AUTOMATIC METHODS FOR EXTRACTING STRUCTURES FROM 3D ELECTRON MICROSCOPE IMAGES

A thesis submitted to the University of Manchester
for the degree of Doctor of Philosophy
in the Faculty of Science and Engineering

2018

By
Yassar Almutairi
School of Computer Science

Contents

List of Tables

List of Figures

Abstract

Automatic methods for Extracting structures from 3D electron microscope images
Yassar Almutairi
A thesis submitted to the University of Manchester
for the degree of Doctor of Philosophy, 2018

Recent advances in cellular microscopy show the capability of gathering large volumes of data for examining small structures in biology. This thesis describes algorithms for locating, segmenting and measuring structures in microscope images. Collagen fibres form important structures in tissue, and are essential for force transmission, scaffolding and cell addition. Each fibre is long and thin, and large numbers group together into complex networks of bundles, which are little studied as yet. Serial block-face scanning electron microscopy (SBFSEM) can be used to image tissues containing the fibres, but analysing the images manually is almost impossible - there can be over 30,000 fibres in each image slice, and many hundreds of individual image slices in a volume. In this thesis we describe a system for automatically identifying and reconstructing the individual fibres, allowing analysis of their paths, how they form bundles and how individual fibres weave from one bundle to another. We also describe and evaluate a method for segmenting cell nuclei from SBFSEM, an important task for many studies. We use a Convolutional Neural Networks to locate the boundary of the nuclei in each

image slice. Geometric constraints are used to discard false matches. The full 3D shape of each nucleus is reconstructed by linking the boundaries in neighbouring slices. We demonstrate the system on several large image volumes.

Declaration

No portion of the work referred to in this thesis has been submitted in support of an application for another degree or qualification of this or any other university or other institute of learning.

Copyright

i. The author of this thesis (including any appendices and/or schedules to this thesis) owns certain copyright or related rights in it (the "Copyright") and s/he has given The University of Manchester certain rights to use such Copyright, including for administrative purposes.

ii. Copies of this thesis, either in full or in extracts and whether in hard or electronic copy, may be made **only** in accordance with the Copyright, Designs and Patents Act 1988 (as amended) and regulations issued under it or, where appropriate, in accordance with licensing agreements which the University has from time to time. This page must form part of any such copies made.

iii. The ownership of certain Copyright, patents, designs, trade marks and other intellectual property (the "Intellectual Property") and any reproductions of copyright works in the thesis, for example graphs and tables ("Reproductions"), which may be described in this thesis, may not be owned by the author and may be owned by third parties. Such Intellectual Property and Reproductions cannot and must not be made available for use without the prior written permission of the owner(s) of the relevant Intellectual Property and/or Reproductions.

iv. Further information on the conditions under which disclosure, publication and commercialisation of this thesis, the Copyright and any Intellectual Property and/or Reproductions described in it may take place is available in the University IP Policy (see `http://documents.manchester.ac.uk/`

DocuInfo.aspx?DocID=487), in any relevant Thesis restriction declarations deposited in the University Library, The University Library's regulations (see http://www.manchester.ac.uk/library/aboutus/regulations) and in The University's policy on presentation of Theses

Acknowledgements

Firstly, I would like to express my sincere gratitude to my supervisor prof.Tim Cootes for his guidance, consistent encouragement and patients throughout my PhD. I greatly appreciate his help without his supervision and and his guidance this thesis would not have been completed.

I would like to thank my co-supervisor Karl Kadler for providing all the data in this project and also for his continues help throughout this project.

Last but not the least, I would like to thank my parents and my wife, Ahlam, for almost unbelievable support and encouragement to complete my PhD. I also dedicate this Ph.D. thesis to my son Rakan who has given me extra strength and motivation.

Chapter 1

Introduction

There are many unanswered questions surrounding the organisation of biological samples at the cellular and sub-cellular level. For example, still unanswered questions concerning the formation of collagen fibres were posed at the end of the 19th century. Indeed, one of the great unsolved questions in biology is how cells generate tissues from collagen fibres [4]. The key to answering these questions lies in the exploration of small structures ranging in size from a few nanometres to hundreds of nanometres. An optical microscope cannot capture such structures because their size is less than the wavelength of light [5]. Electron microscope (EM) is the only technique capable of capturing structures only a few nanometres across.

