ABSTRACT

Title of master thesis:	Cellular Pattern Quantification and Automatic Bench-marking Data-set Generation on confocal microscopy images
	Chi Cui, Master of Engineering, 2010
Thesis directed by:	Professor Joseph JaJa Department of Electrical and Computer Engineering

The distribution, directionality and motility of the actin fibers control cell shape, affect cell function and are different in cancer versus normal cells. Quantification of actin structural changes is important for further understanding differences between cell types and for elucidation the effects and dynamics of drug interactions. We propose an image analysis framework to quantify the F-actin organization patterns in response to different pharmaceutical treatments.The main problems addressed include which features to quantify and what quantification measurements to compute when dealing with unlabeled confocal microscopy images. The resultant numerical features are very effective to profile the functional mechanism and facilitate the comparison of different drugs. The analysis software is originally implemented in Matlab and more recently the most time consuming part in the feature extraction stage is implemented onto the NVIDIA GPU using CUDA where we obtain 15 to 20 speedups for different sizes of image. We also propose a computational framework for generating synthetic images for validation purposes. The validation for the feature extraction is done by visual inspection and the validation for quantification is done by comparing them with well-known biological facts. Future studies will further validate the algorithms, and elucidate the molecular pathways and kinetics underlying the F-actin changes. This is the first study quantifying different structural formations of the same protein in intact cells. Since many anti-cancer drugs target the cytoskeleton, we believe that the quantitative image analysis method reported here will have broad applications to understanding the mechanisms of candidate pharmaceutical.

Cellular Pattern Quantification and Automatic Benchmarking Data-set Generation on confocal microscopy images

by

Chi Cui

Thesis submitted to the Faculty of the Graduate School of the
University of Maryland, College Park in partial fulfillment
of the requirements for the degree of
Master of Engineering
2010

Advisory Committee:
Professor Joseph F. JaJa, Chair/Advisor
Professor Shihab A. Shamma
Professor Min Wu

Acknowledgments

I owe my gratitude to all the people who have made this thesis possible and because of whom my graduate experience has been one that I will cherish forever.

First and foremost I'd like to thank my advisor, Professor JaJa for giving me an invaluable opportunity to work on challenging and extremely interesting projects over the past three years. He has always made himself available for help and advice and there has never been an occasion when I've knocked on his door and he hasn't given me time. It has been a pleasure to work with and learn from such an extraordinary individual.

I would also like to thank my collaborator, Dr. Stephen Lockett. Without his extraordinary theoretical ideas and computational expertise, this thesis would have been a distant dream. Thanks are due to Dr.Thomas Turbyville, Dr.John Beutler, Dr.Prabhakar Gudla and Mr.Kaustav Nandy for providing the original data set, insightful suggestions on my work and generous help.

I owe my deepest thanks to my family - my mother and father who have always stood by me and guided me through my career, and have pulled me through against impossible odds at times. Words cannot express the gratitude I owe them.

I would like to acknowledge financial support from University of Maryland Institute for Advanced Computer Studies(UMIACS), for all the projects discussed herein.It is impossible to remember all, and I apologize to those I've inadvertently left out.

Lastly, thank you all and thank God!

Table of Contents

List of Figures v

List of Abbreviations vii

1 Introduction 1
- 1.1 Thesis Goals 1
- 1.2 Major Challenges 1
- 1.3 Previous Work 2
- 1.4 Outline of Thesis 3

2 Background 4
- 2.1 Brief Biological Introduction 4
  - 2.1.1 Actin 4
  - 2.1.2 Schweinfurthin A 4
  - 2.1.3 Green Fluorescent Protein(GFP) 5
  - 2.1.4 DAPI 5
  - 2.1.5 Dimethyl sulfoxide (DMSO) 5
- 2.2 Confocal Microscopy 6
- 2.3 Image Formation 6
- 2.4 Data Preparation 7

3 Feature Extraction and Quantification 8
- 3.1 General Analysis Framework 8
- 3.2 Top Down Feature Extraction 9
  - 3.2.1 Tier 1: Tri-band segmentation 9
  - 3.2.2 Second Tier: Sub-Cellular Feature Extraction 15
    - 3.2.2.1 Sub-cellular feature definition 15
  - 3.2.3 Cortical Actin Versus Punctuate Dots 16
    - 3.2.3.1 Stress Fiber Extraction 17
- 3.3 Multi-scale Cytoplasm Texture Analysis 18
  - 3.3.1 Definition of cellular**texture** 18
  - 3.3.2 Why multiscale? 18
  - 3.3.3 Detailed Methodology 19
- 3.4 Feature Quantification 20
  - 3.4.1 Overview 20
  - 3.4.2 Scalar Quantification 21
  - 3.4.3 Cytoplasm Nuclei relative distribution Quantification 22

4 Parallel Tri-band Segmentation using CUDA 25
- 4.1 Why CUDA? 25
- 4.2 CUDA Introduction 25
  - 4.2.1 What is CUDA 25
  - 4.2.2 Programming Model 25
  - 4.2.3 Memory Model 27
  - 4.2.4 Global Memory Access Coalesce 28
- 4.3 CUDA Based Tri-band Segmentation 28

4.3.1 Framework 28
4.3.2 Point wise Feature Generation 29
4.3.3 Fuzzy C Means(FCM) 33
4.4 Experiments and Results 35

5 Automatic Synthetic Data set Generation 37
5.1 Related Work 37
5.2 Methodology Overview 38
5.3 Algorithms 39
5.3.1 Cell nuclei generation 39
5.3.2 Cell Body Generation 40
5.3.3 Cellular Feature generation 42
5.3.4 Assembling Components 44
5.3.5 Post Processing 45

6 Validation and Results 46
6.1 Feature Extraction Validation 46
6.2 Quantification Validation 48
6.3 Automatic Benchmark Generation 52

7 Conclusion and Future Work 56

Bibliography 58

List of Figures

2.1 This figures illustrates the principle of confocal microscopy. 6

3.1 The works shows the framework of our image analysis algorithms 9
3.2 This figure shows a comparison of the original image and the color labeled segmented image after applying the tri-band segmentation 11
3.3 The number of segments for both algorithms was set to 3. The original image is the first one from the left. In the EM segmented image which comes the second, color yellow denotes the WBA regions, color cyan denotes the cytoplasm regions and color blue denotes the background. In the image segmented by our method which shows on the right most, color red denotes the WBA region, color green denotes the cytoplasm region and color blue denotes the background 13
3.4 The sub-figure on the right gives an example of the two sub-structures belonging to white bright actin and the sub-figure on the left gives an example of the stress fibers which belongs to the cytoplasm component generated from the first tier decomposition. 15
3.5 This figure shows the visualization of cytoplasm nuclei relative distribution quantification 23
3.6 The left sub-figure shows the cells treated under cytoB in 10um/ml and the right sub-figure shows the cells treated under DMSO. 23

4.1 This figures illustrate the CUDA Programming Model. Courtesy Image from NVIDIA CUDA Programming Guide 2.0. 26
4.2 This figure shows the hierarchy of the CUDA memory model. Courtesy image from CUDA Programming Guide 2.0. 27
4.3 This figure shows the computation step and memory layout of the point wise feature extraction step. 32
4.4 This figures shows the memory layout for the membership matrices 33

5.1 The figure shows the computational framework of our F-actin highlighted confocal microscopy image simulation 39
5.2 This figure shows an example of the generated nuclei mask using the method described above. 40
5.3 The images from left to right are: L) The distance map generated from the cell nuclei mask, 0 is mapped to black and the maximum distance in the image is mapped to white. M) Marker controlled watershed segmentation result color labeled according to patch indices after visibility filtering. R) The final cell body mask after the erosion 41
5.4 The first sub-figure from the left shows an example of the generated cortical actin mask. The second sub-figure shows an example of generated stress fiber mask. The last sub-figure from the left shows the final result of our automatic benchmark generation framework 43

6.1 This figure shows the results for SA at 10 nm/ml. 49
6.2 This figure shows the results for SA at 100 nm/ml. 50
6.3 This figure shows the results for SA at 1000 nm/ml. 50

6.4 The subfigure on the left shows the quantification result of the ratio of the stress fiber area to the cell area of SA; the one on the right shows an example of the box plot for the quantified texture energy of ceph. 51
6.5 This figures shows the images corresponding to the quantification result in the left sub-figure of Fig.6.4. The two sub-figures on the first line correspond to the DMSO, sa(10nM/ML) and the other two on the second line correspond to sa(100nm/ml) and sa(1000nm/ml) 54
6.6 This figure shows the pair-wise distance map of all the treatment conditions in our experiment. Darker color means smaller distance, in other word, greater similarity and black means the same. 55
6.7 A: Image of F-actin labeled cells. B: tri-band segmentation using the method introduced in reference [1]. C: Simulated image of F-actin in cells. D: Tri-band segmentation of the simulated image. 55

List of Abbreviations

SA	Schweinfurthin A
CLSM	Confocal laser scanning microscopy
GFP	green fluorescent protein
DMSO	Dimethyl sulfoxide
cytoB	cytochalasin B
FCM	Fuzzy C Means
CUDA	Compute Unified Device Architecture

Chapter 1

Introduction

1.1 Thesis Goals

The advent of confocal microscopy brings to the biologist new possibilities to obtain images of the detailed structures within a cell. Also the development of fluorescent labeling technique makes targeting one protein at a time much easier. Separated channel recording of different fluorescent dyes enables us to capture the image of different proteins independently in parallel. In this thesis we extract intensity, morphology and textual patterns from confocal microscopy images of giloma cells.We build generalized pattern descriptor and compute quantification measurements from these descriptors. Different quantification results are combined further to profile the effects drugs with different type or concentration have on the cells. The profiling is validated through prior observed facts and is used to predict how close the behavior of a new treatment is to the well known ones. This technique could be widely applicable in drug discovery. In this thesis we focus on one protein inside the cell, namely the F-actin. On discovering the effect of a particular drug, namely **Schweinfurthin A(SA)** on the F-actin in the hope to find a powerful inhibitor for brain tumors.

1.2 Major Challenges

There are two major technical challenges to be addressed in this work:

- Which features to extract and how to extract them

- How do we quantify these features so as to understand the effects of drug treatments on the F-actin

Here, in item 1, the requirement for the generated features is that they should not capture a suitable description of the critical components of the protein structure while also being capable to be used to derive the quantification results. The extraction method needs to be robust to varying image conditions, e.g. contrast, signal to noise ratio and etc. The measurements generated from the features should have low uncertainty with regard to different imaging conditions, e.g. size of the samples, resolution, contrast and etc. The quantification results computed from these measurements should be accurate enough to be validated at least by visual observation.

1.3 Previous Work

Previous work has shifted more to automatic quantification, such as in [2],[13]. Most of these quantification methods computed some statistics of F-actin covered regions, lengths, orientations and subcellular localizations. However the F-actin images used(e.g.[9]) are often quite simple containing only one or two geometric primitives with minor deformation.Recent work in this area[20][12][5][24][11], which is similar to ours includes the recognition of locations of different kinds of proteins in a single cell or multiple cell images and the "cell profiler"[4][16][15][14], a machine learning powered interface which extract rules from user specified data-set and distinguish positive samples from the negative ones. While our work shares some similarities with the previous work we highlight the differences here as shown in Table. 1.1.

	cell profiler	Protein localization	Our work
Feature Extraction	cell based	cell based	image based
Result Format	Binary Indicator	Discrete Class labels	Continuous Numerical Results
Classification Goal	Tell positive from negative	Different types of proteins	Different organizational and morphological pattern of a single protein (F-actin)
Motivation	Highlight the difference	Highlight the difference	Show the conditional variation trend

Table 1.1: This table shows a comparison between our method and two previous work

1.4 Outline of Thesis

This thesis is organized into 7 chapters. The first chapter introduces the goal, the major challenges and some relevant previous work. Chapter 2 provides the biology and imaging background as a foundation to understand the whole work. Chapter 3 describes the design and implementation of the feature extraction and quantification methodology. Chapter 4 details the CUDA implementation of a time consuming part of the feature extraction algorithm, which is the Tri-band Segmentation. Chapter 5 deals with the design framework of automatic benchmark dataset generation. Chapter 6 gives validation results from experiments to show the effectiveness of our methods and we draw our conclusion in the last chapter.

