is on the right-bottom

After that, the Segmentation module will pass the information to the Metadata Unit for local storage and sharing the information to the internet.

3.3.2 The Server

The main role of the server is to store and distributed shared metadata information. It includes two parts, the upload part and the request part. Users can generate metadata and upload it or download shared metadata from it.

The Server will use PHP to process request and use MySQL as the backend storage. This PHP+MySQL system is chose mainly because that

PHP and MySQL are popular and well developed tools. There are lots of documentations around the world so it is more easy to get support. Also, the are mature libraries and frameworks on PHP, this shorten the development time and prevent reinventing the wheels. Moreover, lots of investigation on optimization has been done by other professionals. This save a lot of time on tweaking.

Besides the backend MySQL, the Server Module can be further divided into two parts, upload.php and request.php

When the user open a comic, the viewer app will automatically submit a request to request.php. request.php then check whether there are metadata. If yes, return it and return not found if no.

When the Viewer app generate a metadata file, it will upload the data through upload.php and the metadata will be saved into the backend database.

Chapter 4 Project Progress

We have tested some algorithms in PCs as well as implemented a simple comic viewer for iOS in Mac. Therefore, in this chapter, we will break down into two main focuses, talking about our project progress as the following list:

- Algorithm Study
  - Histogram
  - Contour Finding
  - Hough Lines
  - Watershed Segmentation
  - Flood-fill
- Implementation
  - Manual Panel Extractor
  - Viewer Implementation on iOS

4.1 Algorithm Study

The following algorithms are implemented in OpenCV, most of them are function calls. However, it requires a lot of pre-processing and post-processing on the target image, therefore we have written several separated programs to examine the algorithms respectively. It is helpful for us to:

- Get familiar with these functions provided by OpenCV,
- Examine the effect of these functions and,
- Determine whether a specific algorithm can provide useful information for panel extraction.

4.1.1 Histogram

For analysis purpose, we have calculated the Histogram of most target test cases. This helps us to cluster the test cases. For instance, we can cluster the test images once we know their color distribution.

In this program, we assume:

- Color range: 0-255, where 0 means totally black and 255 means totally white.
- Frequency of a certain color: 0-5000

4.1.2 Contour Finding

Contour provides useful information to panel extraction. No matter we use Watershed Segmentation or Flood-fill, contours should be pre-calculated.

This is the second program we wrote at the very beginning. To examine the image processing without Watershed Segmentation, we have written this program which can achieve the following functions:

- ◆ Image Binarization
- ◆ Contour Detection
- ◆ Bounding Box Calculation

We have tried and compared two methods for image binarization:

Method1: Directly use cvThreshold()

Method2:

- ◆ Get background color through 4 sample points and take average value
- ◆ Calculate distance between background color and the image
- ◆ Binarized the image by a threshold, same as the one cvThreshold() used

After image binarization, we perform contour detection and calculate the bounding box to the two binary results with same parameters. Before actual test cases, we have a pre-test case to evaluate the threshold we are going to use.

Case: Pre-test
Purpose: to evaluate the threshold for further testing

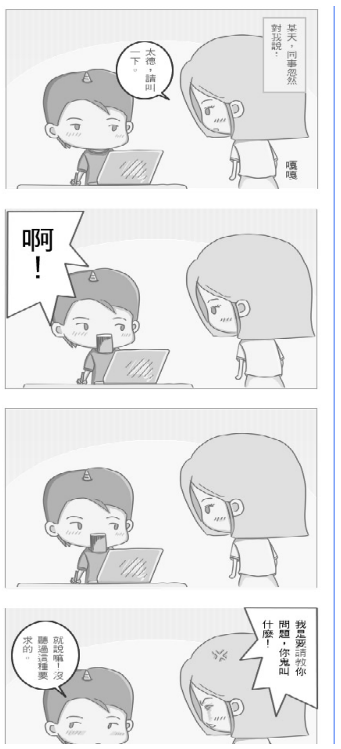


Figure 4.1 - Input image for Pre-test case

Case: #1
Purpose: to find the contours as well as bounding box for a simplest Four Column Comic
Remark: Same as Pre-test case input

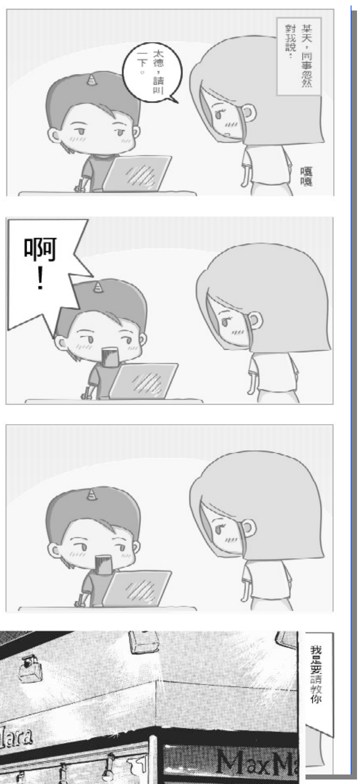


Figure 4.2 - Input Image for contour finding, Test case 1

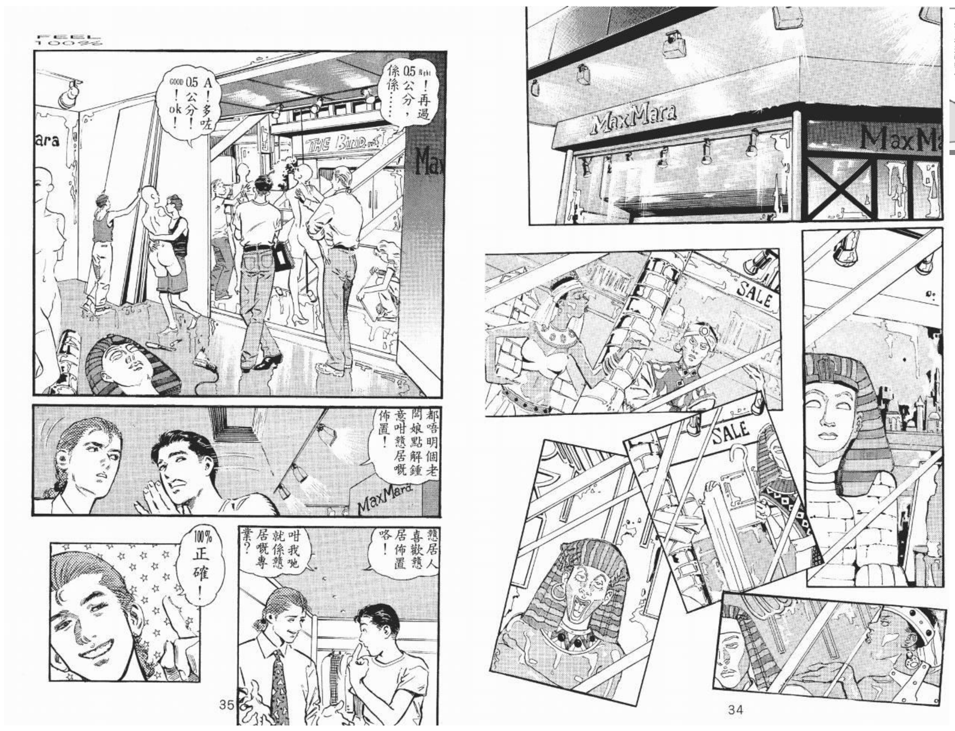
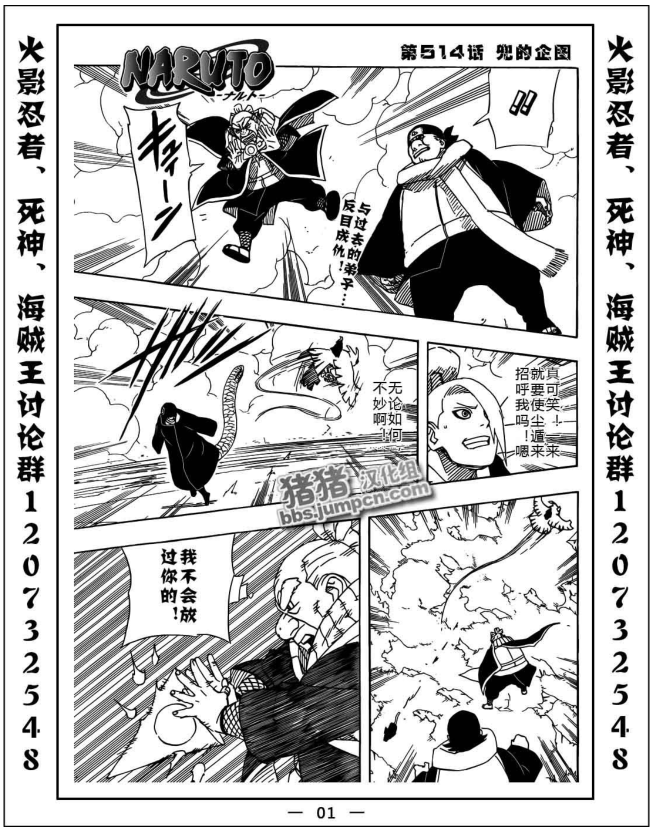


Figure 4.3 - Input image for contour finding, Test case 2

Case: #2
Purpose: to try finding the contours as well as bounding box for image with overlapped panels


火影忍者、死神、海贼王讨论群120732548
NARUTO
第514话 兜的企图
!!
キィーン
与过去的弟子…反目成仇！
真可笑！来就要使尘遁来招呼我吗！嗯
无论如何不妙啊！
我不会放过你的！
猪猪汉化组 bbs.jumpcn.com
— 01 —

Case: #3
Purpose: to find the contours as well as bounding box for image with strong strokes and crossed-panel-blobs

Figure 4.4 - Input image for contour finding, Test case 3

This test image is chosen because:

- There exist some speech bubbles crossing two panels in general.
- Strong strokes of this image might affect the result.

Case: #4

Test case 4 makes use of the original image of case 3, with threshold on 255 for demonstration purpose only.

4.1.3 Hough Lines Finding

We have also examined on Hough lines finding. Hough lines are different with contours as contours contain arbitrary lines but what we can get from Hough line detection are straight lines only.

Case: #1
Purpose: to find the Hough lines in a simple Four-Column Comic page

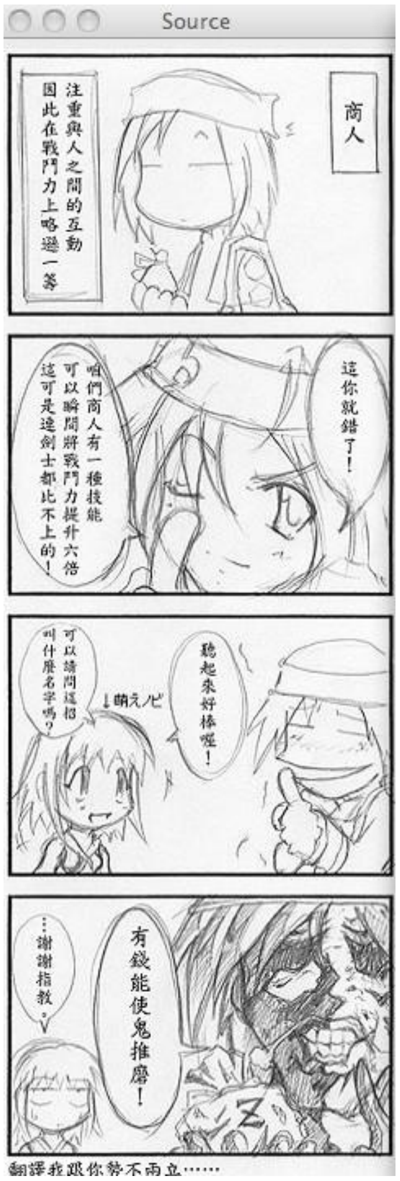


Figure 4.5 - Input image for Hough Line finding, Test case 1

Case: #2
Purpose: to find the Hough lines in a traditional comic page with open panels

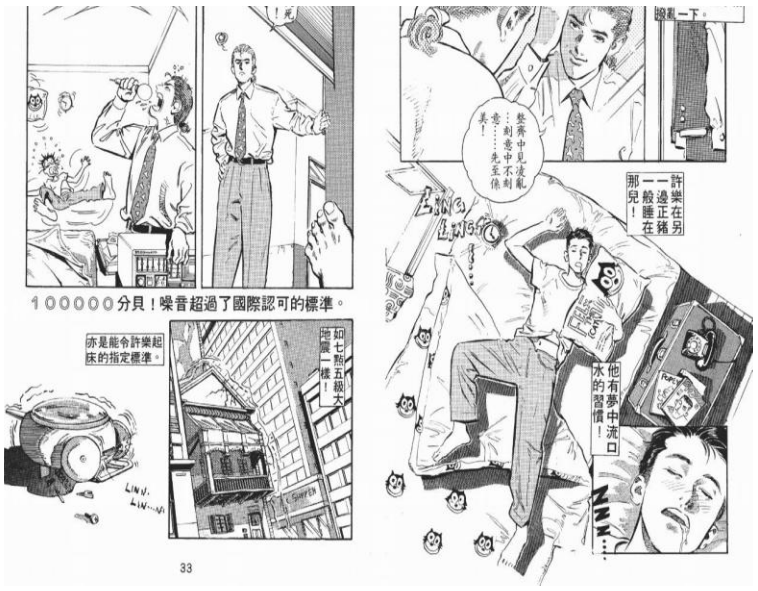
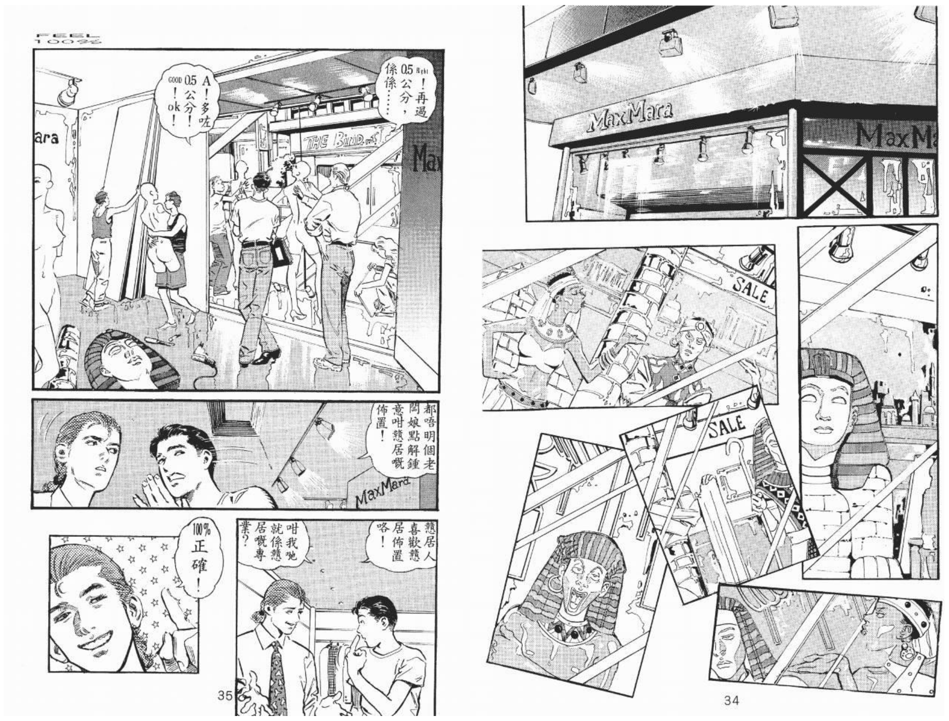

撥亂一下。
整齊中見凌亂…刻意中不刻意……先至係美！
許樂在另一邊正豬一般睡在那兒！
他有夢中流口水的習慣！
NNN….
LING LING
100000分貝！噪音超過了國際認可的標準。
亦是能令許樂起床的指定標準。
如七點五級大地震一樣！
33

Figure 4.6 - Input image for Hough line finding, Test case 2



Case: #3
Purpose: to find the Hough lines in a traditional comic page with overlapped panels

Figure 4.7 - Input for Hough line finding, Test case 3

Case: #4

Purpose: to find the Hough lines in a comic page with strong strokes

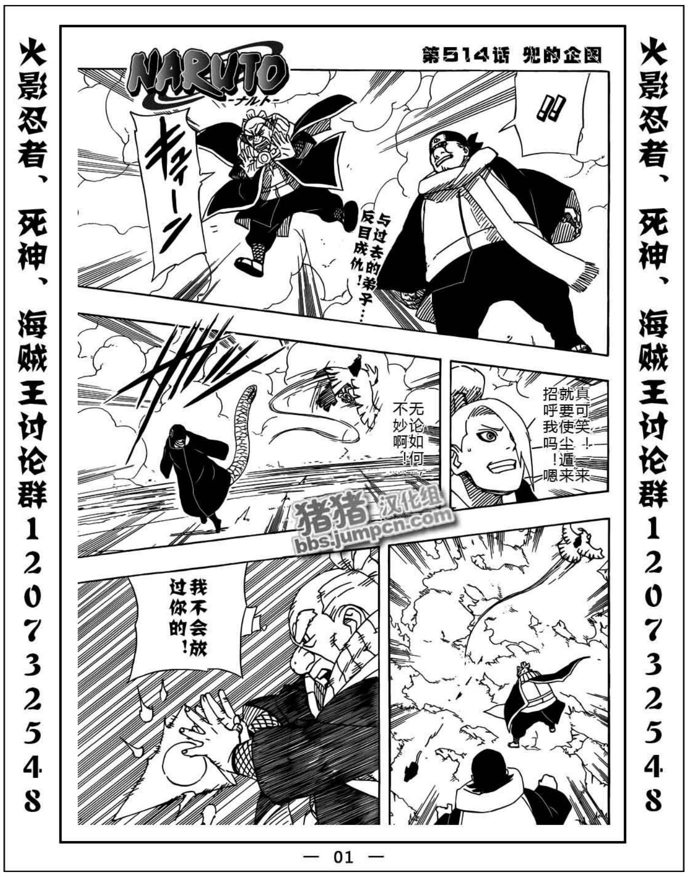
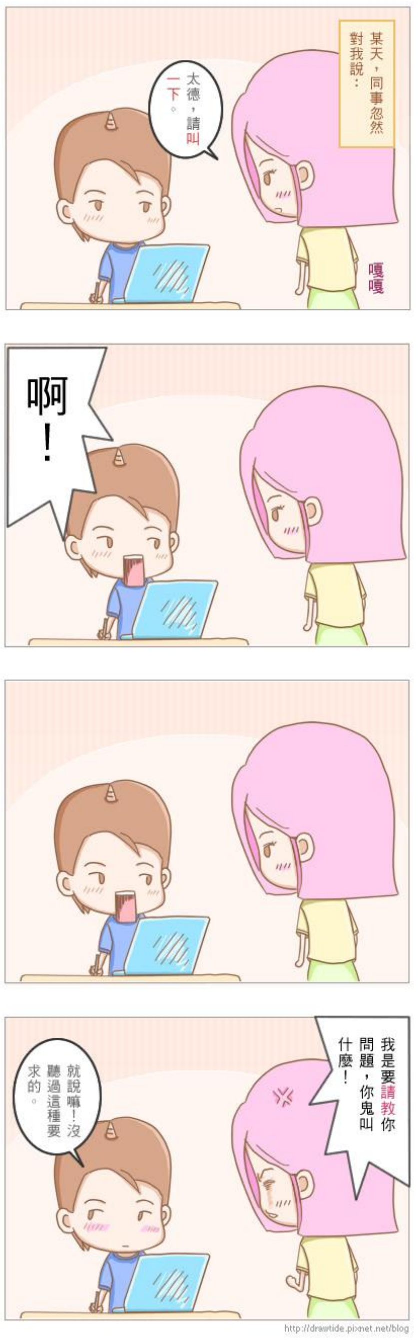


Figure 4.8 - Input for Hough line finding, Test case 4

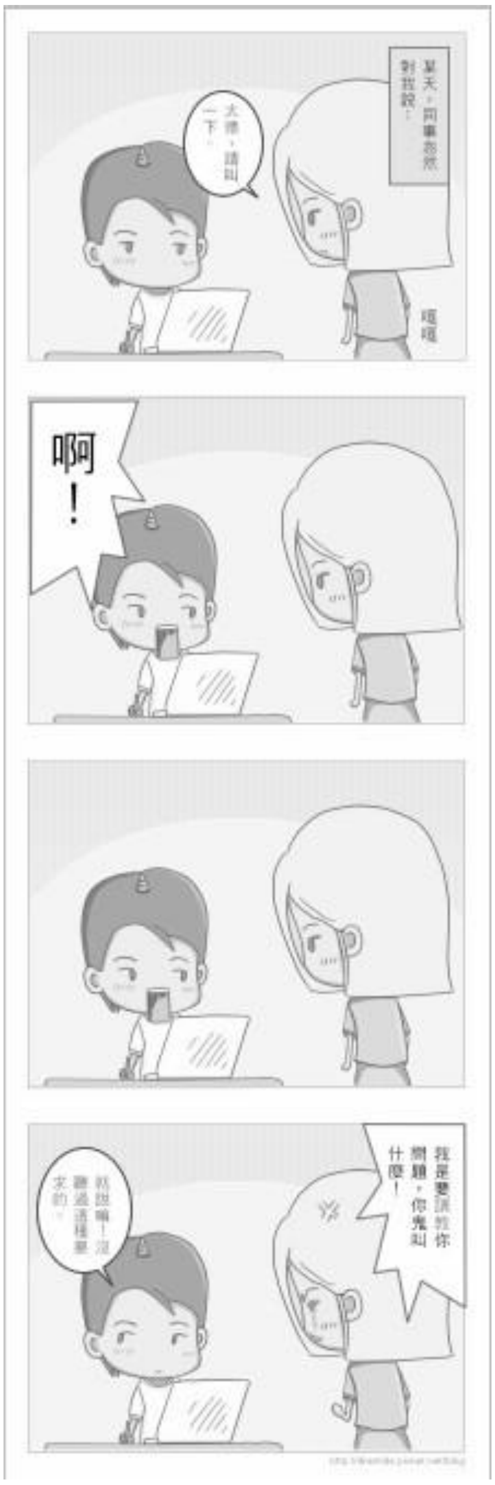
4.1.4 Watershed Segmentation

We have 4 test cases for Watershed Segmentation which requires manually input the markers for panels.



