CELL DETECTION IN KNIFE-EDGE SCANNING MICROSCOPY IMAGES OF NISSL-STAINED MOUSE AND RAT BRAIN SAMPLES USING RANDOM FORESTS

A Thesis

by

SHASHWAT LAL DAS

Submitted to the Office of Graduate and Professional Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

Chair of Committee,	Yoonsuck Choe
Committee Members,	John Keyser
	Louise Abbott
Head of Department,	Dilma Da Silva

December 2014

Major Subject: Computer Science

ABSTRACT

Microscopy has developed into a very powerful medium for studying the brain. The Knife-Edge Scanning Microscope (KESM), for example, is capable of imaging whole rat and mouse brains in three dimensions, and produces over 1.5 terabytes of images per brain. These data can reveal the structure and organization of the brain's internals including neurons and blood vessels. Neuron count and density strongly influence the behavior of an organism, and measuring their spatial distribution is key to a better understanding of the workings of the brain. This kind of analysis involves identifying neurons in large brain regions, for which fast automated detection methods are necessary. Most of the current automated cell detection techniques require complex preprocessing of images, use heuristics that are time consuming to develop, or do not generalize well to three dimensional data. In this thesis, I propose two methods based on random forests for detecting neuron bodies in the rat and mouse brain KESM data.

The proposed methods require a few hundred cell centers to be manually labeled. Random forests are trained to predict if a voxel is a cell center or not by using these labeled data and features derived from orthogonal image patches. They can then be used to locate cell centers in 3-D in other images, aided by a refinement step whose parameters are determined from the training data. Minimal manual input is required, and random forests provide a good combination of accuracy and speed. This is expected to enable fast counting and density measurements of neurons in brain regions. The detected cell centers should also be valuable as seeds for cell segmentation methods.

DEDICATION

to my family

ACKNOWLEDGEMENTS

I sincerely thank Dr. Yoonsuck Choe, my advisor, for sharing his knowledge and experience that were invaluable for my research, while also giving me the confidence and freedom to pursue my own ideas. He deserves huge credit for making the time spent on this research enjoyable. I am grateful to Dr. Louise Abbott and Dr. John Keyser, my committee members, for very helpful discussions from which I always emerged with new perspectives and solutions.

I am thankful for the support from the faculty and staff at Texas A&M University and the Department of Computer Science and Engineering. The CSE department also provided the computational resources where I performed my experiments. Without them, my research would have been much slower. I am indebted to fellow students and friends for some wonderful experiences during my time at Texas A&M.

Everyone in my family has been a source of unceasing encouragement for which I express my deep gratitude. I am fortunate beyond words that in every difficult moment, I have had their complete support.

This research was funded in part by National Science Foundation grant #1208174 and #0905041 (PI: Yoonsuck Choe).

TABLE OF CONTENTS

Page

Page

LIST OF FIGURES

FIGURE Page

FIGURE Page

LIST OF TABLES

TABLE Page

1. INTRODUCTION

1.1 Motivation

Neurons are the fundamental building blocks of the nervous system, which controls and coordinates the body's activities. Studying the organization of neurons can lead to a greater understanding of the workings of the nervous system, and thus, the brain. According to Williams and Herrup [46], "the number of neurons and their relative abundance in different parts of the brain is a determinant of neural function and, consequently, of behavior."

Several studies have found important phenomena to be associated with changes in neuron number. Loss of neurons in certain brain regions has been found to accompany neurological disorders such as Alzheimer's disease [45] and Parkinson's disease [14]. The amygdala of autistic brains has been observed to have significantly fewer neurons than usual [37]. Further, the development of more accurate counting methods has helped show that significant cell loss does not occur with normal aging, contrary to popular belief [5]. A deeper study of the brain can be facilitated by knowledge of the location of neurons. Together with the neuron count, neuron locations can be used for measuring the brain's spatial organization [20] and understanding the brain's circuitry.

Brain microstructure, including neuron number and morphology, is widely studied through microscopic examination of brain tissue. Modern microscopy techniques produce high-quality data in quantities so large that examination by experts is infeasible. This makes the development of automated techniques for tasks such as the detection and counting of neurons a necessity.

One such modern microscope is the Knife-Edge Scanning Microscope (KESM) [8],

[27], developed at Texas A&M University's Brain Networks Laboratory. It has been used to acquire whole rodent brain images in three dimensions at sub-micrometer resolutions, permitting the accurate detection and counting of brain cells across all brain regions.

Current techniques for automated cell detection are mainly adapted for certain kinds of data or require significant preprocessing of images. Only a few techniques have been adapted to data obtained from Nissl staining, which enhance neuronal cell bodies. Moreover, methods that extend to three-dimensional (volumetric) data are rare. These limitations motivate the development of a new automated cell detection technique for KESM Nissl data.

1.2 Goal of Research

This research attempts to develop an automated method for cell detection in KESM images of Nissl-stained rat and mouse brains. The output of the technique will be the three-dimensional location of the cell centers in the given data. This output can be used for cell counting and for seeding cell segmentation algorithms.

Key characteristics of the method will include minimal manual input, including labeling and preprocessing; and computational efficiency so as to be applicable to large data sets.

1.3 Approach and Methods

I use a machine learning-based method for detecting cells in KESM data semi-automatically. This method essentially consists of the following steps:

1. Manual labeling of a cell centers in a small number of subvolumes to create training data,

2. Training random forest classifiers on these data,

3. Finding likely cell centers in unseen data using these classifiers, and

4. Refinement of these centers to get one point per cell.

The details of the procedure for the rat brain data and mouse brain data differ slightly, but follow the same basic template. A brief description of these steps follows.

1.3.1 Manual Labeling

Cell centers in a small set of subvolumes of the data need to be labeled for providing training data to the random forest classifiers, and for testing the accuracy of the proposed method.

Around 5–6 small image volumes are extracted from a brain region and all cell centers (here, centers of neuronal cell bodies) are exhaustively marked in 3-D. The volumes are chosen such that the variation in appearance of cells is sufficiently captured. In total, a few hundred cell centers are labeled.

1.3.2 Training Random Forest Classifiers

The coordinates of the labeled cell centers are used for training the random forest classifiers. Orthogonal image patches are extracted around each of these cell centers and constitute the cell center training set. Patches are also extracted around some of the remaining points (not cell centers), which form the non-cell center training set. A classifier is trained on these training sets so that it predicts different labels for cell centers and non-cell centers.

This idea is implemented differently for rat brain and mouse brain data due to their unique characteristics. For rat brain data, a cascade of random forest classifiers is used, with each stage being more complex than its predecessor in terms of image features used (size of the image patches extracted) or classifier strength. Since the rat brain data have low cell density and consists of relatively large cells, most of the

points are obvious non-cell centers and can be easily rejected with low computation. This happens in the initial, simple stages of the cascade, which thus speeds up the detection process.

In the mouse brain data, cells are small and very densely packed which precludes quick elimination of non-centers using simple features. Therefore, a single complex stage is used with a fixed patch size. In contrast to the rat data classifiers that output center/non-center labels, this stage predicts the probability of a point being a cell center. It is trained on a probability map of the training volumes. This map is generated based on the distance of a point from its nearest cell center.

1.3.3 Finding Likely Cell Centers in Unseen Data

The trained classifiers are applied to unseen data, or test data. For the rat data, the classifiers output a set of points which are cell center candidates. The classifier for the mouse data outputs, for each point in the data, its likelihood of being a cell center.

1.3.4 Refinement of Classifier Output

Seldom do the rat data classifiers output exactly one cell center candidate per cell – typically, the candidates form small clusters around the cell centers. These clusters are identified using the mean shift clustering algorithm. One cluster is assumed to represent one cell. Finding the center of a cluster gives the approximate center of a cell.

Cell centers in the mouse data are extracted by finding local maxima in the probability map output by the classifier.

1.4 Significance

This work proposes a method for cell detection that can be used for counting and segmenting cells in the rat and mouse brain KESM Nissl data sets. The method has been designed to be computationally efficient and parallelizable, which opens up opportunities to study the KESM data on a larger scale. Manual input required is limited and simple in nature, such as marking cell centers in a small number of sub-volumes of the data, making it easy to use. Results from this work can contribute to a greater understanding of the number of neurons in the brain and their morphology. These are essential for research on the structure and function of the brain.

1.5 Outline

This thesis is organized as follows. Section 2 describes prior work in cell detection and the closely related task of cell segmentation that further motivates a separate detection method. Section 3 describes the cell detection methods proposed for the mouse brain and rat brain data sets. Results from applying these methods are presented and discussed in Section 4, including accuracy and speed evaluations. In Section 5, improvements and limitations of the proposed methods are discussed. The conclusions drawn from this work are presented in Section 6.

2. BACKGROUND AND PRIOR WORK*

This section first provides a background on the Knife-Edge Scanning Microscope, which is the source of the data used in this work, followed by a brief review of relevant prior work on cell detection and segmentation.

2.1 Knife-Edge Scanning Microscope

The Knife-Edge Scanning Microscope (KESM) [8], [27], shown in Figure 2.1, was developed at the Brain Networks Laboratory at Texas A&M University for imaging small animal brains in 3-D at high resolution.

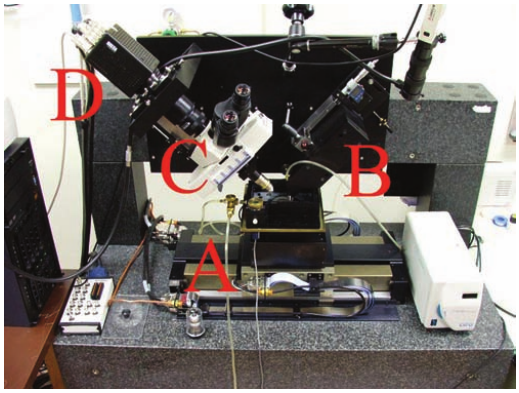


Figure 2.1: The Knife-Edge Scanning Microscope. (A) Movable stages (B) Diamond knife and illuminator (C) Microscope (D) Line-scan camera. Adapted with permission from [27].

It concurrently sections and images tissue at a sub-micrometer voxel size of 0.6 μm $\times$ 0.7 μm $\times$ 1 μm which is good enough to expose the morphology of structures such

*Some figures used in this section are reprinted with permission from D. Mayerich, L. Abbott, and B. McCormick, "Knife-edge scanning microscopy for imaging and reconstruction of three-dimensional anatomical structures of the mouse brain", *Journal of Microscopy*, vol. 231, pp. 134–143, Pt 1 2008, © 2008 The Authors Journal compilation © 2008 The Royal Microscopical Society.

as dendrites, dendritic spines, and axons [8]. Imaging proceeds with high throughput – a 1 cm^3 volume of tissue can be imaged in around 100 hours, producing about 1 to 2 terabytes of volumetric data. The KESM has been successfully used for imaging whole rodent brains stained in Golgi, Nissl, and India ink [9], [10].

2.1.1 KESM Operation

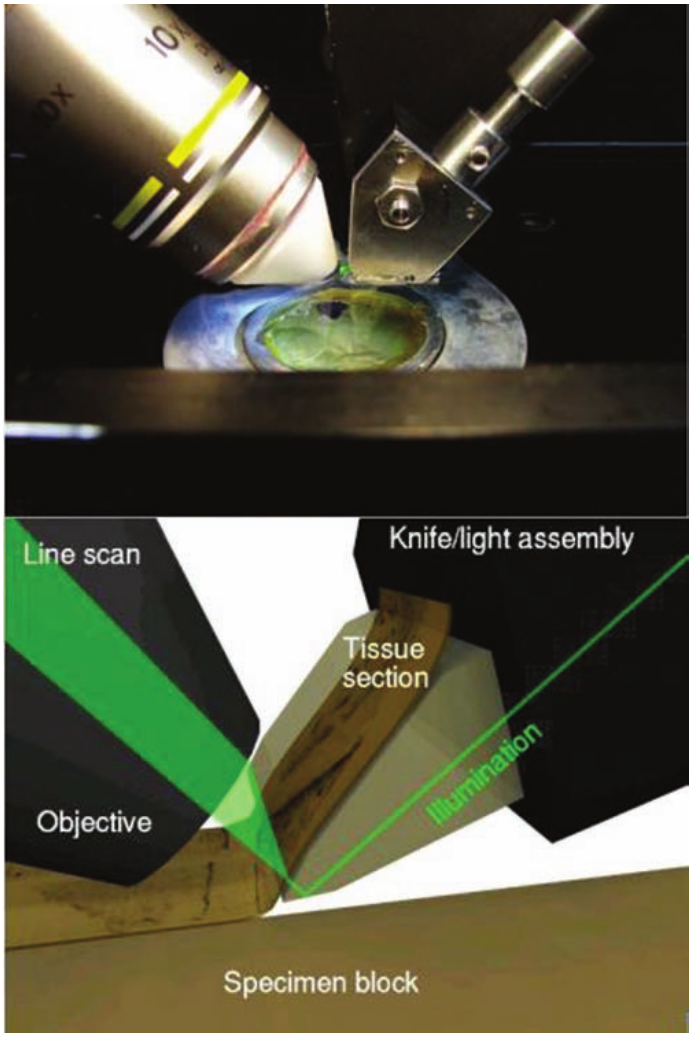


Figure 2.2: Operation of the KESM. The top image shows the objective (left) and knife (right) together used for imaging the specimen. This is made more clear by the illustration at the bottom. Adapted with permission from [27].

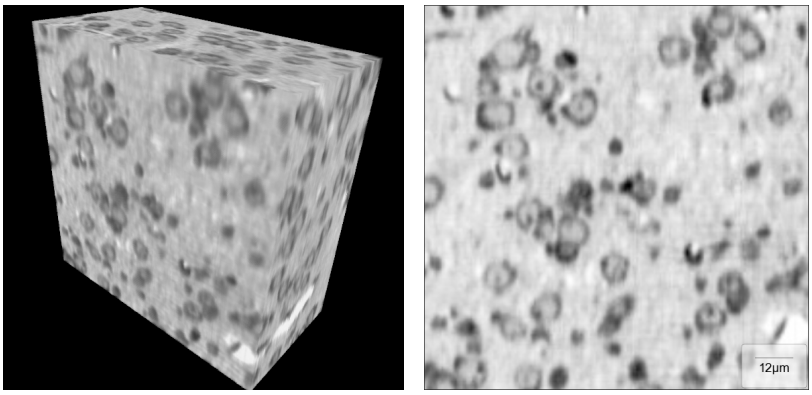
KESM performs line-scan imaging of sections of a specimen. Some of the main components of the instrument include a line-scan camera, microscope objective, diamond knife-collimator assembly, a white light illuminator, a specimen tank, and a server for control and image acquisition [8]. An illustration of its operation can be seen in Figure 2.2.

A hard plastic resin such as araldite is used to embed the specimen which is then mounted on the KESM's specimen ring [9]. A lateral sectioning strategy is used to process the entire volume in multiple sweeps. During sectioning, the specimen is moved towards the diamond knife via a mechanical positioning stage. The diamond knife illuminates and cuts the portion being imaged into a section as thin as 0.5 μm [27]. This physical sectioning allows for higher resolution imaging along the z direction when compared to optical sectioning methods [8]. The camera images the section and sends the data to the server.

2.1.2 Specimen Staining

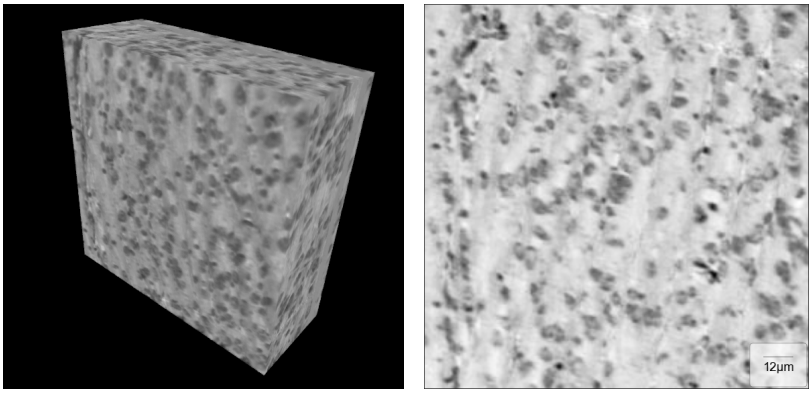
A staining method is chosen for the specimen depending on the cellular features that need to be seen. Whole mouse brains stained in Nissl, Golgi, and India ink have been imaged by the KESM. Nissl staining makes cell soma visible. The Golgi stain is used for the study of neuronal morphology, and India ink enhances the contrast of the vasculature [27].

The Nissl stain is most relevant for this thesis. As it enhances cell bodies, images of Nissl-stained samples can be used for cell detection and counting. This thesis deals with KESM data previously obtained from imaging Nissl-stained mouse and rat brains [9], [27]. Sample rat brain and mouse brain KESM data are shown in figures 2.3 and 2.4, respectively.



(a) Nissl volume (b) Nissl slice

Figure 2.3: KESM rat brain Nissl data. (a) A 200 voxels × 200 voxels × 100 voxels volume with voxel size 0.6 μm × 0.7 μm × 1.0 μm. (b) A single 200 × 200 image slice from the volume.