Chapter 2

Background

2.1 Brief Biological Introduction

2.1.1 Actin

Actin is a globular, roughly 42-kDa protein found in all eukaryotic cells. It participates in many important cellular processes including muscle contraction, cell motility, cell division and cytokinesis, vesicle and organelle movement, cell signaling, and the establishment and maintenance of cell junctions and cell shape. Many of these processes are mediated by extensive and intimate interactions of actin with cellular membranes.Individual subunits of actin are known as **globular actin (G-actin)**. G-actin subunits assemble into long filamentous polymers called **F-actin**.

2.1.2 Schweinfurthin A

Schweinfurthin A(SA) is a natural product isolated and identified at the National Cancer Institute, which can cause dramatic changes in the organization of the actin cytoskeleton [1].Using live-cell imaging, confocal microscopy, and biochemical assays, NIH researchers have been able to determine that this occurs not because of direct action on actin, but rather due to changes in actin regulation. Interestingly, this molecule has over 1000-fold selectivity for central nervous system (CNS) tumor cells. There are other molecules that share this selectivity, and have similar effects on actin.

2.1.3 Green Fluorescent Protein(GFP)

The green fluorescent protein (GFP) is a protein composed of 238 amino acids (26.9kDa), which exhibits bright green fluorescence when exposed to blue light. While most small fluorescent molecules such as FITC (fluorescein iso-thiocyanate) are strongly phototoxic when used in live cells, fluorescent proteins such as GFP are usually much less harmful when illuminated in living cells. In cells where the gene is expressed, and the tagged proteins are produced, GFP is produced at the same time and thus gives fluoresce when observed under fluorescence microscopy.

2.1.4 DAPI

DAPI or 4',6-diamidino-2-phenylindole is a fluorescent stain that binds strongly to DNA. It is used extensively in fluorescence microscopy. For fluorescence microscopy, DAPI is excited with ultraviolet light. DAPI's blue emission is convenient for microscopists who wish to use multiple fluorescent stains in a single sample. There is fluorescence overlap between DAPI and green-fluorescent molecules like fluorescein and green fluorescent protein (GFP), or red-fluorescent stains like Texas Red, but using spectral unmixing or taking images sequentially can get around this.

2.1.5 Dimethyl sulfoxide (DMSO)

DMSO is used in PCR to inhibit secondary structures in the DNA template or the DNA primers. It is added to the PCR mix before reacting, where it interferes with the self-complementarity of the DNA, minimizing interfering reactions. It may also be used as a cryoprotectant, added to cell media to prevent cell death during the freezing process.

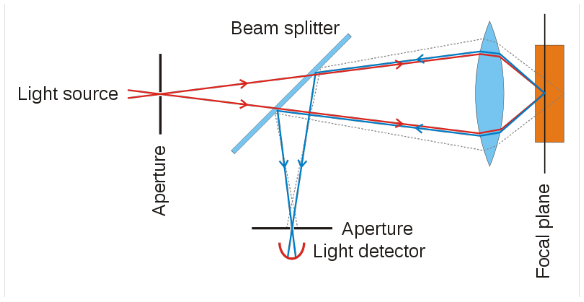

Figure 2.1: This figures illustrates the principle of confocal microscopy.

2.2 Confocal Microscopy

Confocal laser scanning microscopy (CLSM or LSCM) is a technique for obtaining high-resolution optical images with depth selectivity. The key feature of confocal microscopy is its ability to acquire in-focus images from selected depths, a process known as optical sectioning. A conventional microscope ”sees” as far into the specimen as the light can penetrate, while a confocal microscope only images one depth level at a time. In effect, the CLSM achieves a controlled and highly limited depth of focus.

2.3 Image Formation

In CLSM a specimen is illuminated by a point laser source, and each volume element is associated with a discrete scattering or fluorescence intensity. The detector aperture obstructs the light that is not coming from the focal point, as shown by the dotted gray line in the Fig.2.1. The out-of-focus light is suppressed: most of the returning light is blocked by the pinhole, which results in sharper images than those from conventional fluorescence microscopy techniques and permits one to obtain images of planes at various depths within the sample (sets of such images are also known as z stacks).

The detected light originating from an illuminated volume element within the specimen represents one pixel in the resulting image. As the laser scans over the plane of interest, a whole image is obtained pixel-by-pixel and line-by-line, whereas the brightness of a resulting image pixel corresponds to the relative intensity of detected light. The scanning method has a low reaction latency and the scan speed can be varied. Slower scans provide a better signal-to-noise ratio, resulting in better contrast and higher resolution.

2.4 Data Preparation

The images for the primary experiment were from a mouse glioma cell line. In the experiment, cells were treated with two reagents known to inhibit actin: cytochalasin B (cytoB) and Y-27632 (Y), three candidate pharmaceuticals: OSW1, Schweinfurthin A (SA) and a synthetic cephalostatin 1 derivative (ceph), and a vehicle control, DMSO. The inhibitors and the candidate pharmaceuticals have been observed visually to cause discrete changes in F-actin organization. Fixed cell cultures were stained with DAPI to label nuclei and facilitate single cell segmentation, and Green Fluorescent Protein(GFP) to label F-actin. Samples are observed under laser scanning confocal microscope. The cell nuclei and F-actin images are captured by different channels. In the following section we called the images containing cell nuclei as nuclei channel images and the images contain F-actin as actin channel images. We randomly picked four regions for taking images under each condition. Since the z dimension of these images is quite small compared to high resolution x-y, the maximum intensity projections are sufficient for the analysis. Therefore the following analysis is based on 2D images.

Chapter 3

Feature Extraction and Quantification

3.1 General Analysis Framework

In this paper we deal with the problem of what features to extract for quantifying the morphology and spatial distribution of a single species of protein, in this case F-actin. The main motivation for this work is that in biological research, both quantitative analysis and visualization of images are necessary for optimal understanding of the samples. Moreover training the classification model on visually perceived structures limits the potential of computational analysis to quantifying only the visual cues and masks important sub-visual information that potentially exists in the images. In outline, a new methodology is proposed for analyzing F-actin images consisting of 2 parts:

1. Segment each image into different feature regions in a top down fashion. The image was first segmented into 3 major parts by performing point-wise fuzzy c-means (FCM) clustering on extracted feature vectors. FCM has been successfully used for unsupervised segmentation of biomedical images. Previous work by Chuang et al [6] shows its robustness to noise. Then each segmented part was subdivided either by sub-cellular location or by morphological features. We computed the area of different classes of segments per image and quantified their pairwise ratio to obtain scalar quantification results;

2. Quantify multi-scale texture features of the cytoplasm regions with identified sub-cellular structures excluded;

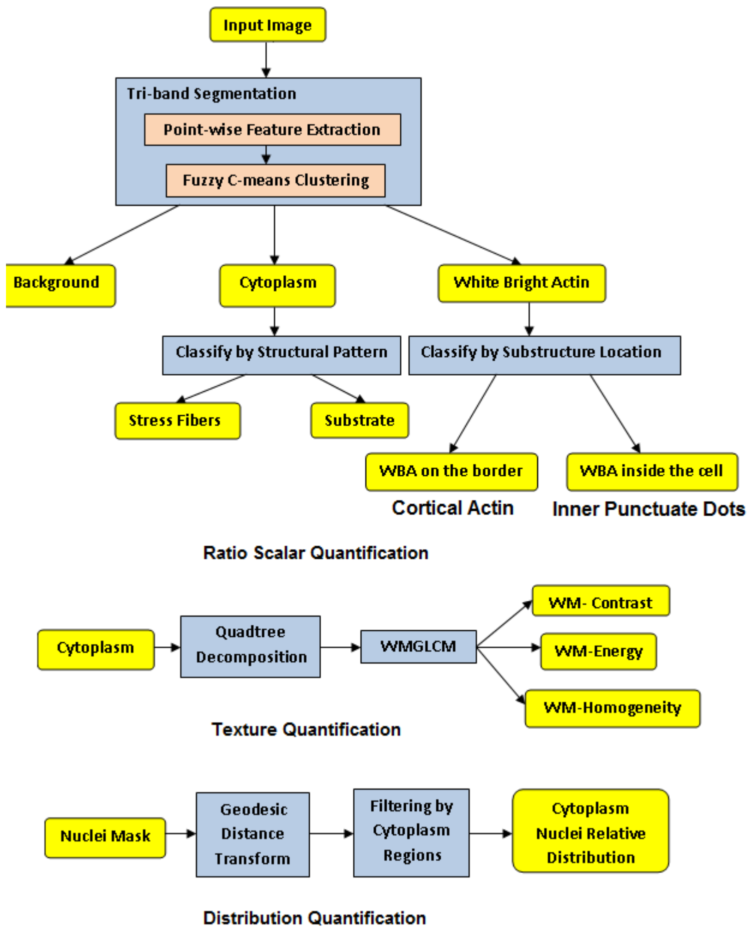

Figure 3.1: The works shows the framework of our image analysis algorithms

Fig.3.1 shows the flow chart of our analysis framework.

3.2 Top Down Feature Extraction

3.2.1 Tier 1: Tri-band segmentation

Our method is robust to the illumination variance over the image by working on a range of intensities and thus there is no need to do the illumination correction before segmentation. Based on the observation of hundreds of confocal microscopy images of the glioma cells we found that all the images has their intensities falling into three easily distinguishable intensity bands, which are background, the darkest region per image,

Intensity	Brightest	intermediate	Darkest
Full Name	White Bright Actin	Cytoplasm with Textual sub-cellular structures	Background
Abbreviation	WBA	Cyto	BK

Table 3.1: This table shows the naming of different bands according to their relative intensity in the image.

highlighted densely organized actin structures, the brightest part per image and cytoplasm with subcellular structure that has the intensities in between. Having highlighted features makes our work differ from previous ones, for which the main targets are mostly differentiate the cells from the background and do the cellular analysis on per cell basis. However, in our work the analysis is done on per image basis since we are interested in a specific structure rather than individual cells. Noticed from our analysis framework flow chart the first step in our analysis pipeline is called "Tri-band Segmentation". We designate the 3 parts shown in the Table.3.1 An example of the segmentation result is shown in. Fig.4. We use Fuzzy C means for the unsupervised classification of all the pixels in the image. The work done by Selinummi et al.[22] has used FCM to segment the image into different components for quantification purpose. However they only had two classes, namely the bright fluorescent dots and the background. This is much easier than our case since the degradation of the results due to a bad random initialization will be less likely to appear. Moreover they used the intensity value of a pixel and the variance of the pixel in its 5x5 local neighborhood to construct the per pixel feature vector.In highly noisy image, this method will fail since it will classify the noise as the bright dot. It is also quite restricted

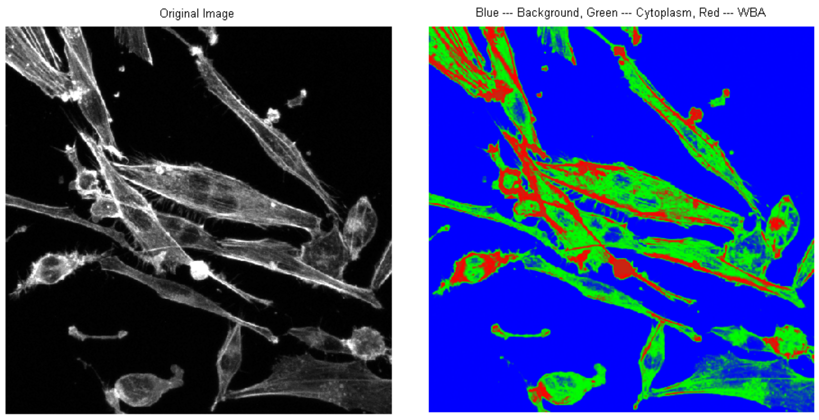

Figure 3.2: This figure shows a comparison of the original image and the color labeled segmented image after applying the tri-band segmentation

to the shape of the foreground features. Since in their case, the bright spot are mostly round, this limitation doesn't cause much problem. However in our case, since the highlighted F-actin regions (WBA) could be any shape, their approach will fail when dealing with long thin lines. We use a multi-scale feature vector per pixel which well captures the geometric structure in a large enough local surrounding region of the pixel. By increasing the number of scales in the feature vector the complexity of the feature space increases and thus decrease the possibility of a degradation in the results due to random initialization.

