

ECE 533 Digital Image Processing
Instructor: Prof. Hu, Yu Hen

Counting Cell

Using Digital Image Processing Technique

December 19, 2005

Prepared By:
Pei Qi
Yang Wang

Electrical&Computer Engineering
University of Wisconsin-Madison

ABSTRACT

This project is focus on developing a computer based software program to be used by the biomedical department to aid their research need. The biomedical department is conducting a research requires an automatic counting program to count the number of iron contained blood cells from a photonic image. The main objective of this project is to explore and develop an image processing program to accomplish automate counting process.

In this project, one of automate counting programs being proposed and developed. The basic idea of this program is to use correlation function to find the largest summation between a sampled image (the wanted cell) and the giving image. Since the largest summation represents the location where the two image shares similarity, it can help us identify the location of wanted cells on a given image. By taking a certain threshold, those wanted cells can be extracted and counted.

This project explains how each process was considered and developed. Because this program is intended to be used by researchers of biomedical department, who has limited knowledge in image processing tool, as a developer, we also have to consider a user friendly interface to make our program more useful to the user.

Furthermore, our scope might go beyond this project. It is also our interests to use this chance to continue our exploration and probably solve a more general task of object recognition in the future.

INTRODUCTION

The biomedical department is conducting a research on the study of iron absorption in the blood cells of an orphan. Studying iron absorption requires work in counting number of iron contained blood cells from a photonic image. So far, hand counting is the only method being used. However, due to high vulnerability in human error and large time consumption, better and more effective image processing software is needed. In this project, an automatic counting program is developed, based on image processing techniques.

APPROACH & IMPLEMENTATION

Since this software will be used by a biomedical researcher, we would like the user to have total control over his or her experiment. First, we would like the user to load an image in the software. Then, it will scan the image and ask the user to select a sample of a specific object of certain color to count with. Based on the range of hue level of user's selection, the program will search and find all the similar shape and color contained in the image. User can choose to delete or add any object with his or her choice. After the selection is finalized, the program will calculate the total number of wanted object in the image.

On the technical side, after an image is loaded and sub-image is selected by the user, first, the program will run an autocorrelation between the image and sub-image. A program fixed threshold is chosen to pick out the largest summation between the two images. By identifying the location of these largest summations, the image can be further changed into binary image to simplify the counting process. Based on the selection, the image is changed into a binary image. Then, the binary image is imposed on the original image so that the user can add or delete any cell of his wish. After the change is finalized, a technique which is similar to the erosion technique is used to change all the wanted shape into a single pixel. The sum of the single pixels will be the number of wanted cell.

Based on the presented approach, the implementation basically involves five critical sections:

1) Choosing Sample

To a great extent, the result of Counting Cell relies on the people who was doing a count task. Experiential determination always significantly affects the result of counting. Our approach takes full advantage of people's experience, therefore we firstly let researcher self-select the sample element for counting. There is a branch of factors that could influence the selection of sample, however among those factors, the most obvious, and straightforward criteria is color choosing. Because the blueness of the blood cell illustrate the concentration of iron in the cell, it is hard to decide how blue the color has to be in order to be counted. This criterion should not be decided by the developer, but the lab researcher. This is the reason why we let the user to select a range of blue color for us to use. This feature enables us to apply digital image processing technique to acquire the sample more easily. In our case, HSI color mode firstly was used to extract specific color elements from original image.

When humans view a color object, we describe it by its *hue*, *saturation*, and *brightness*. Hue is a color attribute that describe a pure color, whereas saturation gives a measure of the degree to which a pure color is diluted by white light. Brightness (intensity) component is a subjective descriptor that is practically impossible to measure. Among those three components, *hue* value significantly determines the color scope for any object. Therefore, in our case we extract the wanted color elements through specifying the range of hue value. Considering the range of hue value should vary according to different image, we also allow researchers to adjust the range of hue value to adapt to particular image.

2) Correlation

Based on the range of hue value and given shape, the correlation function is used to find similarities between original image and the selected sample. The highest peaks in Figure 1 represent the location of the founded similarities. More over, this peak also reveals where the blue cells are located. The height of the peaks in the correlation figure really shows how similar the two images are at a certain area. Higher the peak represents stronger similarity.