Manually reconstructing the connectivity of an EM image is time consuming. In some cases, it is not feasible. This is due to the large number of objects in each slicesuch as neurones, fibres, or nucleiand the fact that slices can number in the thousands. For example, Helmstaedter et al. [6] calculated that to reconstruct the neurites of a mouse retina would require approximately 60,000 hours of annotation. The time available to the researcher is too valuable to be spent on such a task. Therefore, a fully automatic reconstruction algorithm that produces acceptably accurate reconstructions is necessary for the analysis of EM images.

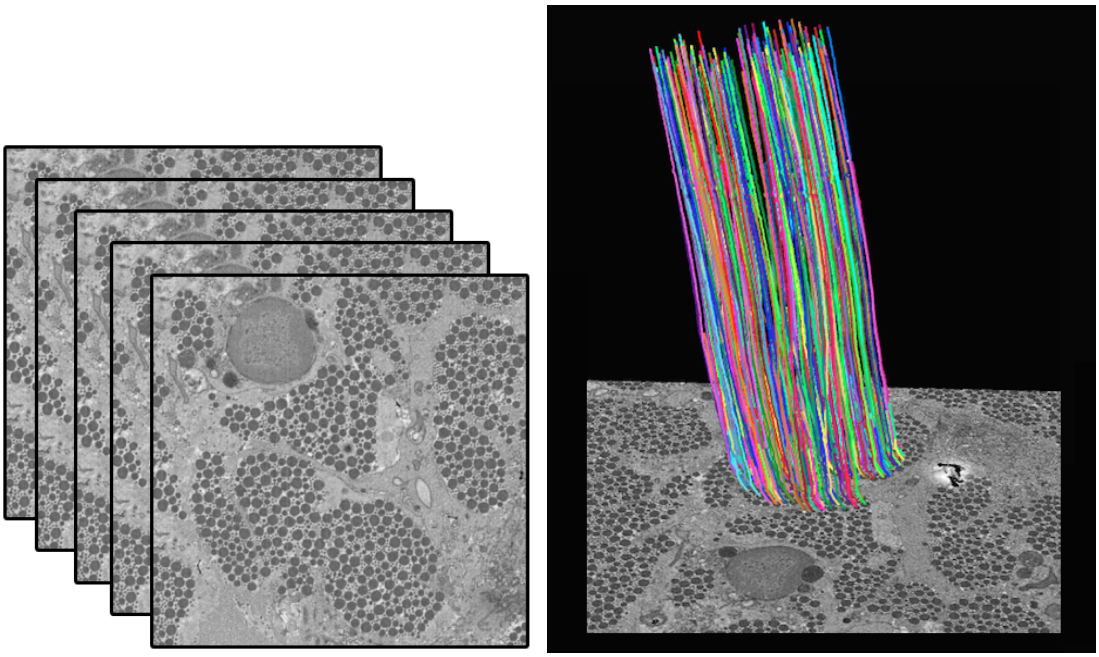
In the last two decades, with rapid improvements in electron microscopy, interest has grown in building an automatic tool that can extract sub-cellular structures. Semi-automatic and fully automatic approaches have been devised to model 3D biological data structures for analysis and visualisation. The semi-automatic method requires considerable time to produce a model for a single object, due to the constant need for human intervention to initialise and correct the algorithm. This limits the number of sub-cellular structures that can be examined. However, most of the automatic methods are designed to segment specific objects and do not generalise to other objects. This thesis will focus on two sub-cellular structures: collagen fibres and nuclei.

1.1 Collagen Fibres

Connective tissues play an important role in the maintenance of bodily structures. A single group of proteins called collagen makes that maintenance possible on a large enough scale to keep us fit. In vertebrates, there are 28 different types of collagen, which are mainly responsible for providing strength on a large scale.