(a) Nissl volume (b) Nissl slice

Figure 2.4: KESM mouse brain Nissl data. (a) A 256 voxels × 256 voxels × 101 voxels volume with voxel size 0.6 μm × 0.7 μm × 1.0 μm. (b) A single 256 × 256 image slice from the volume.

2.2 Prior Work

A wide range of automated and semi-automated techniques for detecting cells in images can be found in the literature. Cell detection typically forms a part of the cell segmentation process which attempts to determine the number of cells, their spatial arrangement, shapes and extents. Due to the considerable variation in images obtained from different sources, no single method has become dominant either for detection or segmentation. Depending on the method chosen, detection can be a prerequisite for segmentation or vice versa. Since one of the key objectives of cell detection is to facilitate segmentation, both tasks merit discussion in the context of this work.

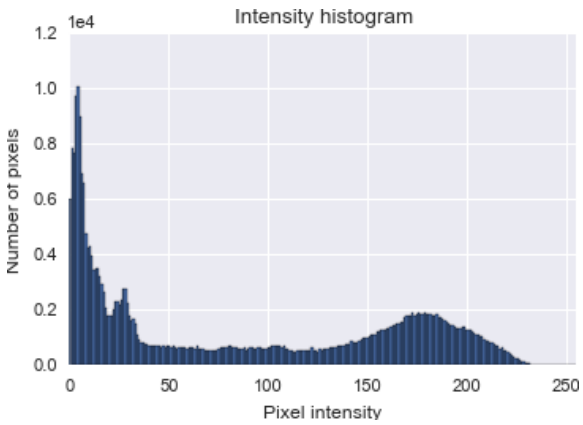
2.2.1 Cell Segmentation

The primary goal of cell segmentation is to extract the contours of cells in an image. A cell detection process can involve segmentation as a first step, following which the centers of the segmented cells can be returned as the detection results.

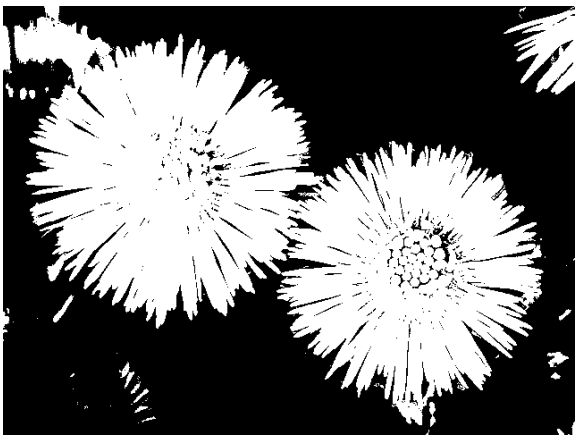
Numerous segmentation methods have been developed for medical images, and are reviewed in [30], [34], [38], [47]. Approaches include simple image analysis techniques such as thresholding and morphological operations; region segmentation techniques like the watershed algorithm, clustering, and region growing; methods based on deformable models such as active contours and level-sets; ideas from pattern recognition such as artificial neural networks; and graph search techniques, most prominently those using graph cuts.

Discussion here is limited to the most common of these methods, namely thresholding, the watershed algorithm, and deformable models.

(a) Grayscale image



(b) Histogram



(c) Threshold set at 95

Figure 2.5: Segmentation by thresholding. Flowers are separated from the background in (c). Original image at pixabay.com/en/tussilago-farfara-flower-macro-100759 by Pixabay user Hans Braxmeier is in the public domain [3].

2.2.1.1 Intensity Thresholding

Intensity thresholding involves first finding an intensity value T, called the *threshold*, which can separate a grayscale image into foreground and background regions. Pixels in the image are then grouped into two classes by setting pixels with intensities above the threshold to one value, and the rest of the pixels to another. If these values are chosen as 1 and 0, the new pixel intensity v_i' for pixel i becomes:

$$v_i' = \begin{cases} 1 & v_i \geq T \\ 0 & \text{otherwise} \end{cases} \tag{2.1}$$

where v_i is the original intensity of the pixel.

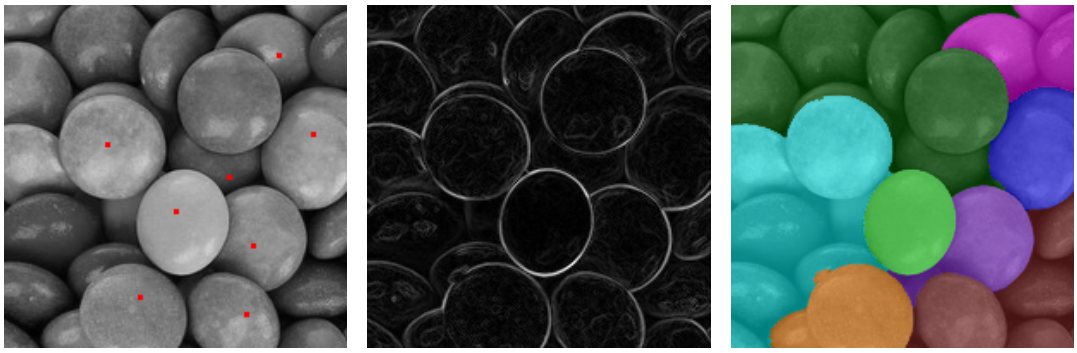
The threshold value T can be automatically determined by statistical analysis of intensity values in the image. Otsu's method [31] analyzes the histogram of grayscale intensities (see Figure 2.5b) to select as the threshold the value that would maximize the separability between the resultant classes. Once an image is thresholded, the foreground can be extracted and further refined by algorithms such as morphological operations or the watershed transform.

Although it has been used for cell segmentation in early work such as [49] and [48], automatic thresholding, in general, has limited success in segmenting microscopy images as the intensity values of the foreground (such as cells) and background overlap. Factors such as uneven illumination in the image, imaging noise, and poor sample staining can lead to low contrast images. Adaptive thresholding, where a threshold value is computed for each local region in an image, or preprocessing steps such as noise reduction and illumination correction, can mitigate this to some extent. Thresholding is thus usually performed not as a sole segmentation step but as an intermediate step after image preprocessing. It is used for speeding up the proposed cell detection method, as described later.

2.2.1.2 Watershed Algorithm

The watershed algorithm is a very popular segmentation technique for grayscale images that partitions an image into non-overlapping regions [23]. It can be understood by interpreting a 2-D grayscale image as a 3-D topographic surface where intensity represents elevation. Mountains are formed from high intensity values in

the image, and low values form valleys [16]. This surface is flooded with a certain level of water, which descends into local minima. Points from which water flows into a given local minimum form its catchment basin. Each basin represents a segmented region in the image.



(a) Image with markers (b) Gradient magnitude (c) Segmented image

Figure 2.6: Marker-controlled watershed segmentation. Each marker belongs to a unique region in the segmented image. Original image at pixabay.com/en/variation-confectionery-coated-69470 by Pixabay user PublicDomainPictures is in the public domain [33].

Watershed segmentation is computationally efficient and can be generalized to n-dimensional images. Depending on the image, it is applied to either the original image, the gradient image, or the distance transform of the thresholded image. Watershed on the distance transform image is frequently used to separate touching objects.

An important drawback of the watershed algorithm is its tendency to oversegment images due to the presence of a large number of local minima caused by factors such as noise. Smoothing the image, post-processing the watershed segmentation, or using the *marker-controlled watershed* (see Figure 2.6) can produce better results.

Marker-controlled watershed is initialized with a set of pixels called *markers* or *seeds*. At least one marker must be provided inside each target object [34]. The algorithm uses these markers as the local minima. While potentially very good results can be achieved, automatically selecting good markers is a challenging problem [23].

2.2.1.3 Deformable Models

In these techniques, a geometric model of target objects is defined which is then deformed to fit objects in an image. In two dimensions, these models behave much like rubber bands under external force, but the forces acting on the model are exerted by the image [18].

The user initializes the model by providing a coarse segmentation of the object to be segmented. Forces guide the model towards image features like edges, eventually settling on the outline of an object. Constraints on the shape of the model help maintain smoothness in the presence of noise. This process can usually be expressed as the minimization of an energy function [29].

Snakes are two-dimensional deformable contours which are constrained to be closed. The energy of a contour is given by the sum of its internal deformation energy and potential energy. The internal energy has terms controlling the tension and rigidity of the contour, while the potential energy determines the image features that attract the contour [29]. Numerical methods are applied to solve the partial differential equation for minimizing this energy.

Other formulations of deformable models such as level sets, and extensions such as active shape models (ASM) [50] are also widely used. They allow segmentation to be performed in three dimensions, enable incorporating prior knowledge such as shape information, and handle merging and splitting naturally. The success of segmentation using deformable models depends on initialization close to the object of

interest [29] and selection of appropriate parameters governing the contour evolution.

2.2.2 Cell Detection

Cell detection is the identification of cells in an image by a unique point per cell, such as the cell center. Once cells are accurately segmented, detection is trivial. However, as described above, most segmentation approaches need an initial set of markers which are critical to the segmentation quality [22].

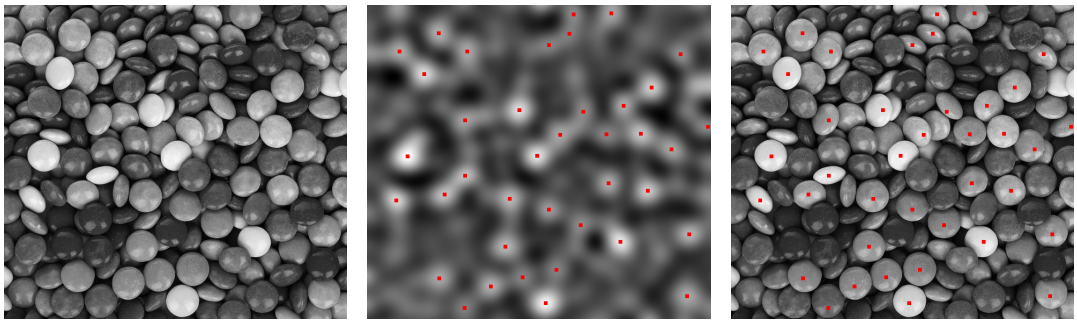
Seed finding procedures exist for generating one or more seed points per object. Here, we focus on cell detection procedures that return only one point per cell. The cell detection output can be directly used for:

1. obtaining the cell count by simply counting the number of cells detected, and

2. segmentation of cells by using the centers of detected cells as markers.

Much of the early work in cell detection employed image analysis techniques such as finding the peaks in a Euclidean distance map of a binarized image [21]. More recently, Al-Kofahi et al. [22] detected nuclei using the multiscale Laplacian of Gaussian (LoG) where the distance map was used for automatic scale selection. The Laplacian of Gaussian is an operation that produces large responses at blob-like structures in an image, and is used extensively for blob detection (see Figure 2.7). Its scale parameter determines the blob sizes it can detect. Al-Kofahi et al. found shape and size cues from the distance map to aid in selecting the scale parameter and detecting nuclei in heterogeneous clusters.

Binarization is a key step in the above methods that use the distance map and in many others [1], [22], [26], [40], [52]. The binary image must accurately separate the cells from the background and preserve their outlines. Unfortunately, this is a hard task for low-contrast images and can require sophisticated preprocessing steps to be



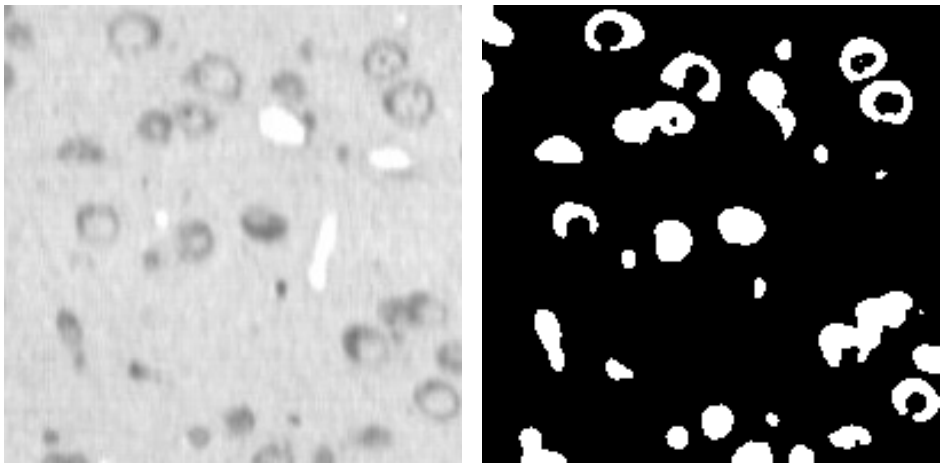
(a) Grayscale image (b) Laplacian of Gaussian (c) Detected blobs

Figure 2.7: Laplacian of Gaussian blob detection. (a) Original 640 pixels × 584 pixels grayscale image with bright and dark blob-like objects. (b) Laplacian of Gaussian with scale 21, intensities inverted. Red dots are local maxima. (c) Image with red dots indicating centers of detected bright blobs.

designed for the particular data set. In the data considered in this thesis, cell bodies and the background have similar intensity values which makes binarization difficult. This is shown in Figure 2.8.

Algorithms which detect interest points based on shape and symmetry have also been applied to cell detection. The Hough transform can detect parametric curves such as circles and ellipses and has been used for nuclei detection [13], but is computationally expensive. Radial symmetry [24] of nuclei was used to detect them in [51] and [43]. These methods need several parameters to be specified for optimal results.

Pattern recognition is gaining a more prominent role in cell detection. D'Souza used template matching followed by refinement as part of a routine for cell detection in the KESM Nissl-stained rat brain data [15]. A 3-D template of a typical cell with size 25 × 25 × 7 voxels was applied on all voxels in a subvolume. The maxima of the matching scores indicated likely locations of cells. Template matching suffers when target objects vary significantly in appearance or size. Matching multiple templates or large 3-D templates can make results more accurate but significantly slow down



(a) Image from rat data set (b) Smoothed and thresholded

Figure 2.8: Poor results from thresholding the rat brain data. (a) Original grayscale image from rat data set. (b) Image smoothed with a median filter and thresholded. Cell bodies and boundaries are poorly separated from the background.

the process.

Artificial neural networks (ANNs) were used in [28] for detecting cells in the KESM Nissl data. Principal component analysis (PCA) was performed on intensities in orthogonal cross sections of each voxel to form input vectors with 20 features. Similar features are used in the proposed method, but without the PCA dimensionality reduction. The detection is performed on a $2X$ downsampled input volume. This method could achieve a recall of 82.4% and precision of 92.8%, surpassing the performance of LoG blob detection and the spherical Hough transform.

A similar set of features was used in [41], where the reconstruction error of an input voxel from PCA basis vectors determines whether it is classified as a cell center or off-center voxel. Han et al. [19] used support vector machines (SVMs) to detect nuclei in 2-D cell culture images. They used intensity and gradient information in 20×20 pixel image patches as features for an SVM classifier, achieving over 90%

accuracy. Other machine learning approaches such as convolutional neural networks have also demonstrated excellent results in cell detection [25], making this a very promising avenue for exploration.

3. APPROACH AND METHODS

Two different but related methods are used for detecting cells in the rat and mouse brain data sets. This section first describes the image acquisition process, and then explains the two proposed methods in more detail. It concludes with the procedure for evaluating the detection results.

3.1 Data Acquisition

Two data sets are used in this work – the Nissl-stained rat brain data set, and the Nissl-stained mouse brain data set. These are equivalently referred to as the "rat data" and "mouse data" in this text. The data sets were acquired previously by other members of the Brain Networks Laboratory using methods detailed in [9] and [27]. A short summary is presented here.

Each data set was acquired from imaging a whole Nissl-stained animal brain at a resolution of 0.6 μm $\times$ 0.7 μm $\times$ 1.0 μm using the KESM. Nissl staining is a method that stains nucleic acids, which results in the cytoplasm and the nucleolus in each neuron being stained due to their acidic content. The Nissl stain overall enhances the structure of soma (cell bodies), making it suitable for cell detection and segmentation.

The stain was introduced into the whole brain by using a novel method described in [27] that includes perfusion of phosphate-buffered saline, followed by phosphate-buffered paraformaldehyde, and then immersion in a solution of thionin dye. Ultimately, the brain was embedded in araldite plastic and imaged by the KESM using a Nikon Fluor 10$\times$ objective with a 0.3 numerical aperture [27]. The KESM has a diamond knife which was used to cut 1 μm thick sections of the tissue using lateral sectioning. Each section was imaged so as to eventually image the entire brain. The

KESM produces volumetric data, making it possible to locate and reconstruct cells in 3-D.

While the KESM images the whole brain, the data analyzed in this work are sampled from the somatosensory cortex in the rat brain and parts of the basal ganglia and thalamus in the mouse brain.

3.2 Approach for Rat Brain Data

This subsection describes the rat brain data set used and the approach proposed for detecting cells in it. A cascade of random forests performs the preliminary cell detection. Mean shift clustering is then applied for refinement of the results. This sequence of steps is shown in Figure 3.1.