Case: #1
Purpose: to perform Watershed Segmentation in a simple colorful Four-Column Comic page

Figure 4.9 - Input image for Watershed Segmentation, Test case 1



Case: #2
Purpose: to perform Watershed Segmentation in a simple Four-Column Comic page in grey scale

Figure 4.10 - Input image for Watershed Segmentation, Test case 2

Case: #3
Purpose: to perform Watershed Segmentation in a comic page with overlapped panels

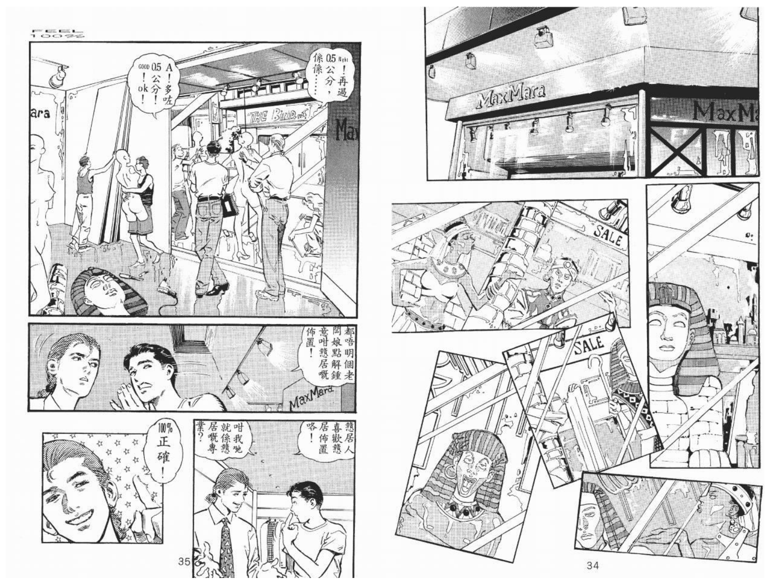
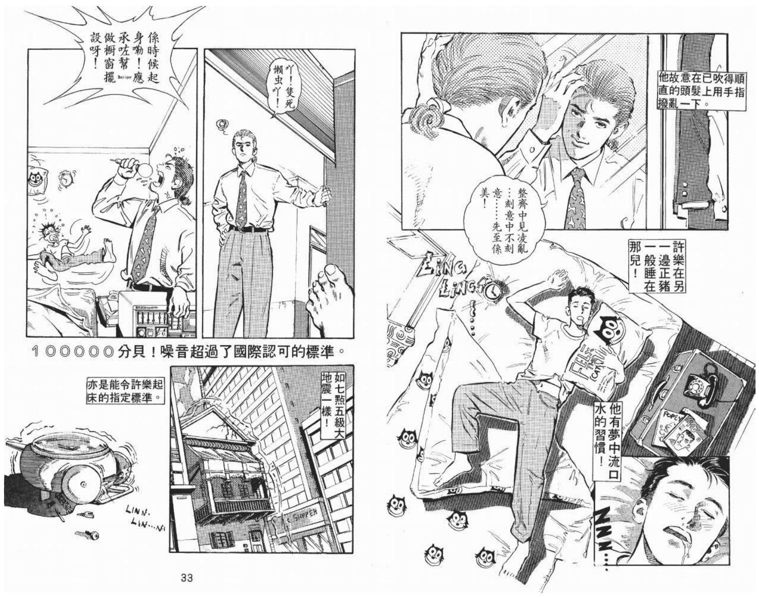


Figure 4.11 - Input image for Watershed Segmentation, Test case 3


係時候起身喇！應承咗幫Boutique做櫥窗擺設呀！
吖！隻死懶虫吖！
100000分貝！噪音超過了國際認可的標準。
亦是能令許樂起床的指定標準。
如七點五級大地震一樣！
LINN. LIN...N
他故意在已吹得順直的頭髮上用手指撥亂一下。
整齊中見凌亂……刻意中不刻意……先至係美！
許樂在另一邊正豬一般睡在那兒！
LING LING
他有夢中流口水的習慣！
NNN....
33

Case: #4
Purpose: to perform Watershed Segmentation in a comic page with open panels

Figure 4.12 - Input image for Watershed Segmentation, Test case 4

Case: #5
Purpose: to perform Watershed Segmentation in a comic page with strong strokes

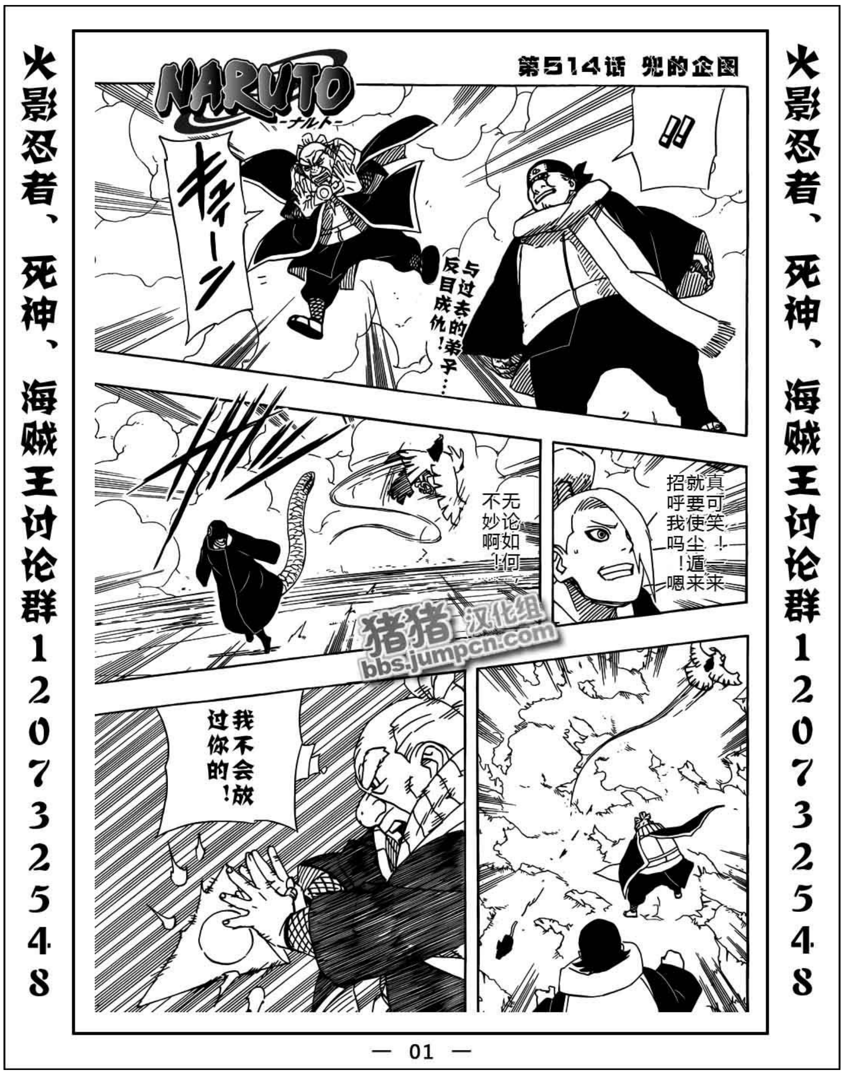
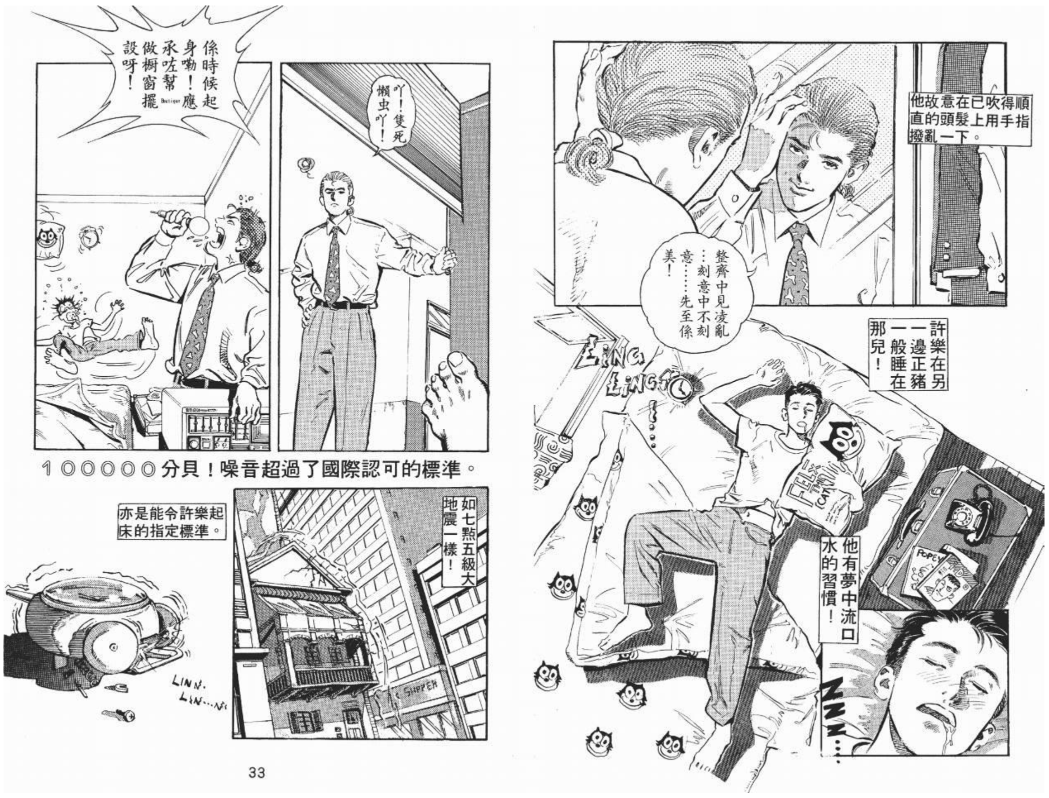


Figure 4.13 - Input image for Watershed Segmentation, Test case 5

4.1.5 Flood Fill

In this part, our concern is how to enhance the accuracy of Watershed Segmentation. Flood-fill is thus helpful to us as it can enhance the contrast between the background and the panels. Since this part does not involve too much, we will only demonstrate some outputs below. The background are manually selected and filled as red.


他故意在已吹得順直的頭髮上用手指撥亂一下。
整齊中見凌亂……刻意中不刻意……先至係美！
LING LING
許樂在另一邊正豬一般睡在那兒！
他有夢中流口水的習慣！
NNN....
係時候起身嘞！應承咗幫Boutique做櫥窗擺設呀！
吖！雙死懶虫吖！
100000分貝！噪音超過了國際認可的標準。
如七點五級大地震一樣！
亦是能令許樂起床的指定標準。
LINN. LIN...N
33

Case: #1
Purpose: to perform Flood-fill in a comic page with open panels
Remarks: Simple Four-Column Comic page is omitted as the result is trivial once Case #1 is done.

Figure 4.14 - Input image for Flood-fill, Test case 1

Case: #2
Purpose: to perform Flood-fill in a comic page with overlapped panels

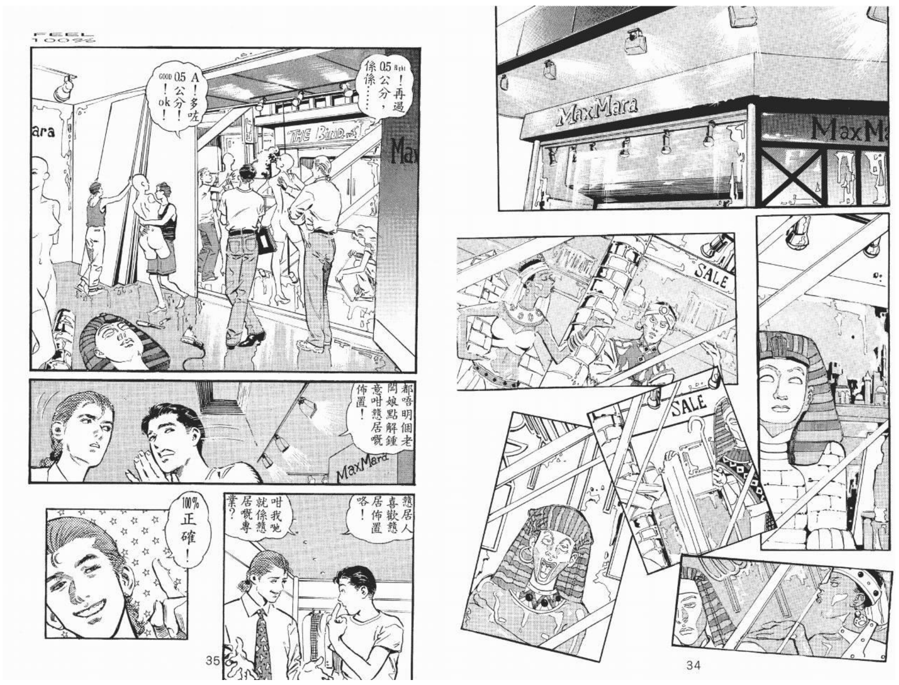
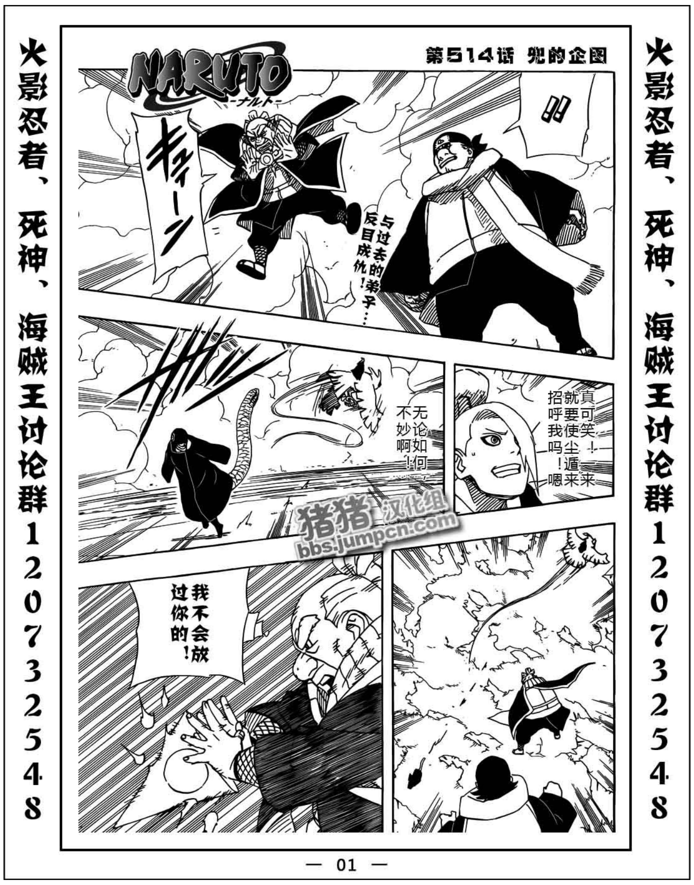


Figure 4.15 - Input image for Flood-fill, Test case 2



Case: #3
Purpose: to perform Flood-fill in a comic page with strong strokes

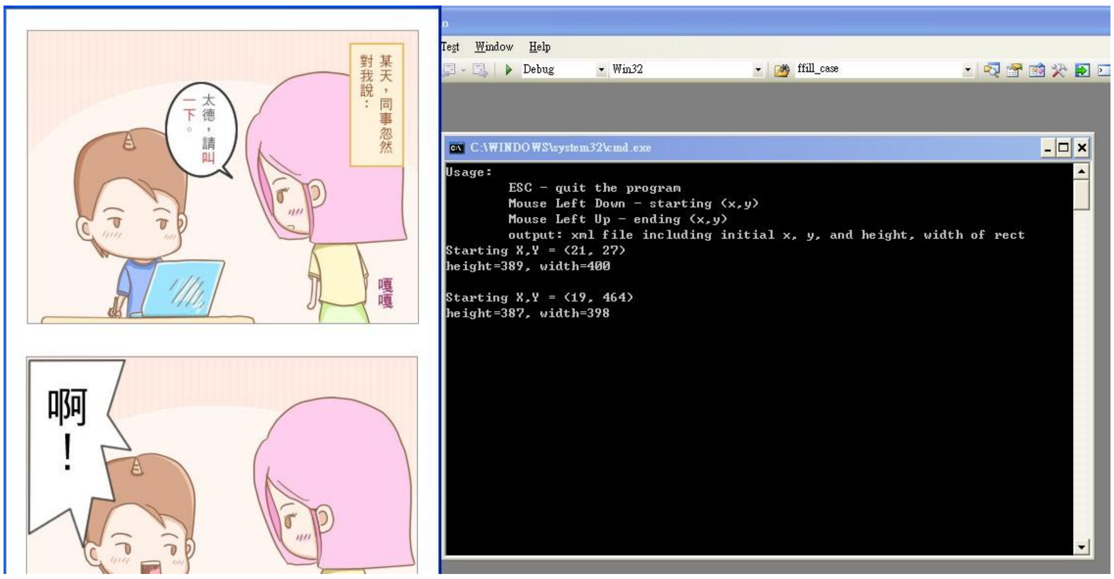
Figure 4.16 - Input image for Flood-fill, Test case 3

4.2 Implementation

4.2.1 Manual Panel Extractor

So far, we have studied some important algorithms and functions for panel extraction. For demonstration, we have also implemented a manual panel extractor in PC which output an XML file describing the structure of the panels on the target image.

