Contour Refinement of Leukocyte Segmentations in Scans of Stained Bone Marrow

Bachelor Thesis

presented by

Yuli Wu

Supervisor:
Philipp Gräbel

Institute of Imaging & Computer Vision
Prof. Dr.-Ing. Dorit Merhof
RWTH Aachen University

Erklärung nach §19 Abs. 7 DPO 2004

Hiermit versichere ich, dass ich die vorgelegte Bachelor Thesis selbständig angefertigt habe. Es sind keine anderen als die angegebenen Quellen und Hilfsmittel benutzt worden. Zitate wurden kenntlich gemacht.

I hereby confirm that I have written this master thesis independently using no sources or aids other than those indicated. I have appropriately declared all citations.

Aachen, 26.10.2018

Contents

List of Figures

List of Tables

List of Listings

Nomenclature

Maths

x	scalar value
$\boldsymbol{x}$	vector
$\|x\|$	absolute value of x
$\|\|\boldsymbol{x}\|\|$	$\mathcal{L}^2$-norm of $\boldsymbol{x}$
$\boldsymbol{A}$	matrix
$\boldsymbol{A}^T$	transposed of matrix $\boldsymbol{A}$
$\inf\{s\}$	infimum of set s

Images

$\boldsymbol{I}$	image
$I(\boldsymbol{x})$	intensity of pixel $\boldsymbol{x}$
$\boldsymbol{x} = (x, y)$	coordinates of a pixel within $\boldsymbol{I}$
$\mathcal{C}$	curve, a set of pixels
$\mathcal{C}(\boldsymbol{x})$	point of curve
w, h	width, height of an image
n_I	number of pixels of an image (or within a shape)

1 Introduction

Classification and counting of different types of white blood cells (leukocytes) in bone marrow play a vital role in the field of hematology. Some diseases, such as leukemia, result in abnormal distribution of leukocytes with different maturation degrees. Automated classification is thus advantageous to efficiently help medical diagnosis.

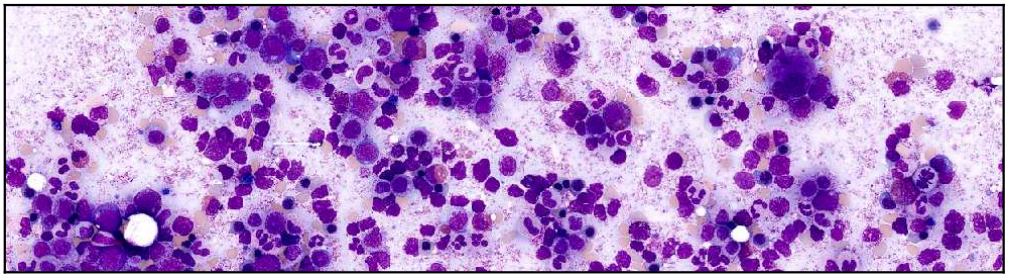


Figure 1.1: Scan of Stained Bone Marrow

Plenty of works have been done to count and to classify leukocytes in blood [1–3] and bone marrow [4]. Most of those use deep learning as a major tool. Prior to the step of classification, the cells should be properly segmented to reduce the difficulty. Suppose that an initial circle is given during the localization of cells in advance, which, however, may be not precise. The task of this thesis is to refine the contours on the basis of these circles to achieve more accurate segmentations of leukocytes using classical methodologies.

Following this introduction, the state of the art is presented in Chapter 2, in which a series of smoothing and segmentation algorithms is introduced. Some candidate methodologies are investigated in the beginning of Chapter 3. The feasibility of several algorithms is analyzed. Further, a pipeline for the entire process of contour refinement based on the initial circles is outlined and performed.

The detailed results including a number of segmentation examples and significant statistics are shown in Chapter 4. Moreover, the methodologies used in this thesis are evaluated based on the experimental results. At the end of this thesis, conclusions are drawn. Following this, improvements and outlook are presented.

2 State of the Art

Before turning to the concrete task of this thesis, some state of the art techniques related to image smoothing and segmentation are introduced in this chapter.

The widely used region-based segmentation algorithm *Watershed* [5–8] is presented in Section 2.1. Active Contour Models, a series of contour-based methods, can be found in sections 2.2 and 2.3, in which two major algorithms, namely *Snakes* [9] and *Active Contours Without Edges* (ACWE) [10], are introduced.

In Section 2.4, the method *SLIC Superpixel* [11] is briefly introduced. Following this, the technique Region Adjacency Graphs *(RAG)* [12] based on the segmentation of SLIC is introduced. Moreover, the smoothing algorithm L_0 *Gradient Minimization* [13] is presented in Section 2.6.

In Section 2.7, several miscellaneous basic techniques are introduced, such as *Gaussian Blur* and *Median Filter*. In addition, a series of histogram equalization techniques [14, 15] are introduced, including the basic Histogram Equalization, the Adaptive Histogram Equalization *(AHE)* and the Contrast Limitied Adaptive Histogram Equalization *(CLAHE)*.

At the end of this chapter, two similarity coefficients are introduced, namely the *Jaccard* [16] and the *Sørensen–Dice* [17, 18] similarity coefficient.

2.1 Watershed

A grayscale image can be considered as a *topographic relief* [8], in which the intensity of a pixel is interpreted as elevation. With this analogy, the features of an image can be metaphorically regarded as peaks, valleys or plateaus. In the field of image segmentation, the desired contours may not be merely related to global sharp changes, but also to characteristics of zones nearby. *Watershed* is an algorithm, in which such features of the neighborhood are also taken into account.

With *Watershed*, seeds as attitude minima are marked in advance. Then "water" is flooding from those seeds gradually, such that catchment basins are developing. When different catchment basins begin to merge, watershed lines appear between these.

The selection of seeds and the sequence of flooding are two important parts during the process. Automatically marked seeds may lack robustness, especially if the image contains noise. Thus, the preprocessing part is challenging and rather problem dependent.

2.1.1 Definitions

Before details of the algorithm are introduced, some definitions relevant to the analogy between image and topographic relief are shown below.

Connectivity Neighborhood of a given pixel (4-connectivity means the pixels to its adjacent above, below, left and right are defined as its connected pixels; 6-connectivity means its adjacent pixels at hexagonal corners; 8-connectivity means 8 pixels in 3×3 of its adjacency, etc.)

Path $\mathcal{P}$ from pixel $\boldsymbol{x}_0$ to $\boldsymbol{x}_n$ of image $\boldsymbol{I}$ is a list of connected pixels: $(\boldsymbol{x}_0, \boldsymbol{x}_1, ..., \boldsymbol{x}_n)$

Length of path $\mathcal{P}(\boldsymbol{x}_0, \boldsymbol{x}_n)$ is denoted as $l(\boldsymbol{x}_0, \boldsymbol{x}_n) = \sum_{i=0}^{n-1} |I(\boldsymbol{x}_{i+1}) - I(\boldsymbol{x}_i)|$

Connected Component $\boldsymbol{\Omega}$ is a set of connected pixels

Geodesic Distance $d_{\boldsymbol{I}}(\boldsymbol{x}_i, \boldsymbol{x}_j) = \inf\{l(\boldsymbol{x}_i, \boldsymbol{x}_j)\}$

Geodesic Influence Zone is a set of pixels in image, from which the geodesic distance to every pixel of some $\boldsymbol{\Omega}_i$ is less than to other connected components: $iz_{\boldsymbol{I}}(\boldsymbol{\Omega}_i) = \{\boldsymbol{x} \in \boldsymbol{I}, \forall j \in [1, n]/\{i\}, \forall \boldsymbol{x}_i \in \boldsymbol{\Omega}_i, \boldsymbol{x}_j \in \boldsymbol{\Omega}_j, d_{\boldsymbol{I}}(\boldsymbol{x}, \boldsymbol{x}_i) < d_{\boldsymbol{I}}(\boldsymbol{x}, \boldsymbol{x}_j)\}$

2.1.2 Immersion Algorithm

The algorithm used in this thesis is the so-called *Immersion Algorithm* [6], which simulates a surface progressively sinking down into a lake. At each predefined minimum of the surface, a well is drilled in advance, such that there already exists some water. With the surface being drowned, water from different wells may flow together to build watershed lines.

With the definitions introduced in Section 2.1.1, this algorithm is formally presented in two steps as follows.

Step 1: Sorting

Every pixel of the image is increasingly sorted by its intensity, and pushed into a FIFO queue (the lowest intensity comes first). Pixels with identical intensities are sorted randomly. Other more efficient sorting techniques can be found in [5, 6, 19].

Step 2: Flooding

Given some minima which are labeled with different positive integers (1,2,3,...) at the beginning. Pixels which have not been visited yet are labeled as 0. Pixels which are regarded as watershed lines are labeled as -1. The flooding step starts with the pixels from the FIFO queue mentioned before. Suppose one pixel is being processed, it is applied with following algorithm.

If this pixel has already been visited, label every pixel in its Geodesic Influence Zone $iz_{\boldsymbol{I}}(\boldsymbol{\Omega}_i)$, where $\boldsymbol{I}$ is the entire image, $\boldsymbol{\Omega}_i$ is the region with the same label as this pixel.

If this pixel has not been visited yet, that means its label is non-positive, then its neighbors as per definition of connectivity are investigated. If all neighbors are not labeled, that means this pixel is a new minimum, which belongs to no discovered region. This pixel gets a new label. If the neighbors are differently labeled, that means this pixel lies on the edge of different regions, thus it is labeled as watershed line. If each neighbor has the identical label, then this pixel gets the same one.

Figure 2.1 shows this in a 1D example visually. A pseudo code of this algorithm can be found in Appendix A.1.

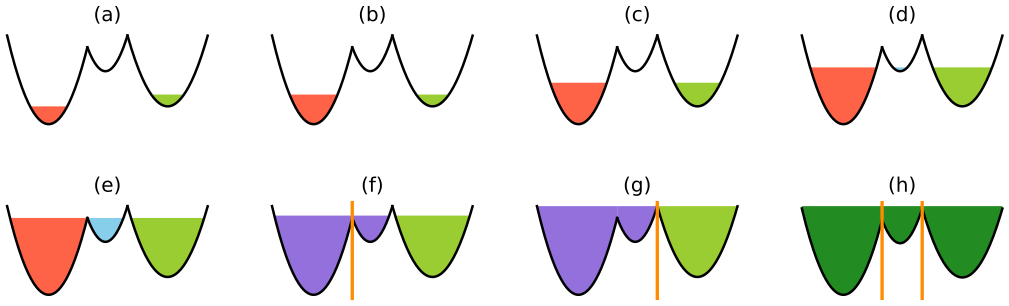


Figure 2.1: Watershed Example in 1D.
There are three catchment basins with two discovered wells (minima). These two minima are marked as orange and green in Figure (a). With increasing water level, there exists a new minimum, which belongs to no discovered catchment basins. Figure (d, e) illustrates this in the middle catchment basin in blue. In Figure (f), water from the left two catchment basins begins to merge, thus there exists a watershed line. The same phenomenon happens in Figure (g) between the right two catchment basins. Therefore, two watershed lines are found, as Figure (h) shows.

2.2 Active Contour Models

Active Contour Model (ACM), also called *Snakes*, is one of the most widely used algorithms in image segmentation. The term *Snake* is used in this thesis to refer to an evolving curve. The basic idea of *Snakes* [9] is to build several mappings from original image to energies, which refer to constraints upon the curve and features of the image, and to minimize the sum of total energy along the curve in the image. The curve with minimal energy is then the desired solution.

The specific algorithm introduced below is based on the main idea in the original paper [9] and the source code in [20], where the energy function is designed as

$$E^*_{\text{snake}} = \sum_{\boldsymbol{x} \in \mathcal{C}} E_{\text{snake}}(\boldsymbol{x}) \tag{2.1}$$

$$= \sum_{\boldsymbol{x} \in \mathcal{C}} E_{\text{int}}(\boldsymbol{x}) + E_{\text{image}}(\boldsymbol{x}) + E_{\text{con}}(\boldsymbol{x}), \tag{2.2}$$

where $\mathcal{C}$ is the curve to be optimized, and E_{int}, E_{image} and E_{con} represent the internal, image and constraint energy respectively.