Collagens are the main structural proteins found in connective tissues. They have many essential functions, including force transmission, scaffolding, cell adhesion, and cell migration. Collagen fibres appear in ordered bundles in the extracellular matrix, where they are the major tensile element in vertebrate tissues [7]. They can be found in skin, tendon, bone, and hollow organs and vary in diameter between 12 and 500 nanometres [8].

These collagen fibres are closely packed together in curvilinear bundles. When imaged, there may be over 20,000 fibres in each slice; furthermore, the number of images in the image stack varies between 1000 and 3000 (see Figure 1.1).



(a) (b)

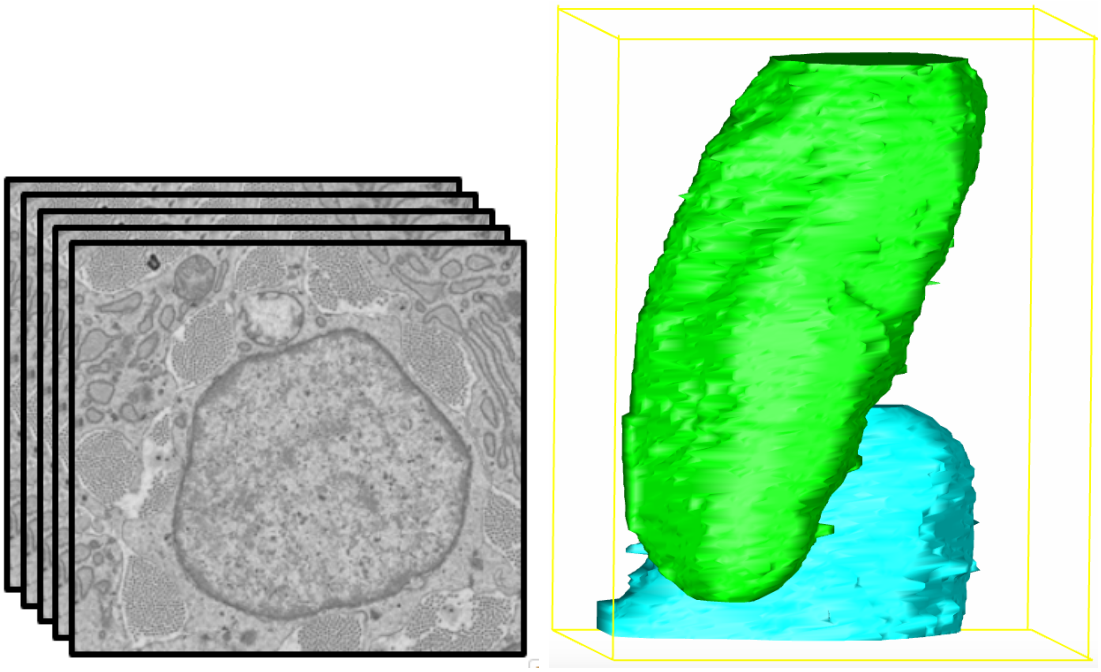
Figure 1.1: (a) Stack of images. (b) 3D paths of some of the detected fibres

These fibres need to be identified and tracked through the image stack in order to reconstruct the 3D shape of each fibre. Little is known about fibre length in tissue and the interactions between the fibres. The length of fibre is difficult to measure because the ends or tips of the fibres are rarely observed in cross-sections [4].

1.2 Nucleus Membrane

The nucleus is the most prominent organelle of the cell and is located in its centre. It appears as a dense, roughly spherical or irregular, organelle. The nucleus contains DNA arranged in chromosomes, which constitute most of the genetic material of the cell. The nucleus occupies approximately 10% of cell volume. The average diameter of the nucleus is about 6μm [9]. The nucleus is surrounded by the nuclear envelope (nuclear membrane). The nuclear envelope is made up of an inner and an outer membrane, which separate chromosomes

from cytoplasm. However, when the nucleus membrane is imaged by EM, the two membranes appeared as one.