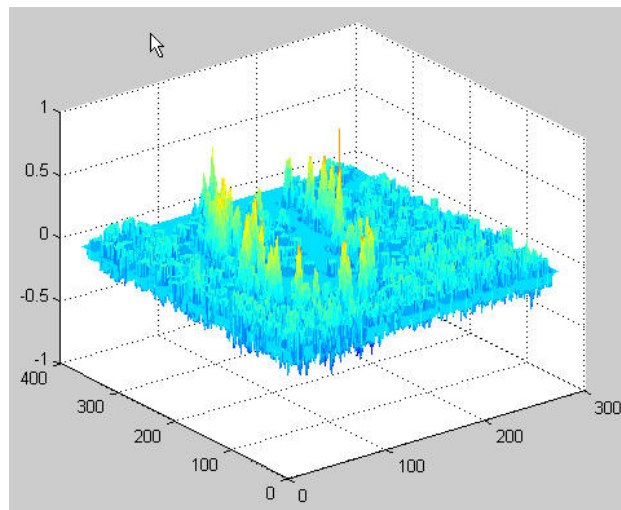


Figure 1. Result after applying correlation function on the image.

3) Thresholding

A certain threshold needs to be picked to extract the location of the blue cell. At this stage of the program, the threshold is chosen by the program; however, this parameter can be given to the user to decide to obtain a better result.

4) Convert to binary image

After a threshold is decided, the program takes any hue values which are above the threshold and assign a RGB to be 255 and any hue value which is below to be 0. Figure 2 below shows the result.

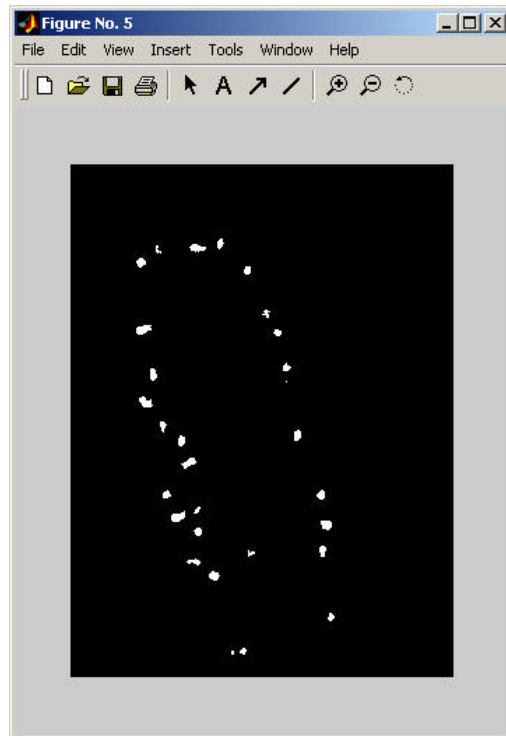


Figure 2. Binary image after thesholding

5) Counting method

Initially, erosion method was considered to be used to count the white shapes in Figure 2. The goal is to use a technique to change all the wanted shape into a single pixel so that they can be added later to count the total number.

Erosion is given by the following the equation:

$$A \ominus B = \{z \mid (B)_z \subseteq A\}$$

However, because the shape of the cells not necessarily the same, erosion technique might be difficult of apply here. Therefore, the following technique was used in this project.

A 'for' loop search, if the current pixel has a RGB of 255 and one of its neighboring pixel is also 255, then change the current pixel to 0. Otherwise, leave the current pixel unchanged. This method shrinks every white shape into a single white dot. By adding the number of white dots, a total number of selected white cell are found. Figure 4 shows the image after the use of this technique.

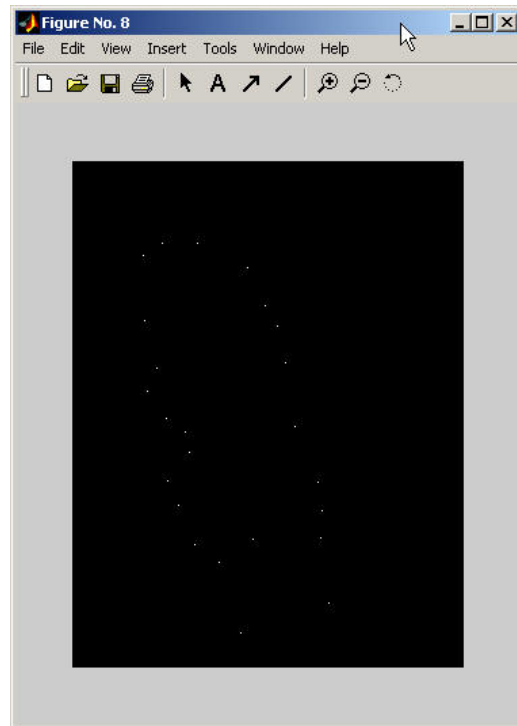


Figure 3 all selected cells are shrunk into dots

Software Instruction

Our software is a Matlab-based application. By using GUI developing tools, we create a simple graphic user interface to assist people to use this software. The whole operation regarding this software is described as the following steps:

Step 1: Run main program (Counting_Cell.m), you will get the following window.