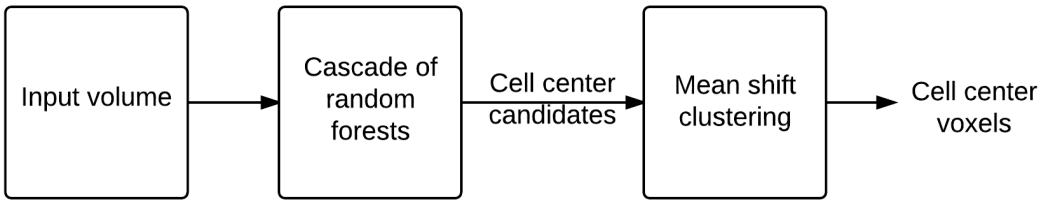


Figure 3.1: Cell detection method for the rat brain data.

3.2.1 Overview of Data

The data under consideration are sampled from the somatosensory cortex of a rat brain. Voxel size is the standard KESM resolution: 0.6 μm $\times$ 0.7 μm $\times$ 1.0 μm. Cell bodies of neurons (the figure on page 22 shows an example) appear as irregular elliptical structures with a dark boundary and a lighter interior region. A nucleolus is typically visible as a dark speckle near the center of a cell body. In this data set,

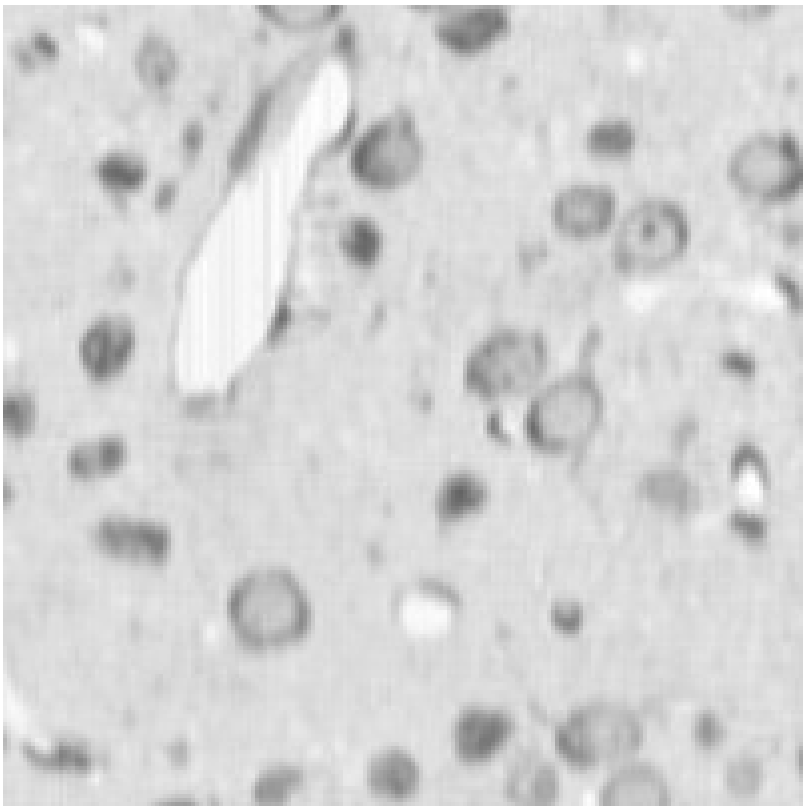


Figure 3.2: Sample 120 μm wide image slice from the rat brain data.

the cell bodies have a diameter of around 20–27 voxels in the x and y dimensions and 10–15 voxels in the z dimension. The variation between the axes is due to the unequal physical dimensions of voxels along them.

A sample 200 pixels $\times$ 200 pixels image (approximately 120 μm wide) can be seen in Figure 3.2. A neuronal cell body from this data set is shown in Figure 3.3. Cells are relatively large and not very densely packed, and objects not of interest such as blood vessels (which are nearly white blobs) are clearly distinguishable. This makes the task of identifying cells easy for a human expert. However, cell boundaries often appear incomplete, and the contrast between cell bodies and the background is low. This poses a challenge for automated cell detection methods.

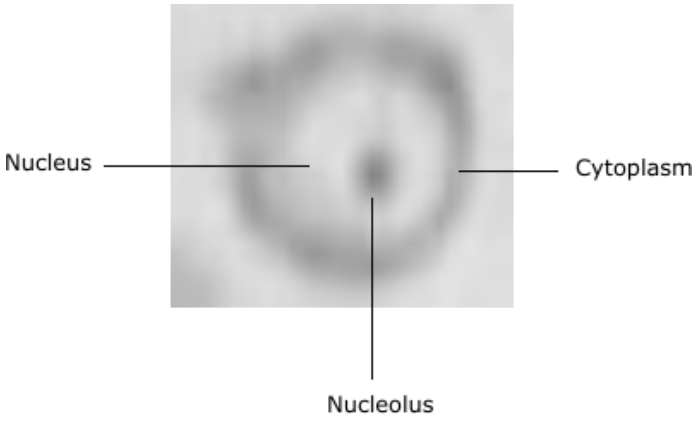


Figure 3.3: A neuronal cell body in the rat brain data.

Apart from occasional low contrast, the images are generally of good quality. They can be further enhanced for downstream processing. It is to be noted that enhancement is not mandatory for the proposed detection approach, but can yield better results at the expense of processing time.

3.2.1.1 Training Data

Three volumes of size 200 voxels $\times$ 200 voxels $\times$ 100 voxels were used as training data for the cascade. The volumes have complementary characteristics in terms of cell size and density. Fiji [35], an image analysis platform, was used to mark the centers of all cells in the three volumes. 396, 182, and 414 cell centers were marked in total, respectively.

To add robustness to orientation, more training data were derived by flipping the original volumes about the coordinate axes. Per volume, three derived volumes are obtained, corresponding to the three axes. The cell center coordinates were also transformed to match the new orientations. Twelve volumes were therefore used for training (three original volumes + 3×3 derived volumes).

3.2.2 Preliminary Seed Detection

The goal of this cell detection method is to identify a unique center for each cell in a given subvolume. It is more convenient to break this down into two stages:

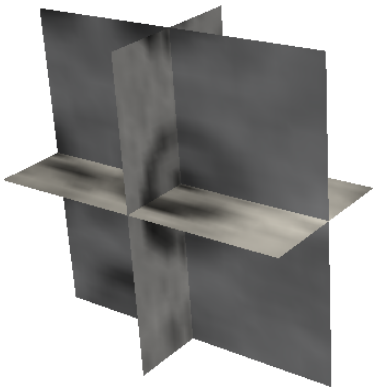
1. Identification of several cell center candidates per cell (seed detection), and
2. Refinement of these candidates to get one unique center per cell (seed refinement).

For seed detection, orthogonal cross-sections of a small subvolume around each voxel was used to classify it as a cell center or not. These cross-sections consist of three 2-D image *windows* or *patches* centered at that voxel.

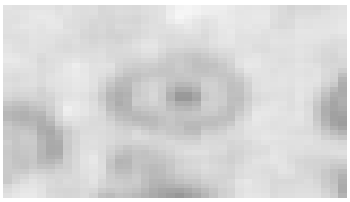
3.2.2.1 Orthogonal Cross Sections

It is common in object detection to classify a pixel by examining a region of the image around it, called a *window* or *patch*. For 2-D images, windows are usually chosen to be rectangular and as big as the largest target object. The *sliding window method* is a popular technique based on this principle. It involves scanning each pixel and extracting a window of predetermined size centered at the pixel. Based on the appearance of its extracted patch, a pixel is classified as a positive or negative pixel.

The natural extension to 3-D images would be to use cuboidal instead of rectangular windows. However, such windows are extremely computationally expensive to extract for each of the possibly millions of voxels in the volume. As an alternative, the proposed method only extracts the three cross sections (three 2-D rectangular windows) of the cuboid centered at a voxel. These are shown in Figure 3.4. They capture the 3-D appearance of the neighborhood reasonably well and are far more computationally efficient to extract.

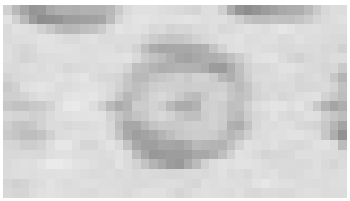


(a) Cross sections



(b) xy plane

(c) yz plane



(d) zx plane

Figure 3.4: Orthogonal cross sections of a cell. (b), (c), (d) Views of the cross sections.

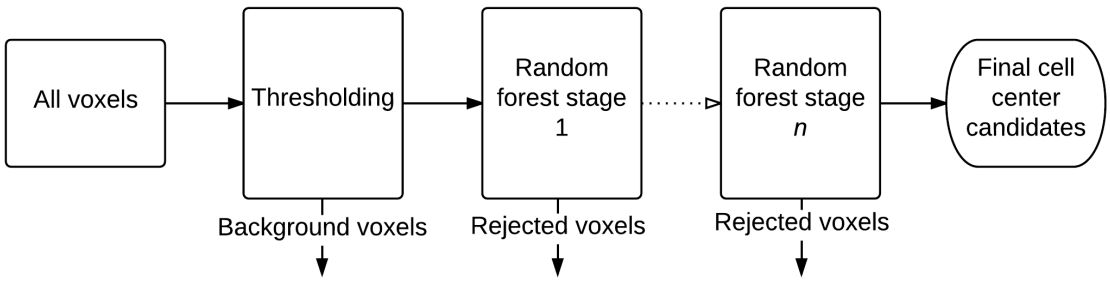
Orthogonal cross sections are represented in this text as (l_x, l_y, l_z) where l_x, l_y, and l_z are the lengths along the x, y, and z axes. Without removing duplicates, the number of voxels in these cross sections is $l_x \times l_y + l_y \times l_z + l_z \times l_x$. The entire cuboid, on the other hand, would contain $l_x \times l_y \times l_z$ voxels.

Based on typical cell diameters mentioned earlier, a cell in the rat brain data would fit in a 3-D cuboidal window with total length of 27 voxels in the x and y dimensions and 15 voxels in the z dimension. This would require extracting $27 \times 27 \times 15 = 10935$ voxels for each voxel in the image. Using only the cross sections reduces this number to $27 \times 27 + 27 \times 15 + 15 \times 27 = 1539$ voxels without removing redundancies.

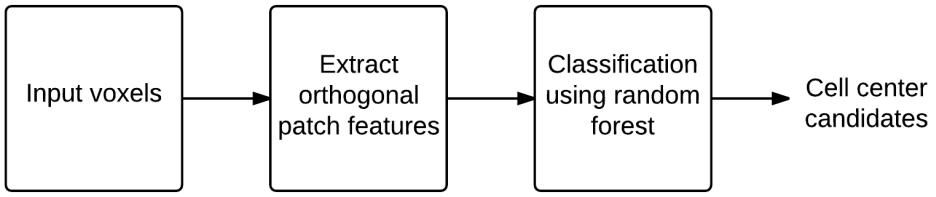
Intensities of voxels in these cross sections are used as features for classifiers in the cascade.

3.2.2.2 Classifier Cascade

To eliminate unnecessary computation, the proposed seed detection approach uses small windows to eliminate obvious portions of the image such as cell boundaries and vessels, and gradually increases the window size to process the remaining portions until a satisfactory set of cell center candidates are obtained. This is implemented as a cascade of classifiers.



(a) Steps in the seed detection cascade



(b) Operations in a cascade stage

Figure 3.5: Cascade for preliminary seed detection. Each stage only operates on voxels accepted by its predecessor.

Cascades of classifiers for object detection consist of a sequence of classifiers where

each classifier updates the predictions made by its predecessors (see Figure 3.5). Weak learners usually form the initial stages of the cascade. They quickly reject large fractions of the data using a small number of features while retaining all the possible matches. Later stages improve on their results by using more sophisticated and expensive features. Rather than computing expensive features such as large windows for all pixels in an image, they are computed only for those pixels that have passed through earlier stages.

The proposed approach uses a cascade of random forest classifiers. Each stage receives as input a set of cell center candidates. The classifier at that stage eliminates less likely candidates from the set; the updated set of cell center candidates is then passed to the next stage and so on until a final set of candidates is obtained from the last stage. Windows used in this cascade start off small in the initial stages and grow until a maximum size is reached at the final stage.

A brief description of random forests is provided next, followed by the strategies for training the cascade and determining the window sizes to be used.

3.2.2.3 Random Forests

Random forests are predictors consisting of multiple decision trees whose predictions are combined to obtain the overall output [4]. Each tree in this ensemble is trained independently on a random sample of the data. At each node of a tree, a small random subset (of fixed size) of the features is used to split the node. The predictions of each tree are usually combined through averaging or by choosing the prediction with the most votes.

Many advantages are offered by random forests over conventional learning algorithms. They can efficiently handle large data sets of high dimensionality and do not overfit [4]. Random forests are capable of performing classification (assigning class

labels) or regression (predicting continuous values). Their predictive performance has been shown to be at par with state of the art supervised learning algorithms such as SVMs and boosted trees [6]. Importantly, they are known to perform well at visual recognition tasks [2], [17], [36]. Finally, they are fast to train and test, and can be efficiently parallelized. Random forests have been implemented both on multi-core CPUs as well as on GPUs [39], [42]. These characteristics make them particularly suitable for the cell detection task as the feature vectors are expected to be high dimensional, and the data set for which prediction needs to be performed is large.

The design of the cascade of random forest classifiers used in the proposed detection method is discussed below.

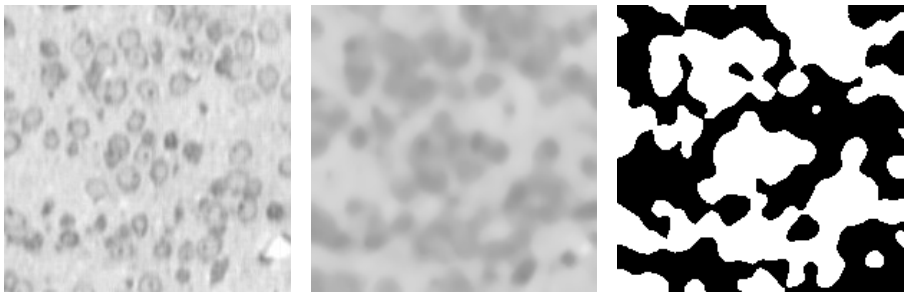
3.2.2.4 Design of the Cascade

- Thresholding

  A coarse thresholding (see Figure 3.6) is performed as the first operation in the cascade. It is not an essential step but provides a speedup by discarding a large fraction of the voxels in a volume (up to half) as the background.

  The volume is eroded using a 3-D ball structuring element with radius $(2, 2, 1)$. Grayscale erosion darkens parts of the cell bodies and also regions surrounding the cells. The eroded image is downsampled by $2X$ and then smoothed using a median filter with radius $(3, 3, 2)$. This produces smooth cell bodies as dark as cell boundaries. This volume is then upsampled back to the original size and thresholded using Otsu's method [31].

  White regions in the thresholded image contain cell bodies (and parts of the background). Black regions contain a significant part of the background and can be quickly discarded by the cascade when making predictions.

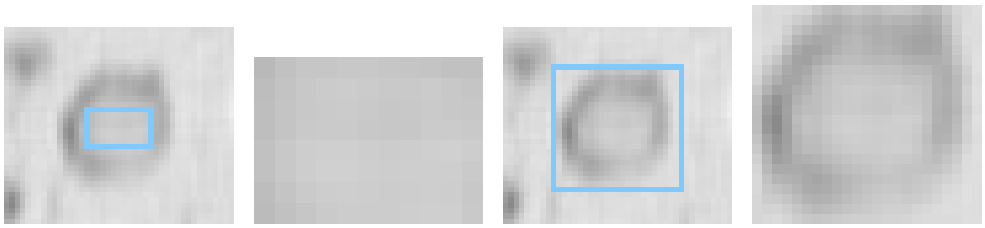


(a) Original (b) Darkened, smoothed (c) Thresholded result

Figure 3.6: Coarse thresholding stage. White pixels in (c) represent the foreground. Cell contours are not preserved, but almost all of the cell bodies are in the foreground. The black background pixels are not processed further by the cascade.

- Random Forest Stages

  The fundamental choice for each cascade stage is the size of the window or patch it uses to classify an input voxel. The window must be at least as large as the largest expected cell, otherwise some cells may not be identified. This is illustrated in Figure 3.7.



(a) (b) (c) (d)

Figure 3.7: Effect of window size being smaller or larger than target objects. (a) A cell with a small window (blue rectangle) at its center. (b) Region inside the small window, which is indistinguishable from a background patch. (c) The cell with a larger window at its center. (d) Region inside the large window, which can now clearly be classified as a cell.

(a) (b) (c) (d)

Figure 3.8: Thin windows for identifying cell bodies. A horizontal patch around the center is shown in (b) and a vertical patch in (d). They can be used to eliminate much of the background whose patches are largely uniform.

Window dimensions of $27 \times 27 \times 15$ are likely to be sufficient for identifying cell centers in this data. Cell bodies, however, can generally be detected by long, thin windows. A horizontal $27 \times 1 \times 1$ window around a cell center (Figure 3.8a) would run from the left cell boundary (dark), through the cell body (light), and up to the right boundary (dark). This dark–light–dark pattern (Figure 3.8b) would indicate the presence of a cell, although it cannot be used to localize the cell center vertically. Due to a cell's ring shape, a similar dark–light–dark pattern would occur along the vertical with a $1 \times 27 \times 1$ window (Figure 3.8d). Using such windows sequentially, each time filtering detections from the previous window, allows a cell center to eventually be localized accurately and more efficiently than using a large $27 \times 27 \times 15$ window.