Input: Drawing bounding box manually by mouse
Output: Initial x and y coordinate as well as height and width of the bounding boxes, saved as XML.



```
Usage:
        ESC - quit the program
        Mouse Left Down - starting (x,y)
        Mouse Left Up - ending (x,y)
        output: xml file including initial x, y, and height, width of rect
Starting X,Y = (21, 27)
height=389, width=400

Starting X,Y = (19, 464)
height=387, width=398
```

Figure 4.17 - Screen dump of the manual panel extractor

The output XML has a simple structure shown as follows, where the depth of the branch represents the level of the node.

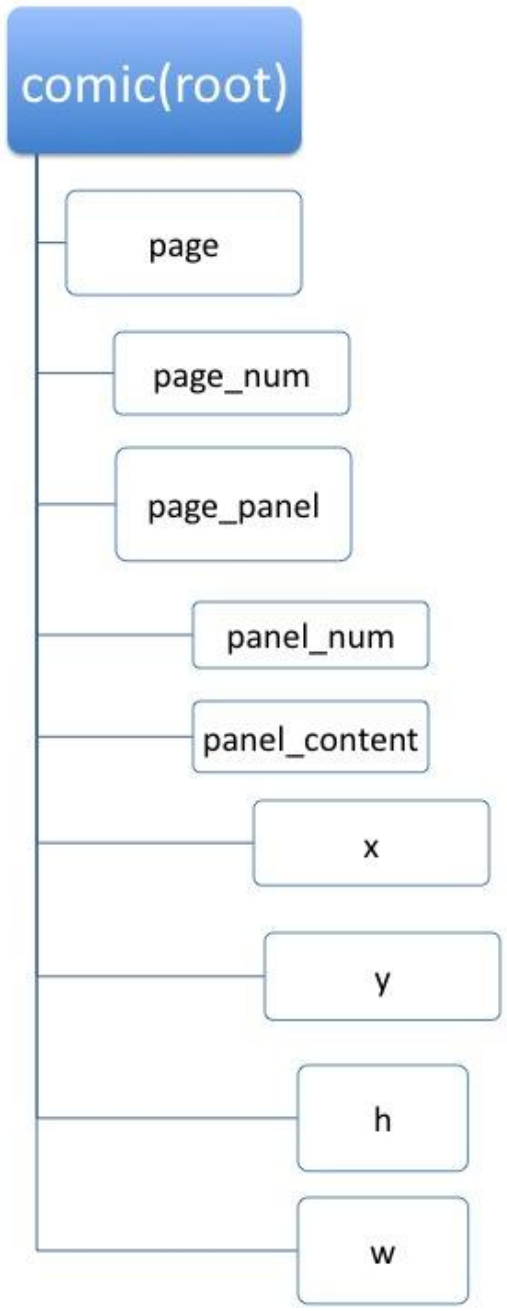


Figure 4.18 - Structure of the output XML

4.1.1 Viewer Implementation

The Viewer App is developed parallel with the algorithm study. The Spiral model is being adopted to provide a balance between fast prototyping and the meeting of the required functions.

As the OpenCV is act as the "heart" of the project, a suitable port on iOS must be ensured.

The porting from Yoshimasa Niwa[9] has been tested and chose. There are several advantages.

- ◆ It is a mature porting. Most of the functions of the OpenCV are working on this porting.
- ◆ There are precompiled version for both iOS Simulator and iOS device.
- ◆ It took use of the Accelerate Framework which is newly ported to iOS 4 by Apple Inc. This improves the performance.

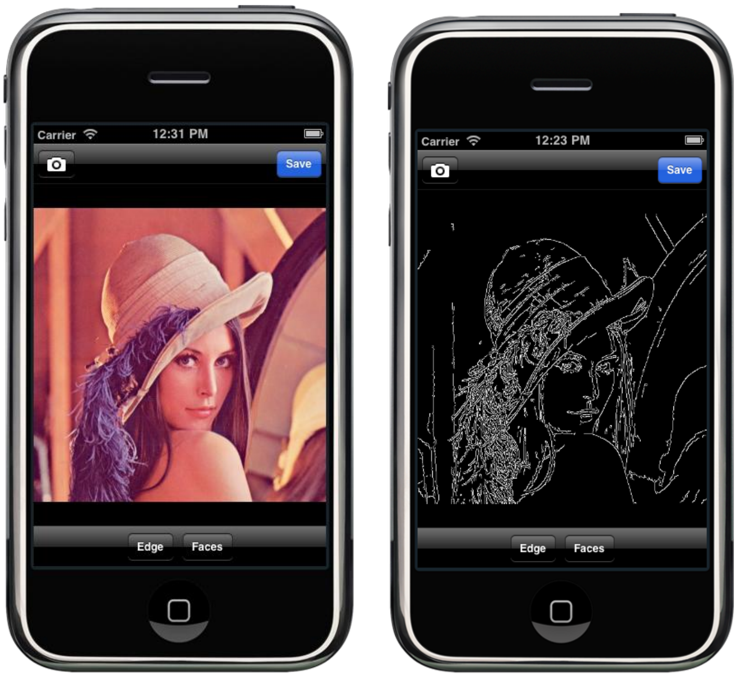


Figure 4.19 – OpenCV program on iPhone

9 http://niw.at/articles/2009/03/14/using-opencv-on-iphone/en

Since the function of the viewer is to display the comic to the user, and the program structure of an iOS app is heavily related to how you organized the display of information, so the interface is being considered and draft first.

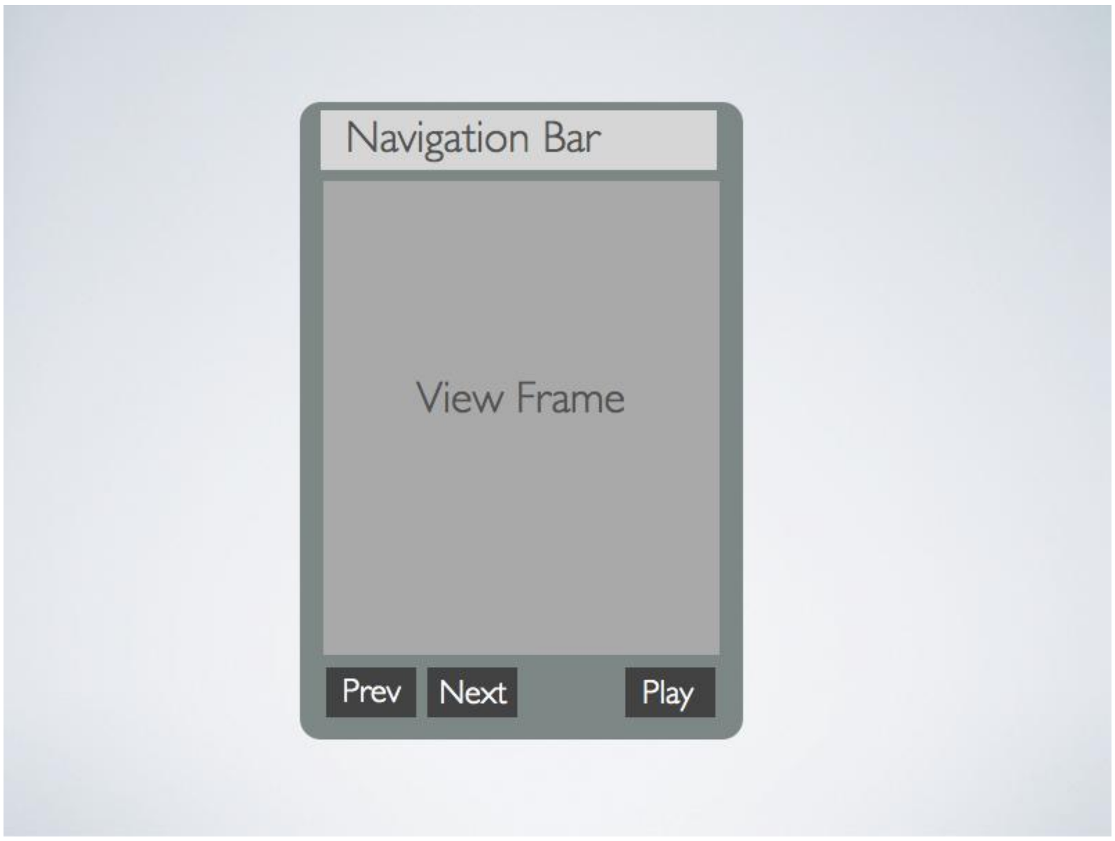
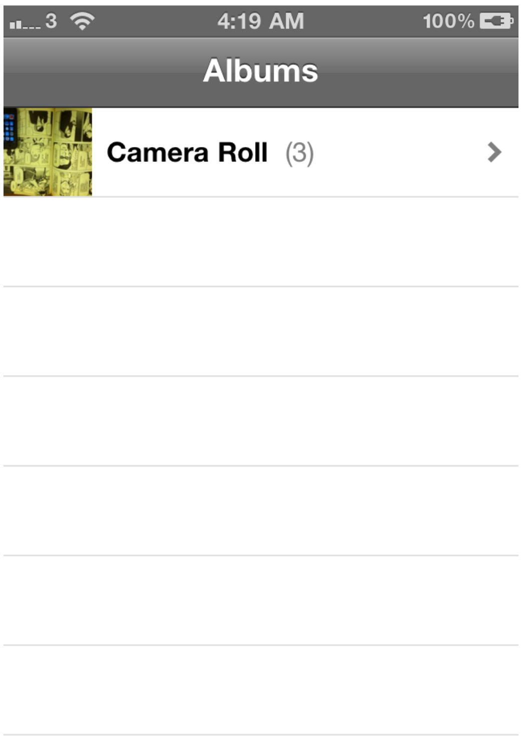


Figure 4.20 - Draft of the interface of the viewer

On top of this, a page is needed for the users to choose which comic to read. As a result, the navigation view mode with table is being chosen.

Table view contains a list of field that you can choose. The navigation view contains a button that can go back to the last page and can provide a “feel” of hierarchy.



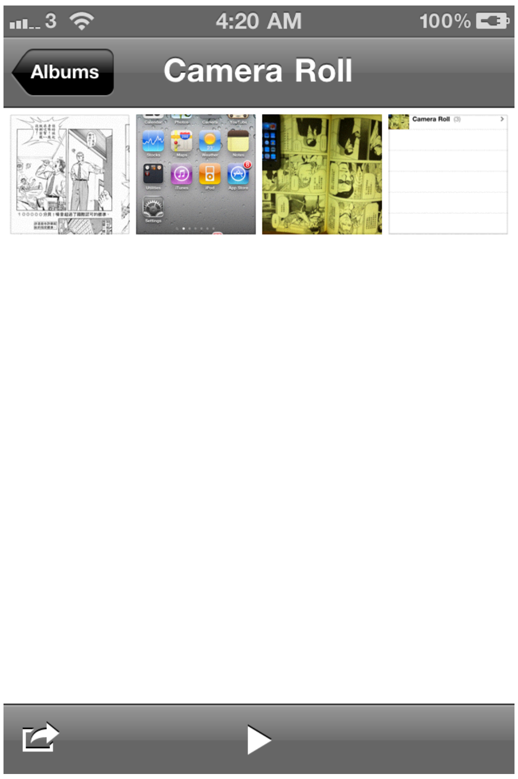


Figure 4.21 - Table view of the viewer

Figure 4.22 - Navigation View.

Note that there is a navigation bar on top

The first version of the viewer provides the following function:

◆ User can choose the comic they want and the comic will shown respectively

◆ The previous page and next page button can be passed to navigating through different pages.

The first version is simple but gives the backbone of the program.

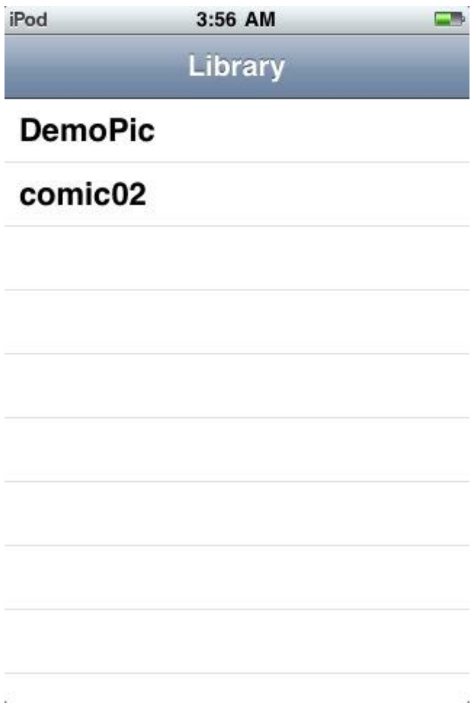


Figure 4.23 - The main view of the app, users can choose the comic they want

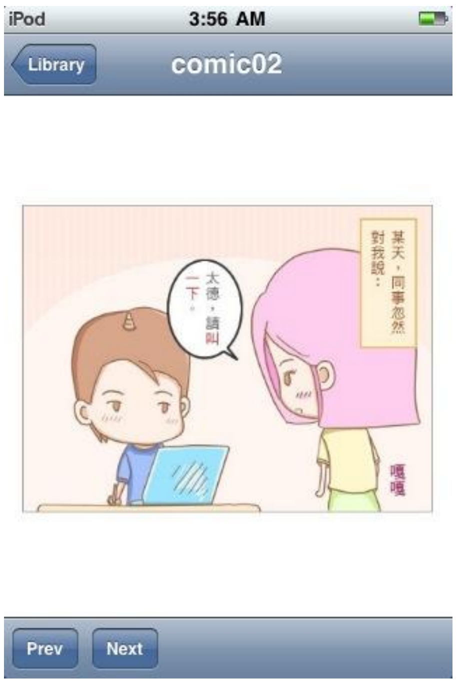


Figure 4.24 - The viewer

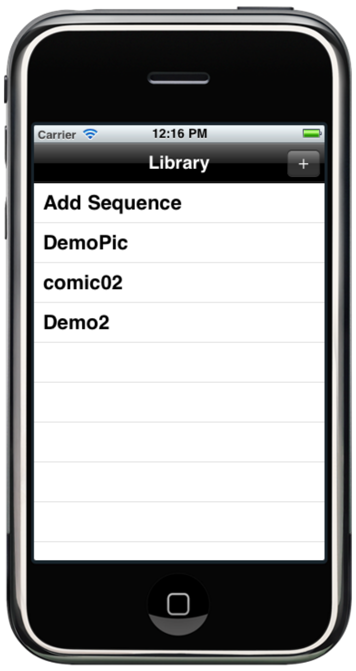
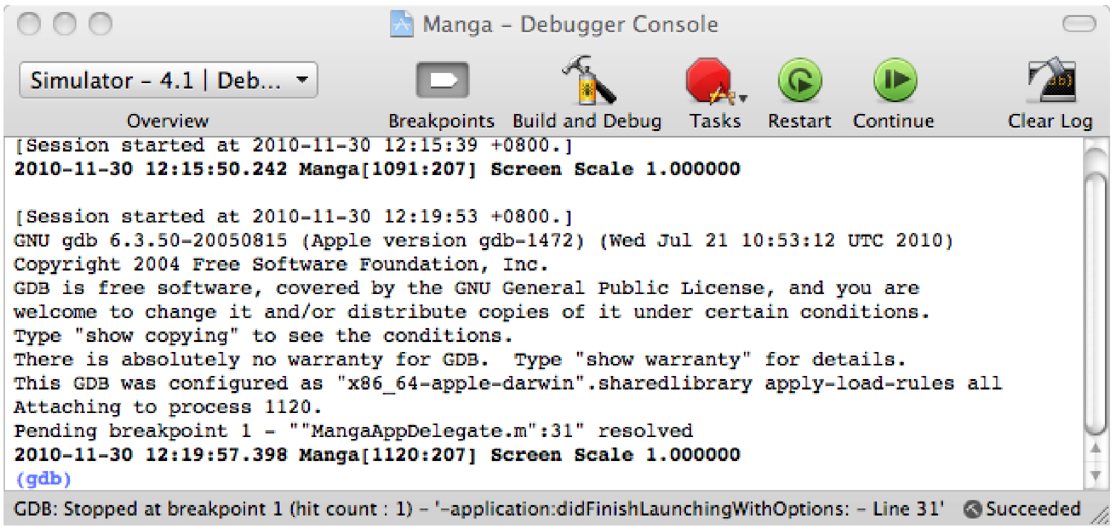


Figure 4.25 - Debugging on the iOS Simulator on Mac OS X



```
[Session started at 2010-11-30 12:15:39 +0800.]
2010-11-30 12:15:50.242 Manga[1091:207] Screen Scale 1.000000

[Session started at 2010-11-30 12:19:53 +0800.]
GNU gdb 6.3.50-20050815 (Apple version gdb-1472) (Wed Jul 21 10:53:12 UTC 2010)
Copyright 2004 Free Software Foundation, Inc.
GDB is free software, covered by the GNU General Public License, and you are
welcome to change it and/or distribute copies of it under certain conditions.
Type "show copying" to see the conditions.
There is absolutely no warranty for GDB.  Type "show warranty" for details.
This GDB was configured as "x86_64-apple-darwin".sharedlibrary apply-load-rules all
Attaching to process 1120.
Pending breakpoint 1 - ""MangaAppDelegate.m":31" resolved
2010-11-30 12:19:57.398 Manga[1120:207] Screen Scale 1.000000
(gdb)
```

GDB: Stopped at breakpoint 1 (hit count : 1) – '-application:didFinishLaunchingWithOptions: – Line 31' Succeeded

Figure 4.26 - Debugging by GDB

After the basic part is implemented, more advanced functions are added to it. One thing is the scrolling and zooming in the viewer. Besides it, the Segmentation module and Storage and Transmission module is also implemented.

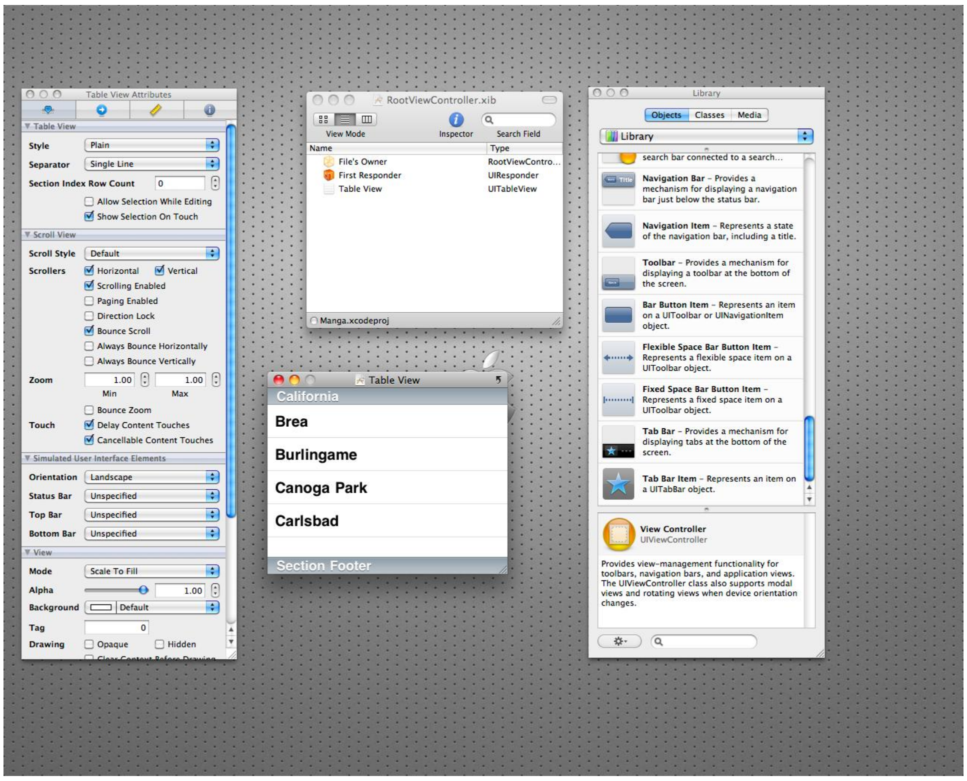


Table View
- Style: Plain
- Separator: Single Line
- Section Index Row Count: 0
- ☐ Allow Selection While Editing
- ✓ Show Selection On Touch

Scroll View
- Scroll Style: Default
- Scrollers: ✓ Horizontal ✓ Vertical
- ✓ Scrolling Enabled
- ☐ Paging Enabled
- ☐ Direction Lock
- ✓ Bounce Scroll
- ☐ Always Bounce Horizontally
- ☐ Always Bounce Vertically
- Zoom: 1.00 Min, 1.00 Max
- ☐ Bounce Zoom
- Touch: ✓ Delay Content Touches
- ✓ Cancellable Content Touches

Simulated User Interface Elements
- Orientation: Landscape
- Status Bar: Unspecified
- Top Bar: Unspecified
- Bottom Bar: Unspecified

View
- Mode: Scale To Fill
- Alpha: 1.00
- Background: Default
- Tag: 0
- Drawing: ☐ Opaque ☐ Hidden

Name	Type
File's Owner	RootViewContro...
First Responder	UIResponder
Table View	UITableView

California
Brea
Burlingame
Canoga Park
Carlsbad
Section Footer

search bar connected to a search...