(a) (b)

Figure 1.2: (a) Stack of images. (b) 3D reconstruction of the nuclei

Our goal is to reconstruct the 3D shape of each nucleus (see Figure 1.2) by finding the nucleus membrane in each slice of the image stack. Biologists are interested in studying the nucleus structure in 3D, in qualifying nuclei, and in measuring their volume, radii, and length. Researchers have found a close relationship during embryogenesis between fibres and cell membranes [10]. This has led to an investigation of their 3D cell shape in order to provide a full understanding of the interactions between fibres and cell membranes and how each affects the others development.

1.3 Aims and Objectives

The aim of this project is to automate the process of identifying and tracking structures through image volumes, thereby allowing these structures to be measured and visualised in 3D. The approach we take to reconstructing collagen fibres and nuclei is to first detect candidates for fibres and nuclei in every image and then to link candidates between neighbouring images to form extended fibres and nuclei. The linking stage can yield mistakes, which we attempt to identify and correct in a third stage. Having tracked all the fibres, we then identify bundles and lone fibres in order to study the networks they form. We also quantify nuclear volume, length, and radii. The main objectives are:

1. Create an automatic system that detects fibres in each slice. To do this, we develop a template matching algorithm and a trained classifier to detect fibres with different radii.

2. Create an automatic system to link candidate fibres between neighbouring images to form extended fibres.

3. Create an automatic system to identify bundles and lone fibres and, once all the fibres have been tracked, to study the networks they form.

4. Measure the fibre length, radii, and curvature.

5. Create an automatic system that detects nuclear membranes in each slice.

6. Create an automatic system to link candidate nuclei between neighbouring images in order to capture their complete structure.

7. Measure nucleus volume, radii, and length.

1.4 Contributions

This project developed a fully automatic system to extract structures in SBFSEM images. This project made the following contributions to knowledge:

- A novel technique to segment collagen fibres and to track thousands of them per slice in order to reconstruct their shape in 3D, thereby enabling biologists to make various measurements.
- A fully automated reconstruction system for nuclear membranes that will allow biologists to analyse 3D shapes and allow them to perform various measurements.

1.5 Outline of the Thesis

The next chapter provides an overview of the related literature. This overview describes the extracellular matrix and the intracellular matrix structures, focusing on collagen fibres and nuclei and on microscopy imaging techniques. There we discuss several approaches to the segmentation of EM images. Chapter 3 demonstrates collagen fibre segmentation and reconstruction in 3D. This chapter also includes the data set, the experiments, and the evaluation. In Chapter 4, we demonstrate our approach to the segmentation of nuclei and their reconstruction in 3D. We conclude in chapter 5 with a summary of our findings and a discussion of the limitations of this work and of its possible extension.

1.6 Publications

Y. Almutairi, T. F.Cootes, and K. Kadler. Tracking collagen fibres through image volumes from sbfsem. In T. Lambrou and X. Ye, editors, MIUA, pages 4045. BMVA, 2015. 3, 5,6.

Y. Almutairi, T. F.Cootes, and K. Kadler. Analysing the Structure of Collagen Fibres in SBFSEM Images. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops; 2016.

Y. Almutairi, T.F.Cootes and K.Kadler. Segmenting Nucleus Membranes in SBF-SEM Volume Data with Deep Neural Networks, MIUA, pages 23-34, Springer, Cham, 2018 .

Chapter 2

Background

2.1 Electron Microscopy

Any tissue consists of two components: the extracellular matrix and the intracellular matrix. These components may be seen only by Electron Microscopy (EM). The Extracellular Matrix (ECM) provides structural support in all cells. It is made of different proteins, collagens, sugars and other components. It is now known that the ECM influences cellular processes including migration, wound healing and differentiation [11]. On the other hand, intracellular matrices are organelles such as the nucleus, mitochondria and the Golgi apparatus. EM technique has increased the number of studies of cellular structures and its connectivity significantly in the last decade.