Accordingly, cross sections of the following windows are used in the five cascade stages: *i*) $1 \times 27 \times 1$, *ii*) $27 \times 1 \times 1$, *iii*) $27 \times 27 \times 1$, *iv*) $27 \times 27 \times 5$, and *v*) $27 \times 27 \times 13$.

3.2.2.5 Training the Cascade

Cascades for object detection are trained with the emphasis that initial stages produce no false negatives (rejecting a foreground pixel) and the latter stages produce

no false positives (accepting a background pixel). This requires stages to be tuned individually for best performance [44]. In the proposed method, a simpler approach is taken where the stages are trained uniformly.

Each stage in the cascade is biased to accept a voxel as a cell center rather than reject it. The expectation is that this will result in a larger number of voxels near cell centers being accepted. The refinement step is capable of handling this increase.

Biasing is done by sampling the training data so that cell centers are overrepresented. The cell center class sampled consists of all the marked cell centers. Non-centers are sampled in three equal-sized chunks of voxels based on distance from the nearest center, d: *i*) $3 \leq d < 5$, *ii*) $5 \leq d < 10$, and *iii*) $10 \leq d < \infty$. The size of each chunk is determined by enforcing the total number of cell centers and the total number of non-cell centers sampled to be equal. Sampling is done independently for each training volume.

All stages are trained using the same data. A stage extracts patches for each training voxel according to its defined window size. If the window extends beyond the volume bounds, the voxel is not used for training by that stage. This can cause some variation in the training data between the stages. Stages are trained independently using the feature vectors that are formed by flattening the extracted patches.

Parameters of the random forest classifier are identical for all stages. 30 classification trees are grown with no depth limit for each forest. The maximum number of features used to split a tree node is $\sqrt{\text{number of features}}$, and 4 samples are required to split internal nodes. The random forest implementation used is the RandomForestClassifier class from scikit-learn [32].

The optional thresholding stage does not participate in training.

3.2.2.6 Prediction Using the Cascade

The test volume presented to the cascade is first thresholded, and foreground voxels are sent to the next stage. Each stage uses its trained random forest classifier to accept or reject a voxel that was accepted by its predecessor. Voxels whose windows would be outside the volume bounds are ignored. Voxels accepted as cell centers by the final stage are considered as the cell center candidates output by the cascade.

3.2.3 Seed Refinement

To reduce the cell center candidates output from the cascade to one candidate per cell, a refinement step is required. The cascade in itself cannot reliably predict one candidate voxel per cell as:

1. The uniqueness condition is hard to directly embed into the classification problem, and

2. Identifying a unique voxel as a cell center needs at the same time its immediate neighbors to be rejected. The choice of features in this approach make this unlikely: windows around a voxel and those around its neighbors are too similar for a classifier to discriminate between them.

Candidates output from the cascade form small clusters near cell centers. Cell centers can be localized by finding the centers of these k clusters. The refinement task is thus reduced to identifying these clusters and replacing them with their k centers.

As the number of clusters k is unknown, it must first be determined before using an algorithm such as k-means, or a clustering algorithm that does not require this knowledge must be used. The proposed refinement approach uses one such algorithm

called *mean shift clustering* that has been successful in computer vision problems. An overview of the mean shift algorithm is provided below.

3.2.3.1 Mean Shift Clustering

Mean shift [7] is a non-parametric density-based clustering procedure. It finds modes on the density surface of the data through steepest ascent.

Density estimation is typically done by radially symmetric kernels such as flat kernels or Gaussian kernels [11]. These kernels are defined by a bandwidth parameter h. The choice of kernel determines how the density is computed at each data point. Points where the gradient of the estimated density is zero are modes of the density surface.

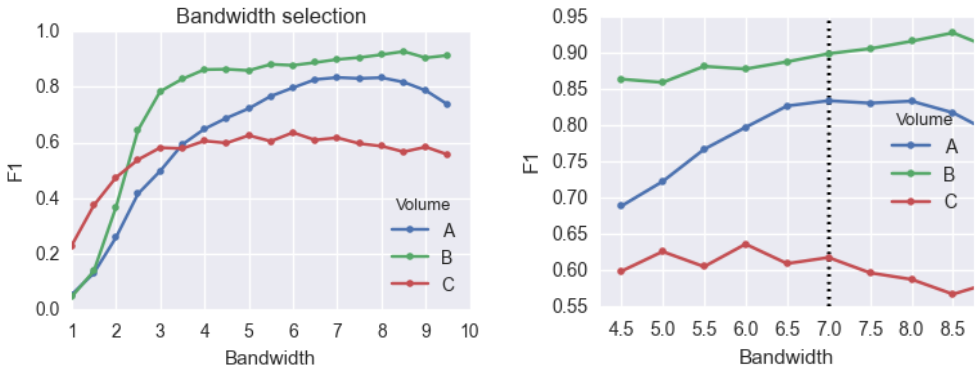
Mean shift is a technique to find these modes and is inherently non-parametric, but requires the kernel to be specified. Essentially, it works by applying the kernel over each data point and moving it along the local gradient. This is done iteratively until convergence to points with zero gradient.

Applications of mean shift to computer vision include image segmentation, smoothing, tracking, and clustering [11]. The performance of mean shift in these applications is strongly influenced by the kernel bandwidth parameter [11]. Automatic bandwidth selection strategies are described in [12]. The proposed technique is a simpler approach similar to cross-validation.

3.2.3.2 Bandwidth Selection

The cascade is trained on two of the three training volumes (A, B, and C) and is used to predict cell center candidates in the other. Mean shift is performed on these candidates by varying the bandwidth from 1 to 9.5 in increments of 0.5. This is repeated with the roles of the training volumes interchanged. The value that leads

to the best detection accuracy scores (see Section 3.4) is used as the bandwidth for novel inputs.



(a) Dependence of F1 score on bandwidth (b) Selecting the best bandwidth

Figure 3.9: Bandwidth selection for the kernel used in mean shift. The bandwidth is selected based on the best detection accuracy (F1 scores) for the training volumes. (a) F1 scores for all bandwidths considered. (b) Narrowed to bandwidths in [4.5, 9] where the highest F1 scores are achieved. A good value of bandwidth is 7.

Figure 3.9 shows the detection accuracy with different choices of bandwidth for the three volumes. A bandwidth of 7 yields a good balance of performance for all the volumes. This value is chosen as the bandwidth for the detection method.

3.3 Approach for Mouse Brain Data

Images from the mouse brain do not look much like the rat brain images presented so far. Cell bodies in these images are smaller and more dense than in the rat brain data. While the cascade method described can be extended to different data sets, the high density and small size of cells in the mouse data are problematic. For a very small cell, the cascade can be prone to eliminating all cell center candidates by the

last stage (for a larger cell size, it much more likely that at least one will survive). High cell density makes it harder for mean shift to separate nearby cells.

However, random forests can perform well even for mouse brain data with a few modifications. In the method proposed, a single random forest predictor is used instead of a cascade, and it is designed to predict the likelihood of a voxel being a cell center. Local maxima of these likelihoods correspond to predicted cell centers. Laplacian features are used in place of intensity features. Lastly, to speed up computation, detection is performed on the volume downsampled by $2X$. This is summarized in Figure 3.10.

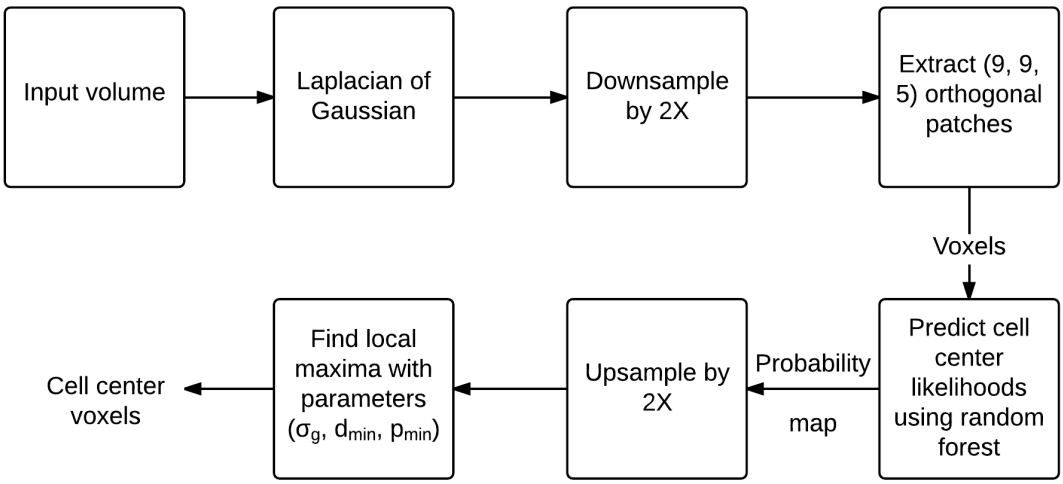


Figure 3.10: Cell detection method for the mouse brain data.

3.3.1 Overview of Data

Mouse brain data sampled from the thalamus and basal ganglia are used to develop and test the proposed method. The resolution of 0.6 μm $\times$ 0.7 μm $\times$ 1.0 μm

remains the same. An example 256 pixels $\times$ 256 pixels image slice (approximately 154 μm wide) from this data set is shown in Figure 3.11. A neuronal cell body from this data set and its components can be seen in Figure 3.12.

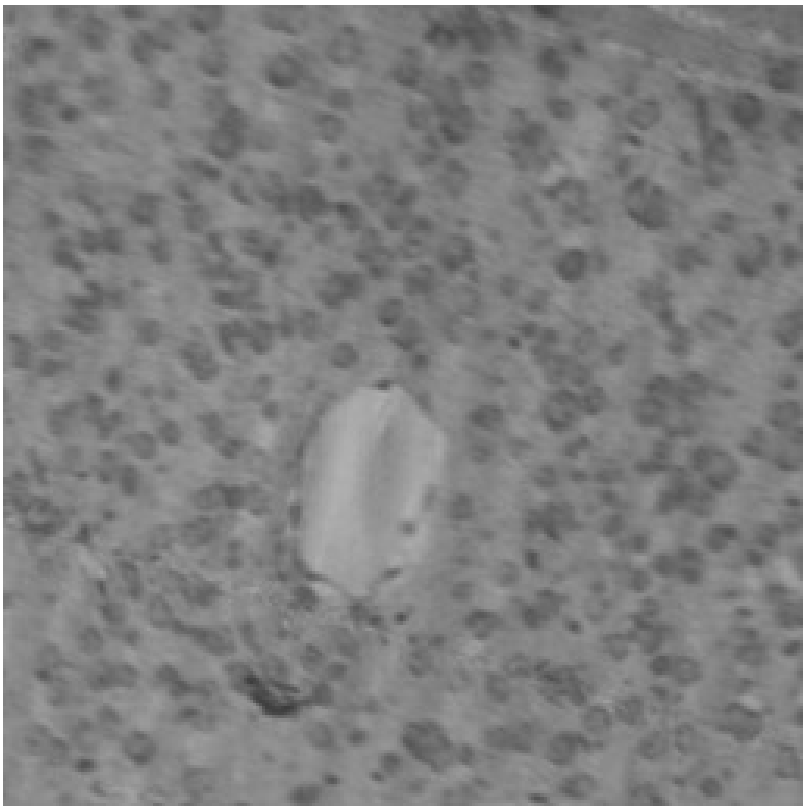


Figure 3.11: Sample 154 μm wide image slice from the mouse brain data.

It is immediately apparent that cells are smaller and far more tightly organized than in the rat brain data. The variation in cell sizes is also more pronounced. Cell diameters can range from about 7 voxels to 15 voxels within small regions.

Besides these differences which originate from biological reasons, another set of differences can be seen due to errors in the imaging process. The four most prominent effects are listed below.

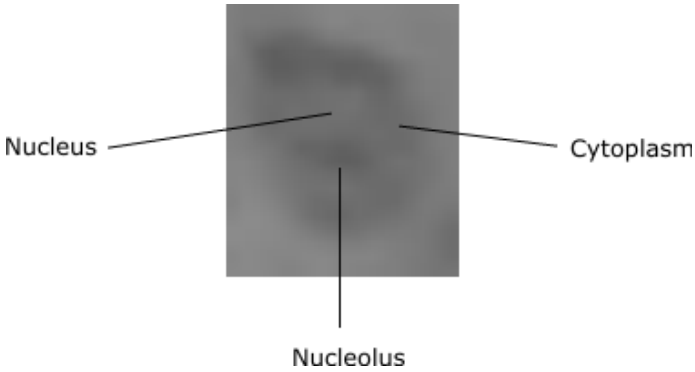


Figure 3.12: A neuronal cell body in the mouse brain data.

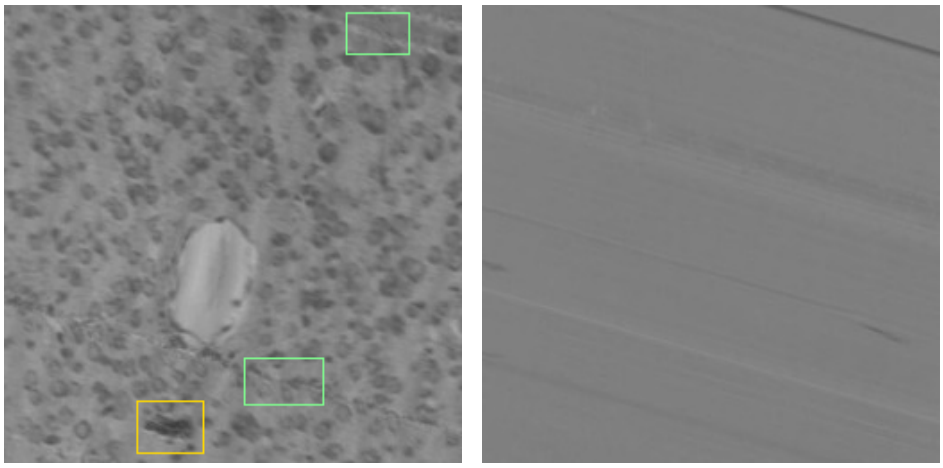
3.3.1.1 Imaging Errors

Loss of focus Focus appears to change between images along the imaging axis (z axis), meaning some images are well focused and the cell contours are clearly visible, and some images are out of focus. These images appear blurred and have low contrast between cell bodies and the background. Especially difficult even manually is recognizing individual cells in cell clusters.

Diagonal stripes Diagonal stripes (see Figure 3.13a) caused by knife chatter extend across the width of the images, persisting along the imaging axis. Cells along the stripes are occluded and sometimes a single cell may be mistaken for two cells due to the separation caused by a stripe.

Dark noisy patches Dark patches about 10 to 15 pixels wide occur unpredictably throughout the data set (see Figure 3.13a). These significantly degrade the image quality and potentially detection performance as the predictor cannot reliably estimate the cell center likelihood for these pixels.

Complete image obscuring Groups of 5 to 10 consecutive (depthwise) image slices exist where the images are almost entirely obscured by imaging errors



(a) Stripes and patches (b) Obscured image

Figure 3.13: Noise in the mouse brain data. (a) Diagonal stripes (green rectangles) and a dark patch (yellow rectangle). These are very common. (b) A completely obscured image slice. This is relatively uncommon but very damaging.

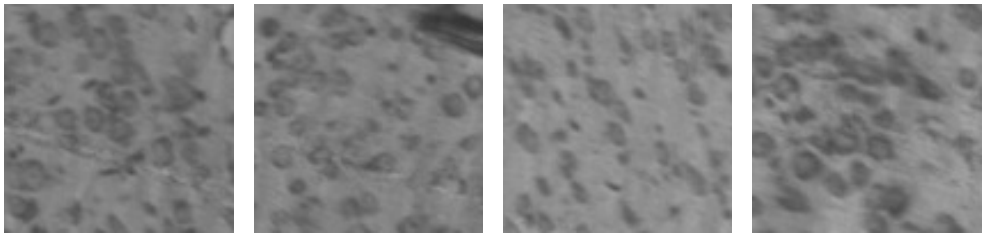
(see Figure 3.13b). Such images assume a uniform grayish appearance where no details are visible.

Of these imaging errors, image obscuring is the most severe to the extent that it is catastrophic. The proposed detection method is not designed to handle this kind of noise, so it is ensured that this noise is absent from the training and testing data. Effects of the other three errors can be diminished by preprocessing the images, but this is not done in this work.

3.3.1.2 Training Data

Seven 100 voxels × 100 voxels × 25 voxels volumes with manually labeled cell centers form the training set. They are sampled to incorporate cells with diverse appearances in terms of size, orientation, density, and contrast. Sample images from these volumes are shown in Figure 3.14.

Labeled cell centers in each training volume ranged from 90 to 103 in number, with the total count being 696. This suggests a more even cell density than the rat data where the variation between training volumes was 182 to 396.



(a) (b) (c) (d)

Figure 3.14: Sample images from the mouse brain training set.