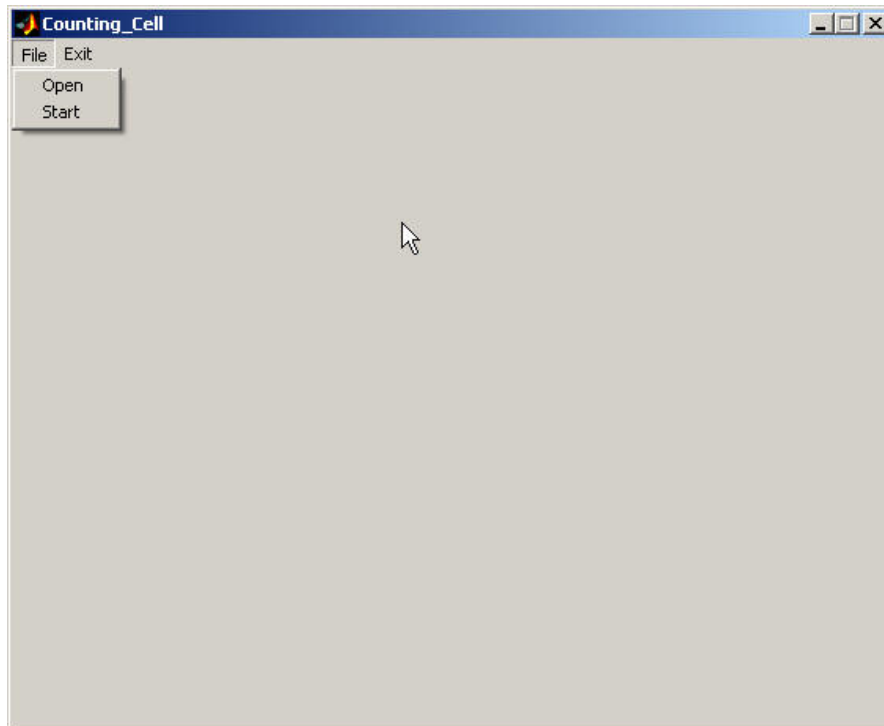


Figure 4

Step 2: Click “Open”, input the image file name you want to count in the textbox, and then click “OK” to confirm.

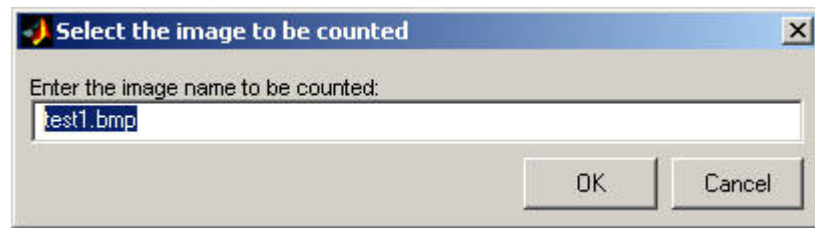


Figure 5

Step 3: There will be two windows in the screen. One (Figure 6) is the main window which includes the image to be counted. The other (Figure 7) is the instruction window that will stay on the upper left corner of screen and exhibits the whole procedure you need to follow during counting process.



Figure 6

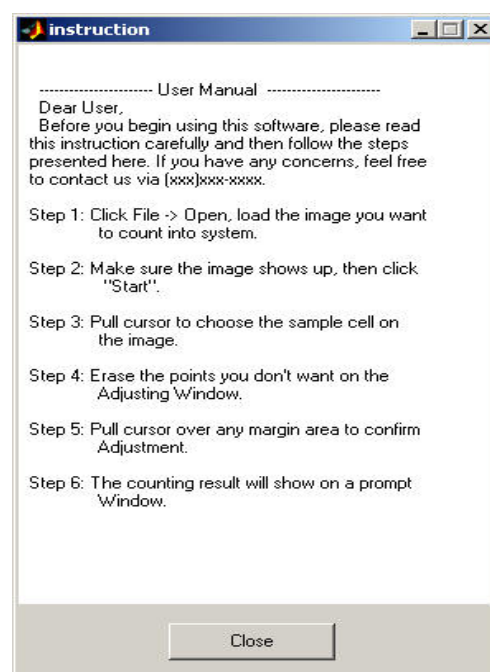


Figure 7

Step 4: Click “Start” to begin counting process. Firstly, pull your mouse cursor, select a sample of specific object of certain color to count with. In this case, we only care the blue dots which indicate those cells having absorbed iron element.