2.2.1 Internal Energy

The internal energy interprets the continuity E_{cont} and the smoothness E_{curv} of a *Snake*. These can be represented by the first and the second derivatives, i.e. gradient and curvature. In the discrete case, which is common in image processing, the *finite difference* can be used to approximately calculate the derivatives. Thus, the internal energy function can be written as:

$$E_{\text{int}} = E_{\text{cont}} + E_{\text{conv}} \tag{2.3}$$

$$= \frac{1}{2}\alpha_i(\boldsymbol{x})\|\boldsymbol{x}_i - \boldsymbol{x}_{i-1}\|^2 + \frac{1}{2}\beta_i(\boldsymbol{x})\|\boldsymbol{x}_{i-1} - 2\boldsymbol{x}_i + \boldsymbol{x}_{i+1}\|^2, \tag{2.4}$$

where $\boldsymbol{x}_{0,1,\ldots,n}$ are $n+1$ points on the curve $\mathcal{C}$. Two parameters, $\alpha_i(\boldsymbol{x})$ and $\beta_i(\boldsymbol{x})$, can be adapted to weigh the influences of two constraints. To simplify the problem, these two parameters are often chosen as constant.

The continuity term serves to minimize the distance of adjacent points. It has the effect that the *Snake* is forced to shrink. A larger α makes the *Snake* contract faster. The smoothness term avoids oscillations and corners, as the curvature of a corner is theoretically infinite. In other words, the parameter β can be set to 0 if the desired solution is allowed to contain a corner.

2.2.2 Image Energy

The desired curve is closely related to the features of the image itself, such as intensity and boundary. The image energy can be defined as a weighted combination of two energy functionals to attract *Snakes* to salient features in images:

$$E_{\text{image}} = w_{\text{line}}E_{\text{line}} + w_{\text{edge}}E_{\text{edge}} \tag{2.5}$$

Line Functional

The simplest way to design a line functional is the image intensity: $E_{\text{line}} = \boldsymbol{I}(\boldsymbol{x})$. Depending on the weight w_{line}, the *Snakes* are attracted to lighter or darker regions. Smoothing or noise reduction filters can be applied to the image, such that the line functional is denoted as $E_{\text{line}} = \text{filter}(\boldsymbol{I}(\boldsymbol{x}))$.

Edge Functional

Edges can be found using the gradient of an image. A simple edge functional can be set as $E_{\text{edge}} = -|\nabla \boldsymbol{I}(\boldsymbol{x})|^2$ with a negative sign, since a minimized energy is the goal.

Plenty of methods can detect edges. Most of them use a convolutional gradient filter, such as Sobel, Scharr, Laplacian etc. Another filter is specifically introduced in the original paper [9]: the Marr–Hildreth edge detection method [21], which operates by convolving the image with the Laplacian of Gaussian function (LoG). After smoothing with Gaussian filter, a larger edge region (or "energy well", as in [9]) is detected and attract the *Snakes* more strongly. In this thesis, the Sobel Operator is used.

2.2.3 Constraint Energy

In some cases, specific areas with fewer salient features for the *Snakes* are still meant to be part of the final curve. An extra term such as $-k(\mathbf{x_1} - \mathbf{x_2})^2, k > 0$ can be added to the function to compose a "spring" between $\mathbf{x_1}$ and $\mathbf{x_2}$. Further, a "volcano" can be created, for example, by putting a point into denominator, such that the *Snake* is repelled by it.

The numerical approximation of this algorithm is discussed in the end of the next section, together with *ACWE*.

2.3 Active Contours Without Edges

Another type of energy functional is introduced by Chan and Vese in [10] to segment images with different intensity features but without clear edges. Compared to classical ACMs, this method has the same philosophy but with distinct focal points.

2.3.1 Energy Functional

Assuming that an image is divided into two areas by a curve $\mathcal{C}$. The energy functional of ACWE is defined as

$$\begin{aligned} E_{ACWE}(\overline{I_1}, \overline{I_2}, \mathcal{C}) = \mu \cdot l(\mathcal{C}) \\ + \lambda_1 \int\limits_{inside(\mathcal{C})} |I(\boldsymbol{x_0}) - \overline{I_1}|^2 d\boldsymbol{x} \\ + \lambda_2 \int\limits_{outside(\mathcal{C})} |I(\boldsymbol{x_0}) - \overline{I_2}|^2 d\boldsymbol{x} \end{aligned} \tag{2.6}$$

where $\overline{I_1}, \overline{I_2}$ are the average intensities inside and outside curve $\mathcal{C}$ respectively and $l(\mathcal{C})$ represents the length of the curve $\mathcal{C}$. The weights μ, λ_1 and λ_2 are, as in *Snakes*, problem dependent and user-designed: a larger μ makes the curve more circular; if

$\lambda_1 < \lambda_2$, the region inside of the curve $\mathcal{C}$ contains a larger range of intensities than outside, and vice versa.

This algorithm is not based on edge features, but rather on the regional intensity. Thus, an image without salient edges can also be well segmented.

2.3.2 Numerical Approximation

In [9, 10], numerical approximation methods are introduced. The main task is to solve a minimization problem.

Finite differences are used to approximate the derivatives and *Gradient Descent* is used to find the local minimum. *Morphological Snakes*, introduced in [22], is another variant of Active Contour Models, where morphological operators are performed to solve partial differential equations.

2.4 SLIC Superpixels

A cluster of similar pixels next to each other can be grouped to a larger pixel, a so-called superpixel. *SLIC (simple linear iterative clustering)* [11] is one of the simplest and most efficient algorithms and is briefly introduced below.

Assume, k superpixels are wanted. First of all, k cluster centers are evenly initialized in a CIELAB color space image, such that the gap between every two centers is $l = \sqrt{n_I/k}$, where n_I is the number of pixels in the image $\boldsymbol{I}$. Then the distance D between cluster center and each pixel in a $2l \times 2l$ square around it can be calculated as:

$$d_c = \sqrt{(l_c - l_i)^2 + (a_c - a_i)^2 + (b_c - b_i)^2}, \tag{2.7}$$

$$d_s = \sqrt{(x_c - x_i)^2 + (y_c - y_i)^2}, \tag{2.8}$$

$$D = \sqrt{{d_c}^2 + \left(\frac{d_s}{l}\right)^2 m^2}, \tag{2.9}$$

where center $I_c = [l_c, a_c, b_c, x_c, y_c]^T$ and pixel $I_i = [l_i, a_i, b_i, x_i, y_i]^T$. The constant m represents the compactness of superpixels. The pixels are assigned to the center, to which it has the shortest D. Next, new cluster centers are calculated. These steps are repeated until the residual error for updates of cluster centers less than a threshold.

Using this method, a complex noisy image can be significantly simplified. Therefore, it is increasingly used as a preprocessing tool to reduce redundancy.

2.5 Region Adjacency Graphs

The region boundary based Region Adjacency Graphs (RAG) can merge the superpixels to larger ones. The center of each superpixel is the node. Between every two adjacent superpixels the nodes are connected and the value of the connection is the

average value of the gradient along the edge between these two superpixels. After these steps, a RAG is created.

The adjacent superpixels, whose connection has the lowest value, are merged, and this processing is executed iteratively. To terminate merging well-timed, a predefined threshold is required. If the connection value is smaller than this threshold, these two regions are merged.

2.6 L_0 Gradient Minimization

An efficient smoothing operation should synthesize similarity and maintain salient features. This goal can be realized through L_0 Gradient Minimization [13]. It reduces the number of non-zero gradient pixels with the constraint that the smoothed image should not significantly differ from the original one.

Given an input image $\boldsymbol{I}$ and the result image $\boldsymbol{S}$, a term to show the differences between them can be written as $\sum(\boldsymbol{S}-\boldsymbol{I})^2$. Besides, a counting function $C(\boldsymbol{S})$ counts the non-zero gradients. To implement the idea expressed before, the optimization problem is defined as

$$\min\left\{\sum(\boldsymbol{S}-\boldsymbol{I})^2+\lambda\cdot C(\boldsymbol{S})\right\}, \tag{2.10}$$

where λ gives the weight of numbers of non-zero gradients, which indicates the smoothness.

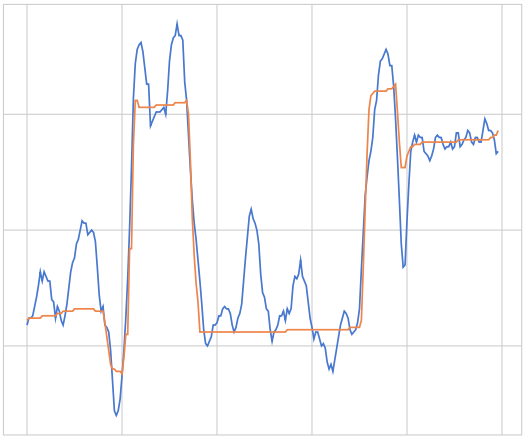


Figure 2.2: L_0 Gradient Minimization in 1D.
The blue curve is the original plot, the orange one is smoothed by L_0 Gradient Minimization. It can be seen that the prominent changes are kept, while slight ones are ignored.

2.7 Miscellaneous Techniques

2.7.1 Gaussian Blur

Gaussian convolutional filter is a commonly used operator for blurring. The following two matrices are examples of the so-called *Discrete Gaussian Kernel* in 3 and 5 dimensions.

$$\frac{1}{16}\begin{pmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{pmatrix} \qquad \frac{1}{273}\begin{pmatrix} 1 & 4 & 7 & 4 & 1 \\ 4 & 16 & 26 & 16 & 4 \\ 7 & 26 & 41 & 26 & 7 \\ 4 & 16 & 26 & 16 & 4 \\ 1 & 4 & 7 & 4 & 1 \end{pmatrix}$$

2.7.2 Median Filter

Median filtering is a powerful operator to remove noise like *salt and pepper*. After sorting the intensities inside a window, the original intensity is replaced by the median value.

2.7.3 Histogram Equalization

The ordinary histogram equalization is based on the Cumulative Distribution Function (CDF). The equalized value of an intensity is defined as below.

$$h(I) = \text{round}\Big(\frac{\text{CDF}(I) - \text{CDF}_{min}}{w \times h - \text{CDF}_{min}} \times (L-1)\Big), \tag{2.11}$$

where $\text{CDF}(I)$ is the value of the intensity I in cumulative histogram, CDF_{min} is the minimum value in cumulative histogram, w, h represent the weight and height of the image. L is the intensity level. It is 256 if the data type is unsigned short integer. Rather than globally in ordinary histogram equalization, the *adaptive* version

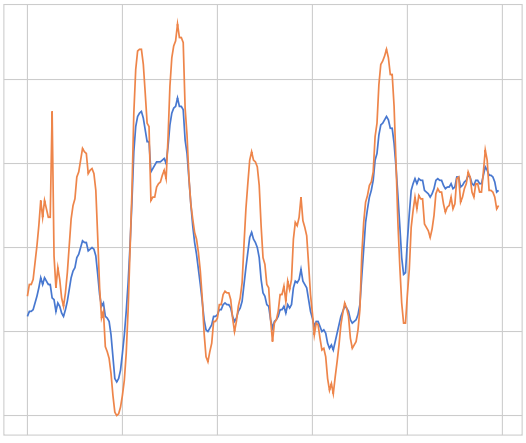


Figure 2.3: CLAHE Example: The blue curve is the original plot, the orange one is applied with CLAHE.

performs this process regionally: the neighborhood region of each pixel is applied with the ordinary histogram equalization. It is advantageous to enhance the local contrast, in this case the contrast between cytoplasm and background. However, this method might amplify noise in the parts which are quasi-constant. To solve this problem, the contrast limited adaptive histogram equalization is introduced. Through adding an amplification threshold, the enhancement of noise is reduced.

2.8 Similarity Comparison

2.8.1 Jaccard Similarity Coefficient

The Jaccard Similarity Coefficient [16], also known as Intersection over Union, is defined as:

$$SC_J = \frac{|\boldsymbol{A} \bigcap \boldsymbol{B}|}{|\boldsymbol{A} \bigcup \boldsymbol{B}|}. \tag{2.12}$$

2.8.2 Sørensen–Dice Similarity Coefficient

Likewise, the Sørensen–Dice Similarity Coefficient [17] is defined as:

$$SC_{SD} = \frac{2(|\boldsymbol{A} \bigcap \boldsymbol{B}|)}{|\boldsymbol{A}| + |\boldsymbol{B}|}. \tag{2.13}$$

2.8.3 Equivalence Relation

For the same $\boldsymbol{A}$ and $\boldsymbol{B}$, the two coefficients are connected through this equation:

$$SC_{SD} = \frac{2 \cdot SC_J}{1 + SC_J}. \tag{2.14}$$

In this thesis, the Sørensen–Dice Similarity Coefficient is used.