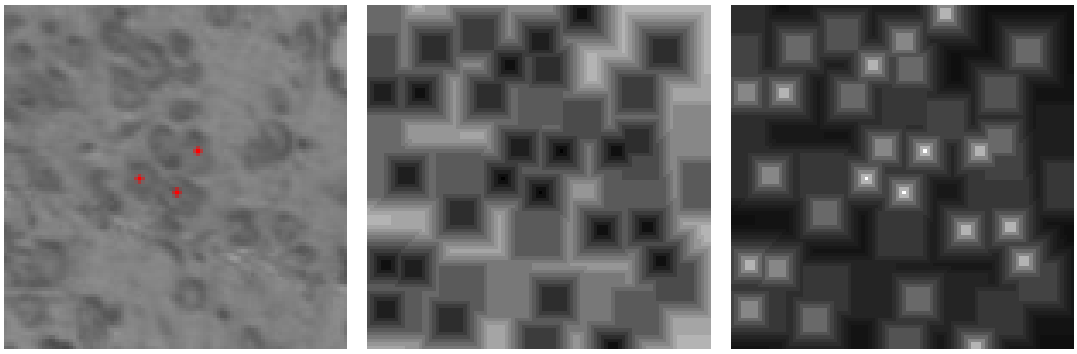
3.3.2 Training the Predictor

Cell detection in the mouse brain data begins with a random forest that predicts the probability of an input voxel being a cell center. This predictor operates as a regressor, as opposed to classifiers used for the rat data.

Probability values are assigned to voxels in a training volume using a kind of inverted distance map. The distance d of each voxel from its nearest center is found (Figure 3.15b). This distance is converted to a proximity value p in the range $[0, 1]$ by the equation

$$p = \frac{1}{1 + \frac{d}{3}}. \tag{3.1}$$

p represents the proximity of a voxel to its nearest cell center; at cell centers, $p = 1$ (Figure 3.15c). Alternate definitions of p also tend to work well as long as p decreases continuously with distance. d used here is the Chebyshev distance (or chessboard distance). Euclidean distance can also be used instead.



(a) Image, cell centers in red (b) Distance map (c) Proximity map

Figure 3.15: Proximity map on which the random forest is trained. (a) is a training image slice with cell centers shown as red dots. Brighter pixels in the proximity map (c) indicate higher proximity to a cell center, p.

50,000 voxels and their proximity values are randomly sampled from the $2X$ downsampled training volumes and used as the training set. Downsampling by $2X$ in each dimension speeds up the detection process greatly as it reduces the total number of voxels by a factor of 8. It also reduces the size of the cross sections that need to be extracted by $2X$ in each dimension. Therefore, orthogonal cross sections of a small $9 \times 9 \times 5$ window are used to extract features for each voxel.

Downsampling also causes a loss of information, especially fine details, from the volume. A loss in accuracy is to be expected as these details can be necessary to detect a cell center. Depending on whether speed or accuracy is more important for a given analysis task, downsampling should be enabled or disabled. In this thesis, it is enabled due to the $8X$ speedup it provides.

A random forest regressor consisting of 50 trees with no depth limit, requiring 2 samples to split internal nodes, and using a maximum of $\sqrt{\text{number of features}}$ features to split a tree node is trained on the feature vectors. Proximity values are used as labels. The random forest implementation used is the RandomForest-

Regressor class from scikit-learn.

For a novel input volume, features for each voxel are extracted from the down-sampled volume and presented to the trained random forest. The random forest predicts the voxel's proximity value, which can be interpreted as the probability of the voxel being a cell center. This "probability map" is resampled to the original size and post-processed to find the locations of cell centers.

3.3.3 Identifying Cell Centers

Cell centers are identified by finding the local maxima of the smoothed probability map.

Isolated peaks are first removed from the probability map by Gaussian smoothing with a standard deviation of σ_g. Voxels that have the maximum probability value in a 3-D neighborhood of length d_{min} are found. Those voxels with probability p less than a minimum value p_{min} are discarded. The remaining voxels are used as the cell center locations.

3.3.4 Feature and Parameter Selection

Classifiers in the rat brain method used graylevel intensities (raw pixel values) in orthogonal patches as features. As shown later, these perform adequately. In [19], among the features evaluated for a 2-D detection task using SVMs, a combination of the raw pixel values and Laplacian edge values performed the best. As detection in the mouse brain data is more complex than the rat data, it is worthwhile to perform a similar feature selection step to find other features that may work better.

Raw pixel values, gradient values, and Laplacian values inside orthogonal patches are chosen as candidate features. The gradient and Laplacian values were smoothed by Gaussians of scales 1 and 1.5, respectively. A grid search was used to determine the best features and best parameter values $(\sigma_g, d_{min}, p_{min})$. The search space for

each parameter is chosen as: $\sigma_g \in \{1, 1.5, 2, 2.5\}$, $d_{min} \in \{1, 2, 3, 4, 5\}$, and $\mathrm{p_{min}} \in \{0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1.0\}$.

For each feature type, random forests are trained on all but one of the training volumes and tested on the remaining volume. Cell centers are found using all $(\sigma_g, d_{min}, p_{min})$ in the parameter search space. F1 scores are found for each run by comparing detected cells with the labels (see Section 3.4). This is repeated for all possible combinations of training and test sets.

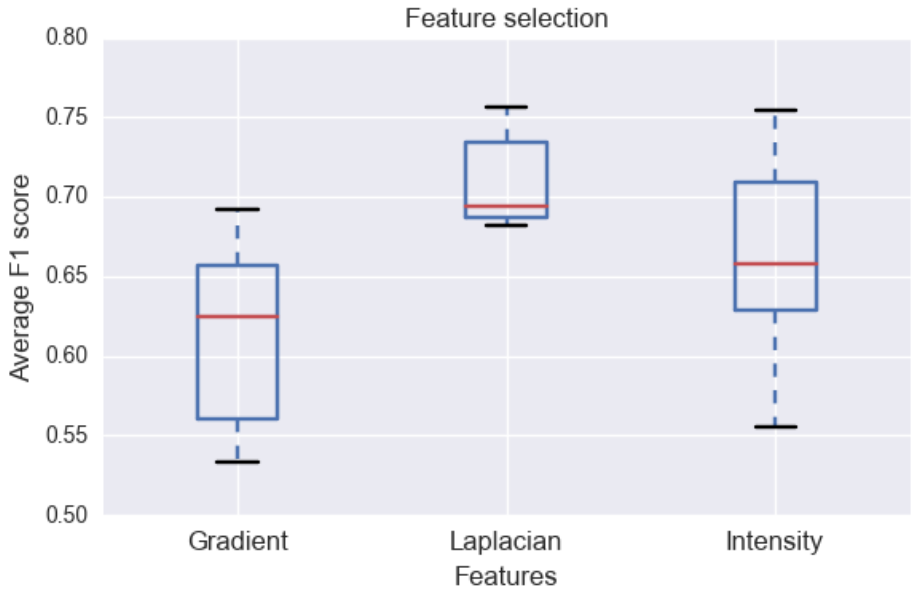


Figure 3.16: Feature selection for the mouse brain method. Sets of six training volumes and one testing volume are created by splitting the seven original training volumes. Detection accuracy is measured for each split for all $(\sigma_g, d_{min}, p_{min})$ parameters and averaged to get a mean F1 score for that split. This box and whiskers plot shows the variation of mean F1 score with the kind of features used by the random forest predictor.

The F1 score averaged over parameter values for each training set/test set combination is plotted in Figure 3.16. Laplacian features are clearly the best-performing

features on average. Intensity features achieve a similar maximum but are unpredictable due to a larger range of variation. Gradient features have a lower maximum and more unpredictability than both. Based on these results, Laplacian features are used instead of intensity features for the mouse data.

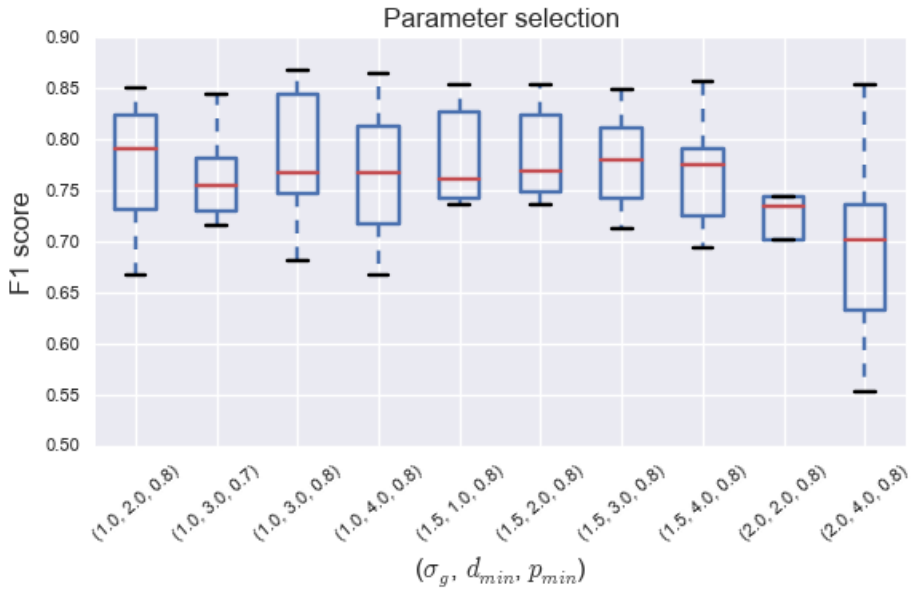


Figure 3.17: Parameter selection for the mouse brain method. The F1 score for each split of the seven training volumes is plotted versus the 10 best $(\sigma_g, d_{min}, p_{min})$ parameters (see Section 3.3.3 for details) using Laplacian features. (1.0, 3.0, 0.8) is selected based on its high performance at the first (lower blue line) and third quartiles (upper blue line), and a high maximum (upper black line).

The performance of different $(\sigma_g, d_{min}, p_{min})$ parameters for Laplacian features on each training set/test set combination is shown in Figure 3.17. Poorly performing combinations are not shown. (1.0, 3.0, 0.8) is chosen due to better performance at the first and third quartiles, and a larger maximum than the others.

In summary, Laplacian features and (1.0, 3.0, 0.8) parameters are used for the mouse brain detection method as they perform the best in the above grid search.

3.4 Evaluation

The proposed cell detection method is evaluated on the basis of accuracy and speed. Speed is an essential metric as the eventual goal is to process large data sets, potentially terabytes in size, in reasonable time.

3.4.1 Detection Accuracy

Detection accuracy is measured by assessing the agreement between the cell centers predicted by the algorithm and the cell centers labeled manually, i.e., the ground truth. In this work, this is done by computing the F1 score, which is an often-used accuracy measure for binary classification problems.

Cell detection can be considered a binary classification problem where the aim is to assign to each voxel one of two labels – cell center ("positive") or not a cell center ("negative"). True positives are those predicted cell centers that are also centers in the ground truth. False positives are those predicted centers that are not centers in the ground truth. True negatives are those voxels predicted as not being cell centers that are not centers in the ground truth. Finally, false negatives are voxels predicted as not being cell centers that are actually cell centers according to the ground truth.

Numerically, voxels that are not cell centers greatly dominate over the cell centers because for every few million voxels, only hundreds will be cell centers. To prevent the skewing of accuracy measures by non-cell centers, the cell detection is reformulated as a one-class classification problem. Here, no true negatives are produced. False negatives still exist when the algorithm misses some cell centers. True and false positives retain their original meanings.

A good detection algorithm must simultaneously maximize two quantities – the fraction of predicted cell centers that are actually cell centers, and the total fraction of cell centers identified. These are related to precision and recall. The F1 score is a

weighted measure of precision and recall, thereby justifying its use as the accuracy measure. It is given by:

$$F_1 = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}} \tag{3.2}$$

or equivalently,

$$F_1 = \frac{2\,TP}{2\,TP + FP + FN}. \tag{3.3}$$

In Equation (3.3), TP is the number of true positives, FP is the number of false positives, and FN is the number of false negatives.

To calculate these values, the correspondence between predicted cell centers and ground truth cell centers must be established. The method used for finding this correspondence is given below.

3.4.2 Matching Predicted Cell Centers with the Ground Truth

A predicted cell center c_p is a true positive if it can be "matched" with a ground truth cell center c_t. The ideal match occurs when the locations of c_p and c_t coincide exactly. This can be extended by adding a tolerance on the separation between c_p and c_t, denoted by d_u. For a match, the separation must not be larger than d_u.

In essence, the correspondence between predicted cell centers P and the ground truth T is found by iteratively matching pairs of points, $c_p \in P$ and $c_t \in T$ such that:

- c_t is the ground truth cell center nearest to c_p,
- c_p is the predicted cell center nearest to c_t, and
- the distance between c_t and c_p is not larger than d_u.

This can be implemented as an iterative procedure that uses trees for efficiently finding mutual nearest neighbors.

The result is the set of matches M,

$$M = \left\{(c_p, c_t) : c_p \in P, c_t \in T, c_p \text{ and } c_t \text{ match}\right\},$$

the set of unmatched predicted cell centers

$$U_p = \left\{c_p : c_p \in P, c_p \text{ does not match any } c_t \in T\right\},$$

and the set of unmatched ground truth cell centers

$$U_t = \left\{c_t : c_t \in T, c_t \text{ does not match any } c_p \in P\right\}.$$

True positives are the predicted cell centers that could be matched to a ground truth cell center: $TP = |M|$. Predicted cell centers that could not be matched form the false positives: $FP = |U_p|$. Ground truth centers that could not be matched are the false negatives: $FN = |U_t|$.

In the evaluation of the proposed detection methods, the separation between points is measured by the Euclidean distance and the tolerance d_u is set to 5 μm.

4. RESULTS

4.1 Methodology

Cell detection methods described so far were evaluated by comparing manually labeled cell centers with automatically detected centers. A set of volumes was sampled, ensuring no overlap with the volumes used for training. The ground truth was established by manually labeling all cell centers in these volumes. These labeled data formed the test set.

Voxels where boundary effects occur were excluded from the evaluation. At these voxels, the orthogonal cross sections would extend beyond the volume bounds. This is why the number of expected cell centers in the tables below is smaller than the total number of labeled cell centers.

The separation tolerance parameter d_u in the point matching procedure was chosen to be 5 μm. This means that if the Euclidean distance between a predicted cell center and a cell center in the ground truth is more than 5 μm, they are not considered to match.

The evaluation results indicate that the proposed methods generally perform well for data that are reasonably similar to the training data. Significant differences, especially in cell size, can hinder detection performance. These findings are explained in more detail below.

4.2 Rat Brain Data

Six volumes (volumes 1–6) of size about 200 voxels $\times$ 200 voxels $\times$ 100 voxels were used as the test data for the rat brain data set. In all, 1540 cell centers were manually labeled in these volumes. The volumes differed substantially in terms of cell size, density, and presence of other structures like blood vessels.

4.2.1 Results

Evaluation metrics for this data set can be seen in Table 4.1. As usual, TP denotes the number of true positives (cells correctly detected); FP is the number of false positives (spurious detections); and FN is the number of false negatives (cells missed). "Expected" and "Predicted" are the number of cells in the ground truth and the number of cells detected, respectively. Apart from volume 1, both precision and recall were found to be about 90% or above.

The *peak performance* is defined in [28] as

$$P_{peak} = \max\left(\frac{TP}{P + FP}\right), \tag{4.1}$$

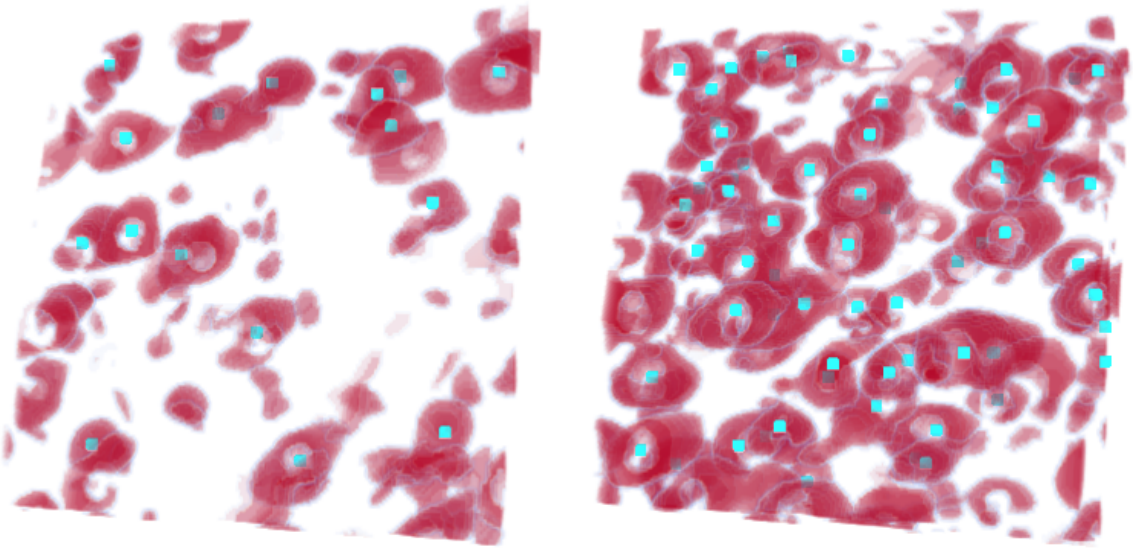
where P is the total number of cell centers in the ground truth. This was calculated to be 84.3% for the proposed method as compared to 77.4% for the method based on artificial neural networks in [28]. It is to be noted that the latter included more testing subvolumes and was designed to be implemented on a GPU.