High-resolution images with minimum distortion are required for objects with diameters ranging from a few nanometres to hundreds of nanometres. Small structures in biological tissues can only be explored in sufficient resolution by using Electron Microscopy (EM). A light microscope is often used in medical imaging to analyse small object structures. It uses visible light and lenses to magnify specimens and their maximum resolution is 200nm. Light microscopes are unable to capture small objects because their size is less than the wavelength of

visible light (e.g. $\lambda = 400$nm to 700nm) [5]. However, EM takes advantage of the electron's wavelength ($\lambda = 0.005$nm), which is much shorter than the wavelength of visible light. Thus, these elements may be explored in unprecedented detail, which will allow biologists to study their properties.

In EM, biological samples are prepared before imaging through several processes. Fixation is the first step in order to preserve sample state, remaining as close to the original state as possible. The sample is dehydrated by replacing water by ethanol or acetone, and then infiltrated and embedded into a liquid resin. Next, the sample is sectioned into ultra-thin slices using an ultramicrotome with a glass or diamond knife [12]. Since most biological materials are almost transparent under an electron beam, a further and critical step is required to enhance the contrast between different structures. Staining the sample using heavy metals such as uranium increases the contrast between different structures [13].

Electron microscope imaging can be broadly categorised into two approaches: Transmission Electron Microscopy (TEM) and Scanning Electron Microscopy (SEM). Each method has its own advantages and disadvantages, and the choice depends on the required resolution and the size of the structures [14].

2.1.1 Transmission Electron Microscopy (TEM)

Historically, Transmission Electron Microscopy (TEM) was the most commonly used technique for examining ultrastructure biological tissue. The tissue sample needs to sectioned before imaging. It was developed by Runska and Knoll in 1932. A beam of electrons is focused on a thin sample to allow electrons to pass through to create a projection. The projection is captured using an electron-sensitive detector [14]. TEM can provide 3D datasets for samples using tilt-series-based tomography [13]. The resolution of TEM can reach 2nm in an image plane, i.e. in the x and y axis. However, the z axis is limited to 50nm, and it is necessary for

the slices to be thick enough to avoid damaging them during the process. As a result it is difficult to match slices through small diameter fibrils from one image slice to the next.

Serial Section Transmission Electron Microcopy (SSTEM)

SSTEM is a well-established technique [15][16] and has affected many studies on neuronal circuitry. The sectioning process is critical in SSTEM; this process is tedious, and those performing it are prone to mistakes. The first step is to perform fixation to the block (several hundred μm) using some type of aldehyde. The block is then embedded in a polymeric material with epoxy resin. This embedded block is serially sectioned by a diamond knife on an ultramicrotome, producing thin sections. The staining procedure can be performed before this sectioning occurs, en bloc, or after sectioning. Figure 2.1 illustrates the process of acquiring an image stack.

A limitation of this technique is that it is time-consuming and needs skilled labour to section the sample into thin slices. Moreover, it is common during sample preparation for some slices to be damaged, making the reconstruction more difficult. In addition, some slices have artefacts generated during sectioning, such as tears, folds and holes, or the thickness varies from slice to slice. These artefacts make alignment of the image stack extremely challenging. Thus, the method is not practical when a large volume needs to be reconstructed.

2.1.2 Scanning Electron Microscopy (SEM)

In Scanning Electron Microscopy (SEM), the sample does not require sectioning prior to imaging. Instead, the surface of the sample is directly imaged by detecting secondary electrons reflected from that surface (see Fig. 2.1.) and collecting these electrons on the right and left sides of the sample. The emission of these electrons, which contain information about the surface topology, is caused by

inelastic interactions between the primary electron beam and the sample. The secondary electrons are isolated from backscattered electrons using special detectors, which detect the higher energy of backscattered electrons.

Serial Block-face Scanning Electron Microscopy (SBFSEM)

SBFSEM was introduced by Denk and Horstmann [5] to automate the process of imaging large volumes. The specimen remains stable during imaging and sectioning, which enhances the alignment of the image stack significantly compared to TEM. SBFSEM involves repeatedly removing a thin slice (as thin as 25nm) from a sample block with a diamond knife then taking an EM image of the remaining face. The individual slices can be put together into a volume [5], and provide large 3D datasets. In this technique, the resolution of the image plane may reach 5nm.