We build the pixel wise multi-scale feature vector by blurring the image with a 5x5 gaussian kernel 3 times and acquire the intensity as one feature component per pixel after each filtering. Finally, after also taking the original pixel value to consideration, we get a length 4 vector for each pixel point. A mathematical representation for this procedure is shown as follows. Let $G^{(n)}_{w\times w}(I)$ denote the application of n gaussian filtering in a w by w neighborhood. Then the feature vector we use for clustering operation is the following:

$$\overrightarrow{V} = [p(i,j), p^1(i,j), p^2(i,j), p^3(i,j)]^T \tag{3.1}$$

where $p^k(i,j)$ denotes the pixel value of $G^{(k)}_{w\times w}(I))$ at location (i,j). We want to compute the probability in which each pixel belongs to a class since maximum projection is used at the preprocessing stage. This could be another reason for us to choose Fuzzy c-means over K-means. The resultant sub-images of each band are computed by multiplying the original intensity with the pixel-wise probability map for each band. Since we don't use hard segmentation, our method works quite well for thin filament structures which are quite common in subcellular environment.

We model the intensity per pixel as a mixture of intensities from different bands. Therefore it seems intuitive to use the Expectation Maximization (EM) segmentation. Which could be mathematically described as follows: **Expectation-Maximization (EM)** algorithm is a method for finding maximum likelihood estimates of parameters in statistical models, where the model depends on unobserved latent variables. EM is an iterative method which alternates between performing an **expectation (E)** step, which computes the expectation of the log-likelihood evaluated using the current estimate for the latent variables, and a **maximization (M)** step, which computes parameters maximizing the expected log-likelihood found on the E step. These parameter-estimates are then used to determine the distribution of the latent variables in the next E step. Given a likelihood function $L(\theta, x, z)$, where θ is the parameter vector, x is the observed data and z represents the unobserved latent data or missing values, the maximum likelihood estimate (MLE) is determined by the marginal likelihood of the observed data $L(\theta, x)$, however this quantity is often intractable.

The EM algorithm seeks to find the MLE of the marginal likelihood by iteratively applying the following two steps:

Step I **Expectation step:** Calculate the expected value of the log likelihood function, with respect to the conditional distribution of z given x under the current estimate of the parameters $\theta^{(t)}$

$$Q(\theta|\theta^{(t)}) = E_{Z|x,\theta^{(t)}[logL(\theta;x,Z)]} \tag{3.2}$$

Step II **Maximization step:** Find the parameter which maximizes this quantity:

$$\theta^{(t+1)} = argmax_{\theta} Q(\theta|\theta^{(t)}) \tag{3.3}$$

The problem with EM is that it is computationally expensive especially for high dimensional data. FCM is more of a compromise between k-means and EM. It is more efficient than EM and often works better than the k-means algorithm. However due to the dependency on the initial prediction of the underlying distribution within each class the EM algorithm is also not a stable algorithm and when applying to some images in the database it fails when the WBA occupies just a small area in the whole image. Fig.3.3 shows an example of this situation. For the choice of feature length more folds of

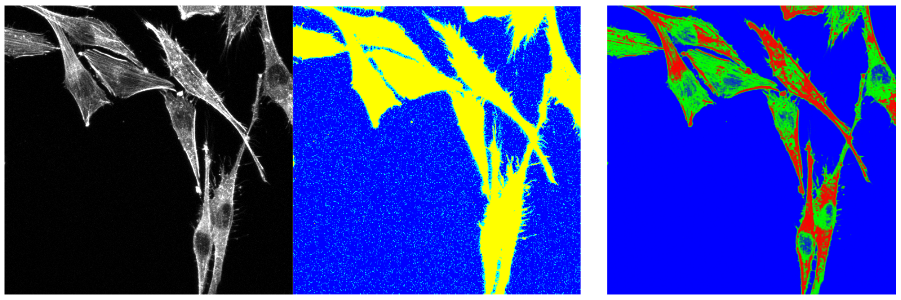

Figure 3.3: The number of segments for both algorithms was set to 3. The original image is the first one from the left. In the EM segmented image which comes the second, color yellow denotes the WBA regions, color cyan denotes the cytoplasm regions and color blue denotes the background. In the image segmented by our method which shows on the right most, color red denotes the WBA region, color green denotes the cytoplasm region and color blue denotes the background

gaussian filtering leads to higher dimension of the feature space and higher accuracy in segmentation. However we need to trade off the accuracy with computation cost and 3 is a optimal choice that strikes a good balance between these two considerations.

Fuzzy c-means (FCM) is a method of clustering which allows one piece of data to belong to two or more clusters. This method is frequently used in pattern recognition. It is based on minimization of the following objective function:

$$J_m = \sum_{i=1}^{N} \sum_{j=1}^{C} \mu_{ij}^m \parallel x_i - c_j \parallel^2 \tag{3.4}$$

where m is any real number greater than 1, $\mu_{i,j}$ is the degree of membership of x_i in the cluster j, x_i is the i^{th} element of d-dimensional measured data, c_j is the d-dimension center of the cluster, and $\parallel * \parallel$ is any norm expressing the similarity between any measured data and the center. Fuzzy partitioning is carried out through an iterative optimization of the objective function shown above, with the update of membership $\mu_{i,j}$ and the cluster centers c_j are given by:

$$\mu_{i,j} = \frac{1}{\sum_{k=1}^{C} (\frac{\|x_i - c_j\|}{\|x_i - c_k\|})^{\frac{2}{m-1}}} \tag{3.5}$$

$$c_j = \frac{\sum_{i=1}^{N} \mu_{i,j}^m \cdot x_i}{\sum_{i=1}^{N} \mu_{i,j}^m} \tag{3.6}$$

The iteration stops when $max_{i,j}|\mu_{i,j}^{(k+1)} - \mu_{i,j}^{(k)} < \varepsilon$, where ε is a termination criterion between 0 and 1, whereas k is the index for the iteration steps. This procedure converges to a local minimum or a saddle point of J_m. The algorithm is composed of the following steps:

1. Initialize $U = \mu_{i,j}$ matrix, $U^{(0)}$

2. At k-step: calculate the centers vectors $C^(k) = [c_j]$ with $U^{(k)}$

$$c_j = \frac{\sum_{i=1}^{N} \mu_{i,j}^m \cdot x_i}{\sum_{i=1}^{N} \mu_{i,j}^m} \tag{3.7}$$

3. Update $U^{(k)}$, $U^{(k+1)}$

$$\mu_{i,j} = \frac{1}{\sum_{k=1}^{C} \left(\frac{\|x_i - c_j\|}{\|x_i - c_k\|} \right)^{\frac{2}{m-1}}} \tag{3.8}$$

4. If $||U^{(k+1)} - U^{(k)}|| < \varepsilon$ then stop; otherwise return to step 2

3.2.2 Second Tier: Sub-Cellular Feature Extraction

3.2.2.1 Sub-cellular feature definition

There are sub-cellular features to be defined here in our framework:

- Cortical actin
- Inner Punctuate Dots
- Stress Fibers

Cortical actin and inner punctate dots together compose the white bright actin. And the stress fibers are dimmer linear structures embedded in the cytoplasm. Fig.3.4 shows an example of each feature mentioned above. Generally cortical actins refer to the white

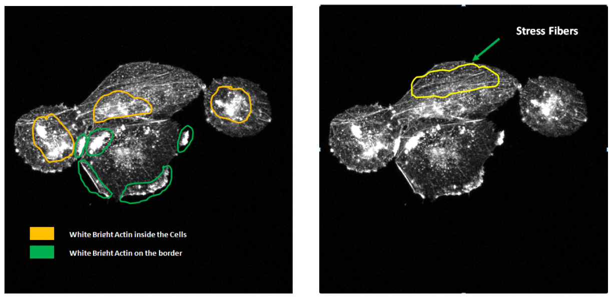
White Bright Actin inside the Cells
White Bright Actin on the border
Stress Fibers

Figure 3.4: The sub-figure on the right gives an example of the two sub-structures belonging to white bright actin and the sub-figure on the left gives an example of the stress fibers which belongs to the cytoplasm component generated from the first tier decomposition.

bright spiky structure lying on the edges of the cells but sometimes they might round up on the corner. Inner punctuate dots refer to the scattered white bright regions inside the cells which look more round.

3.2.3 Cortical Actin Versus Punctuate Dots

Since cortical actin has shape ambiguity we use the cellular locations to distinguish between the cortical actin and inner punctuate dots. The simple criterion is: cortical actins appear on the border while punctuate dots appear inside the cell. The cell border information could be obtained using the procedures as follows:

1. Apply OTSU Method[21] to the Cyto and WBA band component image and then obtain the foreground stencil by combining the binary results of the two
2. Use connected components to identify different regions within the image and save one pixel on the boundaries for each component
3. Apply active contour to trace the outline of each region
4. compute the centroid of each region and building a vector towards the centroid for all the pixels on the border
5. Shrink the border outline along the vector. If the new frontier encountered is the foreground in the WBA region label the pixel as cortical actin and stops when first meeting a background pixel

Here Otsu's method is used to automatically perform histogram shape-based image thresholding,[23] or, the reduction of a gray level image to a binary image. The algorithm assumes that the image to be thresholded contains two classes of pixels (e.g. foreground and background) then calculates the optimum threshold separating those two classes so that their combined spread (intra-class variance) is minimal. By performing the procedures above we can identify all the F-actin structure residing on the border and the remaining will be punctuate dots. And thus we can get a mask for all the cortical actin in the image and at the same time a mask for inner punctate dots. The segmentation of the cell boundaries need not to be very exact since the distance transformation and thresholding will be tolerant to small boundary location variation.

3.2.3.1 Stress Fiber Extraction

Though stress fibers appear as dark lines embedded in the cytoplasm, we don't use anisotropic filters to extract them since we don't have their direction information and in most cases stress fibers are intertwined with each other forming a very complex network. Therefore our strategy here is to compute a line score for each pixel then threshold it using Otsu method[21]. We can directly use the cytoplasm part obtained from the tri-band segmentation. However the present of the halos of cortical actin will be a strong interference for stress fiber detection. Thus we need to remove them first before computing the line score. This is achieved by the following 2 steps:

1. Compute the edges of the bright actin segments using sobel operators

2. Dilate the extracted edges with a disk shape structural element to well cover all the halos that remain in the cytoplasm part for generating a mask to removing them.

Due to the heavy blurring of the stress fiber bodies, we do the following enhancement to the halo free images and the enhanced intensity value are used as the line score per pixel.

1. $I_g = H_{log}(I)_{5\times 5}$

2. $I_r = I - I_g$

3. $I_f = H_{unsharp}(I_r)_{\alpha=0.8}$

Here I denotes the halo free images, I_g denotes the image after LOG filtering and I_f denotes the resultant image. After running the above procedure, we apply OSTU thresholding to the enhanced image I_f which segment two parts with the foreground denoting extracted stress fibers.

3.3 Multi-scale Cytoplasm Texture Analysis

3.3.1 Definition of cellular**texture**

Previous studies often treat everything other than the nuclei inside the cell as **texture** without excluding the subcellular structures. However,based on the observation of a large amount of images and consulting with the biologists, it is considered much more reasonable to think about the "**textures**" and subcellular structures in two different scenarios. Sub-cellular structures, such as cortical actins, punctuate dots or even stress fibers are usually easily identifiable from their surrounding environments and thus it doesn't make much sense to define the sub-cellular structures we already know as the texture of the cells. In our work we use "**texture**" to define something that looks ambiguity in the cytoplasm, e.g. diffusions.