Volume	Expected	Detected	TP	FP	FN	Precision	Recall	F1
1	78	91	73	18	5	0.802	0.936	0.864
2	170	179	159	20	11	0.888	0.935	0.911
3	161	153	146	7	15	0.954	0.907	0.930
4	220	224	200	24	20	0.893	0.909	0.901
5	228	208	203	5	25	0.976	0.890	0.931
6	138	134	127	7	11	0.948	0.920	0.934

Table 4.1: Detection accuracy for volumes in the rat brain test set.

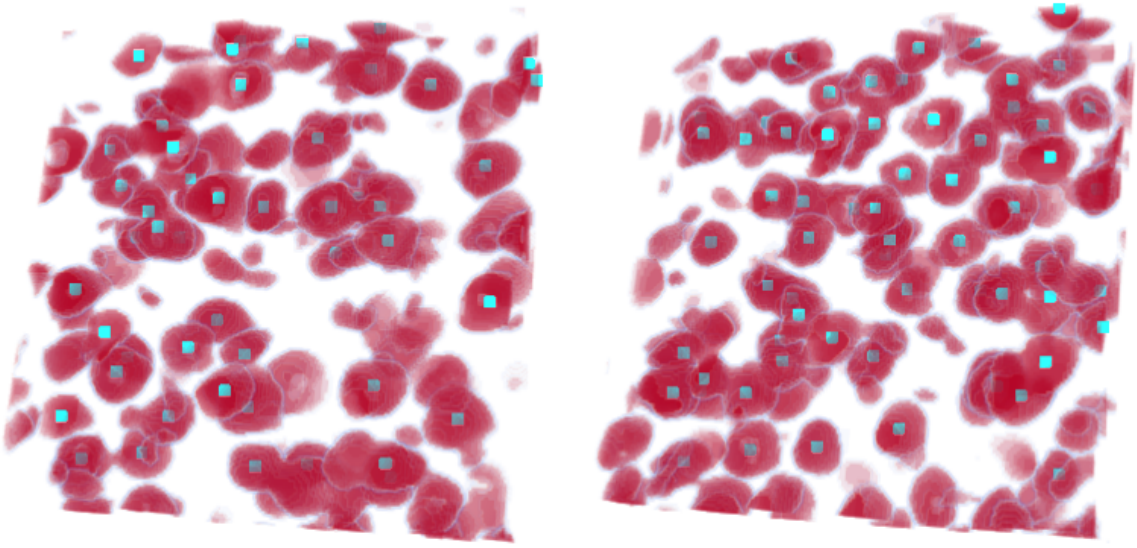
Figures 4.1–4.3 show blocks within the six volumes with the predicted cell centers. These also clearly show the differences in cell sizes and density across the volumes:

volume 1 had large cells with low density; volume 2 had large cells with higher density; volume 3, volume 4, and volume 5 had moderately sized cells with high density; and volume 6 had moderately sized cells with lower density. Cell bodies in these figures (in red) were extracted using a simple smoothing and thresholding method and were therefore only crudely segmented. Large cells like those in volumes 1 and 2 (Figure 4.1) appeared to contain holes and were particularly poorly segmented.



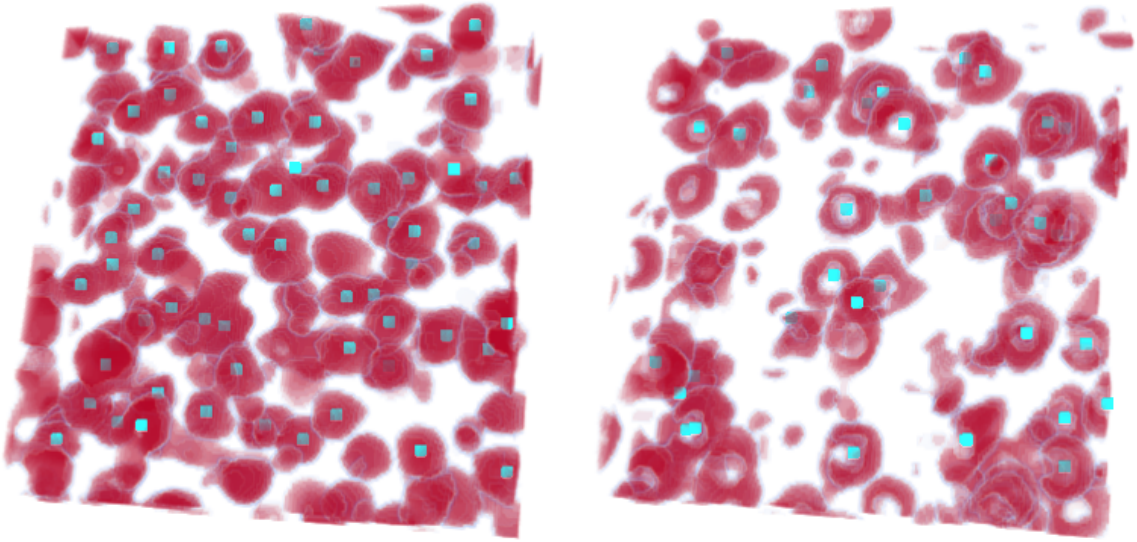
(a) Volume 1 (b) Volume 2

Figure 4.1: Predicted cell centers in blocks of volumes 1 and 2.



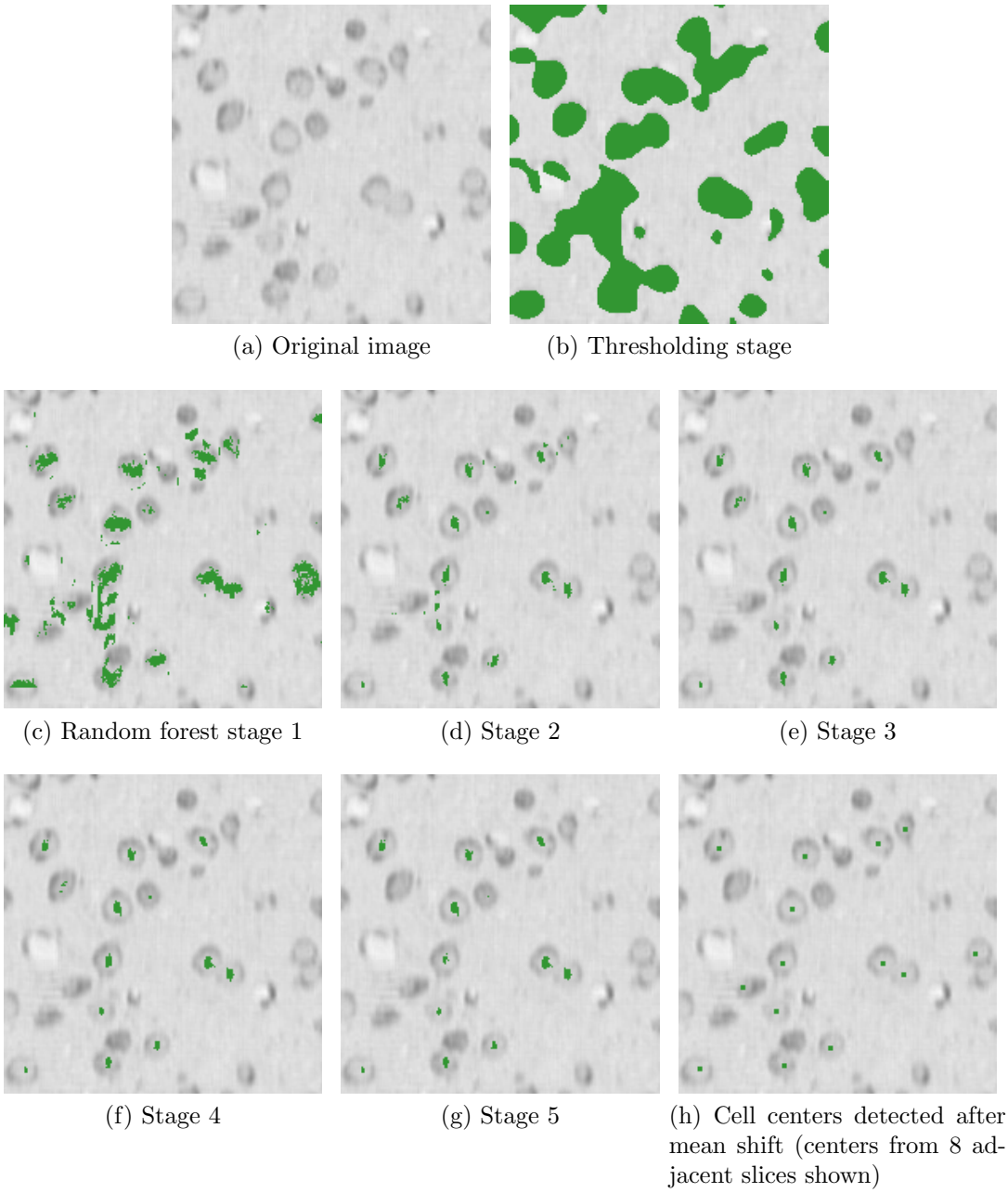
(a) Volume 3 (b) Volume 4

Figure 4.2: Predicted cell centers in blocks of volumes 3 and 4.



(a) Volume 5 (b) Volume 6

Figure 4.3: Predicted cell centers in blocks of volumes 5 and 6.



(a) Original image (b) Thresholding stage

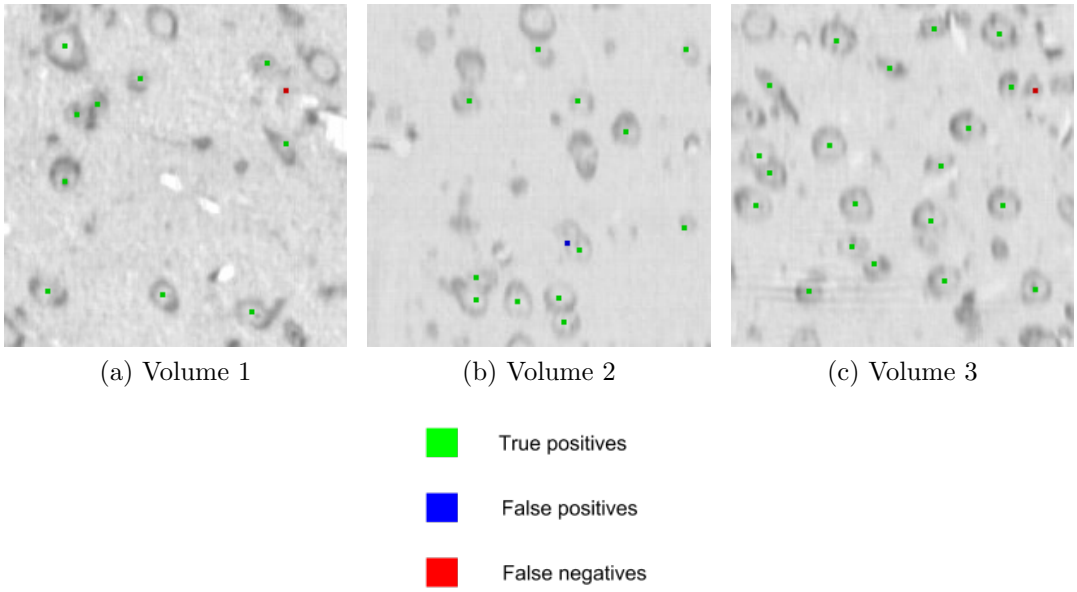
(c) Random forest stage 1 (d) Stage 2 (e) Stage 3

(f) Stage 4 (g) Stage 5 (h) Cell centers detected after mean shift (centers from 8 adjacent slices shown)

Figure 4.4: Candidates at each stage of the cascade in a slice of volume 3. Each stage improves the cell center candidates (in green) output by the previous stage.

The detection process can be tracked by visualizing the cell center candidates at each stage of the cascade. Figure 4.4 shows the stage-by-stage candidate filtering for

a slice of volume 3. Thresholding (Figure 4.4b) rejected obvious background regions. The first random forest stage (Figure 4.4c) used a thin vertical window to filter out candidates from the foreground of the thresholded image. It identified regions near cell centers but, as expected, was unable to horizontally localize cell centers. By the final stage (Figure 4.4g), cell centers were localized accurately enough for mean shift to be applied. Figure 4.4h shows the cell centers found by applying mean shift to the output of the cascade. It includes centers detected within four slices above and below the given slice.



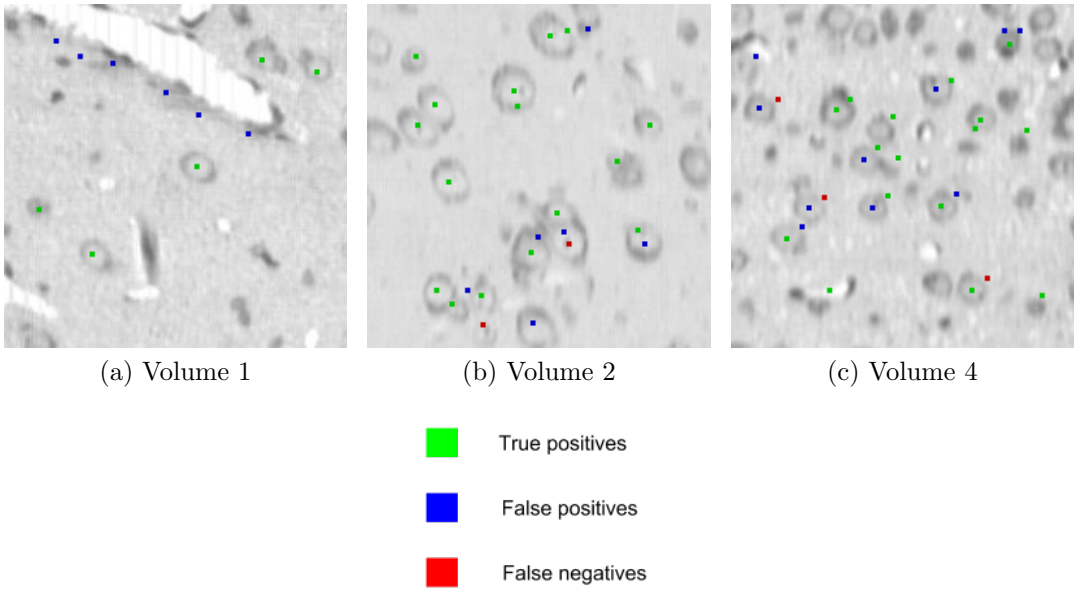
(a) Volume 1 (b) Volume 2 (c) Volume 3

Figure 4.5: Rat brain images with mostly accurate detections.

Figure 4.5 shows some slices for which the detection method exhibited good performance. Cells with different shapes and elongations were handled well. Large cells in Figure 4.5a and smaller cells in the others both seemed to be detected correctly. Further, touching cells were generally correctly separated. Separating touching ob-

jects is a key problem faced by typical segmentation algorithms. With seeds placed accurately at each object, this problem can be alleviated.

The method appeared to be quite robust to variations in image quality. Background regions in each of the slices shown are considerably different, and Figure 4.5c contains dark stripes over a cell near the bottom left. Slices in figures 4.5b and 4.5c also have cells with faint outlines. All these factors did not seem to significantly hinder the detection method's performance at least in the slices shown.



(a) Volume 1 (b) Volume 2 (c) Volume 4

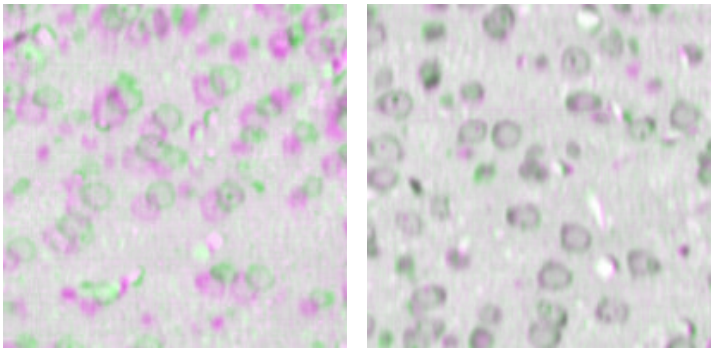
Figure 4.6: Rat brain images with spurious detections (false positives).

Figure 4.6 shows slices with a large number of spurious detections (in blue). In Figure 4.6a, six false positives (of the 18 in the entire volume) occur along the boundary of the long white blood vessel. A straightforward explanation is difficult as no false positives were present near the other vessels in the slice, or in the other

slices shown. One possible factor could be that the endothelial cells that line the blood vessels looked similar enough to the marked cells in the training set.

Spurious detections in Figure 4.6b were made near the periphery of cells that were bigger than those in the training volumes. These cells did not fit completely in the smaller windows used in the cascade, leading to less reliable results.

The numerous false positives in Figure 4.6c (10 out of 24 in the entire volume) were due to a sudden drift in two adjacent slices of the volume, causing misalignment between the slices (shown in Figure 4.7). This imaging error occurred relatively rarely and is not expected to affect much of the rat brain data.