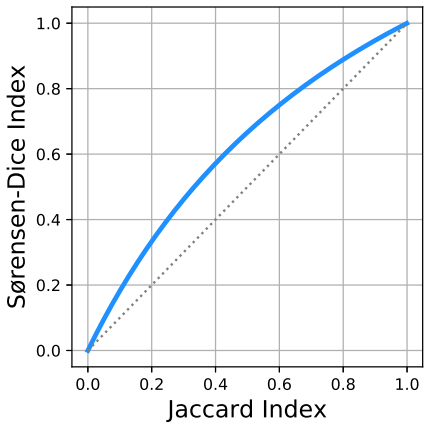


Figure 2.4: Equivalence Relation of Sørensen–Dice and Jaccard

3 Approaches

In Chapter 2, the state-of-the-art methods and algorithms are introduced. This chapter seeks to address how these methods are applied.

First of all, the test conditions and database are presented. In the second section of this chapter, the ground truth contours are investigated to extract the salient features of leukocytes. After that, a series of candidate methodologies are chosen to test the feasibility, among which four sets of methods are proven to be adequate for the task. Their results are shown in Section 3.3. The application of these methodologies and their evaluations are described in sections 3.4 and 3.5 respectively.

3.1 Test Conditions

The aim of this thesis is to refine the segmentation contours of leukocytes based on initial circles. The scans are stained bone marrow, on which the radius and center of each leukocyte are given. With this in mind, single-cell images are cropped from large-scale scans in a first step. Each single-cell image has three channels in the RGB color model. Ground truth contours are provided to evaluate the results of refinement. In this thesis, a pre-classification is not considered.

In Figure 3.1, some examples with ground truth contours are shown.

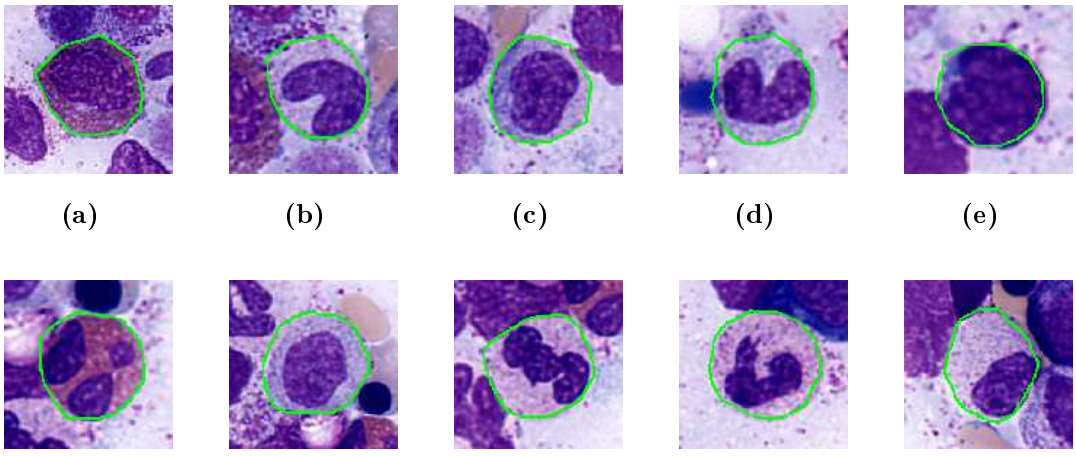


Figure 3.1: Ground Truth Examples: green contours are ground truth.

3.2 Preliminary Considerations

3.2.1 Noise Reduction

Usually, the first step in image processing is to reduce noise. The noise in this case consists mainly of small blobs in the background. Gaussian Blur or Median Filter can be used to reduce this type of noise. The details are discussed in the following.

3.2.2 Color Space

Attention must be paid when using the package OpenCV [23], as the default color model is BGR, while some other packages use RGB color model as default. After applying Pappenheim staining, the main color of scans is purple, which is synthesized from red and blue. In the following experiments, if a single-channel image is needed, the original image is transformed to R channel, B channel or gray channel, which is combined by three channels:

$$\mathrm{GRAY} = 0.299 \cdot \mathrm{R} + 0.587 \cdot \mathrm{G} + 0.114 \cdot \mathrm{B}$$

3.2.3 Ground Truth Analysis

In addition to the irregular shapes of blood cells and the complexity of the background, the leukocytes are of various types, which enhances the difficulty of segmentation. In order to understand more clearly how the cells should be segmented, the ground truth contours are analyzed.

Since the majority of cells is circular, it simplifies the process to transform the original image from cartesian coordinates to polar coordinates. The details of this are introduced in Section 3.3.

The gradient represents the rate of change. Thus, the edges of different regions generally contain large gradients, which makes the gradient one of the most significant features in the field of segmentation.

Two saliently large gradients are located

- between the nucleus and the cytoplasm, and
- on the cell membrane.

The contour of a cell is positioned on the outer salient gradient. Therefore, the main task in segmenting a cell is to find the gradient on the cell membrane.

If a cell is ideally centrosymmetric, the original image, represented as 2D matrix, can be simplified to a 1D vector. This vector represents the values alongside a radius. Due to the symmetry just one vector alone can describe the entire cell.

With this simplification in mind, the contour can be clearly detected. The problem is to find out whether this simplification is fulfilled in reality and how images can be automatically simplified in general.

Another attempt is the Active Contour Model, such as *Snakes* and ACWE, which are extremely flexible and controllable.

The third idea is the Watershed Algorithm, which simulates an image as a topographic relief.

Based on the algorithms in Chapter 2, the details of processing are introduced below.

3.2.4 Standard Operating Procedure

A standard operation procedure can be created to pursue an automatic processing for the task. This means, despite of the differences of input cells, such as the image size, the range of intensity, the shape of cells etc., a chain of methodologies must match all of them and the output contour should achieve better results compared to initial ones.

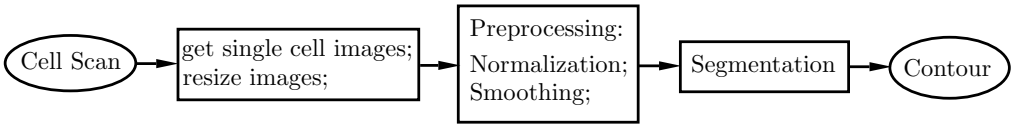


Figure 3.2: Standard Operating Procedure

Two major steps are preprocessing and segmentation. The preprocessing step includes

- noise reduction, such as blur filtering and median filtering,
- smoothing, such as Superpixel, Region Adjacency Graphs (RAG) and L_0 Gradient Minimization,
- normalization, such as image resizing and equalization.

The goal of the segmentation step is to perform the segmenting methodologies upon the preprocessed images and to output the final uniform contours, which include

- analyses of gradients in polar coordinates,
- *Watershed*, such as *Immersion Algorithm*,
- active contour models, such as *Snakes* and ACWE.

In the following sections the details of these two steps are discussed.

3.3 Methodology

In the last section, the preliminary considerations have been taken. Concrete methods are described on the basis of above attempts in this section.

Preprocessing methodologies are introduced in Section 3.3.1. Following this, the primary segmentation algorithms are presented.

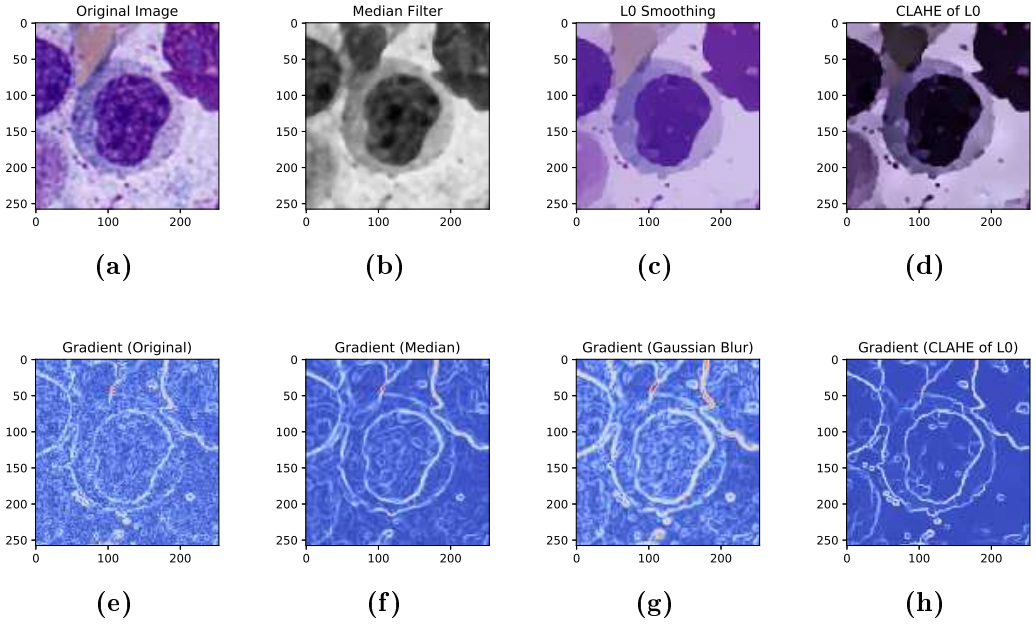


Figure 3.3: Smoothing Examples

3.3.1 Preprocessing

In order to reduce noise and extract features, the original single-cell image is preprocessed in the initial stage. As discussed in the previous section, the gradient is the main feature during the process. Therefore, image smoothing is chosen as a noise reduction method to eliminate redundant gradients. Some examples are shown in Figure 3.3. It is obvious that preprocessing extracts the prominent gradients and reduces noise.

In the following, some smoothing algorithms are introduced and compared.

Smoothing Filters

As mentioned in Section 2.7, two basic filters, Gaussian Blur and Median Filter, are used. As structuring element of Median Filter a diamond is used. In this thesis, these two methods are applied to reduce blob-like noise in the background and smooth the superpixels inside of cells. Figure 3.3 shows some examples of them. It can be seen that the gradient images after applying with smoothing filters contain less noise. However, the results are not as good as other smoothing techniques, such as L_0 Gradient Minimization.

L_0 Gradient Minimization

L_0 Gradient Minimization is a method to make images smoother while keeping the salient features of the original image. By reducing the amount of non-zero gradients, the indistinct ones within nuclei and cytoplasms are wiped away, while the salient gradients maintain.

It is known from Equation 2.10 that the weight of the counting function λ controls the smoothness. Figures 3.3 (c,d,h) show the results after applying with L_0 Gradient Minimization using default setting $\lambda = 0.02$. As the default setting of this algorithm produces desired results, further procedures are unnecessary.

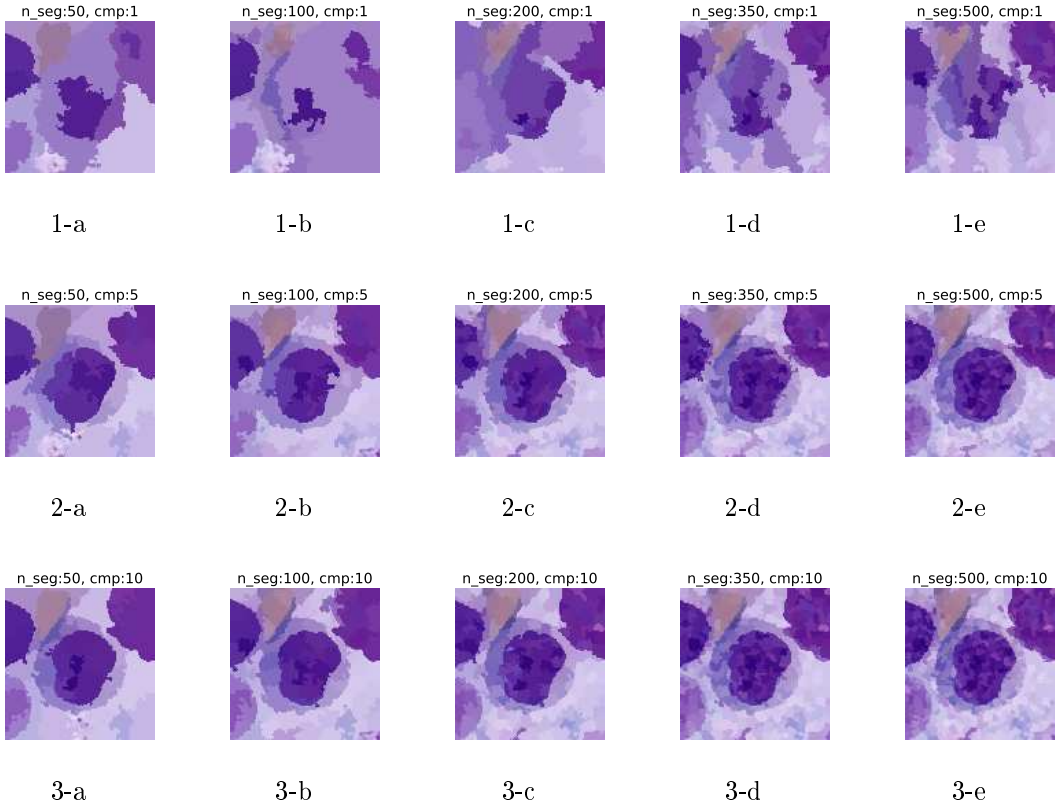


Figure 3.4: SLIC with different parameters - Cell 1

SLIC Superpixels

As discussed in Section 2.4, SLIC can be used to reduce noise by combining adjacent and similar pixels into a larger superpixel. The intensity of each superpixel is replaced by the mean value of all original pixels. After these steps, a cell is divided into several regions. Within each region, the intensity is constant. Furthermore, the gradient is non-zero only on the edges of each superpixel.