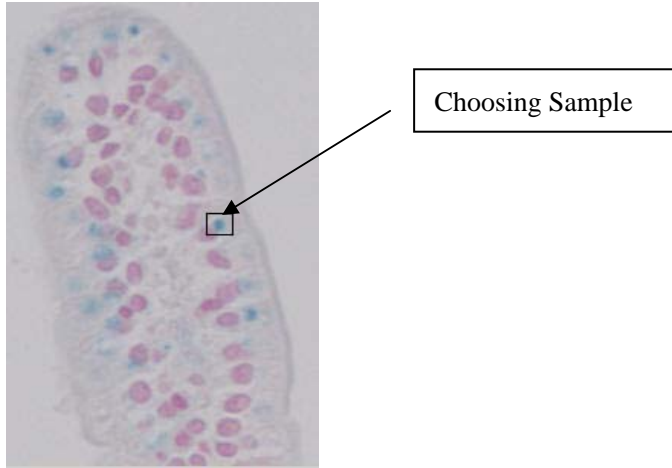


Figure 8

Step 5: After application obtains the sample chosen by researchers, it will begin to deal with image according to the methods we mentioned in this article, and return a correlation image window. Here the application gives researcher the last chance to adjust their selection. Researchers could erase some dots they believe they shouldn't be counted.

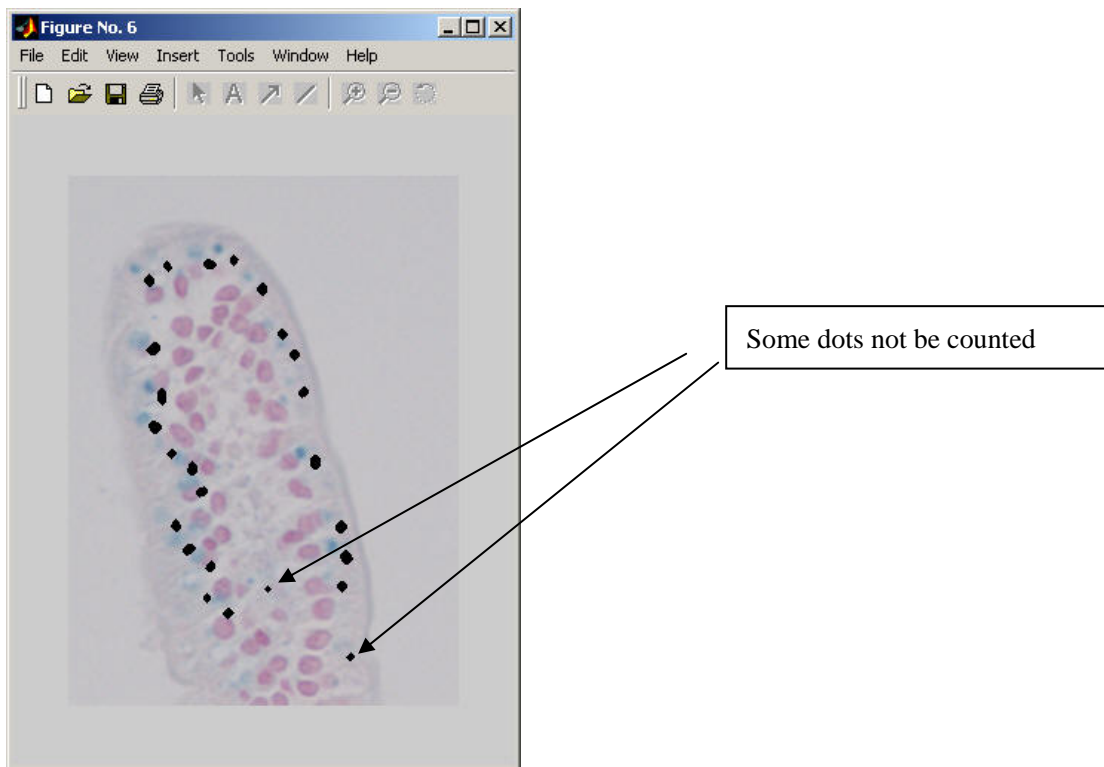


Figure 9

Step 6: After you finish your correction, the counting result will be shown in the window like the following.