3.3.2 Why multiscale?

We integrate the texture analysis into a multi-scale framework to effectively get rid of the interference of the sub-cellular structure during the texture analysis. After performing a quad-tree decomposition on the image we get the subcellular structures and the real **texture**, which we focus on, are separated to different patches at different scales. Since the intensity within the subcellular structure is more homogenous than that of the textural regions in the cytoplasm, their measurements from the texture analysis based on co-occurrence matrix will be very low, most of the time very close to 0 and thus their effects will be almost filtered out. To accommodate the difference in scales, we use a weighted scheme to sum up the results from different sized patches.

3.3.3 Detailed Methodology

A co-occurrence matrix, also referred to as a co-occurrence distribution, is defined over an image to be the distribution of co-occurring values at a given offset. Mathematically, a co-occurrence matrix C is defined over an n x m image I, parameterized by an offset $(\Delta x, \Delta y)$, as:

$$C_{\Delta x,\Delta y}(i,j) = \sum_{p=1}^{n}\sum_{q=1}^{m} \begin{cases} 1, & \text{if I(p,q) = i and } I(p+\Delta x, q+\Delta y) = j; \\ 0, & \text{otherwise.} \end{cases} \tag{3.9}$$

Since the cytoplasm is a highly textured region in which stress fiber is the structure of interest we develop algorithms for quantifying the stress fibers. However, texture analysis of the cytoplasm is still needed to indicate the morphology distribution of stress fibers. Here we introduce a new term called a weighted multi-scale GLCM (Gray Scale Co-occurrence Matrix) (WMGCLM) to characterize the inner complexity of the cytoplasm. The algorithm for WMGCLM is described as below:

1. Set the smallest patch scale to 1 and apply quadtree decomposition the cytoplasm part
2. Filter out all the patches at scale 1. Then, at each scale compute the GLCM for all then square segments belonging to that scale and get their contrast, energy and homogeneity shown as in Eq.3.10 to Eq.5 3.12

3. Use the average of measurements from all the segments at one scale to form the measurement at that scale

4. Weight the measurement from different scales by diving the elementary area at that scale and sum them up as shown in Eq.3.13

$$GLCM_1 = \sum_{i,j} \mid i - j \mid^2 p(i,j) \tag{3.10}$$

$$GLCM_2 = \sum_{i,j} p(i,j)^2 \tag{3.11}$$

$$GLCM_3 = \sum_{i,j} \frac{p(i,j)}{1 + \mid i - j \mid} \tag{3.12}$$

$$WGLCM_k = \sum_{s=1}^{S-1} \frac{\frac{1}{N_s} \sum_{i=1}^{N_s} GLCM_k^{i,s}}{W_s^2} \tag{3.13}$$

where k denotes the type of the measurement; S denotes the maximum decomposition depth;N_s denotes the number of patches associated with scale s, W_s denotes the patch size at decomposition level s and $GLCM_k^{i,s}$ denotes the type k GLCM measurement of block i at decomposition level s

3.4 Feature Quantification

3.4.1 Overview

We divided the quantification into two parts, scalar quantification, which is one numerical result per image and vector quantification, which is a stack of numerical results per image. All the scalar quantification results are visualized via box plot. We put three concentrations of each drug in one image and list them from low to high. The DMSO is put on the left most side in each figure as a reference.

3.4.2 Scalar Quantification

Since we have extracted 5 kinds of feature regions, WBA, cytoplasm, stress fibers, cortical actin and punctate dots, there are many possibilities to build the quantification measurements based on the ratios between combination of the intensity or area of these regions. Here we only selected the top 10 ones which are the most effective to differentiate the drug dependant and concentration dependant changing behaviors for profiling purpose. They are:

1. Integrated Actin (WBA + cytoplasm) Intensity to Nuclei Area
2. Average Cytoplasm Intensity per Image
3. Average Actin (WBA + cytoplasm) Intensity per Image
4. Integrated Actin (WBA + cytoplasm) Intensity to cell area
5. Stress fibers to cell area
6. Cortical actin area to the punctuate dots area
7. Cortical actin area to Nuclei area
8. Multi-scale Cytoplasm Texture Contrast
9. Multi-scale Cytoplasm Texture Homogeneity
10. Multi-scale Cytoplasm Texture Energy

These measurements are named scalar quantification since there is only one number associated with each image. And they are oblivious to the image resolution. Since the metrics are built on per image basis no individual cell separation technique is needed and less uncertainly in the computed results will be incurred. These measurements are picked based on a large amount of experiments for testing the effectiveness of different metrics to profile the visualized changes within cells under different treatments. In the validation part we will demonstrate the power of the scalar quantification for drug profiling on different cell lines.

3.4.3 Cytoplasm Nuclei relative distribution Quantification

Similar to work done by B.Roysam [8] which uses distance transformation to associate different cellular structures, we model the relative distribution of cell nuclei to the rest part of the cell, namely the white bright actin and cytoplasm by computing the geodesic distance transform for all the points in the bright actin segments and cytoplasm segment to the cell nuclei using the method introduced by O. Cuisenaire in [7].

Here the geodesic distance between two pixels p and q is the length of shortest path from p to q. Suppose Path P ($P = p_1, p_2, p_3, \cdots, p_n$) is a path between pixels p_1 and p_n, i.e. p_i and p_{i+1} are connected neighbors for $i \in \{1, 2, \cdots, n-1\}$ and p_i belong to the domain for all i. The path length l(P) is defined as:

$$l(P) = \sum_{i=1}^{n-1} d_N(p_i, p_{i+1}) \tag{3.14}$$

the sum of the neighbor distances d_N between adjacent points in the path. A particular geodesic metric is defined by which neighbors are considered to be connected and the values of d_N for each pair of connected neighbors.

The geodesic distance transformation statistics was organized into 257 bins. In the first 256 bins distances that were evaluated between i and i+1 fall into the i^{th} while the last bin represents a count of the distances that are larger than 256. This could be named as vector quantification. We don't use histograms to show the distribution since we need to compare 4 conditions, 3 concentrations for one pharmaceutical and the DMSO, in one

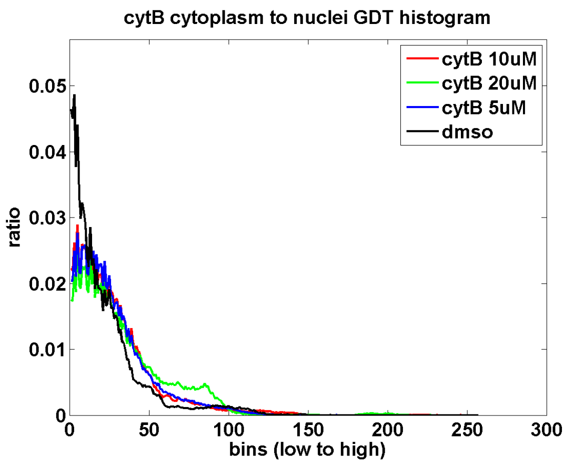

Figure 3.5: This figure shows the visualization of cytoplasm nuclei relative distribution quantification

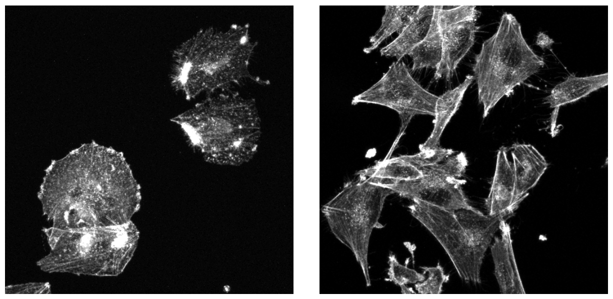

Figure 3.6: The left sub-figure shows the cells treated under cytoB in 10um/ml and the right sub-figure shows the cells treated under DMSO.

graph.

Therefore it is easier to use linear plots to show the trend of multi-variables simultaneously than histogram. Most of the distinguishable differences between subjects appear when x is small and the visualization is very helpful to illustrate these differences. An example is shown in Fig.3.5 This relative distribution plot is vert effective to profile the morphology differences between cell images. Cells that have more spiky shape and have uneven distribution of cellular structures shows sharp changes in their geodesic distance profiles when increasing the index of bins while cells that have more round shape or have relative even distribution of cellular structures shows smooth changes in their profiles.This explanation could be well proved by the Fig.3.6, which compares the images of cells treated with cytoB and the ones treated with DMSO.

Chapter 4

Parallel Tri-band Segmentation using CUDA

4.1 Why CUDA?

Tri-band Segmentation is a time consuming step in our automatic analysis framework, especially when the size of the image is very large, say 512 by 512 or 1024 by 1024. Since most of the steps are done per pixel basis or in a quite small local neighborhood, e.g. 5 by 5 Gaussian Filtering. Therefore it is natural to map the algorithm to Single Instruction Multiple Data(SIMD) architecture for a significant speedup. In this chapter we described the implementation of our tri-band segmentation algorithm on the NVIDIA CUDA architecture. The performance improvement will be shown in the later section.

4.2 CUDA Introduction

4.2.1 What is CUDA

CUDA (an acronym for Compute Unified Device Architecture) is a parallel computing architecture developed by NVIDIA. CUDA is the computing engine in NVIDIA graphics processing units or GPUs that is accessible to software developers through industry standard programming languages.

4.2.2 Programming Model

CUDA extends C by allowing the programmer to define C functions, called kernels, that, when called, are executed N times in parallel by N different CUDA threads. Each of the threads that execute a kernel is given a unique thread ID that is accessible

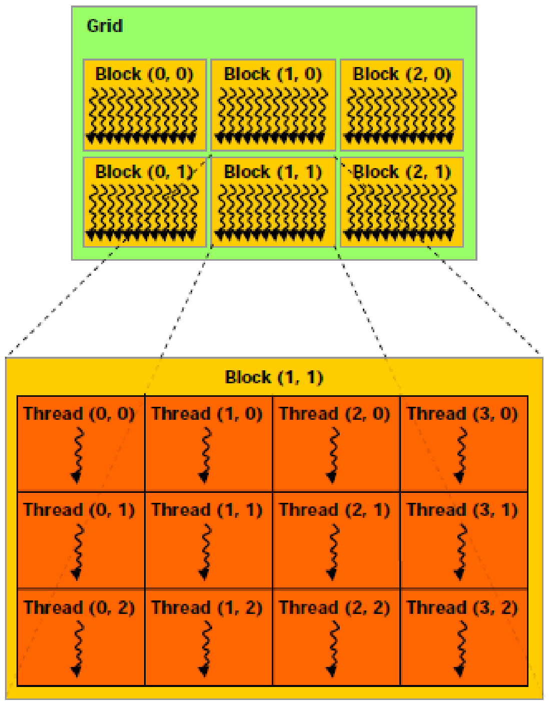

Figure 4.1: This figures illustrate the CUDA Programming Model. Courtesy Image from NVIDIA CUDA Programming Guide 2.0.

within the kernel through the built-in threadIdx variable. For convenience, threadIdx is a 3-component vector, so that threads can be identified using a one-dimensional, two-dimensional, or three-dimensional index,forming a one-dimensional, two-dimensional, or three-dimensional thread block. Threads within a block can cooperate among themselves by sharing data through shared memory. The number of threads per block is restricted by the limited memory resources of a processor core. On the NVIDIA G80 architecture, a thread block may contain up to 512 threads. A kernel can be executed by multiple equally-shaped thread blocks. These multiple blocks are organized into a one-dimensional or two-dimensional grid of thread blocks as illustrated by Fig.4.1

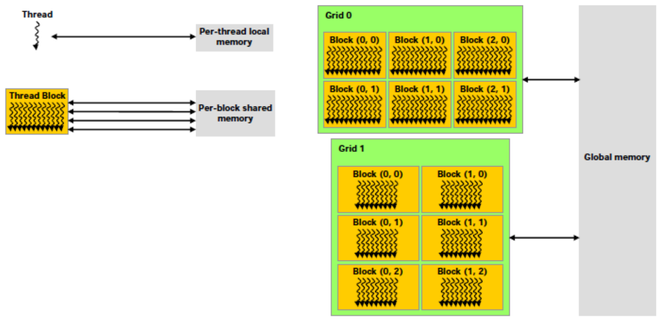

Figure 4.2: This figure shows the hierarchy of the CUDA memory model. Courtesy image from CUDA Programming Guide 2.0.