Navigation Bar – Provides a mechanism for displaying a navigation bar just below the status bar.

Navigation Item – Represents a state of the navigation bar, including a title.

Toolbar – Provides a mechanism for displaying a toolbar at the bottom of the screen.

Bar Button Item – Represents an item on a UIToolbar or UINavigationItem object.

Flexible Space Bar Button Item – Represents a flexible space item on a UIToolbar object.

Fixed Space Bar Button Item – Represents a fixed space item on a UIToolbar object.

Tab Bar – Provides a mechanism for displaying tabs at the bottom of the screen.

Tab Bar Item – Represents an item on a UITabBar object.

View Controller
UIViewController

Provides view-management functionality for toolbars, navigation bars, and application views. The UIViewController class also supports modal views and rotating views when device orientation changes.

Figure 4.27 - Screen capture of the development of the interface.

Currently, the following functions are being supported in the Viewer App:

- Sequence and Panel Zooming Generating
- Viewing of Comics, including zooming and scrolling through pinching. Automatic playing is also supported with predefined metadata
- Sharing of Metadata of sequence and zooming information through internet

Chapter 5 Experimental Result

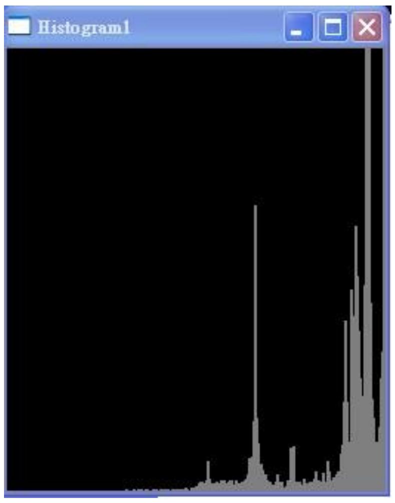
In this chapter, we are going to talk about the experimental results of the algorithm study as the following list:

- Histogram
- Contour Finding
- Hough Lines
- Watershed Segmentation
- Flood-fill

5.1 Histogram

We will describe the Histograms of 5 test cases we chosen in this part.

Image: #1



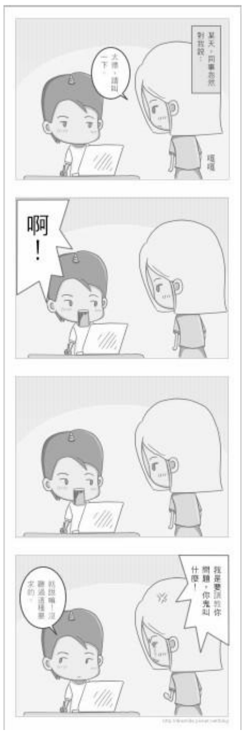


Figure 5.1 - Histogram for Image #1 *Figure 5.2 - Input image #1*

Image #1	Data
Max Value	762.277588
Index of Max. Value	242
Min Value	0.000000
Index of Min. Value	0

Table 5.1 - Peak value of Histogram for Image#1

Image: #2

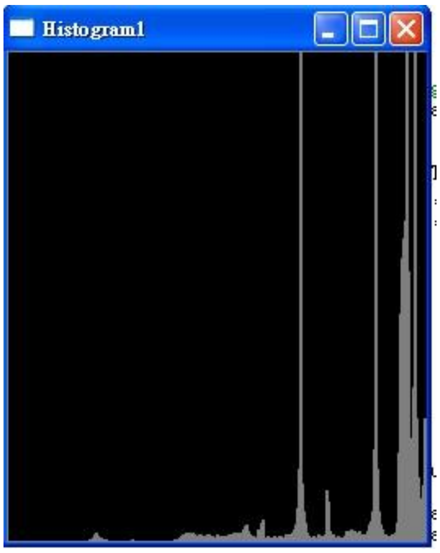


Figure 5.3 - Histogram for Image #2

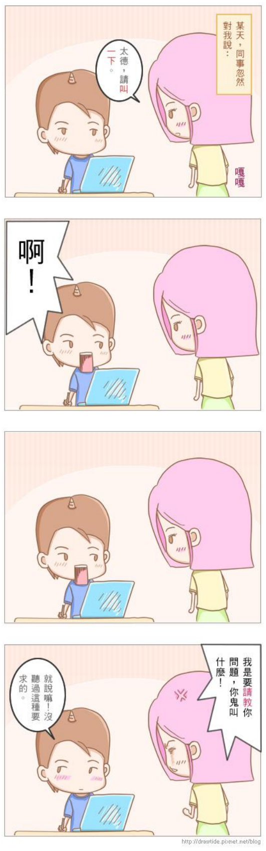


Figure 5.4 - Input image #2

This is another special case as we assume all images are in grey scale for development. We still include this image as it is used for testing Watershed Segmentation.

This image will be converted to grey scale before calculating the Histogram.

Image #2	Data
Max Value	1018.862000
Index of Max. Value	224
Min Value	0.000000
Index of Min. Value	0

Table 5.2 - Peak value of Histogram for Image#2

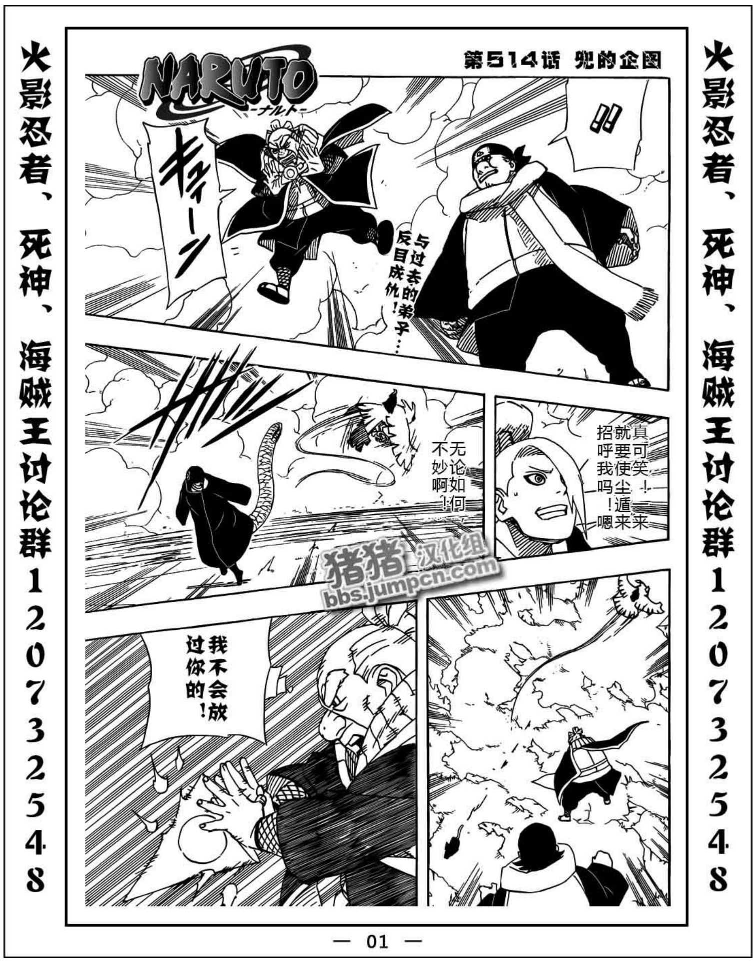

火影忍者、死神、海贼王讨论群120732548
NARUTO
第514话 兜的企图
!!
与过去的弟子反目成仇!……
真可笑!来就要使尘遁来招呼我吗!嗯
无论如何不妙啊!
我不会放过你的!
猪猪汉化组 bbs.jumpcn.com
— 01 —

Image: #3

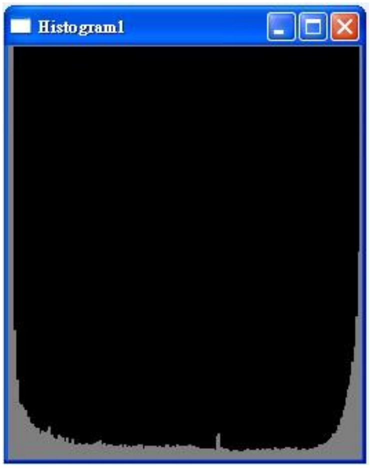


Figure 5.6 - Histogram for Image #3

Figure 5.5 - Input image #3

Image #3	Data
Max Value	656.016724
Index of Max. Value	0
Min Value	4.958656
Index of Min. Value	180

Table 5.3 - Peak value of Histogram for Image#3

This histogram is more special comparing with others as its index of max value is 0 which means pure black appears most frequently in this image. This is also related to the drawing style.

Image: #4

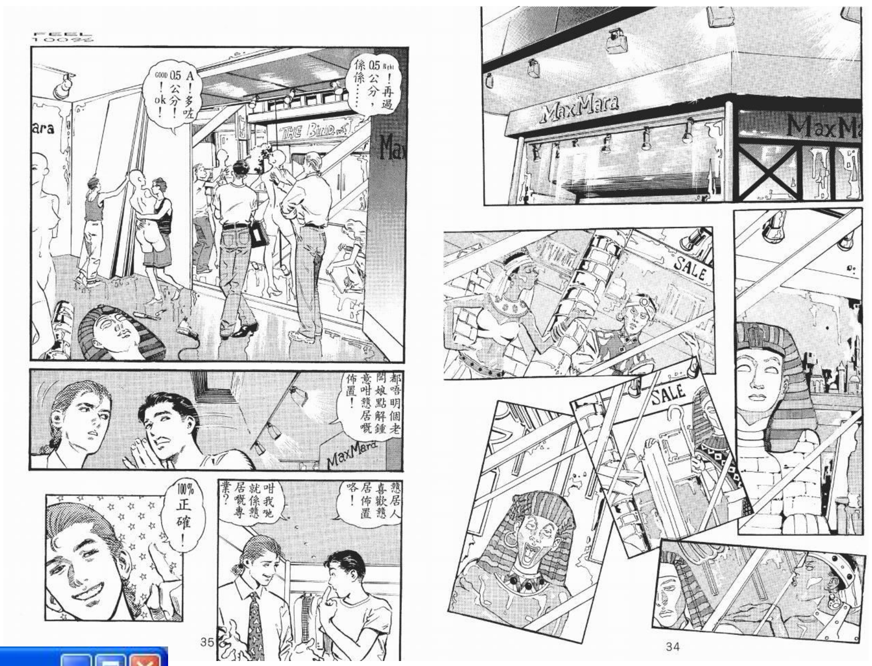


Figure 5.7 - Input image #4

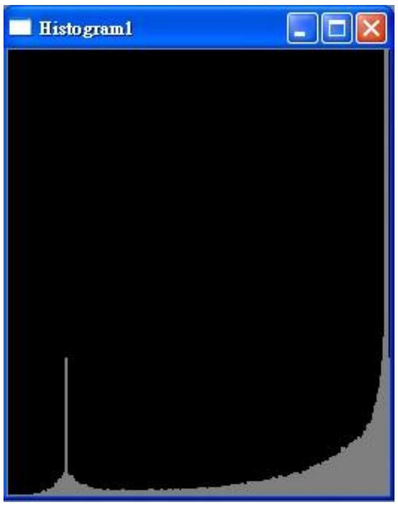


Figure 5.8 - Histogram for Image #4

Image #3	Data
Max Value	988.389404
Index of Max. Value	252
Min Value	0.000000
Index of Min. Value	1

Table 5.4 - Peak value of Histogram for Image#4

Image: #5

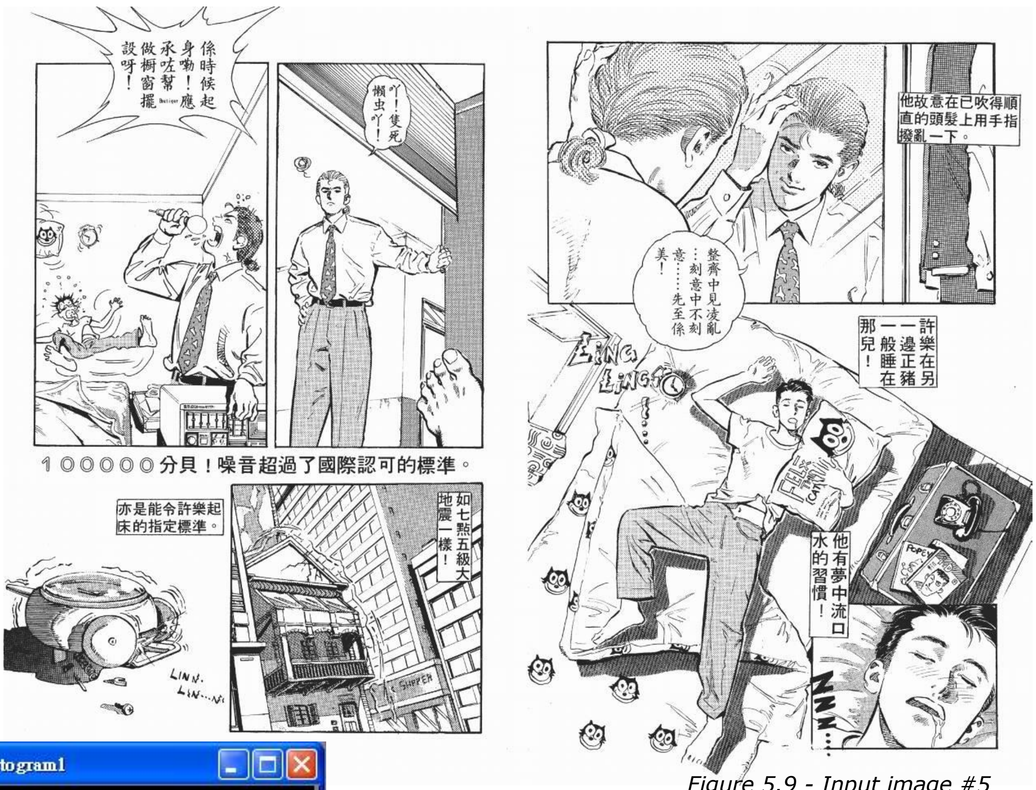

他故意在已吹得順直的頭髮上用手指撥亂一下。
整齊中見凌亂……刻意中不刻意……先至係美！
許樂在另一邊正豬一般睡在那兒！
LING LING
他有夢中流口水的習慣！
NNN…
係時候起身嘞！應承咗幫Boutique做櫥窗擺設呀！
吖！隻死懶虫吖！
100000分貝！噪音超過了國際認可的標準。
如七點五級大地震一樣！
亦是能令許樂起床的指定標準。
LINN. LIN…N

Figure 5.9 - Input image #5

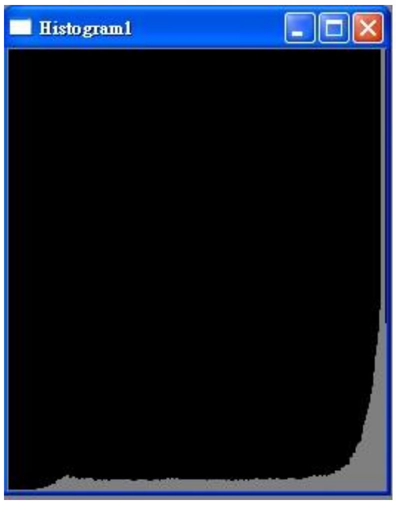


Figure 5.10 - Histogram for Image #5

Image #3	Data
Max Value	1286.586304
Index of Max. Value	252
Min Value	0.000000
Index of Min. Value	0

Table 5.5 - Peak value of Histogram for Image#5

Except image #3, max value relates to color index 220 to 255. This means, in general, white or a color near to white appears most frequently.

5.2 Contour Finding

Recalling that we have tried and compared two methods for image binarization:

Method1: Directly use cvThreshold()

Method2:

- Get background color through 4 sample points and take average value
- Calculate distance between background color and the image
- Binarized the image by a threshold, same as the one cvThreshold() used

We will describe the results of several test cases for the two methods respectively here.