Two main parameters for this process are the maximal number of superpixels and compactness of superpixels. An example with different parameters is shown in Figure 3.4.

It is obvious that the compactness should not be less than 10. Otherwise, even a large number of segments can not guarantee the correctness of segmented image. For instance, if compactness is set to 10 and the number of segments is small, the shape of cell 1 is well segmented and the image is smoothed (see Figure 3.4, columns a-b).

Figure 3.5: SLIC with different parameters - Cell 2

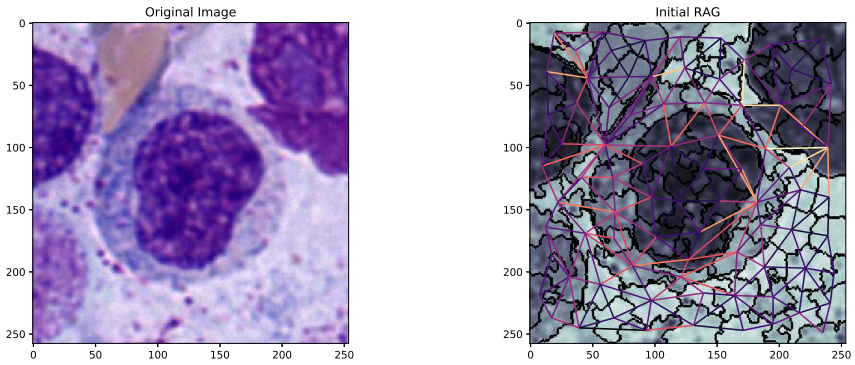
Another example in Figure 3.5 shows, if the cytoplasm is similar to the outside region, an inadequate number of superpixels can cause the "birth defect": the low gradient between cytoplasm and outside region is swiped away and these two regions are merged, so that a proper segmentation is not possible anymore.

In this thesis, the number of segments is set to 100 and compactness is set to 10. A relatively large number of segments prevents the pixels around indistinct cell membranes from merging. The compactness balances the regularity of shapes and preservation of smooth arcs on the cell membrane.

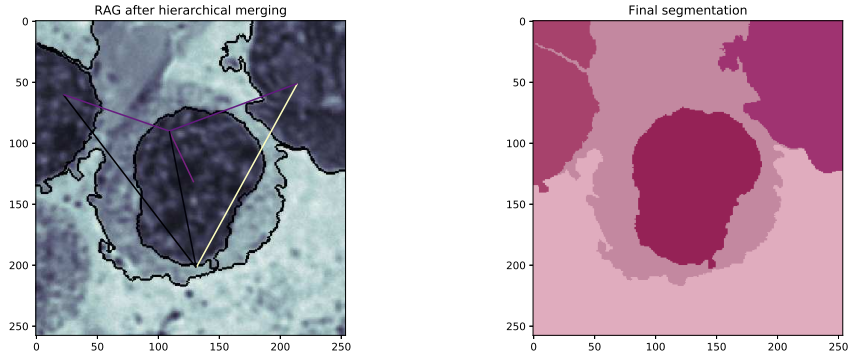
Region Adjacency Graphs

As introduced in Section 2.5, RAG can be used to merge superpixels to a larger one, such that the images can be further smoothed. The algorithm stated in Section 2.5 is presented with an example cell in Figure 3.6.

During the step of merging, the predefined parameter threshold represents, less than which connection value two superpixels are merged. Figure 3.7 shows the merged RAGs with different thresholds. Some conclusions can be drawn from these examples. First, the unique constant threshold is difficult to define for all cells. Therefore, it could ruin the automaticity. Moreover, it tends to under-segmentation, which results in irreparable defects. To overcome this disadvantage,



(a) Original **(b)** RAG



(c) Merged RAG **(d)** After smoothing

Figure 3.6: RAG Examples.
Figure 3.6 (b) is performed SLIC Superpixel with n_segments=100, compactness=10; The merging in Figure 3.6 (c) uses thresh=0.07.

a small threshold is necessary. However, it makes this step less meaningful, as only very similar superpixels are merged. With this in mind, the hierarchical merging will not used. Instead of this, Gaussian Blur and Median Filter are applied after SLIC Superpixel to smooth the image.

Normalization

As stated previously, an automatic algorithm is the goal of this task. To achieve this, normalization is a necessary step. Three features of images, that may vary, can influence the uniformity of images:

- image size;
- image contrast;
- range of intensity.

In order to provide the segmentation mechanism with uniform images, the initial input images are scaled to the same size on the basis of the radius of cells. In

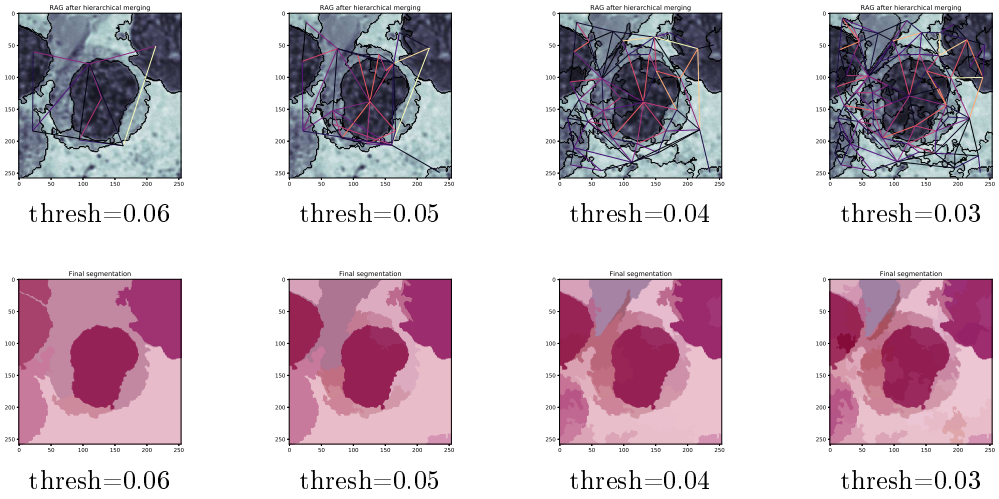


Figure 3.7: RAG Examples with different thresholds. Images above are merged RAGs, below are final images after merging.

this thesis, the radius of cells are given as parameter and the scaled image size is 200×200.

It is known that one of the challenges of this task is to distinguish the nucleus, the cytoplasm and the region outside of the cell. However, the difference of intensity between the cytoplasm and the region outside of the cell may not be prominent enough, so that some algorithms consider these regions as one. This makes the following segmentation unsuccessful. The technique to enhance the contrast of images is Contrast Limited Adaptive Histogram Equalization, as introduced in Section 2.7. In this thesis, images are applied with this technique prior to segmentation.

Some algorithms, like *Snakes*, use energy functionals, which are bound up with the intensities of images. However, the some cells are brighter, while others are darker. In order to pursue the uniformity of images, the range of intensities are rescaled homogeneously to a fixed one. In this thesis, the normalized range is $[0, 255]$. In addition, gradient images are also rescaled to this range to show the results visually.

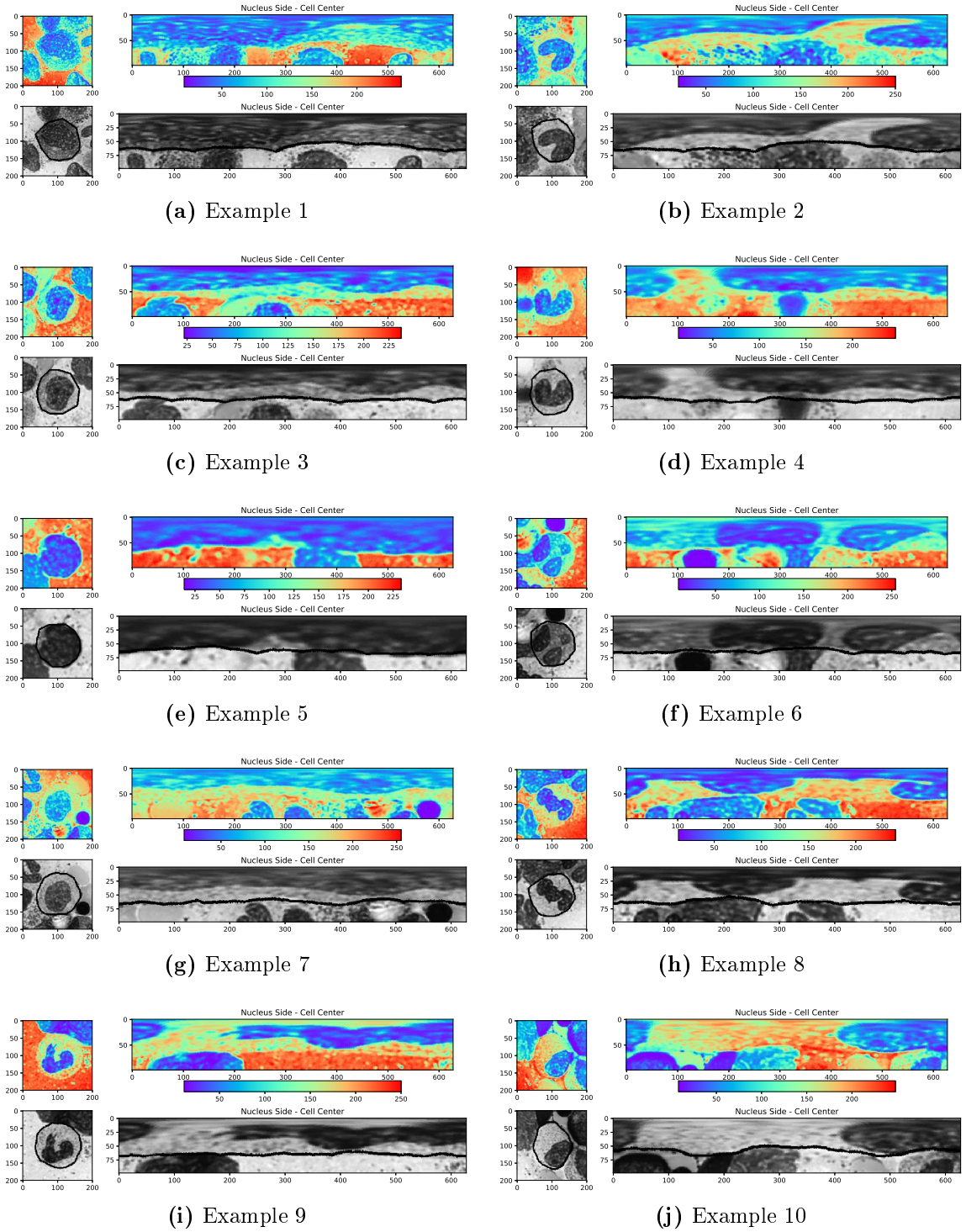
Figure 3.3 shows some examples using various smoothing methods and represents the effect of normalization.

3.3.2 Segmentation

Gradient in Polar Coordinates

The idea is to transform the image from cartesian to polar coordinates and analyze each column of pixels, which represents the radius line. In the ideal case, the desired position within each column should have salient features, such that it can be easily found, as shown in [24].