4.2.3 Memory Model

CUDA threads may access data from multiple memory spaces during their execution as illustrated by Fig.4.2. Each thread has a private local memory. Each thread block has a shared memory visible to all threads of the block and with the same lifetime as the block. Finally, all threads have access to the same global memory. There are also two additional read-only memory spaces accessible by all threads: the constant and texture memory spaces. The global, constant, and texture memory spaces are persistent across kernel launches by the same application. CUDA assumes that the CUDA threads may execute on a physically separate device that operates as a coprocessor to the host running the C program. CUDA also assumes that both the host and the device maintain their own DRAM, referred to as **host memory** and **device memory**, respectively. Therefore, a program manages the global, constant, and texture memory spaces visible to kernels through calls to the CUDA runtime. This includes device memory allocation and deallocation, as well as data transfer between host and device memory.

4.2.4 Global Memory Access Coalesce

Global memory access pattern affects the overall performance of CUDA application. Though there are also other issues related to performance optimization, we only mention the global coalesced memory access here since this is relevant to our work. The global memory access by all threads of a half-warp is coalesced into one or two memory transactions if it satisfies the following three conditions:

- Threads must access:

    1. Either 32-bit words, resulting in one 64-byte memory transaction
    2. Or 64-bit words, resulting in one 128-byte memory transaction
    3. Or 128-bit words, resulting in two 128-byte memory transactions

- All 16 words must lie in the same segment of size equal to the memory transaction size (or twice the memory transaction size when accessing 128-bit words)
- Threads must access the words in sequence: The k^{th} thread in the half-warp must access the k^{th} word.

If a half-warp does not fulfill all the requirements above, a separate memory transaction is issued for each thread and throughput is significantly reduced.

4.3 CUDA Based Tri-band Segmentation

4.3.1 Framework

Our CUDA based tri-band segmentation contains 4 kernels. Their names and functionalities are listed in Table.4.1 as follows. The implementation includes two major steps:

Step I Point Wise Feature Extraction

Kernel Name	Functionality
GaussianFilterKernel	Generate one component image in the feature map
NormalizeMembershipKernel	Normalize the random group assignment after the initialization
UpdateCentroidKernel	Update the centroid location of each group using the new group assignment
UpdateMembershipKernel	Update the membership assignment using the new centroids

Table 4.1: This table shows the name of the kernels and their corresponding functionalities.

Step II Fuzzy C means Segmentation

The pseudo code of these two steps are shown in Table.4.2 and Table.4.3 respectively. In these two tables, the parts done on the CPU and GPU are explicitly labeled with all the kernel function calls highlighted in bold font. The array stored in the device memory are highlighted in bold font while the array stored on the CPU memory are highlighted in capital letter font. They could also be distinguished by the array name prefix **d-**(device) and **h-**(host CPU). Table.4.4 charts the block and grid size configurations of all the kernels.

4.3.2 Point wise Feature Generation

In point wise feature generation step, we apply a 5 by 5 gaussian filter on the image for 3 times. The 3 resultant images plus the original image build the feature map for all

```
Point Wise Feature Extraction

Read in the image ORGIMG(CPU)
*Transfer the data from CPU to GPU*
$I_0 \leftarrow$ ORGIMG
for i:= 1 to iterNum
    $I_i$ = **GaussianFilterKernel**($I_{i-1}$) (GPU)
end for
**d-image**$\leftarrow[I_0, I_1, I_2, I_3]$
```

Table 4.2: This table shows the pseudo code of the Point Wise Feature Extraction Step.

the pixels. We call any of resultant images or the original image a sub-image and store all the sub-images consecutively in the device memory, which needs 4 MB for a 512 by 512 image and 16 MB for a 1024 by 1024 image. During the computation of each sub-image we call **GaussianFilterKernel**. The computation step and the memory layout are illustrated in Fig.4.3 For the **GaussianFilterKernel** we decompose the image into 16 by 16 tiles and assign each tile to a thread block. The size per thread block is 16 by 20 and the width is equal to the size of a half warp which allows global memory access coalescing. Besides fetching the data to be updated we also need to fetch the boundary regions for each tile. Therefore our problem is to how to use a 16 by 20 thread block to fetch a 20 by 20 tile from the device memory. In the y direction we have extra threads to complete the work. However on the X direction we only have 16 threads for a line that has 20 elements. This could be achieved by either assigning more work to some of the threads or letting all of them do extra work. Here, in the CUDA programming scenario,

```
Fuzzy C Means Segmentation

Step 1. Initialization

Random Generate $width \times height \times N$ matrix h-membership (CPU)
Transfer data from CPU to GPU
d-membership ← NormalizeMembershipKernel(d-membership)(GPU)
d-centers ← UpdateCentroidKernel(d-image,d-membership)(GPU)

Major Iteration

while (! Converge)
d-membership ← UpdateMembershipKernel(d-centers, d-image)(GPU)
d-centers ← UpdateCentroidKernel(d-image,d-membership)(GPU)
memcpy(H-PREV-CENTERS, H-CENTERS)(CPU)
Transfer data from GPU to CPU
H-CENTERS←d-centers
CheckConvergence(H-CENTERS,H-PREV-CENTERS)(CPU)
end
```

Table 4.3: This table shows the pseudo code of the Fuzzy C Means Segmentation Step.

Kernel Name	bDimX	bDimY	bDimZ	gDimX	gDimY
GaussianFilterKernel	16	20	1	$\frac{Width}{16}$	$\frac{Height}{16}$
NormalizeMembershipKernel	32	16	1	$\frac{Width}{32}$	$\frac{Height}{16}$
UpdateCentroidKernel	32	16	1	$\frac{Width}{32}$	$\frac{Height}{16}$
UpdateMembershipKernel	32	16	1	$\frac{Width}{32}$	$\frac{Height}{16}$

Table 4.4: This table shows the execution configurations of all the kernels. "Width" and "Height" denotes those of the input image.

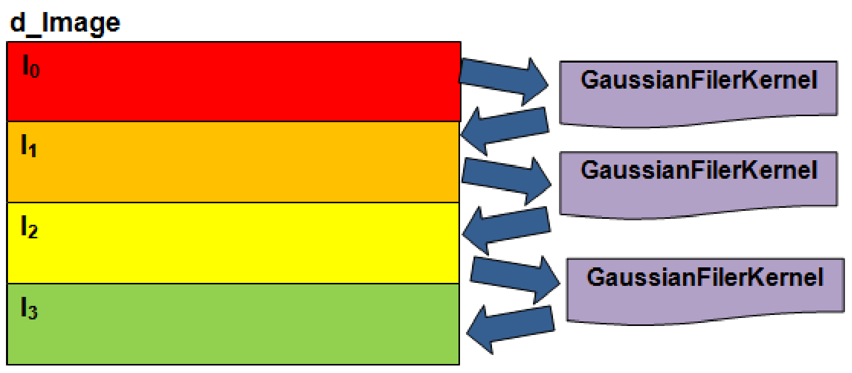

Figure 4.3: This figure shows the computation step and memory layout of the point wise feature extraction step.

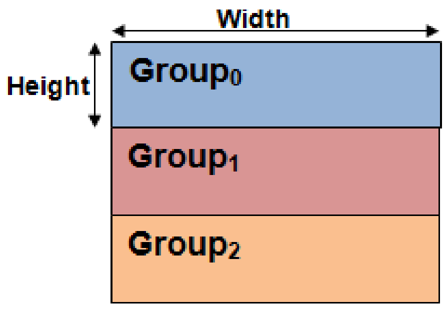

Figure 4.4: This figures shows the memory layout for the membership matrices

we prefer the latter since having part of the threads in a consecutive 16 unit idle will bring execution divergence inside a warp. This will hurt performance. Therefore we first use all the threads in a warp to fetch the corresponding 16 contiguous elements starting at the address that has -2 offset from the 16 memory locations that receiving the output from the threads later, then shift the starting address 4 byte higher and read the corresponding 16 elements there. Though there is an overlap between the threads in these two batches of memory reads, this strategy works faster than the first one since each of the two big reads is fully coalesced.

4.3.3 Fuzzy C Means(FCM)

In addition to storing the feature map for the image in the device memory, we also need to store the membership matrices for all the groups on the GPU. The size of each membership matrix is the same as the size of the input image. The memory layout is illustrated is Fig.4.4 The first step in FCM is to generate the random membership map. In our case, three random floating point numbers between 0 and 1 are generated per pixel. Then we transfer the data from the CPU to the GPU and perform the membership vector normalization on the GPU. We try to avoid doing the normalization on the CPU since elements in different membership matrices need to be accessed for the processing of one pixel and two neighboring elements in the membership vector belonging to one pixel has one membership matrix size gap between their storage locations. This memory access pattern will incur a lot of cache misses which negatively affect the performance. But on the GPU no such issues arise.

In **NormalizeMembershipKernel**, the size of a thread block is 32 by 16 and the size of a grid is $\frac{width}{32}$ and $\frac{height}{16}$, where width and height denote the dimensions of the image. All the threads read in membership numbers from three matrices simultaneously. After obtaining the new membership assignment on the device memory, we use the new member ship to update the centroids of the group. This is a gathering operation and we need at least two parallel reduction passes through the whole dataset, one for computing $\sum_{i=1}^{N} u_{ij}^{p}$ and the second for computing the new centroid. Though there are 4 components in the centroid vector they share the same $\sum_{i=1}^{N} u_{ij}^{p}$. Therefore we use segmented parallel reduction, which performs parallel reductions on separated rows simultaneously to compute the 4 components per center at the same time. We then start the iterative procedure. During each iteration, the centers are first updated using new membership matrices calculated in the last step then the membership matrices are updated according to the computed new centroid. We have two separate kernels, **UpdateCentroidKernel**, **UpdateMembershipKernel** for these two steps respectively. The **UpdateCentroid-Kernel** kernel is the same one used in the initialization step. The update of membership is done per pixel basis, and hence we use a pixel as the grain size. The same domain decomposition strategy is applied in **UpdateCentroidKernel**. The feature vector per pixel and its three reference centroid are stored in per thread registers. Since all the threads need to read from the same memory locations for the reference centroid the broadcasting

Image Size	CUDA Version (sec)	Matlab Version	Speedup	Iteration No.
512 x 512	1.6406	26.0785	15.896	45
1024 x 1024	10.2628	205.4623	20.020	80

Table 4.5: This table compares the runtime results of the CUDA version and Matlab implementation of our tri-band Segmentation method.

mechanism for the global memory reads makes the data available to all the threads at the same time. After getting its own feature vector and the three reference centroid, each thread could generate the new membership values per pixel in all the three membership matrices.

4.4 Experiments and Results

In this section we compare our CUDA based tri-band segmentation implementation with the original Matlab implementation. The CUDA version is tested on a machine with the hardware configuration:

CPU Intel(R) Core(TM)2 Duo T5800 @ 2.00GHz each, 2.00GB memory, 32 bit Windows Vista OS

GPU NV9600 MGT, 512MB device memory, 1.25GHz clock rate, 4 multi-core processor per die.

We apply the CUDA version to two images. The first image is of 512 by 512 and the second is of 1024 by 1024. The running time of the Matlab version and the CUDA version are listed in Table.4.5 below. We include the resulting speedup in the fourth column. It can be seen from the table that we obtain a higher speedup on the larger sized image since more data parallelism can be exploited in that case.

Chapter 5

Automatic Synthetic Data set Generation

To confirm the reliability and accuracy of the automatic analysis method, we need a large amount of test images under diversified conditions. However such a process is intractable for human labeling. Therefore synthetic benchmark generation is needed for algorithm validation purposes. In this chapter, we propose a computational framework for simulating the confocal microscopy images at cell level, in which the F-actin structures are stained and highlighted by the Green Fluorescent Protein (GFP). The generated images and used for validating our F-actin feature extraction and quantification algorithms described in the previous chapters.