Figure 10

RESULT and DISCUSSION

The counting result is much closer to the hand counting result. To verify the accuracy of our application, we invited some students from biology labs to help us count cells manually. We chose several different images published in public domain, and then put them into different category which is determine by their size, shape, and numbers of cell etc.

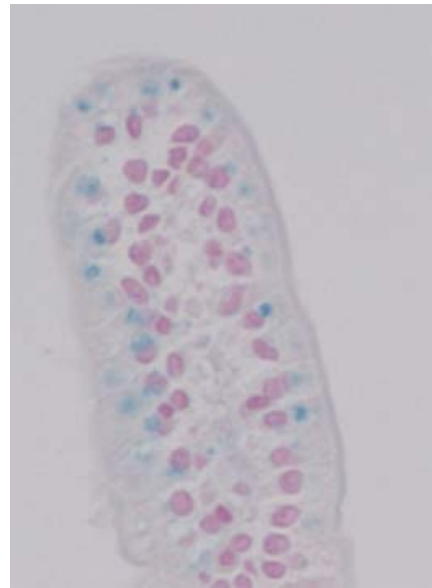
Two very important premises here,

- i) All experimenter don't know the counting results generated by our application.
- ii) Experimenter chooses the sample using our software on the same image he or she had counted manually.

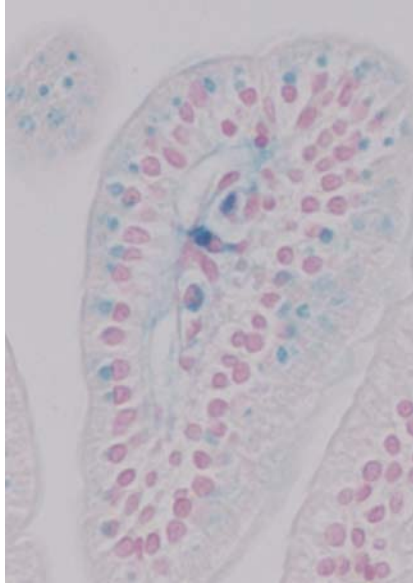
Category I (Small Size Image)