In reality (see Figure 3.8), however, it is ambiguous to determine which pixel belongs to the contour. In (a-c), the contour lines are relatively easy to find, as



(a) Example 1 **(b)** Example 2

(c) Example 3 **(d)** Example 4

(e) Example 5 **(f)** Example 6

(g) Example 7 **(h)** Example 8

(i) Example 9 **(j)** Example 10

Figure 3.8: Gradient in Polar System Examples - Ground Truth Analysis. Different intensities are illustrated as different colors. For images in polar coordinates, the top line represents the cell center in cartesian coordinates. Ground truth contours are shown in both coordinates. To show the intensity distribution and the location of the contours in polar coordinates, different color maps are used.

the salient gradients of columns are quasi horizontal. The contour lines in (d-g) are not the most salient gradient lines, because nuclei are much darker than cytoplasm regions. It makes the detection of contours difficult. In (g-j), the nuclei are irregular, such that it is hard to find an algorithm to recognize contours.

In summary, the examples show, that the contour is easy to find, if the cytoplasm region is as dark as the nucleus. If the cytoplasm is more similar to the background than to the nucleus, it makes this local method work less well. In the worst case, the nucleus is irregular.

Watershed

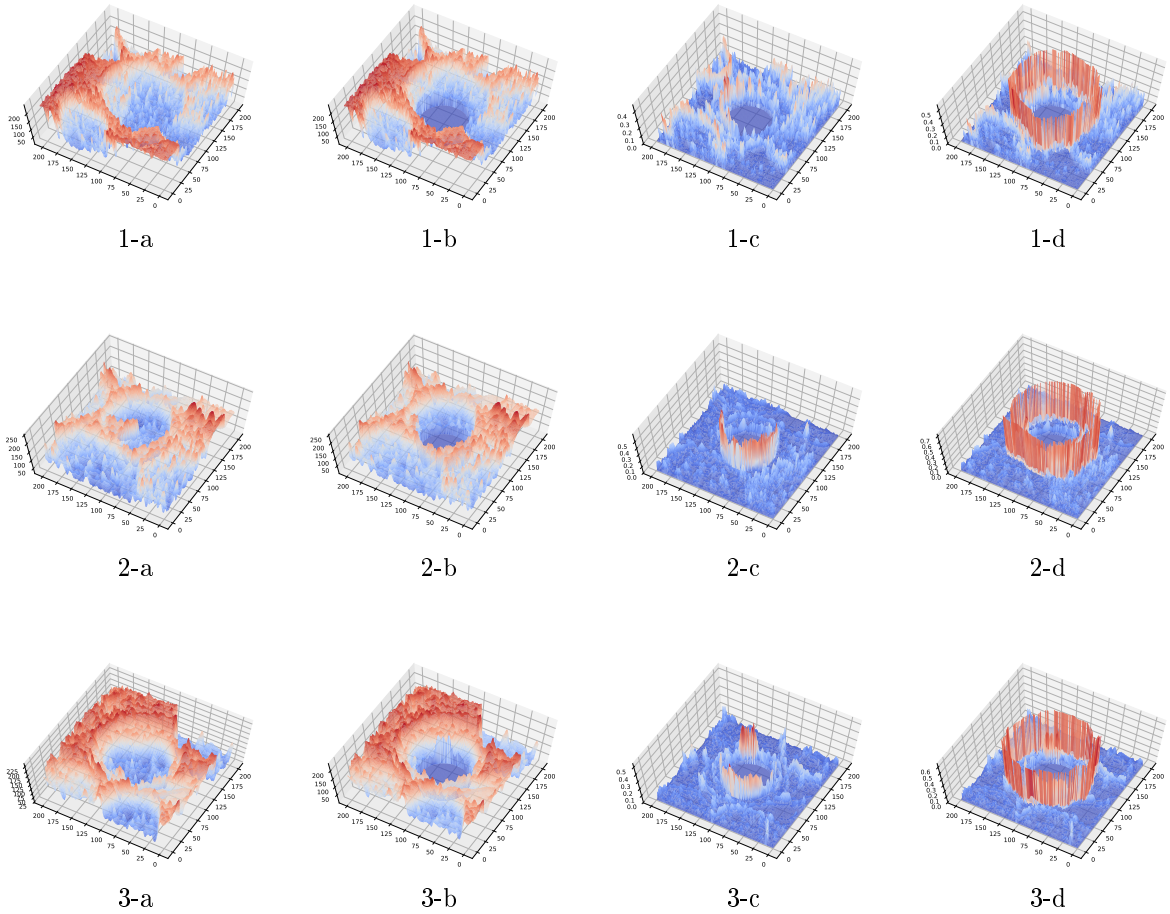


Figure 3.9: Watershed Examples 3D.
(a-d) represent original images, original images with marked minima, gradients of original images with marked minima and gradients of watershed lines results respectively. Z-Axis represents intensities.

The algorithm used in this thesis is non-parametric marker-based Watershed. As the parameters for this method are input image and the marker image, it is necessary to select the markers and masks properly. Figure 3.9 shows examples using

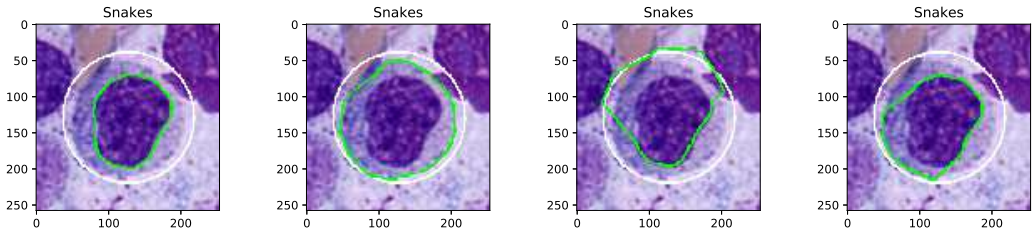
Watershed. Figure 3.9 (b) shows marked images based on the original ones in 3.9 (a). Figure 3.9 (c) shows the gradients of original images, and 3.9 (d) shows the results of *Watershed*. The difficult part of applying *Watershed* is proper preprocessing. The details can be found in previous sections.

In this thesis, 5% of initial radius is added to construct a circular mask. Inside of the cell, a circular marker is defined with a radius of 85% of initial one. If the marker is too small, the watershed line tends to stop on the most salient gradient, which could be the edge between the nucleus and the cytoplasm. If the marker is too big, the process could fail for some irregular cells, as the catchment basins may be overfilled, such that the watershed line disappears. The gradient image of the original single-cell image is chosen as input image. The technique used here is the Sobel Operator.

After the previous steps, *Watershed* can be applied. As output, an image of marking contours as -1 is returned. It is notable that the rectangle boundary of the whole image is also detected as contour. Therefore, this boundary should be ignored to achieve a pure contour for the following similarity evaluation.

Snakes

As stated in Section 2.2, *Snakes* is a segmentation algorithm using the concept of energy minimization. One of the most distinguishing features of *Snakes* is that the final contour is controllable using few parameters. A simple example is shown in Figure 3.10, with different continuity weights α, the initial contour can be set to stop on the edge between nucleus and cytoplasm or on the cell membrane.



(a) α=5.0, β=0.0, w_{line}=1, w_{edge}=1

(b) α=0.3, β=0.0, w_{line}=1, w_{edge}=1

(c) α=0.3, β=0.0, w_{line}=-1, w_{edge}=1

(d) α=2.0, β=0.0, w_{line}=0, w_{edge}=2

Figure 3.10: Snakes Examples with different parameters
The white circles are initial contours, the green are found contours.

Due to the flexibility of this methodology, a set of proper parameters can be obtained by experimental analyses. On the other hand, the complexity of parameter settings brings instability. Figures 3.10 (c,d) show that if the parameters are not properly set, it easily leads to wrong results.

A guidance is developed in [25] to select optimal parameters for this algorithm. However, additional information is necessary, such as the shape of objects. Despite

that circular cells are majority, the number of irregularly shaped cells can not be ignored. Furthermore, they are the most challenging part in this thesis. To find the proper parameters, one of them is variable while the others are fixed. In this thesis, the continuity weight α is chosen as variable, as the others can be reasonably predefined. They are listed and explained below:

- β: smoothness parameter, set to 0, as it allows the shape of cells irregular;
- w_{line}: weight of intensity, set to 1, such that the *Snakes* are attracted to light region. In other words, final contours are repelled by nuclei of the cells to be segmented and the adjacent cells;
- w_{edge}: weight of gradient, set to 3, as the gradient plays a more important roll than intensity.

With increasing continuity weight α from 0.01 to 0.5, the best average value of Dice-Index can be obtained when $\alpha = 0.2$. This experiment is conducted based on 500 cells randomly chosen from dataset. The experimental results are the averages of three approaches, which is shown in Table 3.1.

α	0.01	0.1	0.2	0.3	0.4	0.5
Dice	0.895	0.905	**0.916**	0.912	0.912	0.911

Table 3.1: Average of Dice-Indices with increasing α. Other parameters are set as constants.

Active Contours Without Edges

Another algorithm adapted from *Snakes* is called Active Contours Without Edges. Similar to the previous attempts, some experiments are conducted to investigate if this algorithm can achieve the desired goals.

As stated in Equation 2.6, there are three controllable parameters. Due to the diverse intensity distribution of different images, two weights for intensities are set to constant: $\lambda_1 = \lambda_2 = 1$. The reasons are:

1. The average values of intensities inside and outside of a cell are approximately of same level, as part of nuclei belonging to adjacent cells are also cropped into single-cell images. Offsets can be compensated by the length term;
2. The feasibility of this algorithm can be easily analyzed with increasing value of the length parameter μ.

At this point the results with various μ are shown in Figure 3.11. It is clear that this algorithm is not suitable for this task. Here are some explanations:

- This algorithm is globally oriented, therefore the adjacent cells can influence the results. As shown in Figure 3.11 Cell 3, the dark region on the left side is regarded as a part of cell;

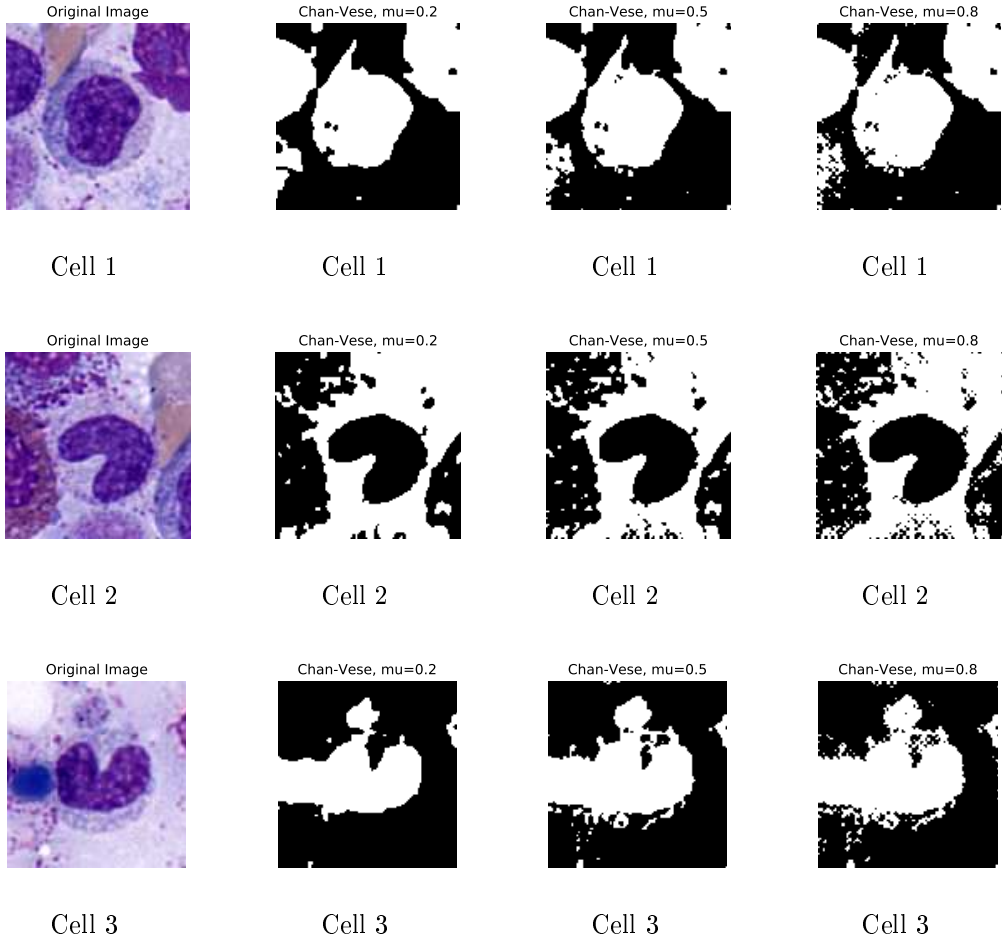


Figure 3.11: ACWE Examples with different parameters

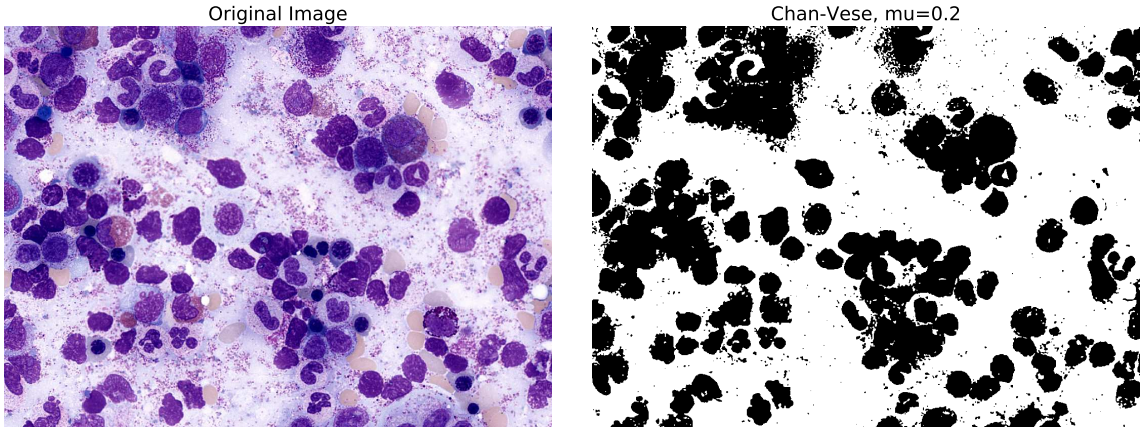


Figure 3.12: Examples for Global ACWE

- This algorithm is very intensity dependent, but not sensitive to gradients, such that it is appropriate for images without salient edges, as its name implies.