Case: Pre-test
Purpose: to evaluate the threshold for further testing

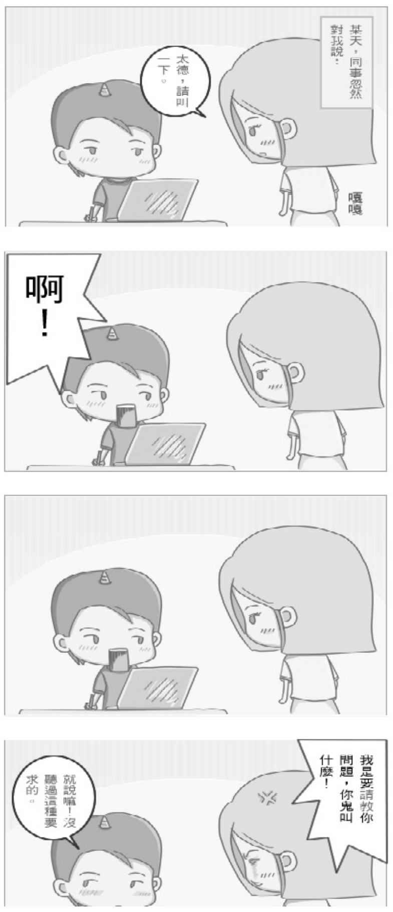
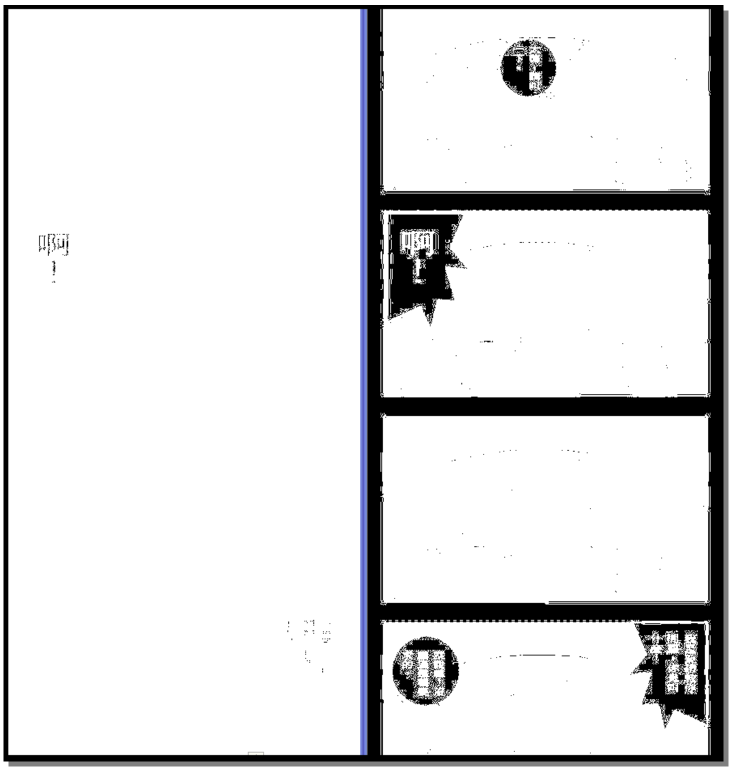


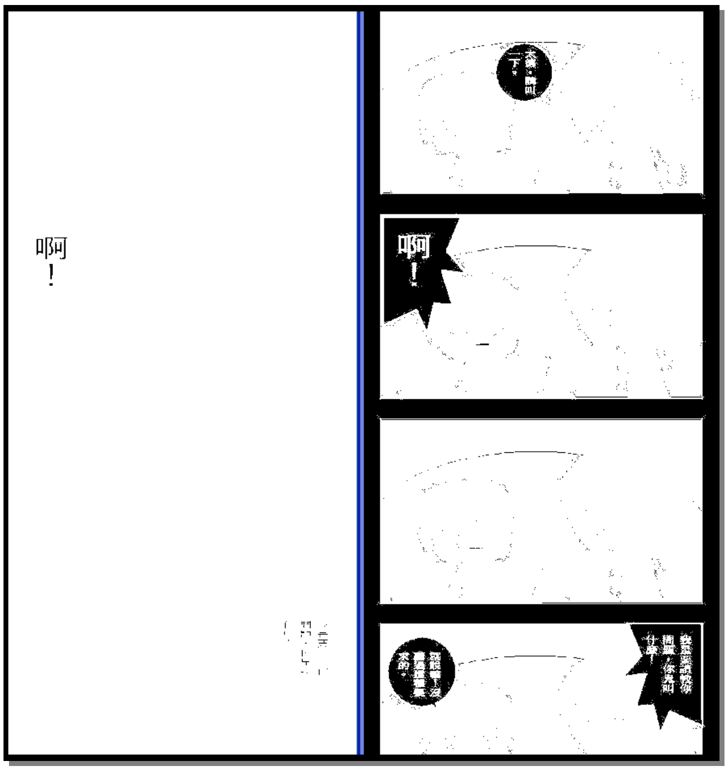
Figure 5.11 - Input image for Pre-test case

We have tried 5 sets of parameters for binarization. The following figures show the results of different parameters for the two methods. The left part is the result of method 1 while the right one is of method 2.



Case: Pre-test
Remarks:
Contour result
on threshold 0

Figure 5.12 - Binary results of Pre-test case on threshold 0



Case: Pre-test
Remarks: Contour result
on threshold 10

Figure 5.13 - Binary results of Pre-test case on threshold 10

Case: Pre-test
Remarks: Contour result on threshold 100

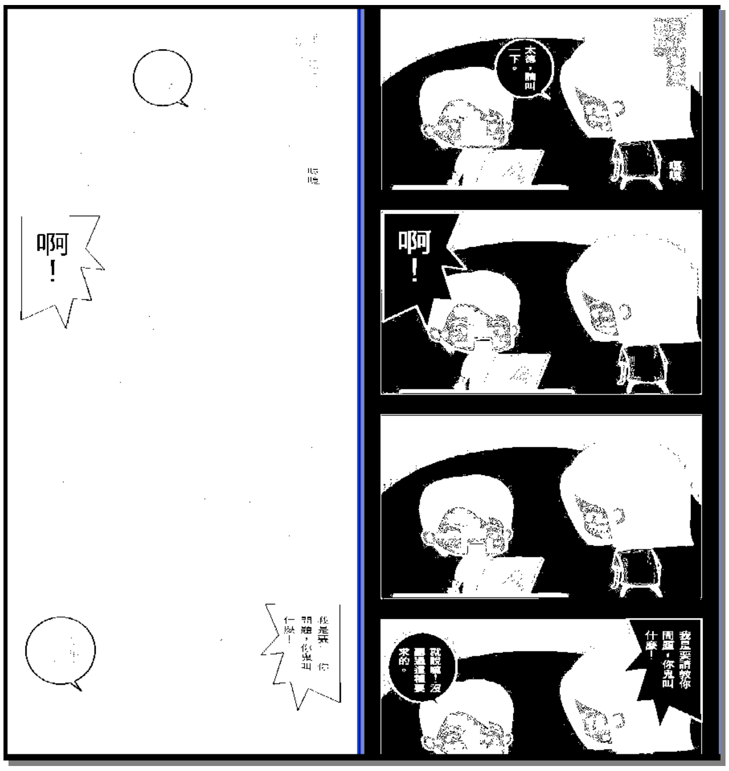
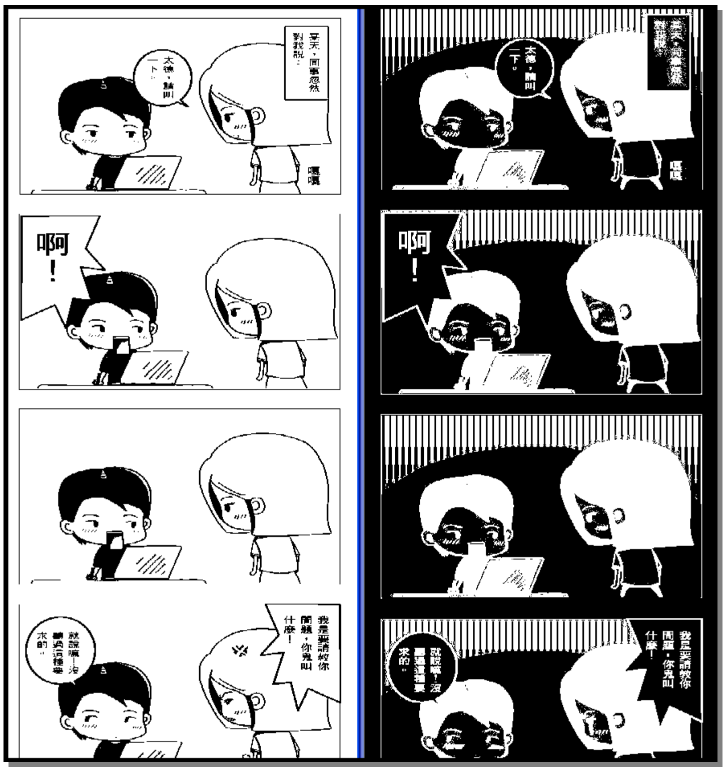


Figure 5.14 - Binary results of Pre-test case on threshold 100



Case: Pre-test
Remarks: Contour result on threshold 200

Figure 5.15 - Binary results of Pre-test case on threshold 200

Case: Pre-test
Remarks: Contour result on threshold 255

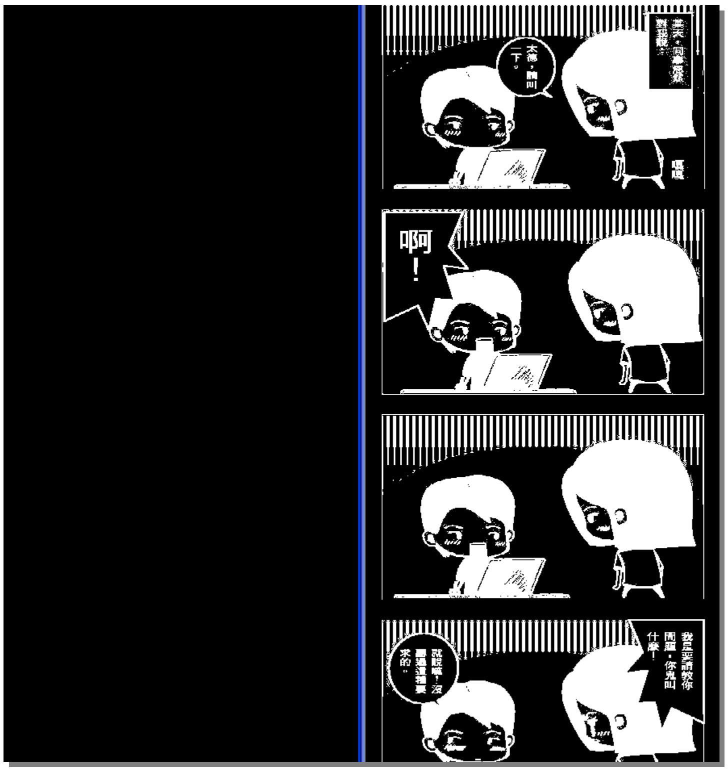


Figure 5.16 - Binary results of Pre-test case on threshold 255

By the pre-test case, we have chosen the threshold on 10 out of 255 for the following test cases. We focus on the right part of the result of the threshold on 10. With this result, we can get the outline of the four panels in a relatively good quality. Besides, we can also get the bounding box of the panels, which is some important information for zooming. However, as this is a very simple input image, we have to try some more complicated ones.

Following test cases is the full testing, including contour finding and bounding box locating.

Case: #1
Purpose: to find the contours as well as bounding box for a simplest Four Column Comic
Remark: Same as Pre-test case input

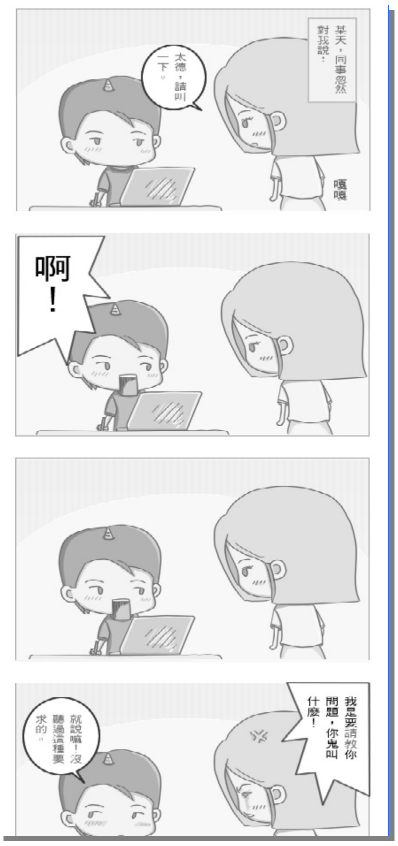


Figure 5.17 - Input Image for contour finding, Test case 1

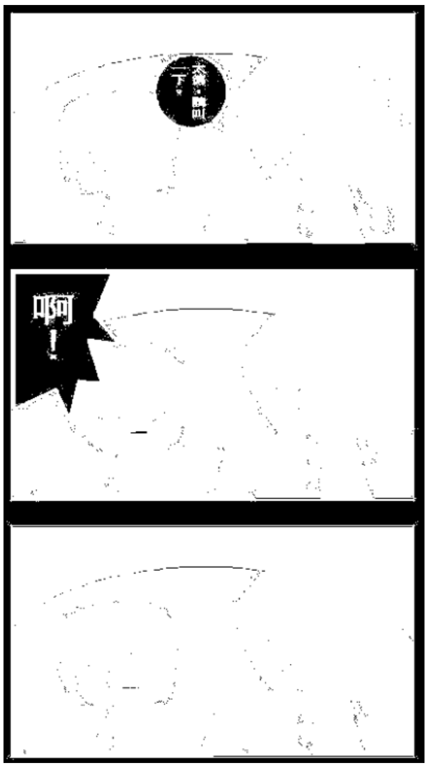


Figure 5.18 - Binary Image for contour finding, Test case 1

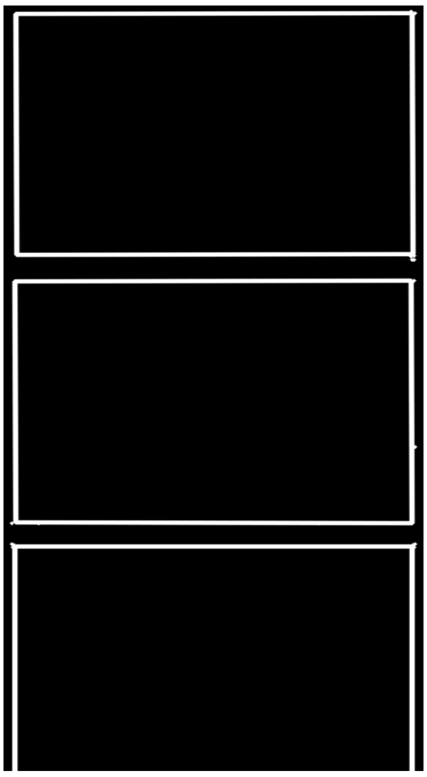
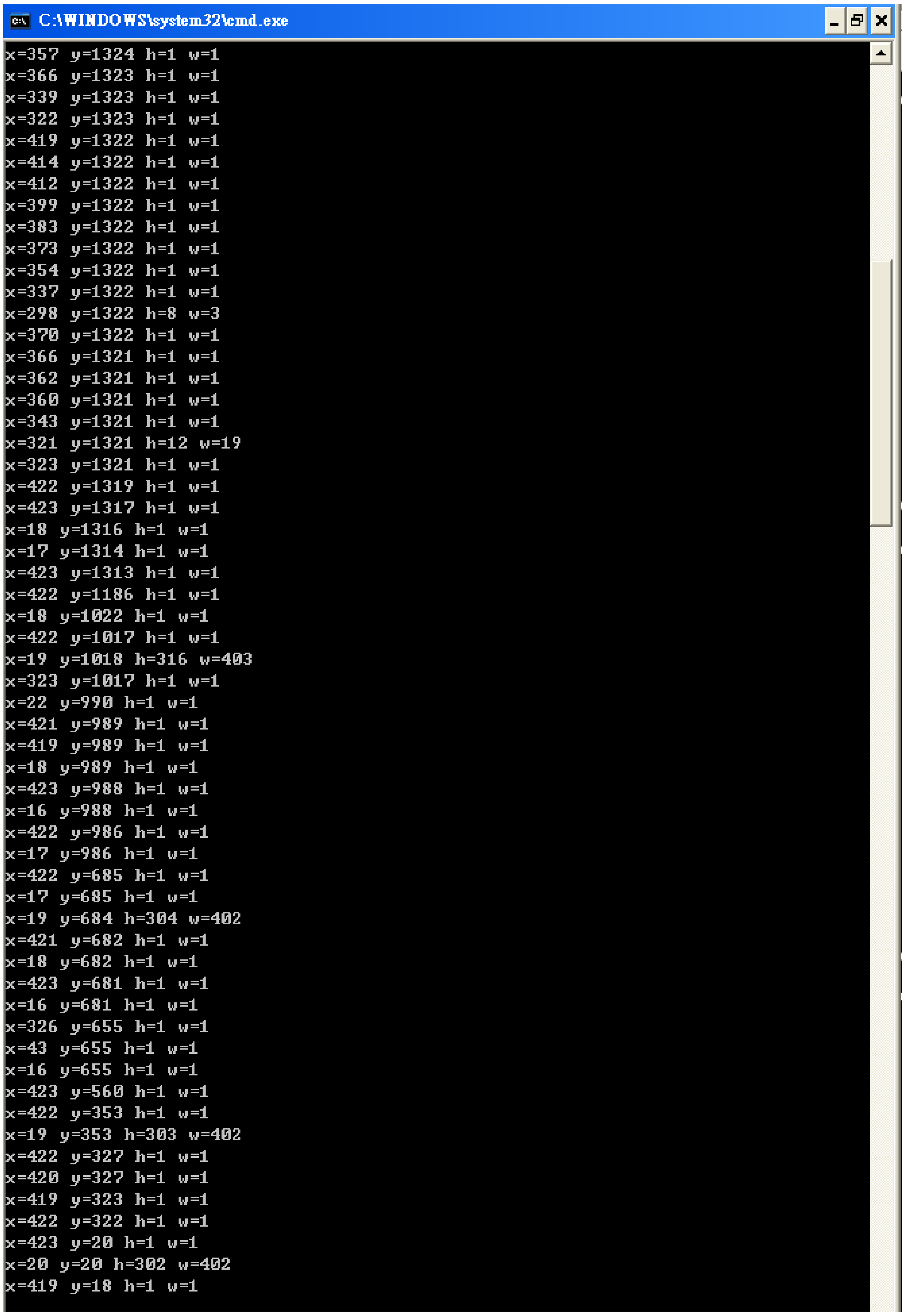


Figure 5.19 - Contour result (Partly), Test case 1

From the contour image shown above, we can have a very clear panel outline of the test case 1. The following screen dump shows the statistics of the bounding box.



```
x=357 y=1324 h=1 w=1
x=366 y=1323 h=1 w=1
x=339 y=1323 h=1 w=1
x=322 y=1323 h=1 w=1
x=419 y=1322 h=1 w=1
x=414 y=1322 h=1 w=1
x=412 y=1322 h=1 w=1
x=399 y=1322 h=1 w=1
x=383 y=1322 h=1 w=1
x=373 y=1322 h=1 w=1
x=354 y=1322 h=1 w=1
x=337 y=1322 h=1 w=1
x=298 y=1322 h=8 w=3
x=370 y=1322 h=1 w=1
x=366 y=1321 h=1 w=1
x=362 y=1321 h=1 w=1
x=360 y=1321 h=1 w=1
x=343 y=1321 h=1 w=1
x=321 y=1321 h=12 w=19
x=323 y=1321 h=1 w=1
x=422 y=1319 h=1 w=1
x=423 y=1317 h=1 w=1
x=18 y=1316 h=1 w=1
x=17 y=1314 h=1 w=1
x=423 y=1313 h=1 w=1
x=422 y=1186 h=1 w=1
x=18 y=1022 h=1 w=1
x=422 y=1017 h=1 w=1
x=19 y=1018 h=316 w=403
x=323 y=1017 h=1 w=1
x=22 y=990 h=1 w=1
x=421 y=989 h=1 w=1
x=419 y=989 h=1 w=1
x=18 y=989 h=1 w=1
x=423 y=988 h=1 w=1
x=16 y=988 h=1 w=1
x=422 y=986 h=1 w=1
x=17 y=986 h=1 w=1
x=422 y=685 h=1 w=1
x=17 y=685 h=1 w=1
x=19 y=684 h=304 w=402
x=421 y=682 h=1 w=1
x=18 y=682 h=1 w=1
x=423 y=681 h=1 w=1
x=16 y=681 h=1 w=1
x=326 y=655 h=1 w=1
x=43 y=655 h=1 w=1
x=16 y=655 h=1 w=1
x=423 y=560 h=1 w=1
x=422 y=353 h=1 w=1
x=19 y=353 h=303 w=402
x=422 y=327 h=1 w=1
x=420 y=327 h=1 w=1
x=419 y=323 h=1 w=1
x=422 y=322 h=1 w=1
x=423 y=20 h=1 w=1
x=20 y=20 h=302 w=402
x=419 y=18 h=1 w=1
```

Figure 5.20 - Screen dump of statistics of bounding box, Test case 1

In this screen dump, x and y mean the starting x, y coordinates of the bounding rectangle while h and w are the height and width of the rectangle respectively. We can see that most of the bounding boxes have width of 1 unit and height of 1 unit. These are all noise as they are single points. In addition, we have a rectangle with width of 19 units and height of 12 units. Its size is still too small to be a bounding box. Hence, it is noise too. All these needed to be eliminated by the algorithm when performing panel extraction. Noted that there are only 4 boxes in a reasonable size, this is what we want.