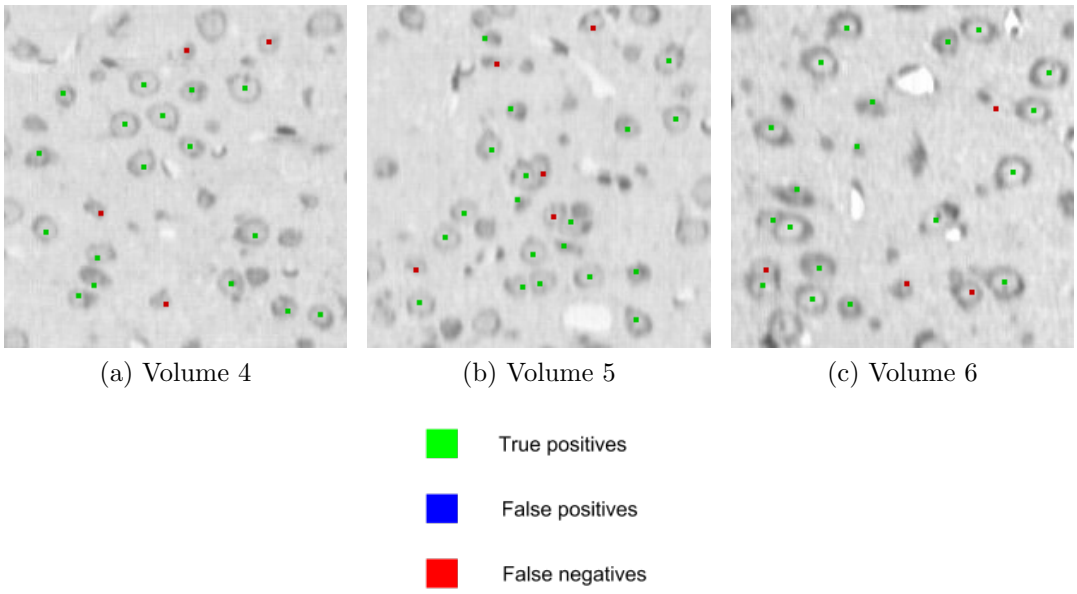


(a) Slices 19 and 20 (b) Slices 1 and 2

Figure 4.7: Errors caused by drift in the rat brain images. (a) Abrupt drift in neighboring slices (colored differently, in magenta and green) of volume 4 that caused detection errors. (b) Slices without drift that change smoothly.

Lastly, slices with cells that were not detected (in red) by the proposed method are shown in Figure 4.8. A majority of these errors could be explained by the size of the undetected cells. The undetected cells were quite small, and it is likely that all cell center candidates were eliminated by the final stage of the cascade. The bandwidth used in the mean shift clustering step is also influential. For clusters of

small cells, a high bandwidth can lead to a single detection near the center of the cluster instead of the centers of the individual cells. This occurred near the center of the slice in Figure 4.8b, where there are two false negatives, and is explained in Figure 4.9.



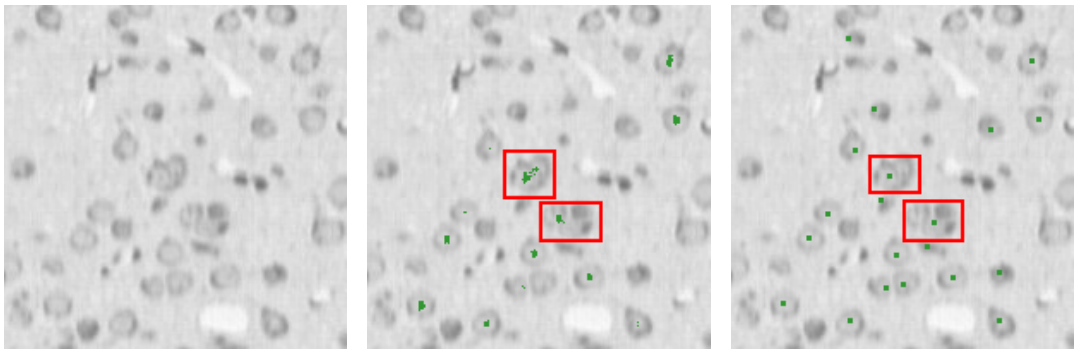
(a) Volume 4 (b) Volume 5 (c) Volume 6

Figure 4.8: Rat brain images with undetected cells (false negatives).

4.2.2 Speed

Speed tests were performed on a laptop running 64-bit Arch Linux. The laptop featured a 2.5 GHz Intel Core i5-3210M processor with 2 physical (4 logical) cores and 3 MB L3 cache, and 6 GB DDR3 SDRAM. Random forests and feature extraction were both set to single-threaded operation.

The time taken for training the cascade for the above runs was between 10 s to 11 s, with a mean of 10.24 s.



(a) Original image (b) Cell center candidates (c) Mean shift (8 adjacent slices)

Figure 4.9: Effect of high bandwidth in a slice of volume 5. (a) Original image. (b) Cell center candidates output from the cascade. (c) Cell centers detected after mean shift using a kernel with bandwidth 7. Centers from 8 adjacent slices are shown. In the red rectangles, mean shift results in one predicted cell center for each cluster of densely packed cells. A lower bandwidth is required to predict centers at individual cells rather than one for the entire cell cluster.

Predicting cell centers on one volume of the test set ($200 \times 200 \times 100$, which is about 2 MB in size) using the cascade took between 11 s and 14 s, with a mean of 11.85 s.

Using multiple random forest or feature extraction threads can provide a further boost to speed. If the volumes are downsampled first (as done in [28], [41]), the speed can improve by an order of magnitude with a small possible loss of accuracy, making it closer to the speed of the ANN method in [28] which had a throughput of 2.19 MB/s using the CPU implementation.

4.3 Mouse Brain Data

A 256 voxels × 256 voxels × 88 voxels volume with 2024 labeled cell centers was used for evaluating detection performance in the mouse brain data.

4.3.1 Results

Table 4.2 summarizes the accuracy of the method on the test volume using parameters found previously in Section 3.3.4. Due to boundary effects, the algorithm was only able to process voxels in a 240 voxels × 240 voxels × 80 voxels subvolume of the original. Voxels outside this subvolume were not considered in the evaluation.

Expected	Detected	TP	FP	FN	Precision	Recall	F1
1662	1521	1352	169	310	0.889	0.813	0.850

Table 4.2: Detection accuracy for the mouse brain test volume. The parameters used were $(\sigma_g, d_{min}, p_{min}) = (1, 3, 0.8)$.

A visualization of the cell centers detected in a block of the volume can be seen in Figure 4.10. A simple but imperfect smoothing and thresholding method was used to extract cell bodies (in red). It was poorer at separating them than in the rat data due to the higher density of cells in the mouse brain data.

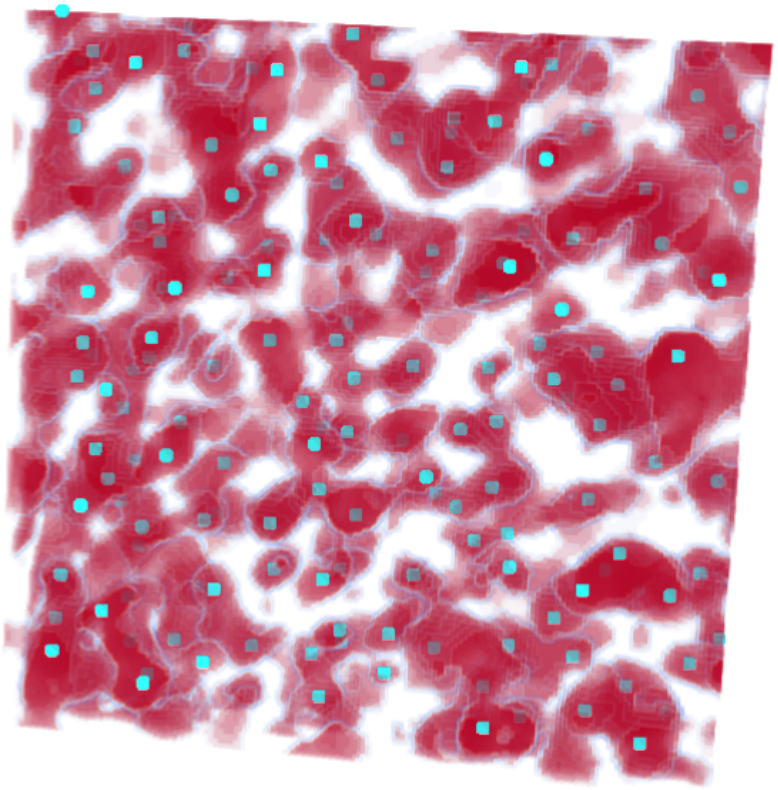
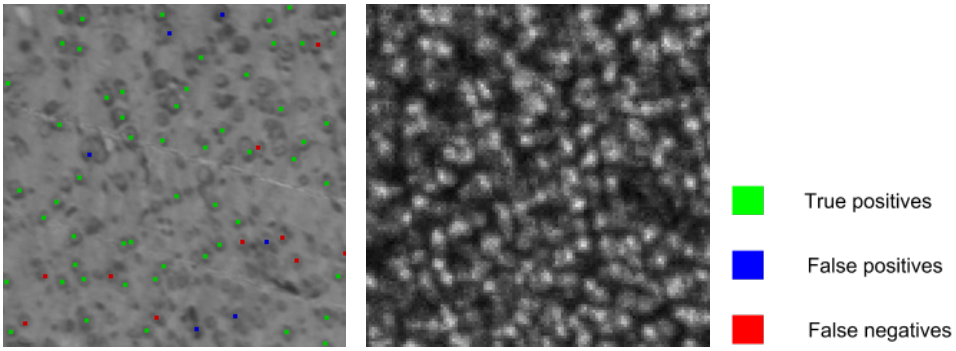


Figure 4.10: Predicted cell centers in a block of the mouse brain test volume.

Figure 4.11a shows the detected cell centers for one slice of the volume. Bright "peaks" can be seen in Figure 4.11b, which is the probability map, near the centers of cell bodies.



(a) Image slice (b) Probability map

Figure 4.11: Probability map and detections in a mouse brain image. (a) A slice from the test volume with detections (see legend). (b) Probability map generated by the random forest. Bright pixels indicate high probability.

A larger view of another slice is shown in Figure 4.12. As expected, a majority of the cells were detected correctly (in green). Low contrast was overcome reasonably well. Small cells were detected with few false positives. Nearby cells were also successfully separated on most occasions.

A sizeable number of errors were also made. The larger error rate was not surprising since the data were of poorer quality than the rat data. Possible deficiencies in the method itself also contributed to errors. Two commonly observed causes of failure were:

Large cells Centers of large cells were sometimes poorly localized (yellow rectangles in Figure 4.12); they were predicted closer to the periphery rather than the

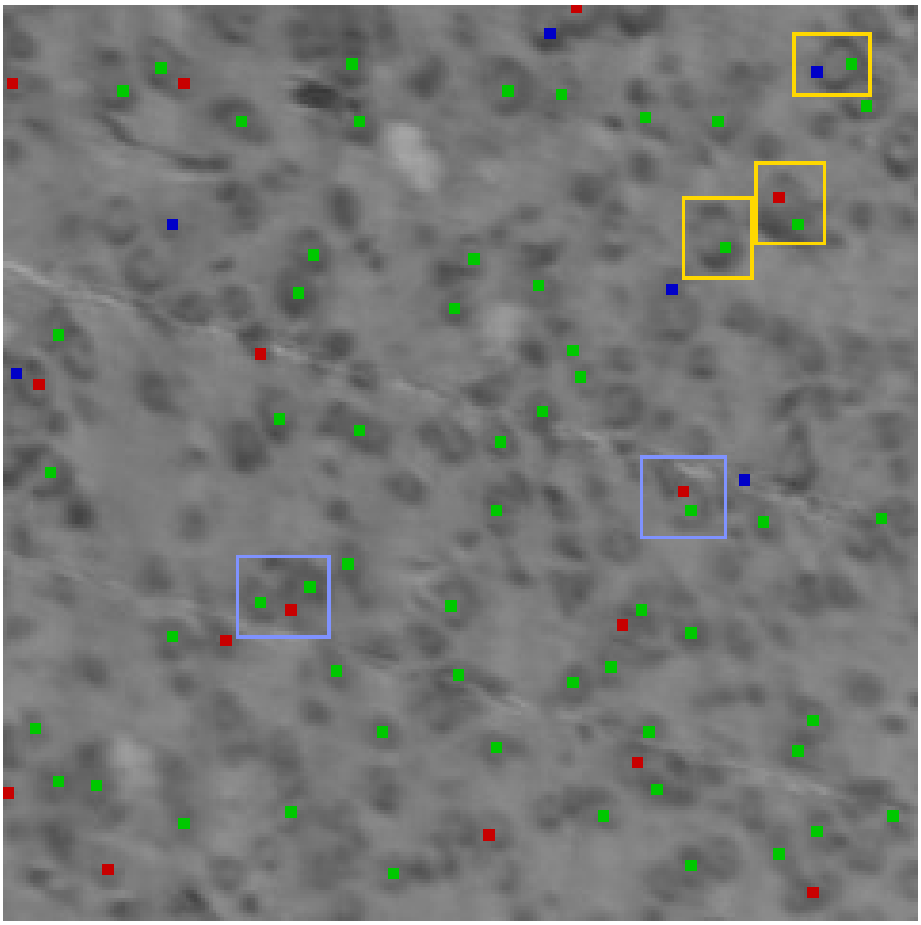


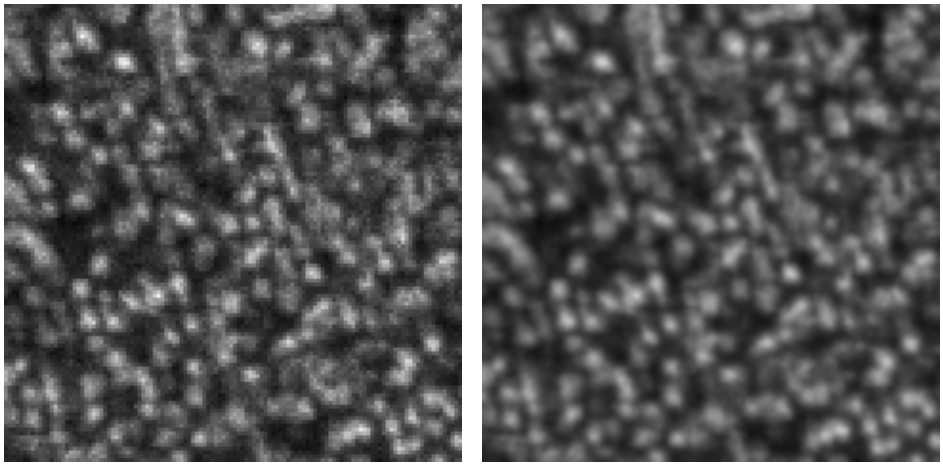
Figure 4.12: Detections and sources of error in a mouse brain image. Yellow rectangles: Large cells where centers are predicted near the boundary or completely missed. Blue rectangles: Clusters where boundaries between cells are weak. Some cells are missed. See Figure 4.11 for plotting conventions.

center. Such cells were relatively uncommon, and there was a lack of training examples due to the small training set. Performance could be improved by adding more training examples and slightly increasing the window size.

Clusters with weak boundaries The method had trouble identifying all cells in clusters with weak boundaries between the cells (blue rectangles in Figure 4.12). In such clusters, one cell could appear like an extension of another. This was

reflected in the probability map by a single peak instead of two, or dimmer peaks that were rejected by the maxima finding procedure.

Errors inevitably occurred when extracting the maxima of the probability map. Smoothing the map (see Figure 4.13) flattened some peaks and caused them to sink below the probability threshold p_{min}. The parameters chosen during the training stage worked well enough to produce a balanced set of detections. However, there is potential for improvement.



(a) Probability map (b) Smoothed

Figure 4.13: Smoothing the probability map. (a) Probability map produced by the predictor. (b) Smoothed probability map ($\sigma_g = 1$) in which local maxima are found in 3-D with parameters $d_{min} = 3$ and $p_{min} = 0.8$.

4.3.2 Speed

Training the random forest predictor took around 460 s compared to 10 s for the cascade used for rat data. Predicting cell centers in the test volume (256 $\times$

256 × 88) took about 9 s on the same machine. All operations were performed in single-threaded mode.

Downsampling decreases the required window size and the number of voxels to be tested. With multithreading and sufficient main memory, downsampling may not be necessary.

4.4 Summary

The proposed methods detected cells they were designed for with good accuracy. For the rat brain data, F1 scores were about 0.9 on average. Precision and recall scores were fairly evenly balanced, however precision scores exceeded about 0.95 for three of the six testing volumes while the peak recall was around 0.93. The F1 score for the mouse brain data was about 0.85, again with a higher precision score (about 0.89) than recall (about 0.81).

Improvements can be made to reduce the number of undetected cells (up to 10% for the rat data and 20% for the mouse data). Performance took a hit by cells in the test data that were significantly different from the training data, especially in terms of size. This is to be expected since both methods fundamentally rely on the training data for making predictions. They do not incorporate other measures to improve generalizability such as scale invariance and noise reduction. Thus the training data need to capture most of the expected variations.