However, it can be used as a localization tool globally, as it shows in Figure 3.12.

3.4 Application

Turning now to the application with the chosen methodologies:

$$\text{Preprocessing} \begin{cases} L_0 \text{ Gradient Minimization (L0)} \\ \text{SLIC Superpixel (SP)} \end{cases}$$

$$\text{Segmentation} \begin{cases} \text{Watershed (WS)} \\ \text{Snakes (Snakes)} \end{cases}$$

After the parameters of these methods have been set, a series of experiments can be carried out. A detailed pipeline for the entire processing is shown in the next chapter.

4 Evaluation

Experiments and results using algorithms discussed previously are presented in this chapter. A range of segmented examples with different qualities are shown. Following this, advantages and disadvantages for different procedures are discussed based on the results. Furthermore, some statistical tables and plots are shown to give a direct view of evaluation.

4.1 Experiments

The results are based on 3500 cells. Algorithms are processed as shown in Figure 4.1.

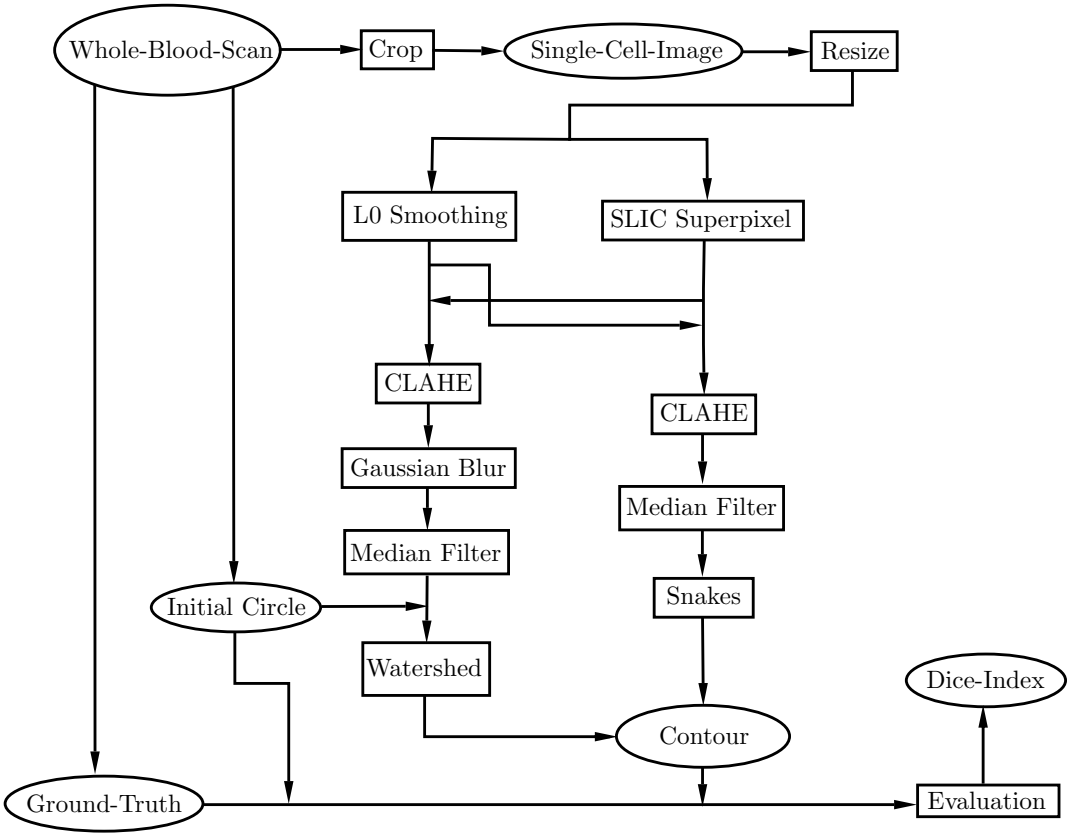


Figure 4.1: Outline of Contour Refinement

4.1.1 Stability

The initial circle is given as the minimal enclosing circle of the ground truth contour, which is quite accurate. In the real world, the initial circle may not be as precise as in this case. In addition, the parameters of each methodology are generated only once. Therefore, the stability of these methods is a big challenge. To test whether they still work for initial circles with errors is necessary.

In this chapter, test results are not only based on the untouched initial circles, but also ones with randomly added errors to the coordinates of the circle center and radius, whereas the parameter settings stay unchanged to evaluate the stability of algorithms.

Initial circles are artificially distorted with additive errors to test the stability of these methods, which are solely optimized for the initial circles *without* errors. Now the test data has three groups, as shown in Table 4.1.

Table 4.1: Errors Added to the Initial Circles

No Errors	Test data is untouched and contain no errors
Light Errors	max. 10% and 5% to the length of radius and coordinates of circle center respectively; Random errors are uniformly added
Heavy Errors	max. 15% and 10% to the length of radius and coordinates of circle center respectively; Random errors are uniformly added

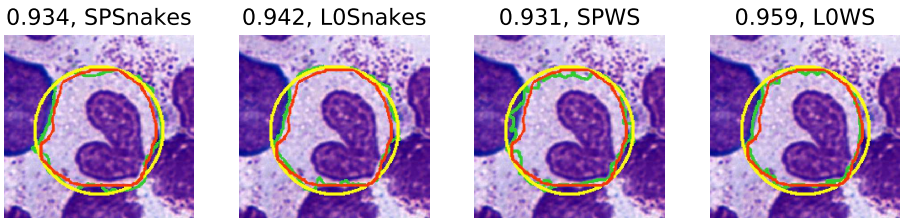
Data sets with errors are tested three times repeatedly and the results are calculated from their average to reduce the influence of randomness. If not specified, the results are presented using the data *without* artificial errors.

Single-cell images are resized to 200×200 pixels. Smoothness weight λ in L_0 Smoothing is set to 0.02 as default. Number of segments and compactness of SLIC Superpixel are set to 100 and 10 respectively. The detailed parameters of *Watershed* and *Snakes* can be found in Section 3.3.2.

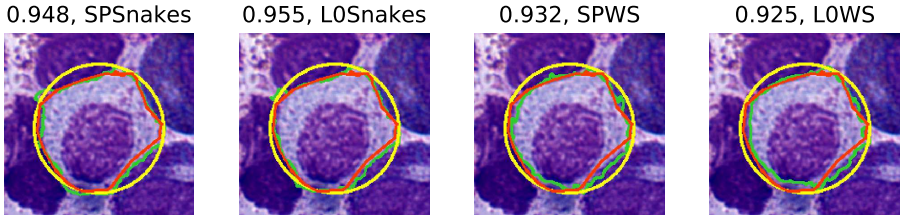
4.2 Results

Now turning to some examples of refined contours compared to initial circles and ground truth contours. First of all, several well refined examples are shown in Figure 4.2. In this case, all four chains of methods obtain desired results.

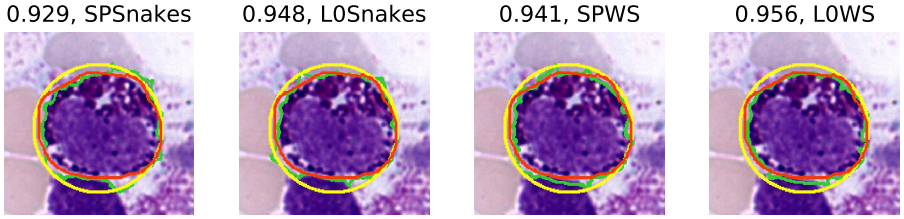
Sometimes, the refined contours using *Snakes* are more accurate than *Watershed*. However, it is possible that *Snakes* might fail. Figures 4.3 and 4.4 show some examples respectively.



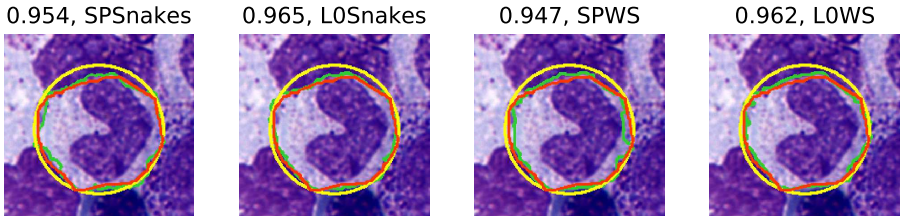
(a) Example 1



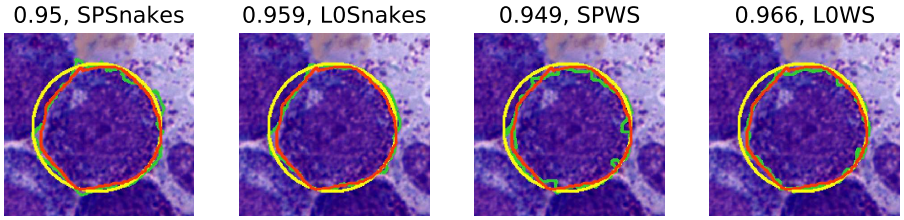
(b) Example 2



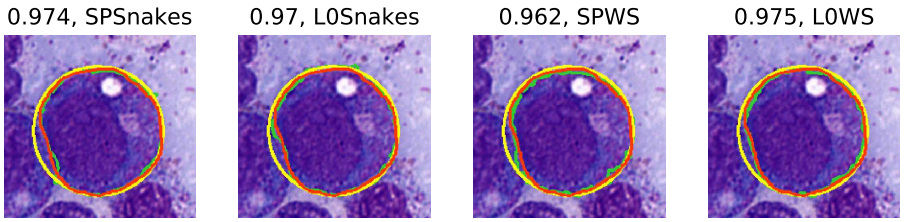
(c) Example 3



(d) Example 4

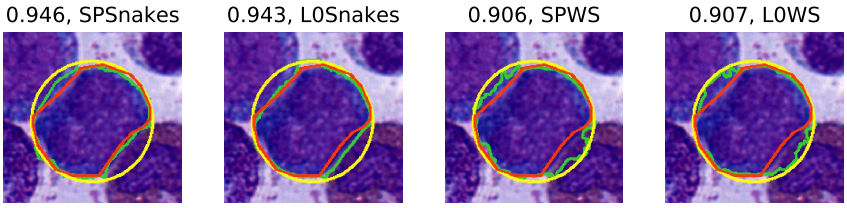


(e) Example 5

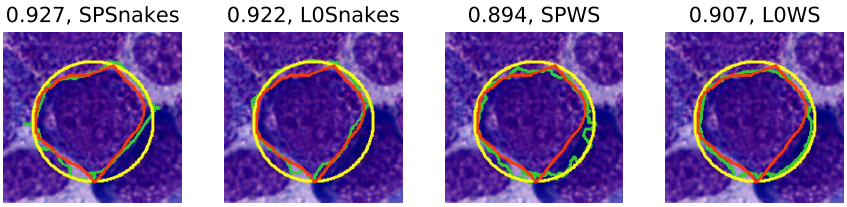


(f) Example 6

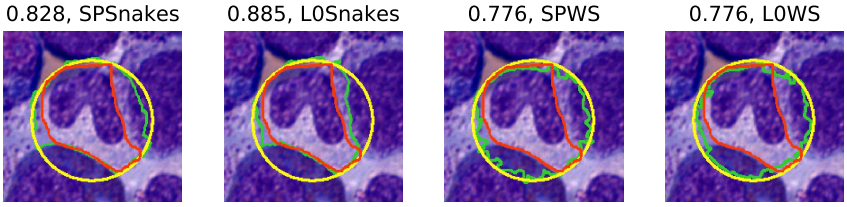
Figure 4.2: Refined Contour Examples - A
Red contours are ground truth contours; yellow ones are initial circles; green ones are refined contours. Dice-Index and methods are shown in the title.