Case: #2
Purpose: to try finding the contours as well as bounding box for image with overlapped panels

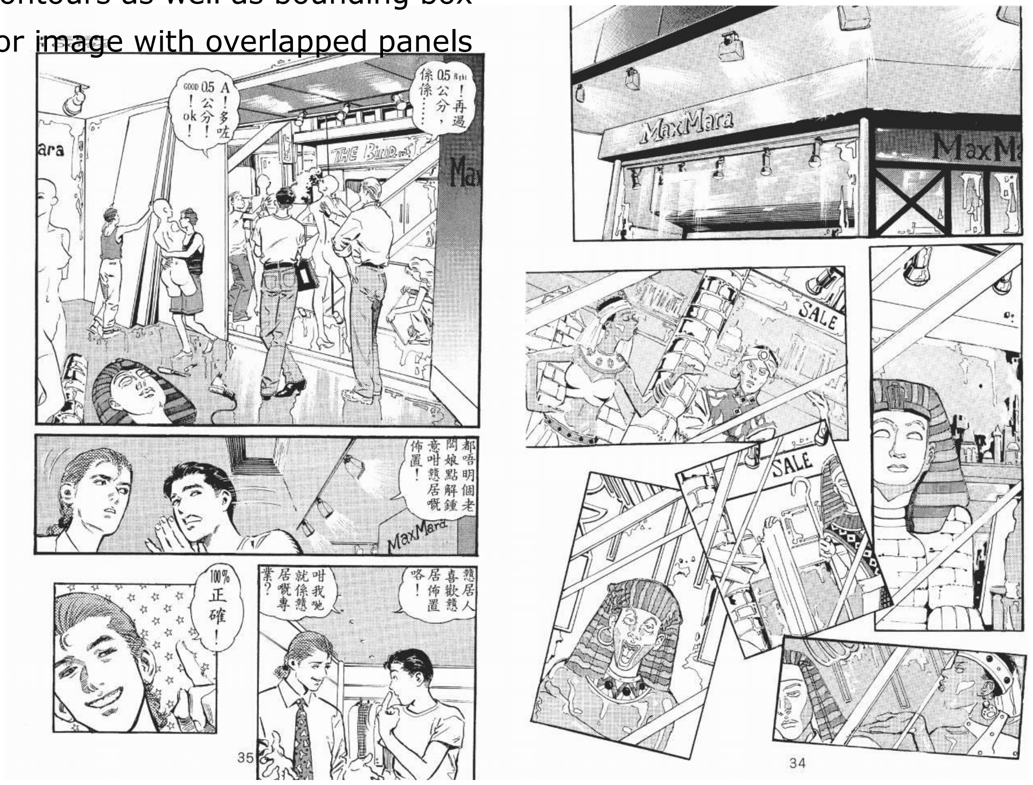


Figure 5.21 - Input image for contour finding, Test case 2

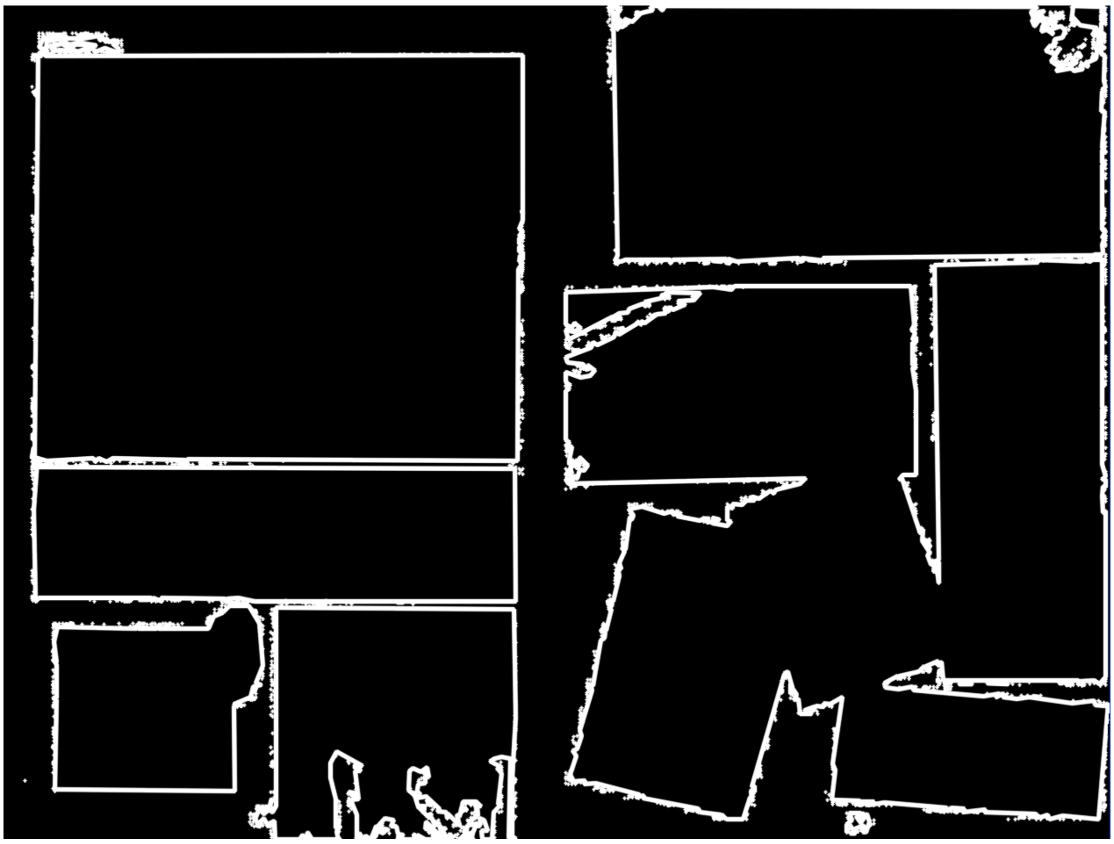
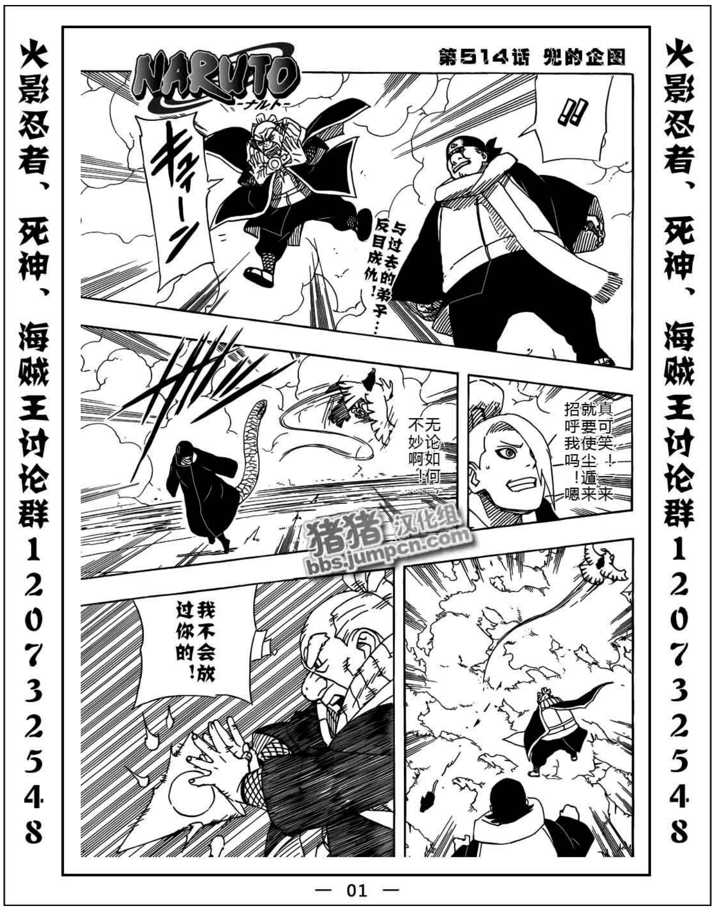


Figure 5.22 - Contour result, Test case 2

Test case 2 outputs a decent contour image as the one shown above. However, when it comes to bounding box, it cannot provide sufficient information as there are still some diagonals and some of the panels are not disjointed.


火影忍者、死神、海贼王讨论群120732548
NARUTO -ナルト-
第514话 兜的企图
与过去的弟子反目成仇……
真可笑！来就要使尘遁来招呼我吗！嗯
无论如何！不妙啊！
猪猪汉化组 bbs.jumpcn.com
我不会放过你的！
— 01 —

Case: #3
Purpose: to find the contours as well as bounding box for image with strong strokes and crossed-panel-blobs

Figure 5.23 - Input image for contour finding, Test case 3

This test case gave a total black result. There is only one bounding box which means only the universal bounding box is detected and no contour is found. This is because of the threshold. The threshold for test case 1 cannot fit this test case. Therefore, we have other test cases with the same image and threshold tuned. However, the results are not encouraging.

Case: #4

Test case 4 makes use of the original image of case 3, with threshold on 255 for demonstration purpose only. Other results are not shown here as either they have too complicated contours or the contours found not informative enough.

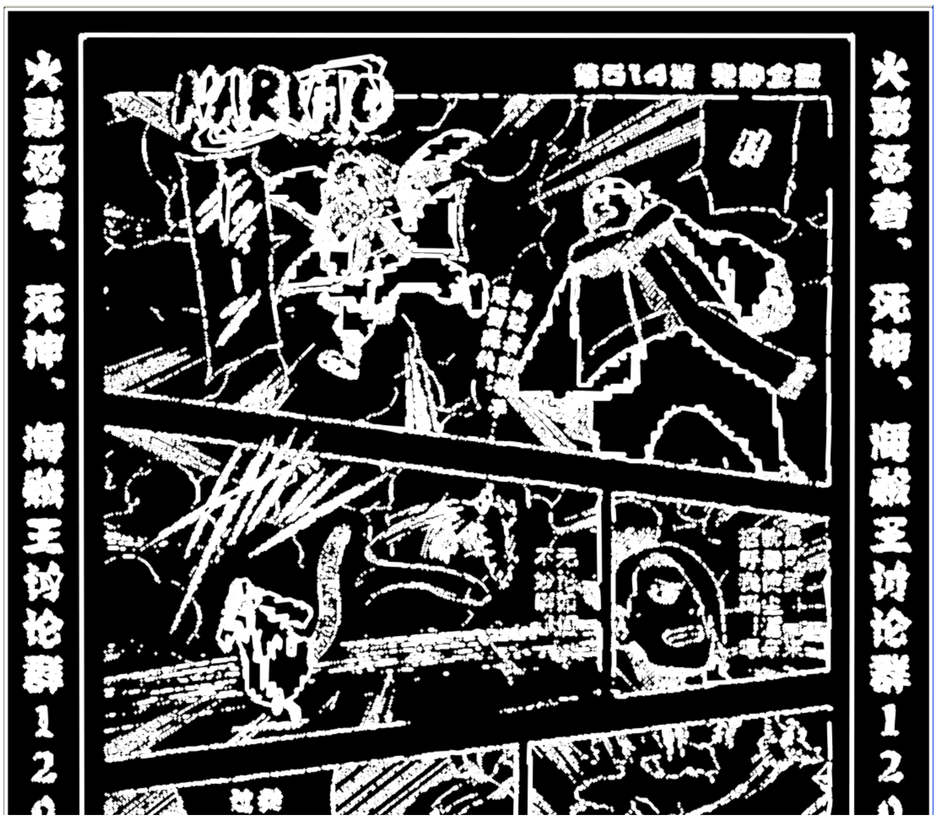


Figure 5.24 - Contour result, Test case 4

The above test cases show one thing – contour finding is a good tool to extract the panels. Yet, it is not powerful enough as we have to manually tune the threshold for comic pages in different style. Not only this, contour detection is not informative enough and we have to make use of other tools to gather more information. When we add up all results, we hope that some meaningful data can be retrieved to achieve the goal.

Summary:

- Comics with different style need different parameters to find out the panels.
- Contour is not enough for finding out the panel for all comic images.

5.3 Hough Lines Finding

As expected, the results cannot cover every edge of the panels.

Case: #1
Purpose: to find the Hough lines in a simple Four-Column Comic page

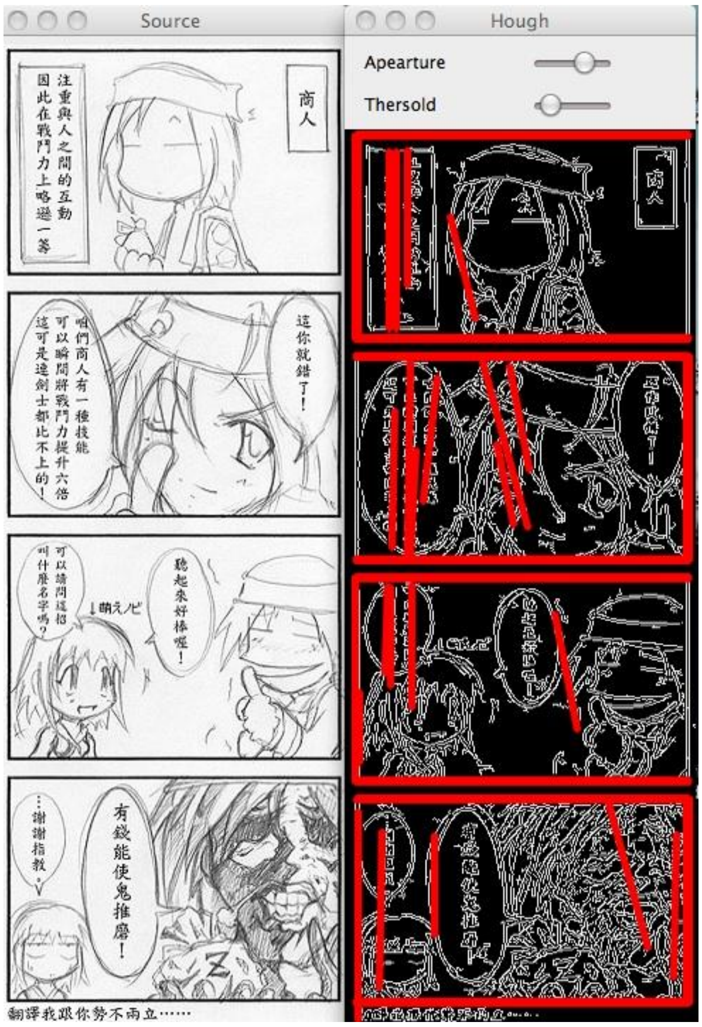


Figure 5.25 - Input and result for Hough Line finding, Test case 1

Case: #2

Purpose: to find the Hough lines in a traditional comic page with open panels

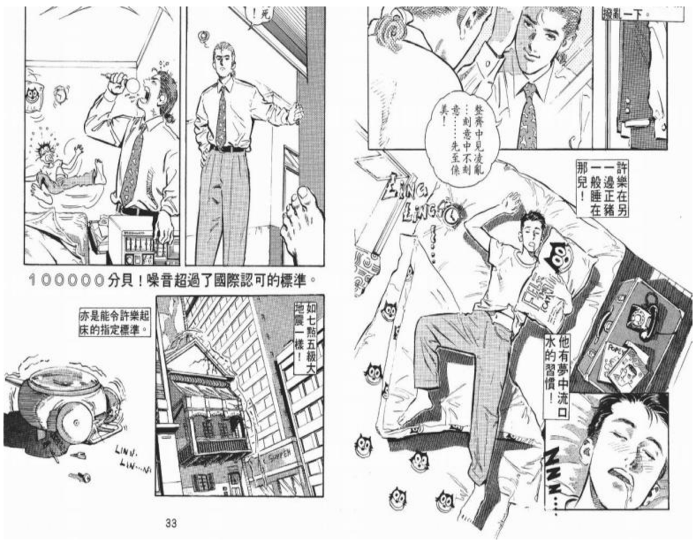

!死
撥亂一下。
整齊中見凌亂…刻意中不刻意……先至係美！
許樂在另一邊正豬一般睡在那兒！
LING LING
100000分貝！噪音超過了國際認可的標準。
亦是能令許樂起床的指定標準。
如七點五級大地震一樣！
他有夢中流口水的習慣！
LINN. LIN…N
NNN…
33

Figure 5.26 - Input image for Hough line finding, Test case 2

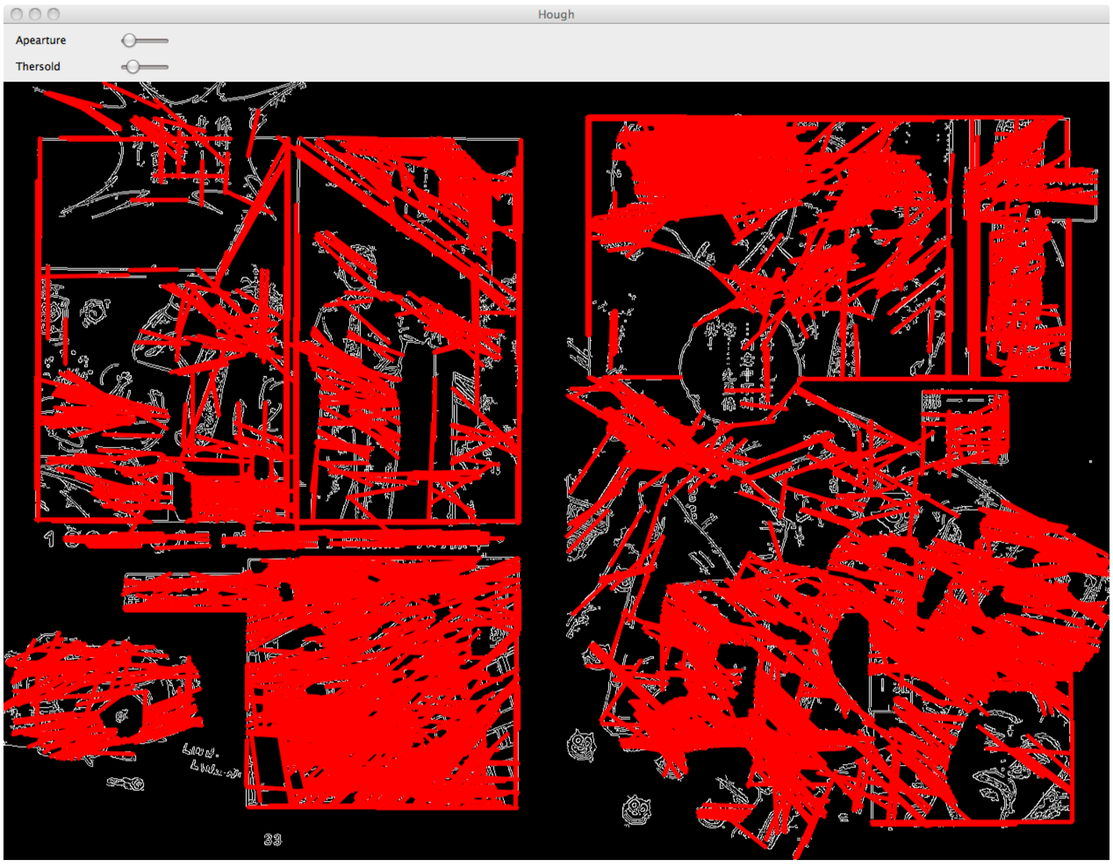


Figure 5.27 - Result of Hough line finding, Test case 2

Case: #3

Purpose: to find the Hough lines in a traditional comic page with overlapped panels