5.1 Related Work

Some previous studies have utilized simulation when developing algorithm for automatic analysis of microscopy images. For instance, in [19] and [3], simplistic simulation was used when developing analysis methods for confocal microscopy. In [19] only a few types of shape, spheres, disks and ellipsoids were used as simulation models while in [3] these objects are manually plotted by hand. In [10], randomly located spheres were simulated when measuring the performance if 3D-FISH spot counting algorithms. The work done by Antti et.al.[17] provides more flexibility in the shapes that could be simulated by their system and the users could control several parameters in the image generation process. However, their work is catering to fluorescent microscope so the resolution of the cell images is not very high. They didn't deal with specific intra-cellular structures but simply treat them as textures, which is also adopted in Svoboda's work[18] for simulating the 3D fluorescent microscopy images. Our work differs from these previous ones in mainly two aspects:

- We generate test images for confocal microscopy which has much higher resolution so the cellular features could be seen in more details.
- We highlight specific cellular structure in our simulation images which, in reality is the F-actin protein stained and highlighted by GFP.

As a tunable system, we also provide many user controllable parameters as what is done in Antti's paper[17].

5.2 Methodology Overview

The cell population we are going to simulate forms into a clump. They are touching each other on the boundary but don't have much overlapping. The specific F-actin structures highlighted in our framework are cortical actins, which appears as bright lines lying on the boundary of the cells and stress fibers, which appears as dim filament structures inside the cell. Fig. 2 in Chapter 3 shows an example of the appearance of cortical actin inner punctuate dots and stress fibers inside the cells. Based on the characteristics of the imaging systems, we get the images in two different channels, one stores only the F-actin and the other stores the nuclei. We are mainly interested in validating the algorithm for analyzing the F-actin channel images. In order to make the generated images more realistic, the regions where nuclei are located in the simulated F-actin channel images were made darker than the surrounding cytoplasm. An image is built in a bottom up fashion. We start from obtaining different component images, such as the cell body, stress fibers

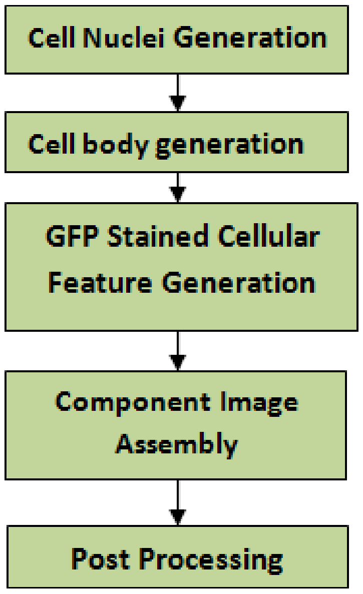

Figure 5.1: The figure shows the computational framework of our F-actin highlighted confocal microscopy image simulation

and cortical actin and later combine them together.Fig.5.1 charts the main steps in our framework. In the following context the binary map for cells or cellular structure will be called a mask of that subject.

5.3 Algorithms

5.3.1 Cell nuclei generation

We use the method for simulating cell bodies described in Antti's paper[17] to generate the nuclei and the shape variation parameter is suppressed to avoid getting very weird shapes. We sample on the circle made by nuclei boundary with a set of coordinates.$x_i, y_i (i = 1 \ldots n)$, where n is the number of sampling points. We use polar coordination system. The equations for obtaining the positions of the sampling points

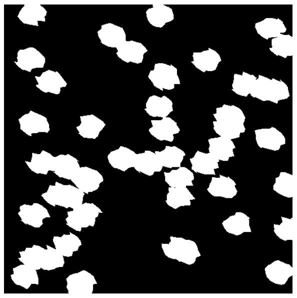

Figure 5.2: This figure shows an example of the generated nuclei mask using the method described above.

could be mathematically represented as follows:

$$\begin{cases} x_i(\theta_i) = r[U(-\alpha,\alpha) + cos(\theta_i + U(-\beta,\beta))] \\ y_i(\theta_i) = r[U(-\alpha,\alpha) + sin(\theta_i + U(-\beta,\beta))] \end{cases} \tag{5.1}$$

where $i = 1 \dots N$, U(a,b) is a uniform distribution on the interval [a, b]. The parameter β controls the randomness of sampling and α controls the randomness of the radius for the circle. We use the following steps to generate the binary mask of cell nuclei.

- Randomly generate N numbers between 1 and W as the X coordinates and N numbers between 1 and H as the Y coordinates for the cell nuclei centroid
- For each centroid, use Eq.5.1 to generate the nuclei body

Here W and H are the width and height of the image and N is a user controllable parameter. Fig.5.2 shows an example of the generated nuclei mask.

5.3.2 Cell Body Generation

In our case the cells in an image are either separated or slightly touching each other on the boundary. Therefore the algorithm for generating the cell bodies described in Antti's paper[17] is not applicable to our case since in their approach users don't have control on the amount of overlaps between cells. However, when the population density is not set too high, their method could be used to generate the cell nuclei . The cell nuclei mask is used to generate the cell bodies. The procedure is detailed as follows.

step 1 Apply distance transformation to the cell nuclei mask and get an intermediate image.

step 2 Use the nuclei mask as markers and apply maker controlled watershed segmentation to the intermediate image.

step 3 Turn on the visibility of different segments with a probability p_{cell}

step 4 Use a radius 3 disk structuring element to erode the resultant image from third step

Fig.5.3 shows an example of the distance map generated from the cell nuclei mask, watershed segmented image color labeled by patch indices and the final cell body mask after erosion. Here the parameter p controlled the density of cells.

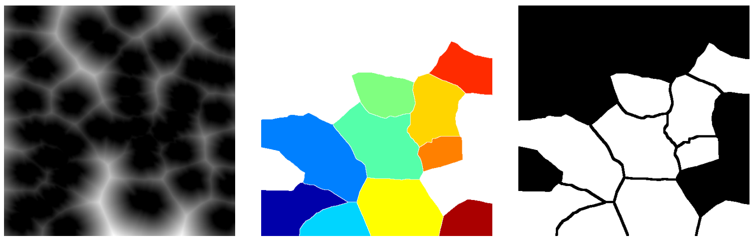

Figure 5.3: The images from left to right are: L) The distance map generated from the cell nuclei mask, 0 is mapped to black and the maximum distance in the image is mapped to white. M) Marker controlled watershed segmentation result color labeled according to patch indices after visibility filtering. R) The final cell body mask after the erosion

5.3.3 Cellular Feature generation

To simulate the GFP staining effects, we need to highlight the cellular features, namely the F-actin. Since we mainly use the synthetic images to validate the extraction algorithms for cortical actin and stress fibers, only the these two cellular features appear in the final simulated images and they are generated independently. We use the following procedures to obtain the cortical actin mask.

- Covert the cell body mask to gray scale image and apply the "Canny" operator to get the edge map
- Dilate the edge map with a radius 3 disk shape structuring element
- Use the segment labels to cut out body and edge mask for each cell
- Erode the sub edge map with a line shape structuring element. The length is set as E_w using Eq.5.2 and the direction is perpendicular to the major axis of the cell

$$E_w = c_w \times A(C_k) \tag{5.2}$$

Here c_w is a constant coefficient determined by the statistics from the real images and $A(\cdot)$ denotes the operation that computes the area covered by cell, C_k. In step 4, we chose the direction of the structuring element to be perpendicular to the major axis due to the fact that in real images, cortical actin are mostly lying parallel to the major axis of the cells. After all these steps we get the cortical actin mask for each cell. An example is shown by the first sub-Figure in Fig.5.4.

The stress fiber filament network, which could be considered as a intertwined complex network are generated using the steps listed below:

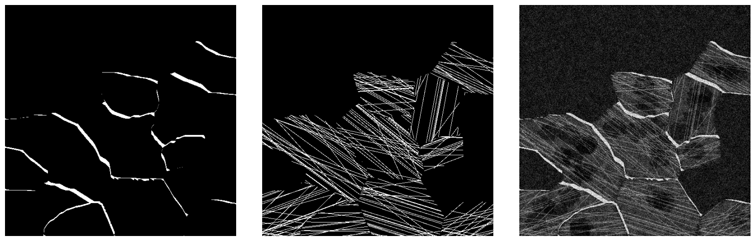

Figure 5.4: The first sub-figure from the left shows an example of the generated cortical actin mask. The second sub-figure shows an example of generated stress fiber mask. The last sub-figure from the left shows the final result of our automatic benchmark generation framework

- Get the major orientation of each visible cell body patch, which should be the direction of the long axis of the bounding ellipse of the patch and store it as θ_k
- Generate a scan line binary image and make visible for each line with a larger probability p_{major} and rotate the scan line image by θ_k.
- Increase the complexity of the network by generating a few more scan line binary images and make visible for each line with a smaller probability p_{major}. The images are rotated by a random angle dithered by δa from θ_k. Here δa conforms to a uniform distribution and the bound is set to [-30, 30] based on a significant number of experiments.
- Combine the scan line images with "OR" operation and filter the result with the submask of the cell body and thus obtain the stress fiber mask for each cell.
- Combine the stress fiber mask for each cell to get stress fiber mask for the whole image

The second sub-Figure from left in Fig.5.4 shows an example of the generated stress fiber mask using the method mentioned above.

5.3.4 Assembling Components

After generating the component images of cell bodies and cellular features, we use the algorithm listed below to combine them together. We start with four sub-images: the cell body mask, the cortical actin mask, the stress fiber mask and nuclei mask.

- Remove the regions covered by cortical actin mask from the cell body mask using cortical actin; assign an intensity I_1 to the remaining foreground and add gaussian noise, 0 mean and 0.05 deviation to the resultant image.
- Blur the image from the last step with a diameter 5 circular averaging filter
- Add the stress fiber by adding an intensity which confirms to gaussian distribution with the mean at I_2 and standard deviation at 1 to the regions covered by the stress fiber mask.
- Add the cortical actin by assigning intensities that confirms to a uniform distribution between $\frac{2}{3}I_3$ and I_3 to the areas covered by cortical actin mask.
- To approximate the effect of the nuclei shadow in the F-actin channel image an intensity I_4 is subtracted from regions covered the nuclei mask after filtered by visibility p.

Based on the visual inspection through lots of experiment with the parameters, the configuration of $I_k(k = 1 \ldots 4)$ is chosen as: $I_1 = 54, I_2 = 82, I_3 = 251, I_4 = 29$. The parameter and distribution models are computed from the statistics of real images. If the intensities for certain cellular features don't vary much in the real images we just use constant in our simulation.

5.3.5 Post Processing

We need to make the generated images noisy and blurry enough to approximate the real situations. To achieve this we post process the clean image obtained after the previous steps by put them through a pipeline that simulates the imaging process of the capture device. We use the following steps to get the final contaminated image.

- Apply diffusion operation to the image $IterN_{diffusion}$ times
- Add Poisson Noise to the image.
- Add Gaussian Noise with 0 mean and deviation 0.01.
- Blur the image with a PSF function described in Eq.5.3.

Here the Poisson Noise simulates the noise brought in by the imaging detector while the Gaussian simulates noise from heat. We used the format of point spread function proposed in Antti's paper[17]. The PSF centered at location(p.q) mapping an image pulse into location(x,y) is given by:

$$PSF(x, y, p, q) = exp(-\frac{0.5}{\sigma_g^2(p, q)}((x - p)^2 + (y - p)^2)) \tag{5.3}$$

where the amount of blurring is controlled by the variance $\sigma_g^2(p, q)$ of the gaussian kernel. Since confocal images use the confocal planes to get rid of much of the out of focus line. The blurring introduced by the PSF should not be very high.The last sub-figure in Fig.5.4 shows an example of the final result of our synthetic image generation framework.