(a) Example 1

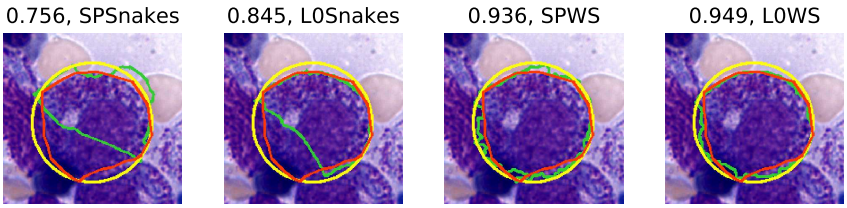


(b) Example 2

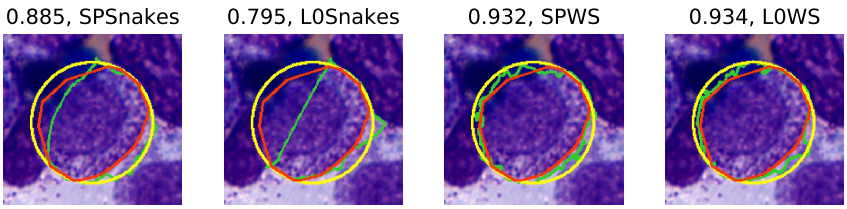


(c) Example 3

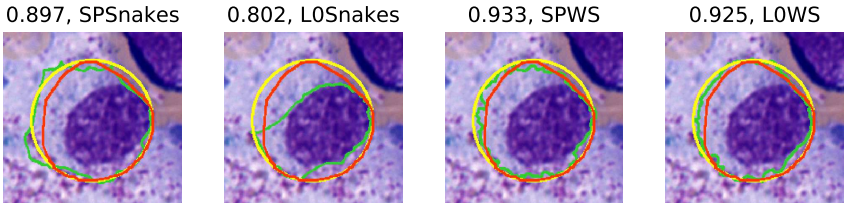
Figure 4.3: Refined Contour Examples - B. Snakes work better than Watersheds.



(a) Example 1



(b) Example 2



(c) Example 3

Figure 4.4: Refined Contour Examples - C. Watersheds work better than Snakes.

Some statistical indices are worth investigating to understand the quality of the results and to compare different methods quantitatively. The most direct way to evaluate the quality of segmentation is to calculate the Dice-Index. The average, median and variance of each method are shown in Tables 4.2-4.4. The best results in a series of experiments are highlighted in bold.

It is shown that the average and median of Dice-Index are improved to above 0.9 in the cases of lower errors. Generally speaking, *Watershed* brings better results than *Snakes* and L_0 brings better results than *Superpixel*. If the initial circles contain heavy errors, the quality of refinement is not as good as in the other two cases, but the average and median of Dice-Index are still improved by approximately 2%. The variances are decreased in all three cases compared to initial circles and the method using Superpixel as preprocessing and Watershed as segmentation brings the most centralized Dice-Indices.

Table 4.2: Average of Dice-Index to Ground Truth

	SPSnakes	L0Snakes	SPWS	L0WS	Initial Circle
No Errors	0.912	0.917	0.923	**0.927**	0.896
Light Errors	0.900	0.906	0.908	**0.912**	0.883
Heavy Errors	0.877	0.885	0.878	**0.883**	0.860

Table 4.3: Median of Dice-Index to Ground Truth

	SPSnakes	L0Snakes	SPWS	L0WS	Initial Circle
No Errors	0.923	0.929	0.936	**0.940**	0.909
Light Errors	0.909	0.917	0.918	**0.922**	0.894
Heavy Errors	0.885	**0.896**	0.884	0.890	0.870

Table 4.4: Variance of Dice-Index to Ground Truth ($\times 10^{-3}$)

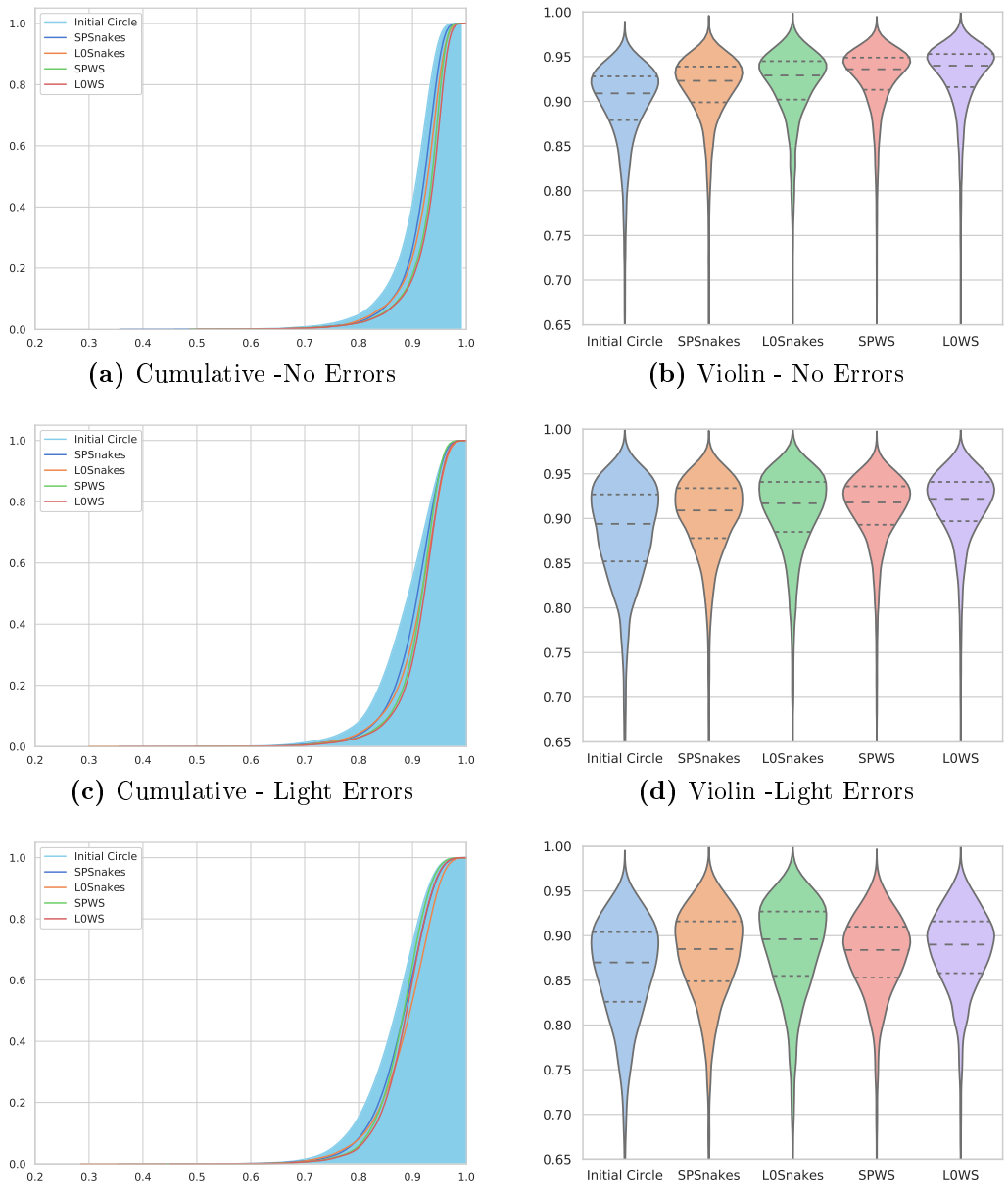
	SPSnakes	L0Snakes	SPWS	L0WS	Initial Circle
No Errors	1.867	1.991	**1.825**	1.876	2.441
Light Errors	2.436	2.609	**2.008**	2.050	3.430
Heavy Errors	3.004	3.322	**2.304**	2.354	3.534

Since the goal of this thesis is to *refine* the contours on the basis of given initial circles, it is vital to compare the Dice-Indices of initial circles and refined contours. Table 4.5 shows the proportion of improved cells in comparison to the initial circle in percent. In the best case, 95.8% of cells are better segmented compared to the initial circles. More than 70% of cells are better segmented, even in the case of heavy errors.

Table 4.5: Percent of Cells with Improved Dice-Index

	SPSnakes	L0Snakes	SPWS	L0WS
No Errors	76.422%	82.452%	94.284%	**95.800%**
Light Errors	73.821%	78.680%	79.508%	**80.595%**
Heavy Errors	73.907%	**78.280%**	68.705%	73.107%

Turning now to cumulative and violin plots of Dice-Indices to compare these methods visually (see Figure 4.5).



(a) Cumulative -No Errors

(b) Violin - No Errors

(c) Cumulative - Light Errors

(d) Violin -Light Errors

(e) Cumulative - Heavy Errors

(f) Violin - Heavy Errors

Figure 4.5: Dice-Index - Cumulative Plots and Violin Plots with Quartiles

The X-axes of cumulative plots are Dice-Indices. The Y-axes represent the normalized cumulative proportion. The areas of curves corresponding to initial circles are filled to have a direct comparison. It is clear that each curve lies in the filled region, which means the refined contours are more similar to the ground truth generally. In the case of no errors, each curve starts rising around 0.9. However, with increasing errors, all curves are shifted to left.

Violin plots are other informative graphs to visualize the distribution of data. Normalized Dice-Index distributions are presented in violin plots in Figure 4.5. The width of each violin represents the normalized number of cells with same Dice-Index. The inner lines are quartiles. The shape of violins represents the distribution of Dice-Indices, such that the quality of results can be visually recognized. It can be seen that all quartile lines are higher for refined contours than for initial circles. However, with increasing errors, the Dice-Indices are dispersed to lower values.

Last but not least, the efficiency plays a significant role to put a theoretical methodology into the real world. Table 4.6 shows the average processing time per cell using different methods. The efficiency of *Watershed* is better than *Snakes* and the efficiency of *Superpixel* is slightly better than L_0.

Table 4.6: Time of Processing per Cell

SPSnakes	L0Snakes	SPWS	L0WS
405.59ms	463.53ms	**82.61ms**	241.99ms

Test environment: OS - Ubuntu 18.04.1 LTS; Processor - Intel Core i3-8100 CPU @3.60GHz×4; Graphics Card - Nvidia GeForce GTX1060 3GB

4.3 Evaluation

At the first stage, some criteria of evaluation are presented. If the Dice-Index is greater than 0.9, this segmentation is regarded as well-segmented. If the refined segmentation is not well-segmented, but the critical part of adjacent cells is not included, it is regarded as acceptable. For instance, some part of background or cytoplasm is inside of the contour, but the nucleus of the adjacent cell is outside. It is reasonable to make this compromise, which alters characteristics of cells negligibly. Within the frame of these criteria, the Dice-Indices, which are not better than that of initial circles, are shown in Figure 4.6.

Following this, the results presented in the previous section are interpreted globally and from the view of single cells.