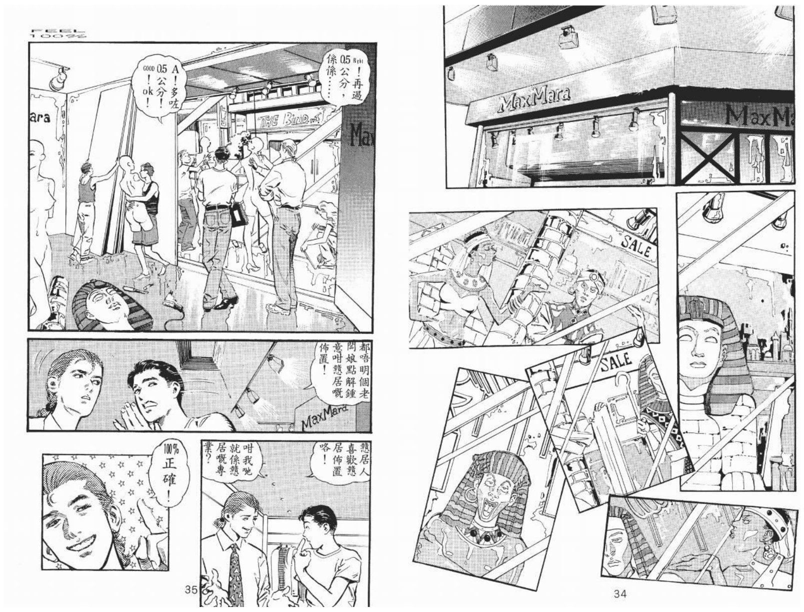


Figure 5.28 - Input for Hough line finding, Test case 3

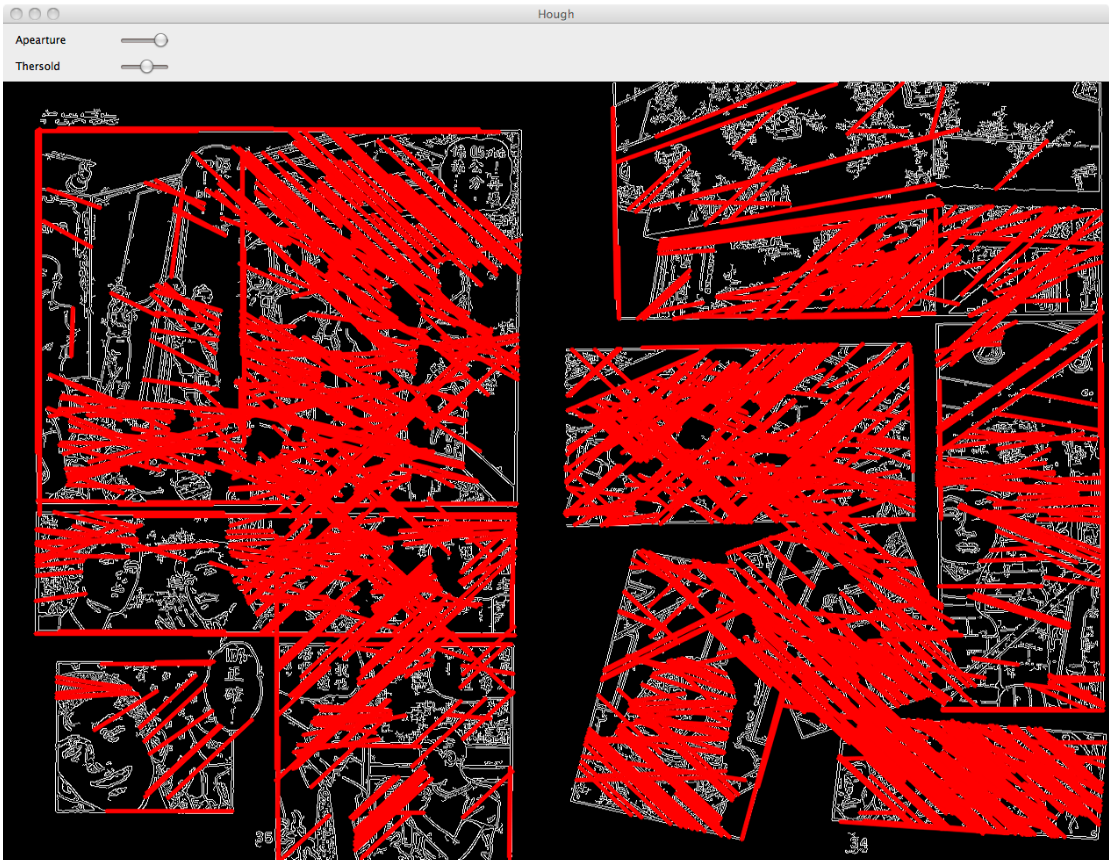


Figure 5.29 - Result for Hough line finding, Test case 3

Case: #4

Purpose: to find the Hough lines in a comic page with strong strokes

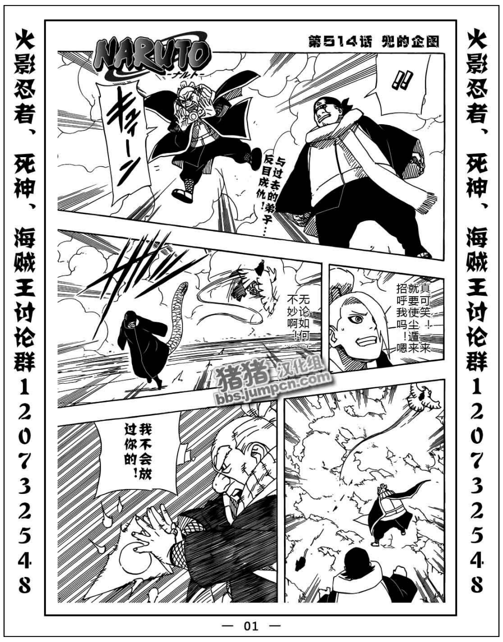

火影忍者、死神、海贼王讨论群1207325 48
NARUTO
第514话 兜的企图
キュイーン
!!
与过去的弟子…反目成仇!
真可笑! 就要使尘遁来招呼我吗! 嗯
无论如何不妙啊!
我不会放过你的!
猪猪汉化组 bbs.jumpcn.com
— 01 —

Figure 5.30 - Input for Hough line finding, Test case 4

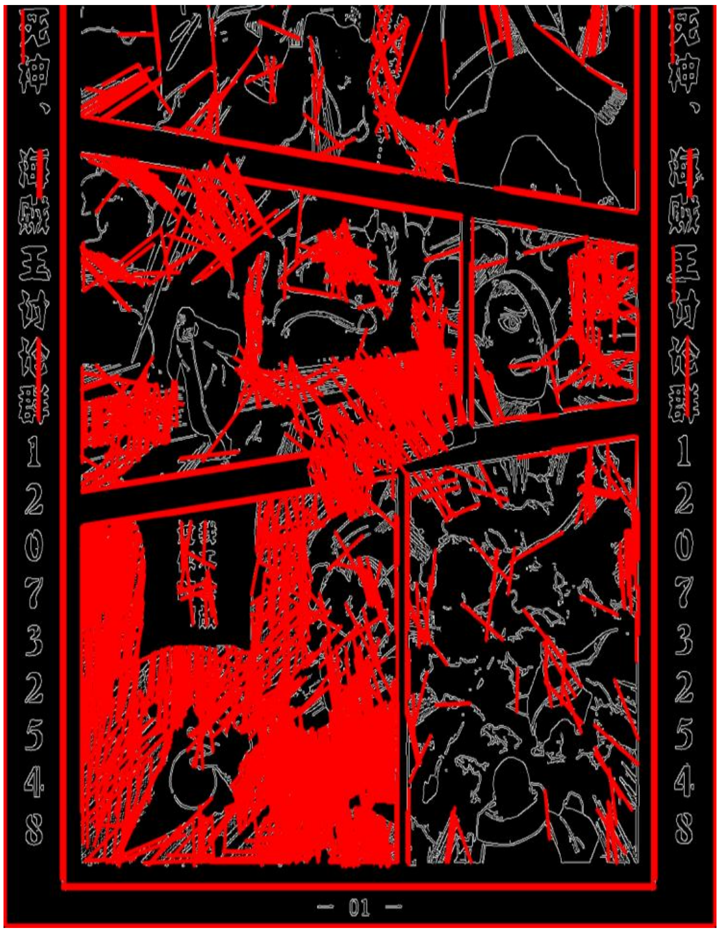


Figure 5.31 - Result for Hough line finding, Test case 4

The above results indicate that the Hough lines may overlap with the contours. Although this is not necessary, it is quite possible.

Another thing is that the longest Hough line in the result image has a strong probability to be a contour of the panel.

Summary:

- The longest Hough line in the result has a strong probability to be a contour of the panel.

5.4 Watershed Segmentation

We have 4 test cases for Watershed Segmentation which requires manually input the markers for panels.

Case: #1
Purpose: to perform Watershed Segmentation in a simple colorful Four-Column Comic page

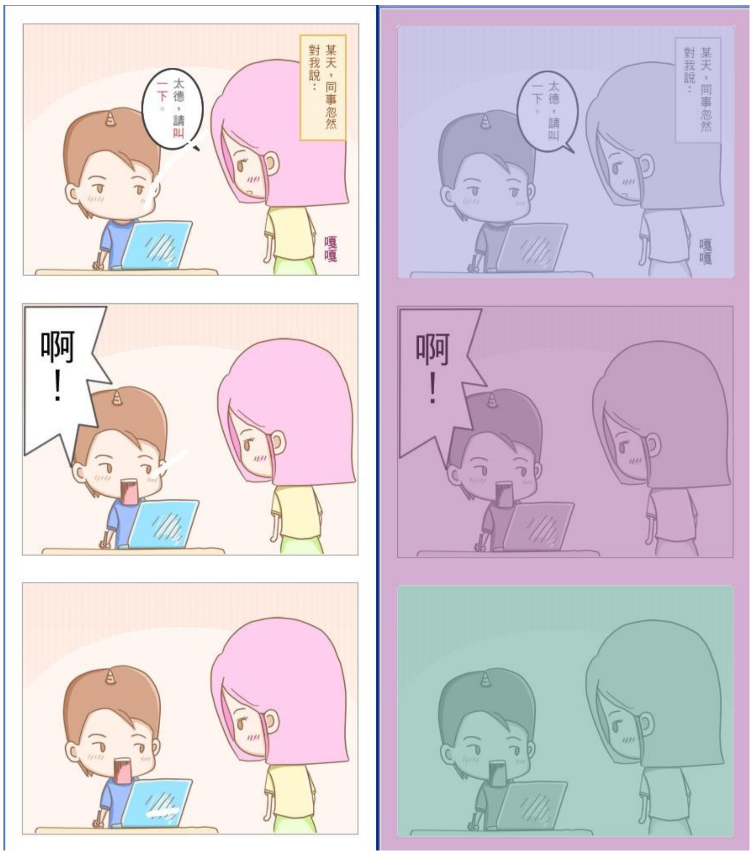
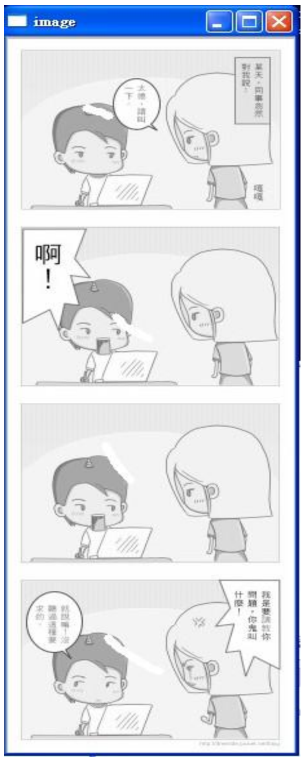


Figure 5.32 - Input and result for Watershed Segmentation, Test case 1

In this program, contours are calculated before performing Watershed Segmentation. Noted that under most situations, it can segment the panels that we have marked, but it is not successful all the time. Test case 1 shows that although we three panels marked, it turns out to have filled only two of them.



Case: #2

Purpose: to perform Watershed Segmentation in a simple Four-Column Comic page in grey scale

Remarks: Watershed Segmentation can be performed on both colorful and grey scale image

Figure 5.33 - Input image for Watershed Segmentation, Test case 2

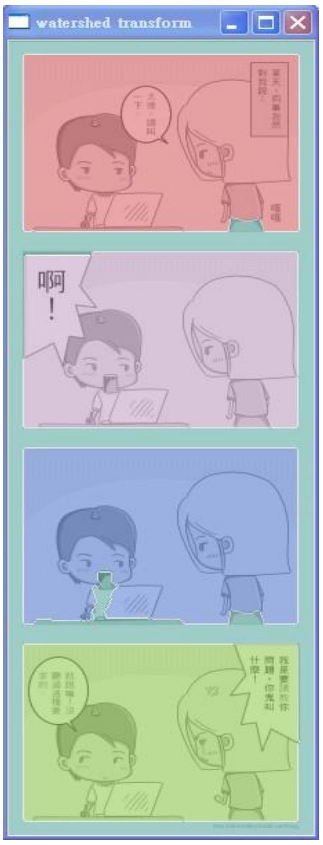


Figure 5.34 - Result image for Watershed Segmentation, Test case 2

Case: #3

Purpose: to perform Watershed Segmentation in a comic page with overlapped panels

Remarks:

a. Open panels not filled

b. Overlapped panels filled

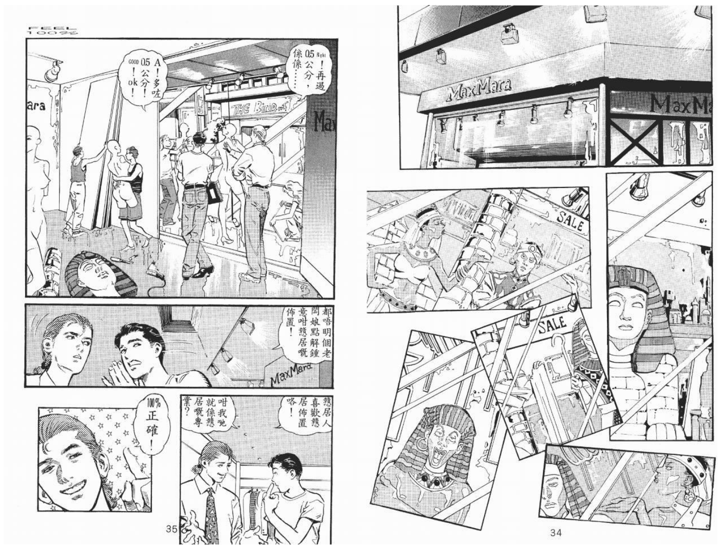


Figure 5.35 - Input image for Watershed Segmentation, Test case 3

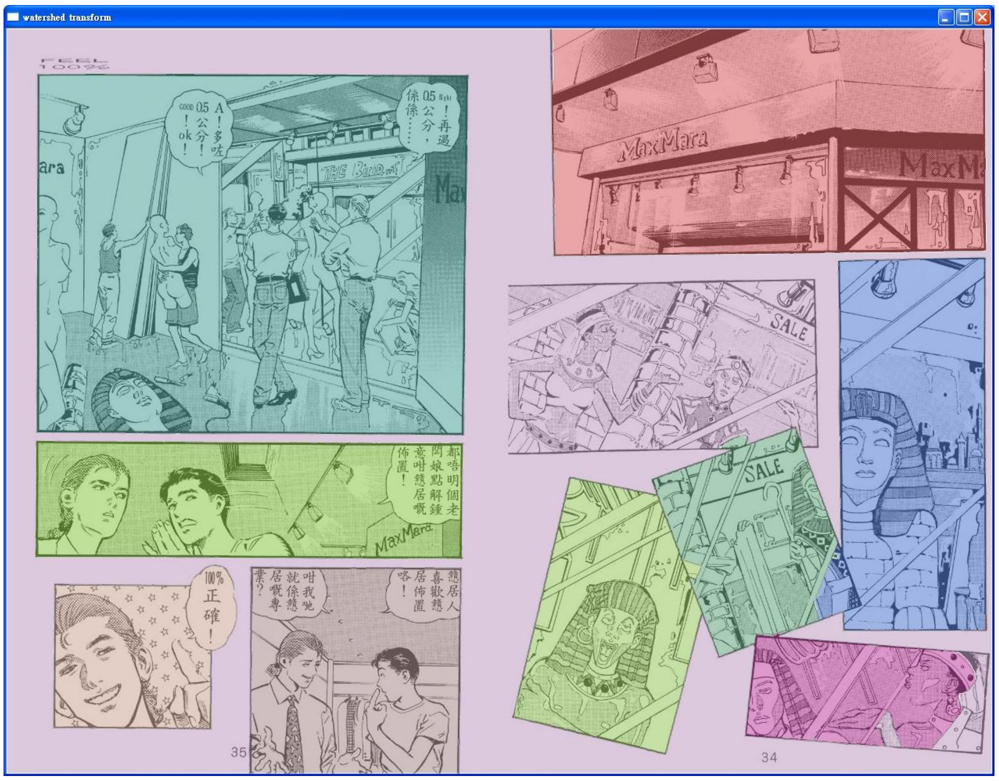


Figure 5.36 - Result of Watershed Segmentation, Test case 3

Case: #4

Purpose: to perform Watershed Segmentation in a comic page with open panels

Remarks:

As expected, open panels not filled

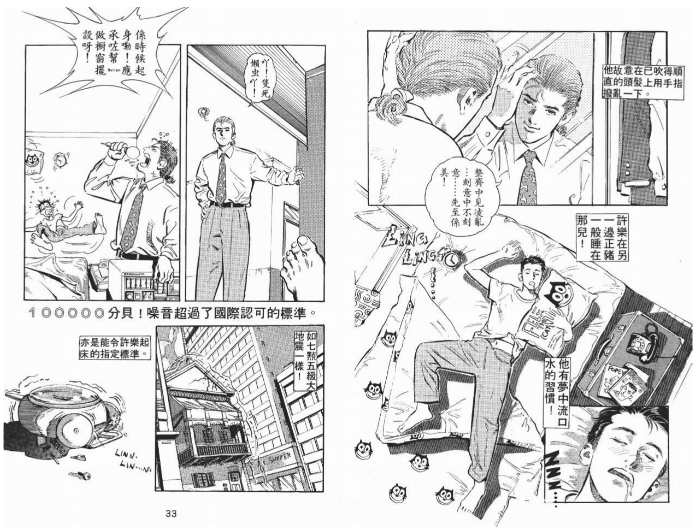

係時候起身嘞！應承咗幫Boutique做橱窗擺設呀！
吖！隻死懶虫吖！
100000分貝！噪音超過了國際認可的標準。
亦是能令許樂起床的指定標準。
如七點五級大地震一樣！
LINN. LIN...N
33
他故意在已吹得順直的頭髮上用手指撥亂一下。
整齊中見凌亂……刻意中不刻意……先至係美！
許樂在另一邊正豬一般睡在那兒！
LING LING
他有夢中流口水的習慣！
NNN…

Figure 5.37 - Input image for Watershed Segmentation, Test case 4

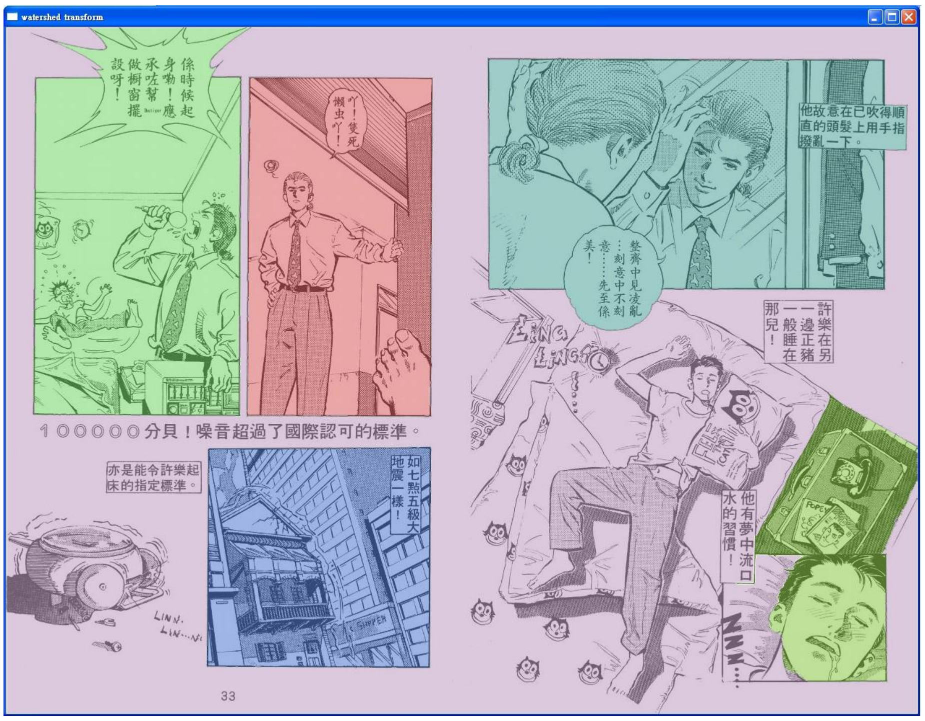

watershed transform
係時候起身嘞！應承咗幫Boutique做橱窗擺設呀！
吖！隻死懶虫吖！
100000分貝！噪音超過了國際認可的標準。
亦是能令許樂起床的指定標準。
如七點五級大地震一樣！
LINN. LIN...N
33
他故意在已吹得順直的頭髮上用手指撥亂一下。
整齊中見凌亂……刻意中不刻意……先至係美！
許樂在另一邊正豬一般睡在那兒！
LING LING
他有夢中流口水的習慣！
NNN…

Figure 5.38 - Result of Watershed Segmentation, Test case 4

As we can see in the above two example, there are some panels or some regions with unclear contour are treated as the background and are filled together with the background. However, some overlapped panels are successfully filled with different colors.