Chapter 6

Validation and Results

6.1 Feature Extraction Validation

The initial validation is carried out by comparing the feature extraction results with the ground truth provided by a domain expert. We ask an expert on actin patterns from National Cancer Institute to selectively label the actin patterns he saw in an image. Since he did not manually delineate all the actin feature regions in the images, the algorithm might find more stress fiber or cortical regions than what are labeled. This is considered correct when the additional detected parts are confirmed by the domain experts. It is considered as making an error when:

- The algorithm misses or misclassifies regions that the expert marked.
- The detected regions don't contain any F-actin patterns at all.

Therefore we compute two numbers corresponding to the different features for all the images. Here the definition for "detected rate" and "mis-detected rate" differ from what they used to be in the previous literatures.

1. **Detected rate**, which is ratio between the algorithm detected correct regions labeled by the expert and the total region labeled by the expert for that feature
2. **Mis-detected rate**, which is the ratio between the algorithm detected regions that are not subcellular structures plus the mis-classified regions and the total detected region for that feature.

	C.A.	I.P.D.	S.F.
detected rate	100%	95.3%	98.7%
mis-detected rate	1.2%	6.5%	8.6%

Table 6.1: This table shows the computed detected rate and mis-detected rate for all the images in the manual validation dataset. Here C.A. is short for cortical actin; I.P.D. is short for inner punctuate dots and S.F. is short stress fibers.

Table.6.1 shows the computed two numbers for all the images in this manual validation set, here we mainly focus on three features as specified in the previous section:cortical actin, inner punctuate dots and stress fibers. Since the domain expert only labeled 19 images for us and he selectively highlighted the cellular structures in each image, the results look much better than general recognition algorithm testings. This validation is somewhat limited. However it is the best that we could do so far due to the lack of fully labeled benchmark data set generated from real images. The mis-detection of cortical actin mainly results from the mis-classification of inner punctate dots as cortical actin. It happens when the inner punctuate dots are too close to the border of the cell or when the cells are touching each other and the boundaries are not extracted correctly. The detected rate for both punctuate dots and stress fibers are less than 100%. The reason for the mis-detection of inner punctuate dots has been mentioned for the mis-detection of the cortical actin and the stress fibers get missed due to the limit in the detection resolution.

Besides of validating our algorithm on the manual labeled data set we also perform testing on the synthetic images generated using the method introduced in the previous section. We test our method for detecting cortical actin, inner punctuate dots and stress

Gaussian std.	0.002	0.004	0.008	0.01	0.02
C.A.	96.4%	94.5%	91.2%	89.7%	86.9%
I.P.D	88.4%	86.8%	86.1%	83.8%	78.5%
S.F.	89.5%	87.5%	86.4%	85.0%	82.2%

Table 6.2: This table shows the detected rate for the computer generated images for the three sub-cellular structures.

fibers on 500 more simulated images and use the ratio between the positively detected region and the ground truth as the benchmark metric. We change the standard deviation of background gaussian noise to create 5 different imaging conditions and generate 100 images under each setting. The numerical results, as shown in Table.6.2 are computed using the average of 100 images under each imaging condition.

6.2 Quantification Validation

To validate the quantification results, we need to show that the measurements we compute provide critical information to the biologists. Therefore as the very first step, our quantification should be consistent with the observation results from experiments. In other words, if the images from two treatments show difference in the amount of stress fibers contained per cell based on human observations, similar results should be reported somewhere in our quantification results. If the visual inspection result from multiple images shows a trend in some sub-cellular structures based on human observation, some parts in our quantification results should tell the same thing. Besides validating the well known facts, it should also have the capability to enable new discovery or highlight hidden

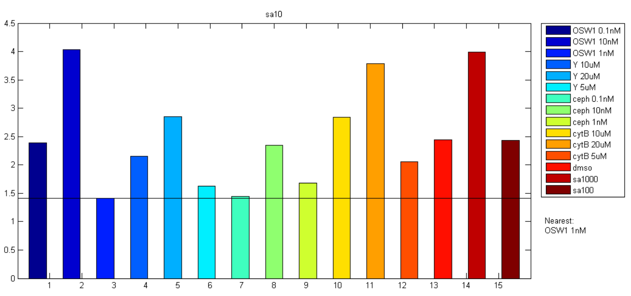

Figure 6.1: This figure shows the results for SA at 10 nm/ml.

changes that are not straight-forward to human observers. These changes can be validated using non-optical methods. Based on the intuitiveness mentioned above we mainly use two ways to validate the quantification results.

- Compare them with the visual observation from the human experts.
- Compare them to the results shown in non-optical experiments.

Since our collaborators are mainly interested in SA, we use the following two facts as our priory validation cases.

1. 23'deoxy-ceph 1 has similar effects on F-actin as SA
2. OSW-1 has similar effects on F-actin as SA

We compute the 10 scalar quantifications for each image and put them into a feature vector. We compute the closeness of 3 SA treatment cases, SA at 10, 100, 1000 nm/ml, to all the rest non-SA treatment cases respectively based on the Mahalanobis distance metrics and generate 3 distance bar graphs (Fig.6.1, Fig.6.2 and Fig.6.3) to visualize which treatment has the highest closeness score to the SA treatment in each concentration. The closest treatment for each case is annotated on the side of the graph. Based on the observation from the bar graph, we could draw the conclusions as follows:

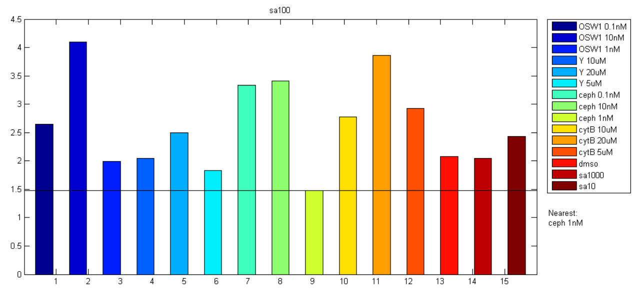

Figure 6.2: This figure shows the results for SA at 100 nm/ml.

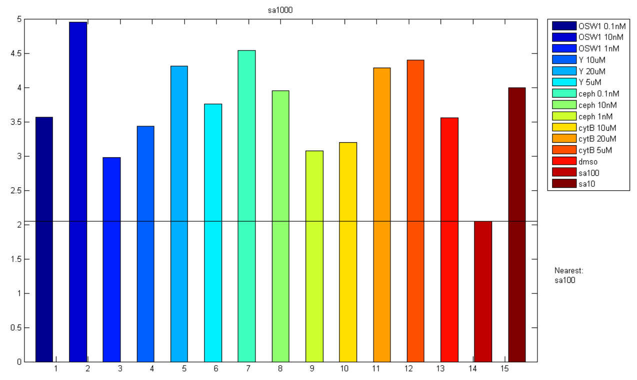

Figure 6.3: This figure shows the results for SA at 1000 nm/ml.

1. SA(10nm/ml) is the most similar to OSW1(1nm/ml)

2. SA(100nm/ml) is the most similar to ceph(1nm/ml)

3. SA(1000nm/ml) is the most similar to SA(100nm/ml)

This result matches quite well with the ready-known biological facts. In addition to the validation based on pair-wise closeness, we also test the capability of our quantification method on highlight the trend discovered in a series of images. Here 3 concentrations for a single drug are put into one group to be shown in one image. The trend is visualized via a box plot and each box displays the mean and standard deviations computed from all 4 images taken under each treatment condition. The boxes are arranged according to concentrations, from low to high, left to right. To enable the comparison between drugs, the DMSO case is put on the left most as a common reference. The left sub-figure in Fig.6.4 shows an example that visualizes the trend for the ratio between the stress fiber area and the cell area for SA. The decrease of the amount of stress fibers could be validated visually from the Fig.6.5.

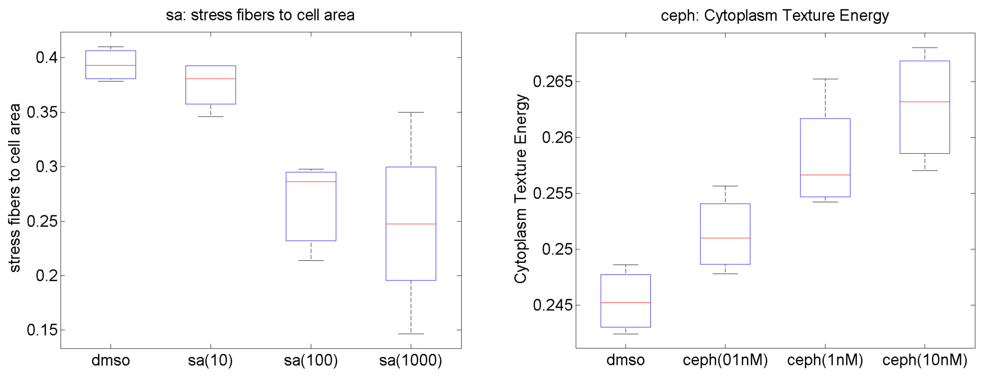

Figure 6.4: The subfigure on the left shows the quantification result of the ratio of the stress fiber area to the cell area of SA; the one on the right shows an example of the box plot for the quantified texture energy of ceph.

The trend quantification can also help the biologists make new discoveries. An example is shown by the right sub-figure in Fig.6.4 which highlights the differences in cytoplasm texture energy when the cells are treated under ceph-1 in 3 concentrations. This result can be validated by non-optical chemical experiments. This proves that quantitative measurements can detect statistically significant differences that are not visually detected with any certainty.

It can be seen from above that our quantification results can provide a good profile of the response of glioma cells to different drugs in varying concentrations and the visualization is straight-forward for the biologists to understand. To generalize the tool to finding the closeness of all the treatment cases, a pairwise distance matrix based on Mahalanobis metrics is generated as shown in Fig.6.6. The square patches are located by the index of the treatment in the X and Y direction. Colors are computed using a ”Heat Map” according to the closeness of different pairs. The blue dots in each row corresponds to the closest case to the drug name on the left of the row. This assembled visualization can be used for the drug discovery on a very large scale.

6.3 Automatic Benchmark Generation

We use the algorithm for detecting different intra-cellular structures developed in house to validate the simulated images. Here the simulated images don't need to look very close to the real images for the validation purposes but they must contain the cellular structures confirming to definition by the biologists and visual insepctions. Fig.6.7 shows an example of a comparison of tri-band segmentation between the real image and the simulated image. Seen from the example, though the two original images look a little bit different, our tri-band segmentation works well on both cases. We do similar tests

	WBA	CYTO	BK	C.A.	S.F.
Average	1.005	0.998	1.00	1.023	1.106
Std	0.02	0.03	0.005	0.03	0.08

Table 6.3: This table shows the preciseness of the tri-band segmentation and subcellular structure extraction algorithm when applied to the artificial images. C.A. denotes cortical actin and S.F. denotes stress fibers.

extended to the sub-cellular structure extraction on 99 more images, and the preciseness, using Eq.6.1 as a metric to show the effectiveness of the simulation algorithm.

$$T^i_{correct} = \frac{\sum_{j=1}^{N} p^i_j}{\sum_{j=1}^{N} \bar{p}^i_j} \tag{6.1}$$

where p^i_j denotes the probability pixel j belongs to class i computed by the algorithm and $\bar{p}^i_j$ denotes the probability pixel j belongs to class i known from the ground truth. The average and deviation of accuracy for the three different bands is shown in Table.6.3. The results from Table.6.3 shows that the simulation framework can generate images that meet the validation need of the automatic analysis algorithms.