Both methods required about 10 seconds on a regular laptop to process a volume approximately 200 voxels × 200 voxels × 100 voxels large. Parallelizing them is relatively trivial and is expected to yield even better speeds.

5. DISCUSSION

This section summarizes the contributions of this research and explores the improvements that can be made in the future.

5.1 Contributions

Two approaches to detect cells in 3-D KESM images of Nissl-stained rodent brains were presented in this thesis. They output the location of one point near the center of each detected cell. This can be used as the starting point of many analysis tasks such as counting cells, studying their spatial distributions, and cell segmentation. Both methods require only a small amount of manual labeling and can process large volumetric data sets without further manual enhancement.

5.2 Future Work

Simple extensions to the proposed detection methods can potentially lead to more accurate results and lower processing time. Key improvements that could be made are listed below.

5.2.1 Robustness to Cell Size

Cells larger than the window size are a major source of detection errors. One solution to achieve scale invariance is to apply the detection methods in multiple resolutions: the training stage remains the same, and in the testing stage the predictors operate on the original volume as well as a downsampled version. This ensures that cells fit inside the window at least in the downsampled volume.

5.2.2 Inexpensive Features

Although orthogonal patch features were successful for detecting cells, they are expensive to extract. This is less of a problem with the cascade method as larger patches are extracted only for a small fraction of the input. Haar-like features have been known to perform well in object detection [44] while being efficient to compute using integral images. Such features can provide speedups for the mouse brain data. Different kinds of features or higher order features should also be tested for improving accuracy while allowing for the window size to be decreased.

5.2.3 Adaptive Parameter Selection

The bandwidth parameter used for mean shift in the rat data detection method was fixed based on the training data. For test data with cell sizes and density significantly different than the training data, an automatic bandwidth selection method [12] would be preferable. The bandwidth selection step showed that different bandwidths are optimal for each of the volumes in the training set: 8.5 for volume A, 7.0 for volume B, and 6.0 for volume C. The choice of 7.0 as the bandwidth was a compromise, making the detection method suboptimal. In the mouse brain method, automated analysis of the probability map could also suggest values for the parameters used to find local maxima, though the details are unclear.

5.2.4 Comprehensive Training Data

Analysis of the results strongly suggests that more training data consisting of differently-sized cells, both small and large, can be of huge benefit. This can be done as an alternative to multiresolution detection once it is verified that cells of all sizes are included in the training set. A large number of training examples are likely not required; it is more important to make cells in the training data diverse than

repetitive.

5.2.5 Further Testing on the Mouse Brain Data

A single volume was used as the mouse brain test set. It is essential to label and test on more data to get a comprehensive idea of the detection method's performance.

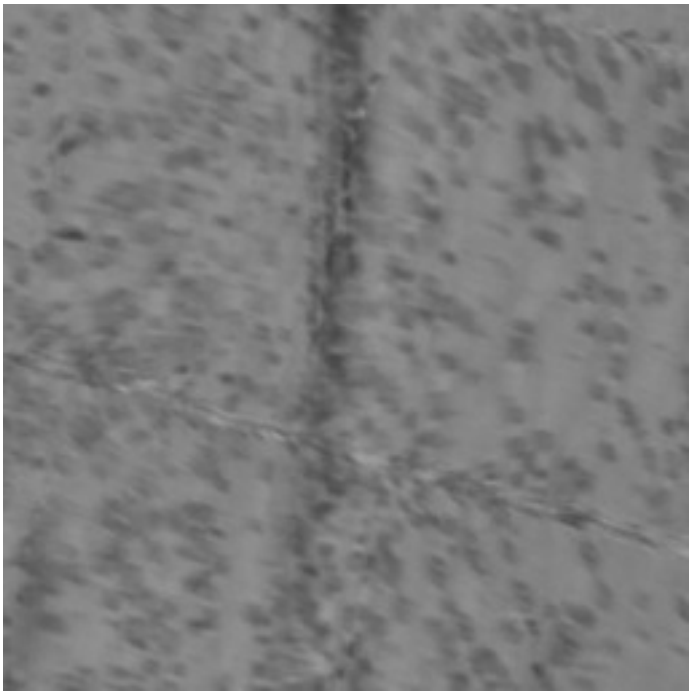


Figure 5.1: Mouse brain image that differs from the data used.

The mouse brain data set contains regions which are more heterogeneous than the data used in this thesis, with several objects that could be mistaken for neurons, such as in Figure 5.1. Testing on these data will help in estimating the extents of the mouse brain over which the method can be successfully applied.

6. CONCLUSIONS

This research was aimed towards developing an accurate and fast automated cell detection method for KESM Nissl images. Two methods were developed, both with random forests at their core, one for each of the rat data set and the mouse data set. Training sets consisting of a small number of volumes with labeled cell centers were required. These data sets were used by each method to output the locations of the detected cell centers given novel volumes. The methods were evaluated by validating the detected cell centers on manually labeled test sets. Results were found to be accurate for data that the methods were designed and trained for. The methods executed fast enough to be run on commodity hardware. A cascade architecture performed particularly well with high computational efficiency. This is expected to enable processing KESM data at large scales and ultimately obtain cell counts and locations in entire rat and mouse brains.

REFERENCES

[1] H. Ancin, B. Roysam, T. E. Dufresne, M. M. Chestnut, G. M. Ridder, D. H. Szarowski, and J. N. Turner, "Advances in automated 3-d image analyses of cell populations imaged by confocal microscopy", *Cytometry*, vol. 25, no. 3, pp. 221–234, 1996.

[2] A. Bosch, A. Zisserman, and X. Munoz, "Image classification using random forests and ferns", in *11th IEEE International Conference on Computer Vision*, Rio de Janeiro, Brazil, 2007, pp. 1–8.

[3] H. Braxmeier. (2013). Free photo: tussilago farfara, flower, macro - free image on pixabay - 100759, Pixabay, [Online]. Available: http://pixabay.com/en/tussilago-farfara-flower-macro-100759/ (visited on 09/15/2014).

[4] L. Breiman, "Random forests", *Machine Learning*, vol. 45, no. 1, pp. 5–32, 2001.

[5] S. N. Burke and C. A. Barnes, "Neural plasticity in the ageing brain", *Nature Reviews Neuroscience*, vol. 7, no. 1, pp. 30–40, 2006.

[6] R. Caruana and A. Niculescu-Mizil, "An empirical comparison of supervised learning algorithms", in *Proceedings of the 23rd International Conference on Machine Learning*, Pittsburgh, PA, 2006, pp. 161–168.

[7] Y. Cheng, "Mean shift, mode seeking, and clustering", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 17, no. 8, pp. 790–799, 1995.

[8] Y. Choe, D. Mayerich, J. Kwon, D. Miller, J. R. Chung, C. Sung, J. Keyser, and L. Abbott, "Knife-edge scanning microscopy for connectomics research",

in *The 2011 International Joint Conference on Neural Networks*, San Jose, CA, 2011, pp. 2258–2265.

[9] Y. Choe, D. Mayerich, J. Kwon, D. E. Miller, C. Sung, J. R. Chung, T. Huffman, J. Keyser, and L. C. Abbott, "Specimen preparation, imaging, and analysis protocols for knife-edge scanning microscopy", *Journal of Visualized Experiments*, no. 58, 2011.

[10] J. R. Chung, C. Sung, D. Mayerich, J. Kwon, D. E. Miller, T. Huffman, J. Keyser, L. C. Abbott, and Y. Choe, "Multiscale exploration of mouse brain microstructures using the knife-edge scanning microscope brain atlas", *Frontiers in Neuroinformatics*, vol. 5, p. 29, 2011.

[11] D. Comaniciu and P. Meer, "Mean shift: a robust approach toward feature space analysis", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, no. 5, pp. 603–619, 2002.

[12] D. Comaniciu, V. Ramesh, and P. Meer, "The variable bandwidth mean shift and data-driven scale selection", in *8th IEEE International Conference on Computer Vision*, vol. 1, Vancouver, Canada, 2001, pp. 438–445.

[13] E. Cosatto, M. Miller, H. Graf, and J. Meyer, "Grading nuclear pleomorphism on histological micrographs", in *19th International Conference on Pattern Recognition*, Orlando, FL, 2008, pp. 1–4.

[14] W. Dauer and S. Przedborski, "Parkinson's disease: mechanisms and models", *Neuron*, vol. 39, no. 6, pp. 889–909, 2003.

[15] A. C. D'Souza, "Automated counting of cell bodies using nissl stained cross-sectional images", Master's thesis, Department of Computer Science and Engineering, Texas A&M University, College Station, TX, 2007.

[16] M. Frucci, G. Ramella, and G. Sanniti di Baja, "Using resolution pyramids for watershed image segmentation", *Image and Vision Computing*, vol. 25, no. 6, pp. 1021–1031, 2007.

[17] J. Gall, A. Yao, N. Razavi, L. Van Gool, and V. Lempitsky, "Hough forests for object detection, tracking, and action recognition", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 33, no. 11, pp. 2188–2202, 2011.

[18] M. A. Haidekker, "Deformable models and active contours", in *Advanced Biomedical Image Analysis*, Hoboken, NJ: John Wiley & Sons, Inc., 2010, pp. 173–210.

[19] J. W. Han, T. P. Breckon, D. A. Randell, and G. Landini, "The application of support vector machine classification to detect cell nuclei for automated microscopy", *Machine Vision and Applications*, vol. 23, no. 1, pp. 15–24, 2012.

[20] A. Inglis, L. Cruz, D. L. Roe, H. E. Stanley, D. L. Rosene, and B. Urbanc, "Automated identification of neurons and their locations", *Journal of Microscopy*, vol. 230, no. 3, pp. 339–352, 2008.

[21] H. Irshad, A. Veillard, L. Roux, and D. Racoceanu, "Methods for nuclei detection, segmentation, and classification in digital histopathology: a review, current status and future potential", *IEEE Reviews in Biomedical Engineering*, vol. 7, pp. 97–114, 2014.

[22] Y. Al-Kofahi, W. Lassoued, W. Lee, and B. Roysam, "Improved automatic detection and segmentation of cell nuclei in histopathology images", *IEEE Transactions on Biomedical Engineering*, vol. 57, no. 4, pp. 841–852, 2010.

[23] I. Levner and H. Zhang, "Classification-driven watershed segmentation", *IEEE Transactions on Image Processing*, vol. 16, no. 5, pp. 1437–1445, 2007.

[24] G. Loy and A. Zelinsky, "Fast radial symmetry for detecting points of interest", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 25, no. 8, pp. 959–973, 2003.

[25] C. Malon and E. Cosatto, "Classification of mitotic figures with convolutional neural networks and seeded blob features", *Journal of Pathology Informatics*, vol. 4, no. 1, p. 9, 2013.

[26] N. Malpica, C. O. de Solórzano, J. J. Vaquero, A. Santos, I. Vallcorba, J. M. García-Sagredo, and F. del Pozo, "Applying watershed algorithms to the segmentation of clustered nuclei", *Cytometry*, vol. 28, no. 4, pp. 289–297, 1997.

[27] D. Mayerich, L. Abbott, and B. McCormick, "Knife-edge scanning microscopy for imaging and reconstruction of three-dimensional anatomical structures of the mouse brain", *Journal of Microscopy*, vol. 231, pp. 134–143, Pt 1 2008.

[28] D. Mayerich, J. Kwon, A. Panchal, J. Keyser, and Y. Choe, "Fast cell detection in high-throughput imagery using GPU-accelerated machine learning", in *2011 IEEE International Symposium on Biomedical Imaging: From Nano to Macro*, Chicago, IL, 2011, pp. 719–723.

[29] T. McInerney and D. Terzopoulos, "Chapter 8 - deformable models", in *Handbook of Medical Image Processing and Analysis (Second Edition)*, I. N. Bankman, Ed., Burlington, MA: Academic Press, 2009, pp. 145–166.

[30] E. Meijering, "Cell segmentation: 50 years down the road", *IEEE Signal Processing Magazine*, vol. 29, no. 5, pp. 140–145, 2012.

[31] N. Otsu, "A threshold selection method from gray-level histograms", *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 9, no. 1, pp. 62–66, 1979.

[32] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and É. Duchesnay, "Scikit-learn: machine learning in python", *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[33] PublicDomainPictures. (2013). Free photo: variation, confectionery, coated - free image on pixabay - 69470, Pixabay, [Online]. Available: http://pixabay.com/en/variation-confectionery-coated-69470/ (visited on 09/15/2014).

[34] J. Rogowska, "Chapter 5 - overview and fundamentals of medical image segmentation", in *Handbook of Medical Image Processing and Analysis (Second Edition)*, I. N. Bankman, Ed., Burlington, MA: Academic Press, 2009, pp. 73–90.

[35] J. Schindelin, I. Arganda-Carreras, E. Frise, V. Kaynig, M. Longair, T. Pietzsch, S. Preibisch, C. Rueden, S. Saalfeld, B. Schmid, J.-Y. Tinevez, D. J. White, V. Hartenstein, K. Eliceiri, P. Tomancak, and A. Cardona, "Fiji: an open-source platform for biological-image analysis", *Nature Methods*, vol. 9, no. 7, pp. 676–682, 2012.

[36] F. Schroff, A. Criminisi, and A. Zisserman, "Object class segmentation using random forests", in *Proceedings of the 19th British Machine Vision Conference*, Leeds, United Kingdom, 2008, pp. 54.1–54.10.

[37] C. M. Schumann and D. G. Amaral, "Stereological analysis of amygdala neuron number in autism", *The Journal of Neuroscience*, vol. 26, no. 29, pp. 7674–7679, 2006.

[38] N. Sharma and L. M. Aggarwal, "Automated medical image segmentation techniques", *Journal of Medical Physics*, vol. 35, no. 1, pp. 3–14, 2010.

[39] T. Sharp, "Implementing decision trees and forests on a GPU", in *Computer Vision – ECCV 2008*, ser. Lecture Notes in Computer Science, D. Forsyth, P. Torr, and A. Zisserman, Eds., vol. 5305, Berlin, Germany: Springer Berlin Heidelberg, 2008, pp. 595–608.

[40] J. Stegmaier, J. C. Otte, A. Kobitski, A. Bartschat, A. Garcia, G. U. Nienhaus, U. Strähle, and R. Mikut, "Fast segmentation of stained nuclei in terabyte-scale, time resolved 3d microscopy image stacks", *PLoS ONE*, vol. 9, no. 2, K. Metze, Ed., e90036, 2014.

[41] C. Sung, J. Woo, M. Goodman, T. Huffman, and Y. Choe, "Scalable, incremental learning with MapReduce parallelization for cell detection in high-resolution 3d microscopy data", in *The 2013 International Joint Conference on Neural Networks*, Dallas, TX, 2013, pp. 1–7.

[42] B. Van Essen, C. Macaraeg, M. Gokhale, and R. Prenger, "Accelerating a random forest classifier: multi-core, GP-GPU, or FPGA?", in *The 20th Annual IEEE International Symposium on Field-Programmable Custom Computing Machines*, Toronto, Canada, 2012, pp. 232–239.

[43] M. Veta, A. Huisman, M. Viergever, P. J. Van Diest, and J. P. W. Pluim, "Marker-controlled watershed segmentation of nuclei in h&e stained breast cancer biopsy images", in *2011 IEEE International Symposium on Biomedical Imaging: From Nano to Macro*, Chicago, IL, 2011, pp. 618–621.

[44] P. Viola and M. Jones, "Rapid object detection using a boosted cascade of simple features", in *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 1, Kauai, HI, 2001, pp. 511–518.

[45] P. Whitehouse, D. Price, R. Struble, A. Clark, J. Coyle, and M. Delon, "Alzheimer's disease and senile dementia: loss of neurons in the basal forebrain", *Science*, vol. 215, no. 4537, pp. 1237–1239, 1982.

[46] R. W. Williams and K. Herrup, "The control of neuron number", *Annual Review of Neuroscience*, vol. 11, pp. 423–453, 1988.

[47] D. J. Withey and Z. Koles, "Medical image segmentation: methods and software", in *Joint Meeting of the 6th International Symposium on Noninvasive Functional Source Imaging of the Brain and Heart and the International Conference on Functional Biomedical Imaging*, Hangzhou, China, 2007, pp. 140–143.

[48] H. Wu, J. Berba, and J. Gil, "Iterative thresholding for segmentation of cells from noisy images", *Journal of Microscopy*, vol. 197, no. 3, pp. 296–304, 2000.

[49] K. Wu, D. Gauthier, and M. Levine, "Live cell image segmentation", *IEEE Transactions on Biomedical Engineering*, vol. 42, no. 1, pp. 1–12, 1995.

[50] C. Xu, D. Pham, and J. Prince, "Image segmentation using deformable models", in *Handbook of Medical Imaging*, J. Fitzpatrick and M. Sonka, Eds., vol. 2, Bellingham, WA: SPIE Press, 2000, pp. 129–174.

[51] Q. Yang and B. Parvin, "Perceptual organization of radial symmetries", in *Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 1, Washington, DC, 2004, pp. 320–325.

[52] Q. Yang and B. Parvin, "Harmonic cut and regularized centroid transform for localization of subcellular structures", *IEEE Transactions on Biomedical Engineering*, vol. 50, no. 4, pp. 469–475, 2003.