Case: #5
Purpose: to perform Watershed Segmentation in a comic page with strong strokes
Remarks: Output not accurate

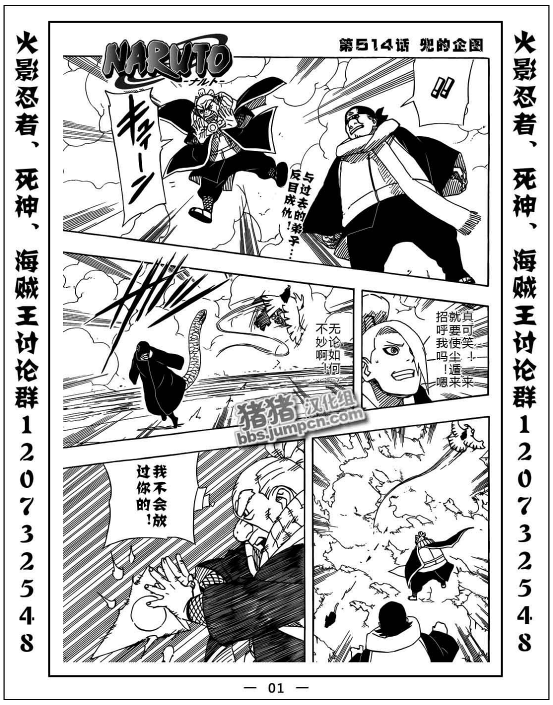
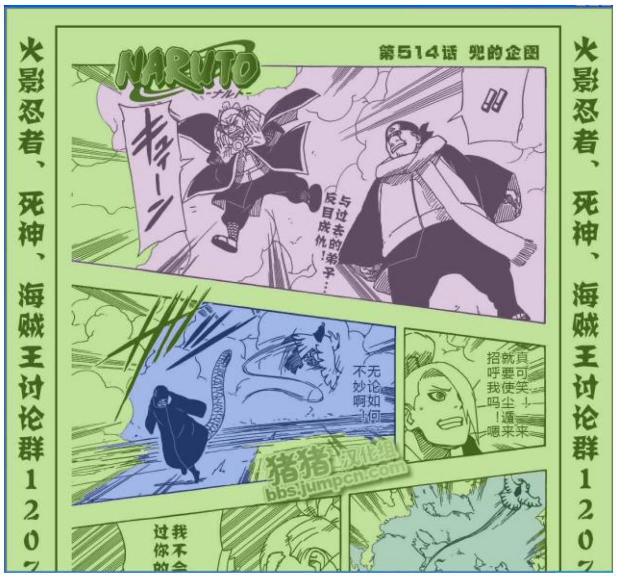

火影忍者、死神、海贼王讨论群120732548
NARUTO
第514话 兜的企图
与过去的弟子反目成仇!…
无论如何 不妙啊!
真可笑!就要使尘遁来招呼我吗!
我不会放过你的!
猪猪汉化组 bbs.jumpcn.com
01

Figure 5.39 - Input image for Watershed Segmentation, Test case 5



The strong strokes of test case 4 affect the output of Watershed Segmentation. This is illustrated on the left figure.

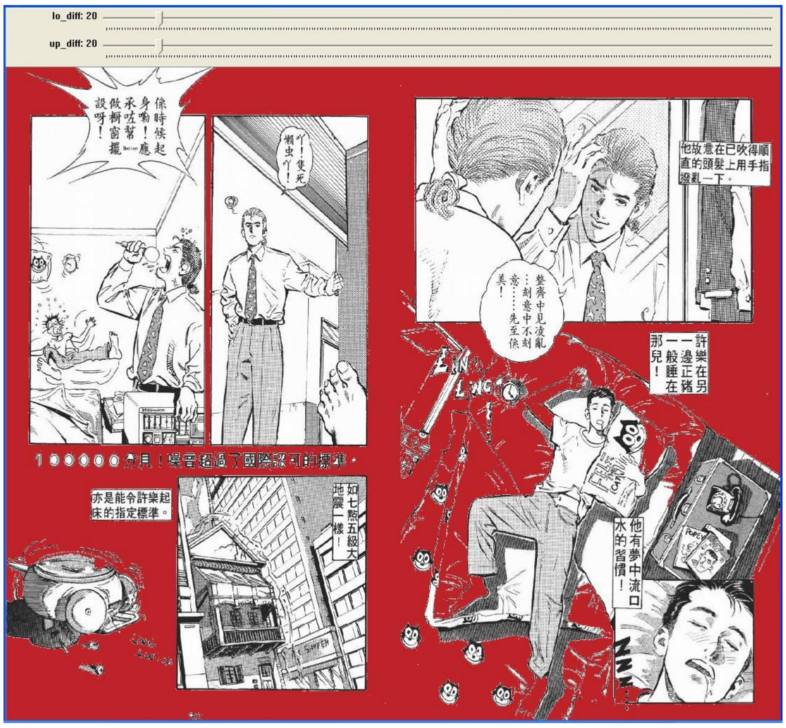
Figure 5.40 - Result of Watershed Segmentation, Test case 5

<u>Summary</u>

- Strong strokes might affect Watershed Segmentation so that a panel cannot be fully filled.
- A region must be strictly closed so that Watershed Segmentation can be properly performed.

5.5 Flood Fill

Since this part does not involve too much, we will only demonstrate some outputs below. The background are manually selected and filled as red.



lo_diff: 20
up_diff: 20

Figure 5.41 - Result of Flood-fill, Test case 1

Case: #1
Purpose: to perform Flood-fill in a comic page with open panels
Remarks: Simple Four-Column Comic page is omitted as the result is trivial once Case #1 is done.

Case: #2
Purpose: to perform Flood-fill in a comic page with overlapped panels

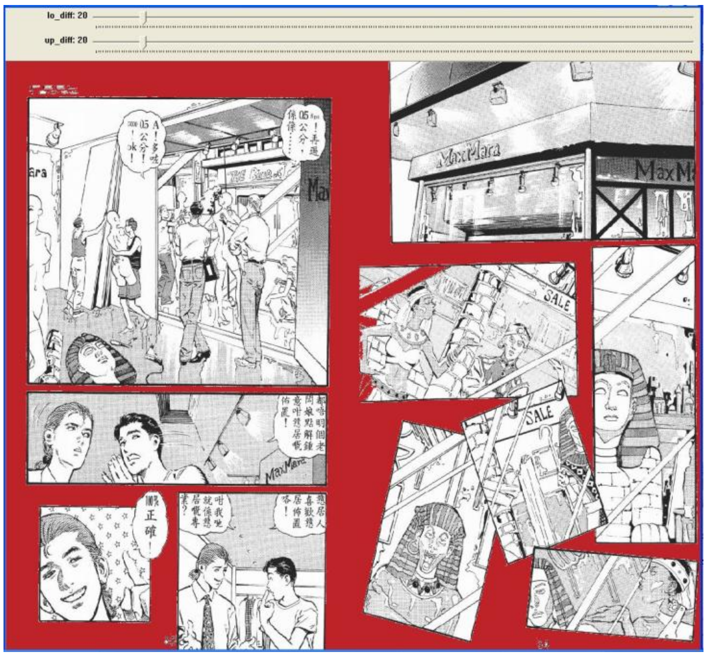
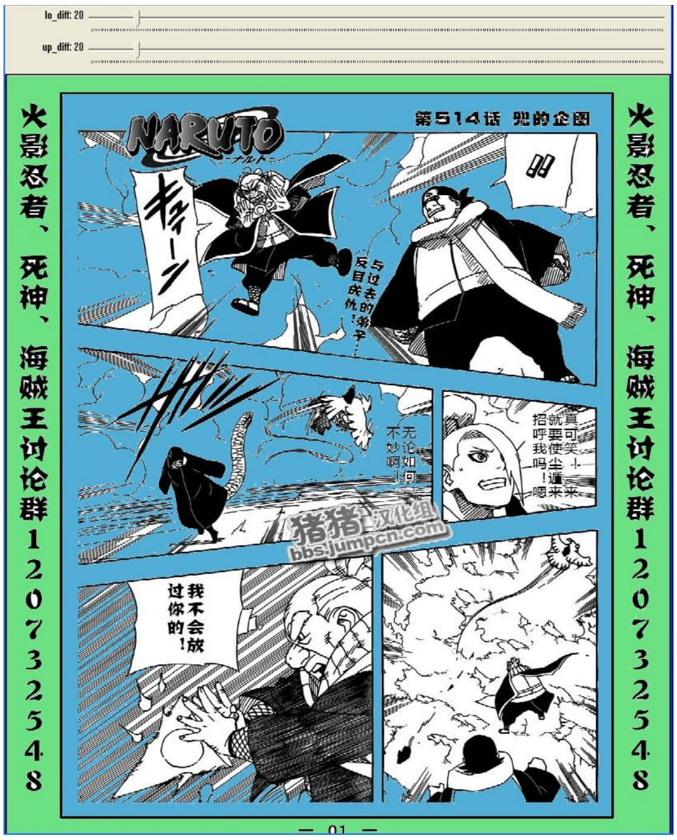


Figure 5.42 - Result of Flood-fill, Test case 2

Case 3:



The result of test case 3 shown on the left may be less helpful when comparing to case 1 and case 2.

Case: #3
Purpose: to perform Flood-fill in a comic page with strong strokes

Figure 5.43 - Result of Flood-fill, Test case 3

Chapter 6 Difficulties Encountered and Current Limitations

We are going to describe some difficulties we encountered while doing this project. There are 4 problems that we have encountered. Let us talk about them one by one.

6.1 Runtime Environment Problem

As we did not have an iPhone available at the beginning, runtime environment was a problem to our research. Hence, we used the iPhone simulator provided by XCode for debugging. However, when it comes to a concrete action, like zooming through multi-touching, it is impossible in simulator.

Therefore, we focus on image processing study on PC in the first term. All this algorithms are not ported into our iPhones due to limited time after our iPhones are available.

6.2 Automating the input

Our Watershed Segmentation needs manual input of some markers. It is a problem to automate this because, once we are able to automatically generate the markers, we have already got enough information for panel zooming. They are similar problems that we are facing.

In addition, the Watershed Segmentation does not always perform perfectly. To enhance the accuracy, we perform Flood-fill first. However, the Flood-fill also requires a manually-input seed point. Since the background pattern is not predictable in general, we cannot do this automatically without further assumptions.

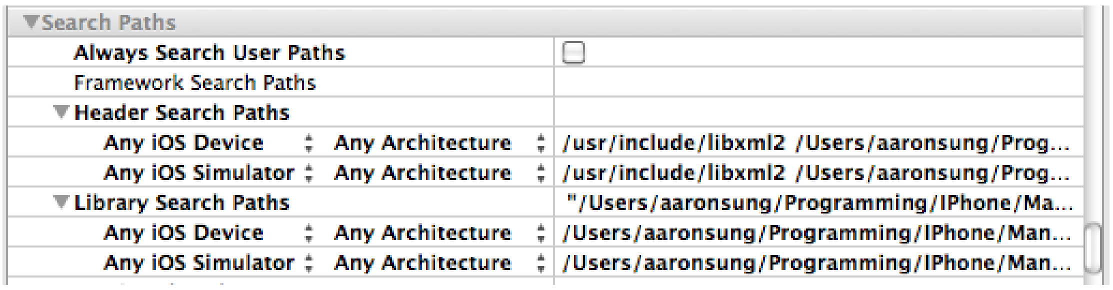
6.3 Insufficient Libraries

iOS is mature enough for development purpose, however image processing libraries are currently not enough. For example, cvBlobsLib, which is for blob tracking, is not available in iOS. In case we need this library in the future, we have to port it into iOS ourselves.

6.4 Insufficient Documentation

Most of the 3rd party frameworks and libraries are lack of proper documentation. Many of them are lack of guidelines and tutorial so a lot of time is spent to understand how to manipulate them.

Two examples are OpenCV and GDataXML. Both of them have only one web page to describe how to use it. A lot of trial and error has been done. For example, there are condition build settings which is different for simulator and iOS device but he documentation are not cleared enough so time has been spent to make it work.



▼Search Paths			
Always Search User Paths			☐
Framework Search Paths			
▼Header Search Paths			
	Any iOS Device	Any Architecture	/usr/include/libxml2 /Users/aaronsung/Prog...
	Any iOS Simulator	Any Architecture	/usr/include/libxml2 /Users/aaronsung/Prog...
▼Library Search Paths			"/Users/aaronsung/Programming/IPhone/Ma...
	Any iOS Device	Any Architecture	/Users/aaronsung/Programming/IPhone/Man...
	Any iOS Simulator	Any Architecture	/Users/aaronsung/Programming/IPhone/Man...

Figure 6.1 - Condition build settings for OpenCV on iOS

Another example is the GDataXML library. There is no documentation to explain what exactly the use of different functions and source code is needed to study before using it.

Chapter 7 Future works

In this chapter, we will talk about our future work.

7.1 Improve the algorithm to support automatic extraction

As mentioned before, our Watershed Segmentation needs manual input of some markers. We try to automate the generation of marker to provide an automatic extraction of panels.

Moreover, the threshold value is entered by user currently. Although letting the user input the threshold can improve the accuracy, we try to find ways to make it smarter and minimize the need of users' input.

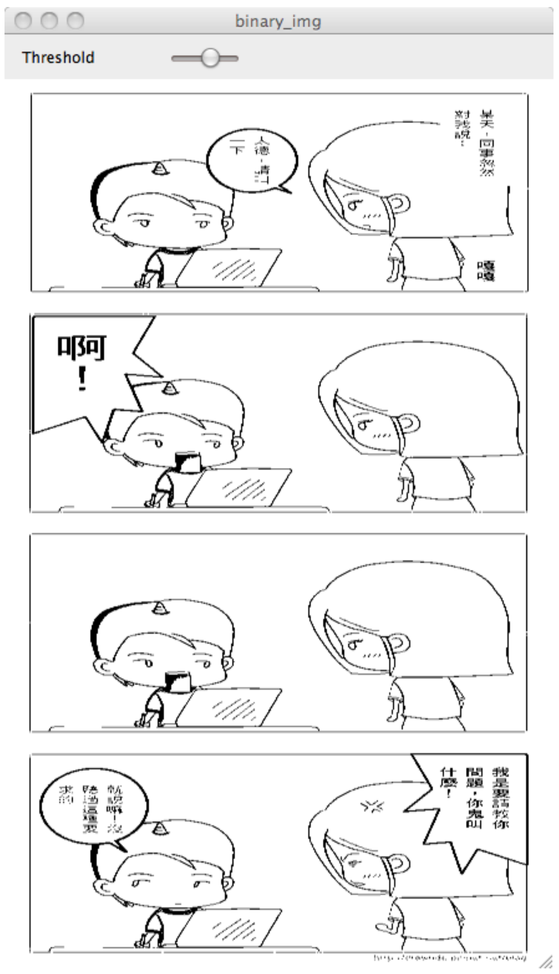


Figure 7.1 - Threshold may be needed to set by user

7.2 Investigate comics that contains out of scope panels

Out of scope is a common phenomenon in comics. Many comics contain out of scope panels. However, due to the nature of most of the segmentation algorithms, such as Flood-fill or watershed, they fail to process out of scope panels. More research and investigation will be done in the future to see whether the problem can be solved.

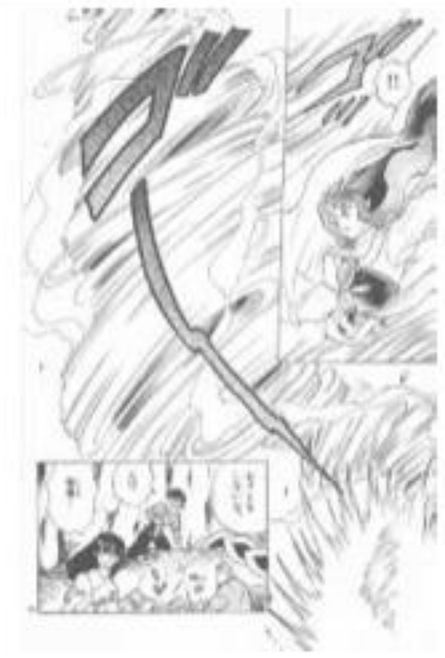


Figure 7.2 - An example of page that out-of-scope panels occurs

7.3 Take advantages from other algorithms

Our work is started from watershed algorithm. As different algorithm has different advantages, we will try to incorporate other algorithm and modified our algorithm better. This can both increase the accuracy and performance of the algorithm.

Chapter 8 Acknowledgement

We would like to take this chance to express our thanks to Prof. Michael Lyu, our supervisor of the final year project. The invaluable guidance and suggestion given from Professor benefit us a lot.

We would also express our thanks to Mr. Edward Yau for the incredible advices and support given by him.

Chapter 9 Reference

[1]http://big5.xinhuanet.com/gate/big5/news.xinhuanet.com/eworl

[2] C. Ponsard, V. Fries - *Enhancing the Accessibility for All of Digital Comic Books*

[3]F. Chang, C-J. Chen, and C-J. Lu. —*A Linear-Time Component-Labeling Algorithm Using Contour Tracing Technique",* Computer Vision and Image Understanding, 93(2):pp. 206-220, 2004.

[4] K. Arai & H. Tolle - *Automatic E-Comic Content Adaptation*

[5]T. Tanaka et el., *Layout Analysis of Tree-Structured Scene Frames in Comic Images*, IJCAI'07 Proceedings of the 20th international joint conference on Artifical intelligence

[6] H.C. Chung, H. Leung & T.Komura, "*Automatic Panel Extraction of Color Comic Images*," Advances Multimedia Inform. Processing – Pcm 2007, vol. 4810/2007, pp. 775–784, 2007.

[7] OpenCV Reference Manual, v2.1, March 18,2010

[8] The watershed transformation for multiresolution image segmentation, S. Wegner, T. Harms, J. H. Builtjes, H. Oswald and E. Fleck

[9]http://niw.at/articles/2009/03/14/using-opencv-on-iphone/en