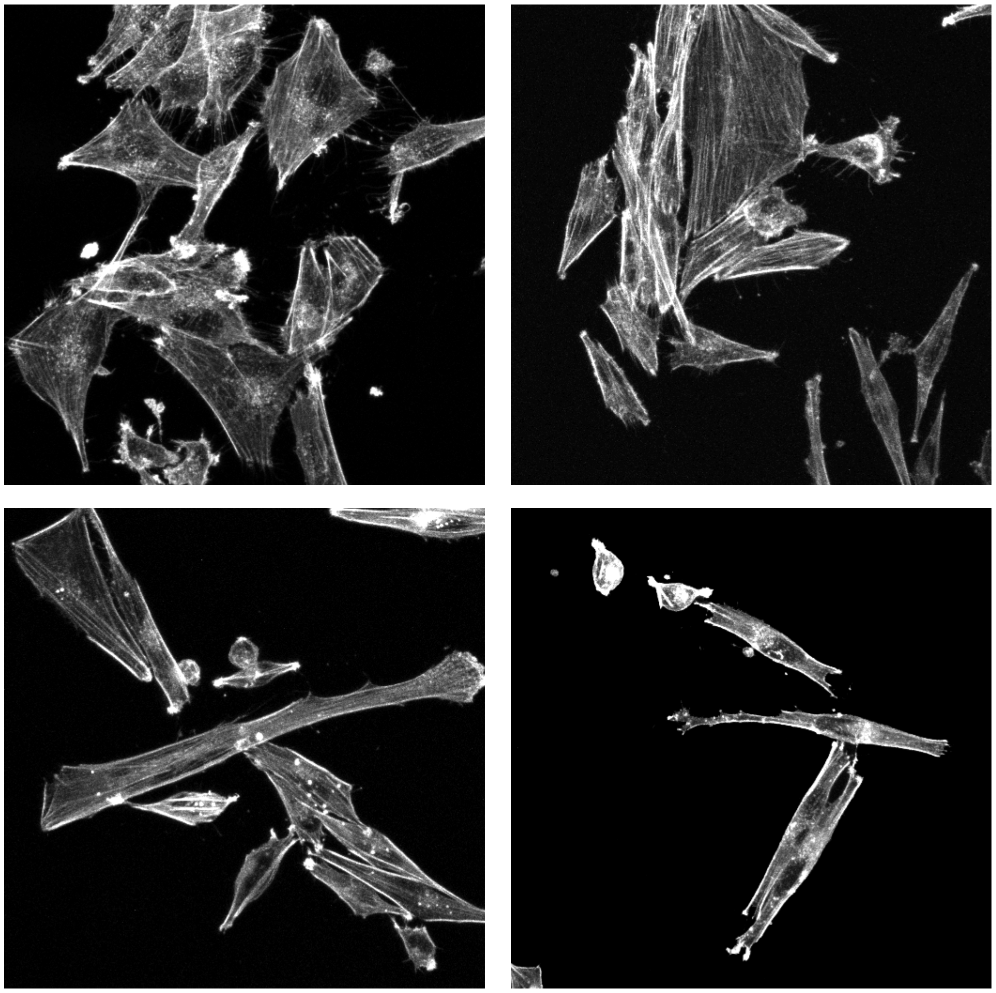

Figure 6.5: This figures shows the images corresponding to the quantification result in the left sub-figure of Fig.6.4. The two sub-figures on the first line correspond to the DMSO, sa(10nM/ML) and the other two on the second line correspond to sa(100nm/ml) and sa(1000nm/ml)

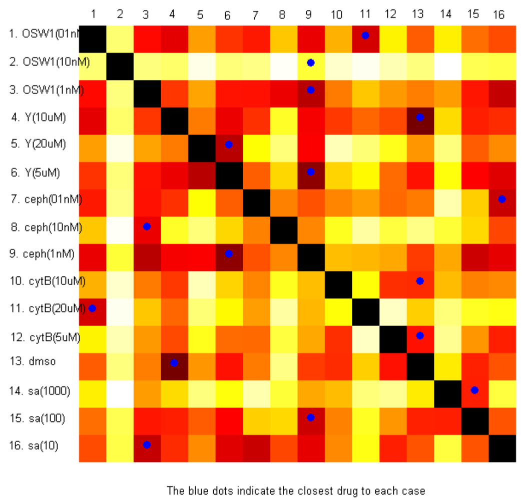

Figure 6.6: This figure shows the pair-wise distance map of all the treatment conditions in our experiment. Darker color means smaller distance, in other word, greater similarity and black means the same.

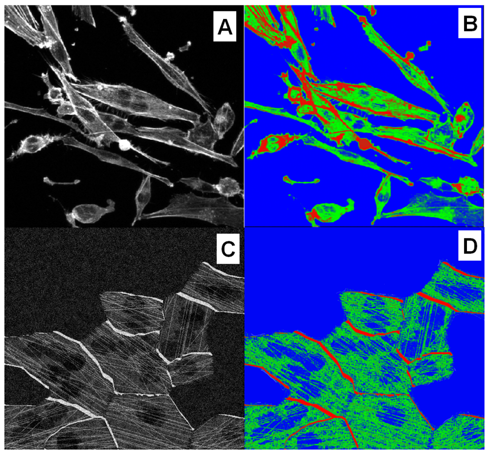

Figure 6.7: A: Image of F-actin labeled cells. B: tri-band segmentation using the method introduced in reference [1]. C: Simulated image of F-actin in cells. D: Tri-band segmentation of the simulated image.

Chapter 7

Conclusion and Future Work

In this thesis we introduce an image analysis framework for quantifying the F-actin organization patterns in confocal microscopy images. The main problem we try to solve is to determine which kind of features to extract and what measures should be taken for quantification so as to make the final results more reliable, descriptive for the observation and discriminant for classification. Our method attempts to minimize the human bias of visual assessment definition as much as possible by choosing the intensity and locations as the main constituents of features. The final quantification results for all the images are grouped based on the basis of individual pharmaceutical. Future work will include running more validations to test the effectiveness of the quantitative measurements we choose and improving the scheme as necessary. Also we seek to find better quantitative descriptions of the features used to quantify F-actin pattern and adapt the system to analyze very large image databases. To speedup the processing of an image we map the most time consuming part in our analysis framework, the tri-band segmentation, to NVIDIA CUDA architecture. Experimental results show that we can obtain 15 to 20 folds of speedup from the CUDA implementation.

We also propose a framework for simulating 2-D F-actin confocal microscopy images. The algorithm is flexible and make use of several parameters that can be controlled by the users. Very little work has been done on the F-actin network simulation in a local region. In addition we are conducting the simulation of images with certain cellular features highlighted, The whole study done for this thesis contributes to the trend in the modern microscopy analysis, which moves from from visual perception to vision based quantification and reasoning which is less biased and easily repeatable.

Bibliography

[1] K. L.; Boyd M. R. Beutler, J. A.; McCall. A novel geranylflavone from. macaranga schweinfurthii. *Nat. Prod. Lett.*, 13:29–32, 1999.

[2] LEPIDI H. ; BENOLIEL A.-M. ; MEGE J.-L. ; BONGRAND P. ; CAPO C. Double localization of f-actin in chemoattractant-stimulated polymorphonuclear leucocytes. *Journal of Cell Science*, 103(1):145–156, 1992.

[3] A. Jones-D. Pinkel J. W. Gray D. Sudar S. J. Lockett C. O. De Solózano, E. G. Rodriguez. Segmentation of confocal microscope images of cell nuclei in thick tissue sections. *Journal of Microscopy*, 193:212–226, January 1999.

[4] Lamprecht MR Clarke C Kang IH Friman O Guertin DA Chang JH Lindquist RA Moffat J Golland P Sabatini DM Carpenter AE, Jones TR. Cellprofiler: image analysis software for identifying and quantifying cell phenotypes. *Genome Biology*, 7(R100), 2006.

[5] S. C. Chen and R. F. Murphy. A graphical model approach to automated classification of protein subcellular location patterns in multi-cell images. *BMC Bioinformatics*, 7, 2006.

[6] K. Chuang, H. Tzeng, S. Chen, J. Wu, and T. Chen. Fuzzy c-means clustering with spatial information for image segmentation. *Computerized Medical Imaging and Graphics*, 30(1):9 – 15, 2006.

[7] O. Cuisenaire. *Distance Transformations: Fast Algorithms and Applications to Medical Image Processing.* PhD thesis, 6 1999.

[8] B. Roysam et al. *Microscopic Image Analysis*, chapter 5, pages 122–131. Artech House, 2008.

[9] Miwaka Ohtani Satoru Nogami1 Fumi Sano-Kumagai Ayaka Saka Masashi Yukawa Taro L. Saito Jun Sese Dai Hirata Shinichi Morishita Genjiro Suzuki, Hiroshi Sawai and Yoshikazu Ohya1. Evaluation of image processing programs for accurate measurement of budding and fission yeast morphology. *Journal Current Genetics*, 49(4):237–247, April 2006.

[10] A. M. Grigoryan, G. Hostetter, O. Kallioniemi, and E. R. Dougherty. Simulation toolbox for 3d-fish spot-counting algorithms. *Real-Time Imaging*, 8(3):203 – 212, 2002.

[11] Yanhua Hu, J. Carmona, and R.F. Murphy. Application of temporal texture features to automated analysis of protein subcellular locations in time series fluorescence microscope images. *Biomedical Imaging: Nano to Macro, 2006. 3rd IEEE International Symposium on*, pages 1028–1031, April 2006.

[12] Kai Huang and R.F. Murphy. Automated classification of subcellular patterns in multicell images without segmentation into single cells. *Biomedical Imaging: Nano to Macro, 2004. IEEE International Symposium on*, pages 1139–1142 Vol. 2, April 2004.

[13] Pieter Sipkema A. B. Johan Groeneveld Ren J. P. Musters Jan R. Leemreis, Amanda M. G. Versteilen. Digital image analysis of cytoskeletal f-actin disintegration in renal microvascular endothelium following ischemia/reperfusion. *Cytometry Part A*, 69A(9):973–978, May 2006.

[14] Lamprecht MR Moffat J Silver S Grenier J Root D Golland P Sabatini DM Jones TR, Carpenter AE. Scoring diverse cellular morphologies in image-based screens with iterative feedback and machine learning. *PNAS*, 106(6), 2009.

[15] Wheeler DB Lindquist RA Papallo A Sabatini DM Golland P Carpenter AE Jones TR, Kang IH. Cellprofiler analyst: data exploration and analysis software for complex image-based screens. *BMC Bioinformatics*, 9(1), 2008.

[16] Carpenter AE Lamprecht MR, Sabatini DM. Cellprofiler: free, versatile software for automated biological image analysis. *Biotechniques*, 42(1):71–75, 2007.

[17] A. Lehmussola, P. Ruusuvuori, J. Selinummi, H. Huttunen, and O. Yli-Harja. Computational framework for simulating fluorescence microscope images with cell populations. *Medical Imaging, IEEE Transactions on*, 26(7):1010–1016, July 2007.

[18] A. Lehmussola, P. Ruusuvuori, J. Selinummi, H. Huttunen, and O. Yli-Harja. On simulating 3d fluorescent microscope images. *Lecture Notes in Computer Science*, August 2007.

[19] Stephen J. Lockett, Damir Sudar, Curtis T. Thompson, Dan Pinkel, and Joe W. Gray. Efficient, interactive, and three-dimensional segmentation of cell nuclei in thick tissue sections. *Cytometry*, 31(4):275–286, January 1998.

[20] R.F. Murphy. Automated interpretation of subcellular location patterns. *Biomedical Imaging: Nano to Macro, 2004. IEEE International Symposium on*, pages 53–56 Vol. 1, April 2004.

[21] N. Otsu. A threshold selection method from graylevel histogram. *IEEE Trans. Syst. Man Cybern*, 9(1), 4 1979.

[22] Jyrki Selinummi, Jertta-Riina Sarkanen, Antti Niemist?and Marja-Leena Linne, Timo Ylikomi, Olli Yli-Harja, and Tuula O. Jalonen. Quantification of vesicles in differentiating human sh-sy5y neuroblastoma cells by automated image analysis. *Neuroscience Letters*, 396(2):102 – 107, 2006.

[23] Mehmet Sezgin and Bulent Sankur. Survey over image thresholding techniques and quantitative performance evaluation. *Journal of Electronic Imaging*, 13(1):146–168, 2004.

[24] Ting Zhao, S. Soto, and R.F. Murphy. Improved comparison of protein subcellular location patterns. *Biomedical Imaging: Nano to Macro, 2006. 3rd IEEE International Symposium on*, pages 562–565, April 2006.