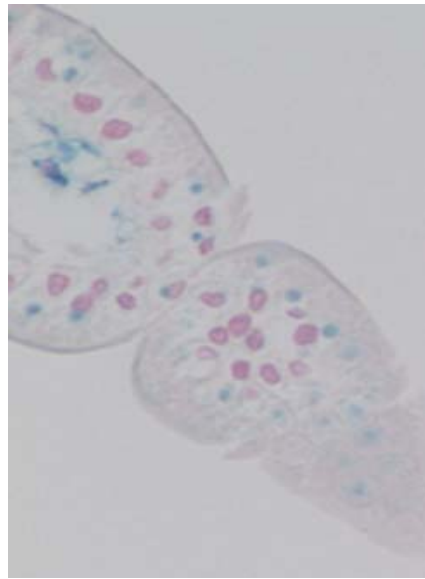
Auto Counting: 19
Hand Counting: 21



Auto Counting: 22
Hand Counting: 22

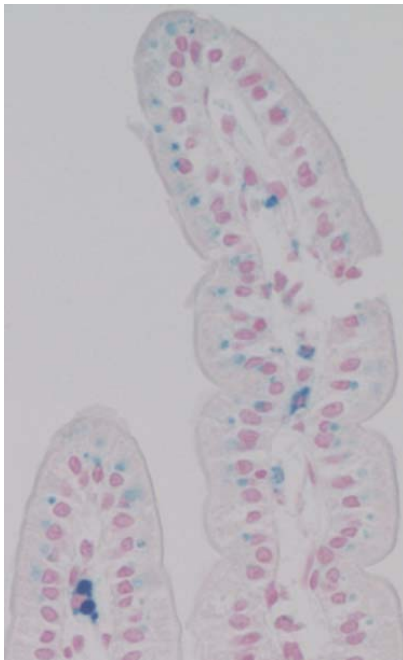


Auto Counting: 26
Hand Counting: 22

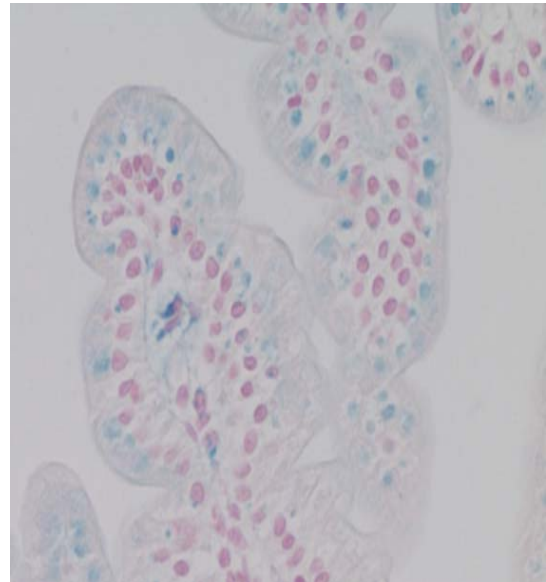


Auto Counting: 21
Hand Counting: 22

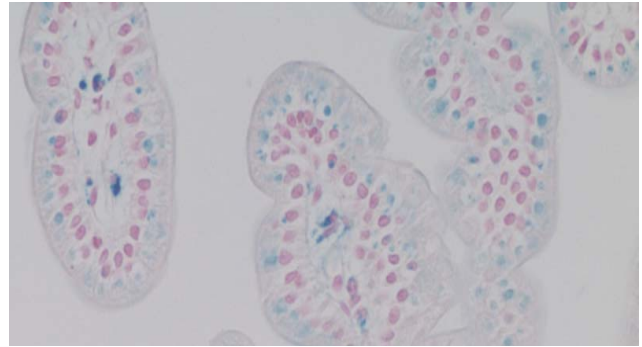
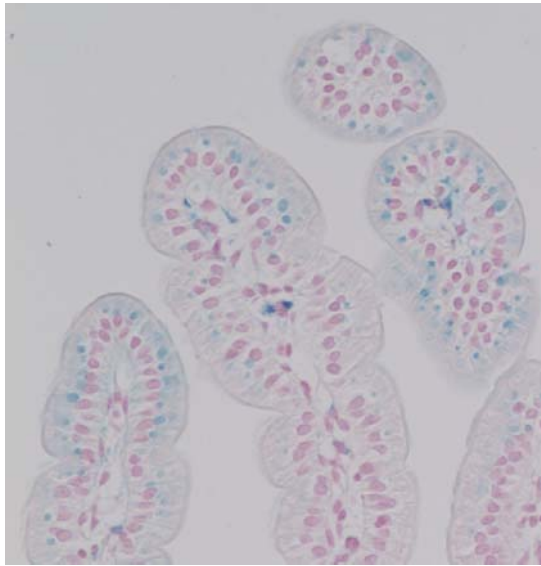
Category II (Large Size Image)



Auto Counting: 50
Hand Counting: 45



Auto Counting: 48
Hand Counting: 55



Auto Counting: 42
Hand Counting: 48

Auto Counting: 41
Hand Counting: 49

The comparison table

Category	Auto Counting	Hand Counting	Difference	Accuracy (%)
I	19	21	2	90.48
I	22	22	0	100
I	26	22	-4	81.82
I	21	22	1	95.45
II	50	45	-5	88.89
II	48	55	7	87.27
II	42	48	6	87.5
II	41	49	8	83.67

(I – Small Image Size, II – Relatively Large Image Size)

From the above comparison, our program is relatively accurate for most of test objects. However, our program is not very efficient on the execution time, especially when handling large size images. A potential solution is to refactory source code using other languages, like C++. Because compared to Matlab, C program has dominant advantage in running loop instructions.

In addition, we don't have a relatively objective measurement to verify our result yet. Whether automate counting or hand counting, subject factor accounts for significant part during the counting process.

REFERENCE

[1] Rafael C. Gonzalez, Richard E. Woods [2004] “*Digital Image Processing*”, Morphological image processing, pp525-527.

[2] Matlab 6.5, images, extended-examples, © 1994-2005 The MathWorks, Inc.

[3] Dubuque SH, Dvorak B, Woodward SS, McCuskey RS, and Kling PJ. 2002. Iron-deficient erythropoiesis in neonatal rats. *Biol. Neonate.* 81:51-57

TASK ASSIGNMENT

Tasks	Pei Qi (%)	Yang Wang (%)
Preparing for project topic	50	50
Programming	45	55
PowerPoint Slides	50	50
Report	55	45
Overall	50	50
Signature		