4.3.1 Macro Level

The refined contours are more similar to the ground truth. This conclusion can be drawn from several aspects. The curves of cumulative plots are on the right side of

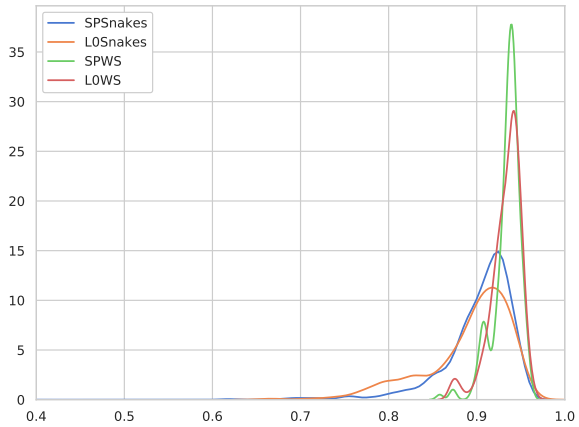


Figure 4.6: Dice-Index: Density Distribution - Worse than Initial

initial ones, which means the numbers of well-segmented cells are larger. In violin plots, all refined quartile lines are higher than the initial ones.

Average and median of Dice-Indices from different methods using different datasets present decent results. The values are around 0.9, which are regarded as well-segmented, as previously introduced. The variances have also been improved. It means these methods are suitable for the vast majority.

It is notable in Figure 4.6 that in spite of the contours being not *refined* compared to initial circles, the Dice-Indices are still located mostly in the region where they are regarded as well-segmented, especially for *Watershed*.

Comparing results from different dataset groups, it is known that the refined contours are always better than initial ones regardless of errors. However, the Dice-Indices depend on the added errors: the more errors initial circles contain, the worse refined results are. It is not surprising for *Watershed* or *Snakes*: with different sizes and positions of initial circles in *Watershed*, the predefined seeds are correspondingly changed. It may cause oversized or undersized seeds, such that the final contours may stop out of the cells or on the edges between nucleus and cytoplasm. The main limitation of *Snakes* is that the initial contour should be near the ground truth. Therefore, a large deviation of positions to ground truth may lead to unstable results. Table 4.4 shows that the more errors an initial circle contains, the less stable the result of *Snakes* is. The reason is that in some cases *Snakes* may result in a semicircle-like shaped final contour even if the ground truth is quasi circular (see Figure 4.4 b). The phenomenon that the processing using *Snakes* fails is further discussed in the next section.

To conclude, most cells are better segmented compared to the initial circles. For those refined contours, which are even less accurate than the initial ones, have great majority of Dice-Indices in the range [0.8,1.0] for *Snakes* or [0.9,1.0] for *Watershed*. Similar refinement can be also obtained for initial circles with errors, but the results become worse with increasing errors.

4.3.2 Micro Level

Evaluation in macro level gives a global point of view, whereas it lacks comparison for the same single-cell image using different methods. As the differences between preprocessing methods are not salient, only the segmentation methods are evaluated.

As shown in Figure 4.2, all four chains of methods obtain desired results. That means, Dice-Indices are greater than 0.9, in this case even around 0.95, and the contours are reasonable. It can be seen in 4.2 (b) that in spite of the irregular shape of this cell, it has still been perfectly segmented, as four adjacent cells are excluded.

Globally speaking, *Watershed* brings in better statistics. After turning to the single-cell examples, the drawbacks of this method are also clear: it works unqualifiedly for irregular cells. As shown in Figure 4.3, some parts of contours are not accurate. Figure 4.3 (c) shows that the contour of the cell below is not correctly recognized. The reason is that the seed is too large for this cell, such that a part of adjacent nucleus is also considered as minimum. On the contrary, if the size of a seed is too small, it tends to stop at the edge between nucleus and cytoplasm. To pursue a global benefit, this compromise is made. Despite the better global results in this segmentation, it can cause a severe consequence for following classification step, as the features of some cells are significantly changed, such as shapes and intensity distribution inside of cells.

Compared to *Watershed*, *Snakes* can find contours in a more reasonable way. It is rare that the edges of adjacent cells are segmented inside of refined contours. However, the global statistical results of *Snakes* are worse than *Watershed*. The reason is that the parameters are set to avoid contours being stuck in dark regions. As shown in Figure 4.4 (a,b), contours are repelled from dark parts. Besides, the smoothness weight for *Snakes* is set to 0, such that irregular shaped cells can also be well-segmented. Some positive examples are presented in Figure 4.2 (b,e) and 4.3 (a,c). Table 4.6 shows that *Watershed* is, not surprisingly, more efficient than *Snakes*, as a optimization problem is to solve in *Snakes*.

5 Discussion

A quick summary of this thesis is recapitulated in this chapter. Following that, conclusions are drawn, which cover comparison of advantages and disadvantages. At the end of the chapter, possible future works related to this thesis are mentioned.

5.1 Summary

As stated in the introduction, the objective of this thesis is to refine the leukocyte contour on the basis of given initial circle. In the initial stage of the process, single-cell images and ground truth contours are extracted from the whole scans of stained bone marrow. A set of preprocessing procedures is applied to these. At this point, the preprocessed single-cell images are normalized and ready to be segmented. Using the initial circles, which are extracted from ground truth contours and optionally added artificial errors, segmentation techniques are applied to the single-cell images. As soon as these steps have been carried out, the segmented contours are compared to the ground truth to evaluate the quality of this process.

According to the results presented in the previous chapter, this chain of methods has achieved the goal of this task. Based on the results of segmentation, the efficiency of leukocyte counting and classification should be improved, compared to the manual operation. It is automatic, because there is no manual interference during the processing.

Nevertheless, it has a number of drawbacks. First of all, the parameters of different methods must be adapted to the specific data sets through experimental analysis. That means, the settings of this processing pipeline are not automatically generated. The stability of this chain of methods is not completely perfect. If the initial circles contain heavy errors, the refined Dice-Indices are still better than initial ones, but they do not remain in the level as with no errors. Moreover, it takes approximately half a second for the step of segmentation per cell. The efficiency needs further improvement, as the number of leukocytes is enormous. The average number of counted cells is 16609 in a bone marrow biopsy [26]. Roughly estimated, it might take hours for such a test.

5.2 Outlook

Machine learning, or more specifically deep learning, is becoming a routine technique for image processing. It is also promising for this task. One of the advantages of deep learning is the accurate extraction of useful features, which is in this case the drawback of classical methodologies. However, deep learning techniques require a

large amount of data as inputs to make a precise decision. Furthermore, it might be unclear what is learned while training.

In this thesis, the results of classical segmentation techniques can serve as baseline of further deep learning techniques. With amassed knowledge from classical methodologies, the selection of data to be trained should be more reasonable. After that, the results of counting and classification for leukocytes with classical and deep learning techniques can be compared to show concrete differences between them.

Bibliography

[1] V. Piuri and F. Scotti, "Morphological classification of blood leucocytes by microscope images," in *Computational Intelligence for Measurement Systems and Applications, 2004. CIMSA. 2004 IEEE International Conference on*, pp. 103–108, IEEE, 2004.

[2] J. Zhao, M. Zhang, Z. Zhou, J. Chu, and F. Cao, "Automatic detection and classification of leukocytes using convolutional neural networks," *Medical & Biological Engineering & Computing*, vol. 55, no. 8, pp. 1287–1301, 2017.

[3] L. Putzu, G. Caocci, and C. Di Ruberto, "Leucocyte classification for leukaemia detection using image processing techniques," *Artificial Intelligence in Medicine*, vol. 62, no. 3, pp. 179–191, 2014.

[4] N. Theera-Umpon and S. Dhompongsa, "Morphological granulometric features of nucleus in automatic bone marrow white blood cell classification," *IEEE Transactions on Information Technology in Biomedicine*, vol. 11, no. 3, pp. 353–359, 2007.

[5] S. Beucher and F. Meyer, "The morphological approach to segmentation: the watershed transformation," *Optical Engineering-New York-Marcel Dekker Incorporated-*, vol. 34, pp. 433–433, 1992.

[6] L. Vincent and P. Soille, "Watersheds in digital spaces: an efficient algorithm based on immersion simulations," *IEEE Transactions on Pattern Analysis & Machine Intelligence*, no. 6, pp. 583–598, 1991.

[7] J. Chanussot, P. Lambert, *et al.*, "Watershed approaches for color image segmentation.," in *NSIP*, vol. 99, pp. 129–133, 1999.

[8] F. Meyer, "Topographic distance and watershed lines," *Signal processing*, vol. 38, no. 1, pp. 113–125, 1994.

[9] M. Kass, A. Witkin, and D. Terzopoulos, "Snakes: Active contour models," *International journal of computer vision*, vol. 1, no. 4, pp. 321–331, 1988.

[10] T. F. Chan and L. A. Vese, "Active contours without edges," *IEEE Transactions on image processing*, vol. 10, no. 2, pp. 266–277, 2001.

[11] R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua, S. Süsstrunk, *et al.*, "Slic superpixels compared to state-of-the-art superpixel methods," *IEEE transactions on pattern analysis and machine intelligence*, vol. 34, no. 11, pp. 2274–2282, 2012.

[12] A. Trémeau and P. Colantoni, "Regions adjacency graph applied to color image segmentation," *IEEE Transactions on image processing*, vol. 9, no. 4, pp. 735–744, 2000.

[13] L. Xu, C. Lu, Y. Xu, and J. Jia, "Image smoothing via l 0 gradient minimization," in *ACM Transactions on Graphics (TOG)*, vol. 30, p. 174, ACM, 2011.

[14] S. M. Pizer, E. P. Amburn, J. D. Austin, R. Cromartie, A. Geselowitz, T. Greer, B. ter Haar Romeny, J. B. Zimmerman, and K. Zuiderveld, "Adaptive histogram equalization and its variations," *Computer vision, graphics, and image processing*, vol. 39, no. 3, pp. 355–368, 1987.

[15] K. Zuiderveld, "Contrast limited adaptive histogram equalization," *Graphics gems*, pp. 474–485, 1994.

[16] P. Jaccard, "Distribution de la flore alpine dans le bassin des dranses et dans quelques régions voisines," *Bull Soc Vaudoise Sci Nat*, vol. 37, pp. 241–272, 1901.

[17] T. Sørensen, "A method of establishing groups of equal amplitude in plant sociology based on similarity of species and its application to analyses of the vegetation on danish commons," *Biol. Skr.*, vol. 5, pp. 1–34, 1948.

[18] L. R. Dice, "Measures of the amount of ecologic association between species," *Ecology*, vol. 26, no. 3, pp. 297–302, 1945.

[19] A. Meijster, *Efficient sequential and parallel algorithms for morphological image processing*. University Library Groningen][Host], 2004.

[20] S. van der Walt, J. L. Schönberger, J. Nunez-Iglesias, F. Boulogne, J. D. Warner, N. Yager, E. Gouillart, T. Yu, and the scikit-image contributors, "scikit-image: image processing in Python," *PeerJ*, vol. 2, p. e453, 6 2014.

[21] D. Marr and E. Hildreth, "Theory of edge detection," *Proc. R. Soc. Lond. B*, vol. 207, no. 1167, pp. 187–217, 1980.

[22] P. Marquez-Neila, L. Baumela, and L. Alvarez, "A morphological approach to curvature-based evolution of curves and surfaces," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 36, no. 1, pp. 2–17, 2014.

[23] G. Bradski, "The OpenCV Library," *Dr. Dobb's Journal of Software Tools*, 2000.

[24] J.-R. Dalle, H. Li, C.-H. Huang, W. K. Leow, D. Racoceanu, and T. C. Putti, "Nuclear pleomorphism scoring by selective cell nuclei detection.," in *WACV*, 2009.

[25] O. V. Larsen, P. Radeva, and E. Martí, "Bounds on the optimal elasticity parameters for a snake," in *International Conference on Image Analysis and Processing*, pp. 37–42, Springer, 1995.

[26] J. Liang, J. A. Malherbe, K. A. Fuller, B. Mirzai, C. George, T. L. Carter, C. H. Cole, B. B. Guo, K. Meehan, and W. N. Erber, "Automated enumeration of lymphoid and plasma cells in bone marrow to establish normal reference ranges," *Journal of clinical pathology*, pp. jclinpath–2018, 2018.

A Appendix

A.1 Pseudo Code of Immersion Watershed Algorithm

```
# LABEL = label of pixel_x
# NEIGHBOR = label of pixel_x's neighbors

if LABEL > 0: # pixel_x is visited
        L1:
        label each pixel in Geodesic Influence Zone of pixel_x with LABEL;
elif LABEL <= 0: # pixel_x is not visited yet
        if all NEIGHBOR <= 0:
                label this pixel with new label; # pixel_x is a new minimum
                GOTO L1;
        elif at least one NEIGHBOR > 0:
                if number of labels of all NEIGHBOR == 1:
                        label pixel_x with NEIGHBOR;
                        GOTO L1;
                elif number of labels of all NEIGHBOR  >1:
                        label pixel_x as watershed;
```

Listing A.1: Pseudo code of Immersion Algorithm