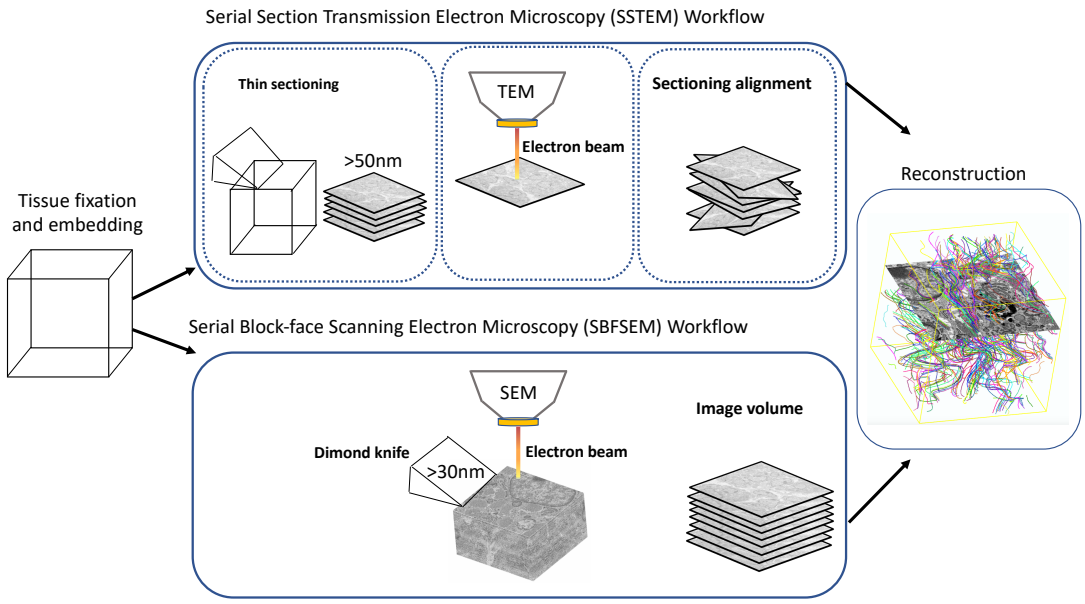


Figure 2.1: A diagram of Electron Microscopy acquisition processes of tissue volume using SSTEM and SBFSEM.

SBFSEM has several advantages over traditional SSTEM. Since sections are discarded, they may be thinner than in TEM. Furthermore, most of the artefacts that occur in TEM, including distortion and compression, are avoided because the block face is imaged, not the cut section. Unlike SSTEM, the sections in SBFSEM are already aligned, and fully automated analysis techniques are much easier. Moreover, the alignment of a large volumehundreds or even thousandsof sequential slices can be imaged without significant operator involvement.

2.2 Structure and Function of Collagen Fibre

Collagens are the main structural protein found in connective tissues. They have many essential functions including force transmission, scaffolding, cell adhesion and cell migration. Collagen fibrils appear in ordered bundles in the extracellular matrix, where they are the major tensile element in vertebrate tissues [7]. They can be found in skin, tendon, bone and hollow organs and vary in diameter between 12 and 500 nanometres [8]. These collagen fibrils are closely packed together in curvilinear bundles. They are not detectable by light microscopy even if the fibril's diameter reaches 500 nanometres, because they are crowded in a bundle. When such a bundle is imaged there may be more than 20,000 fibrils in each slice, and the number of images in the volume varies between 1000 and 3000. These fibrils need to be identified and tracked through the image stack in order to reconstruct the 3D shape of each fibril. It is not feasible to track all the fibrils in the image stack manually.

Trelstad and Hayashi [17] were the first to show the structure of collagen fibrils by serial section Transmission Electron Microscopy (ssTEM). Twenty-eight different types of collagens have been identified in vertebrates, which can be divided into two groups based on their structure: fibrillar and non-fibrillar [18]. Fibrillar collagens consist of four types and provide mechanical strength in connective

tissues [19]. In spite of important differences between different fibrillar collagens, they show similarities in their structure and behaviour. Fibrillar collagens form elongated triple-helical molecules that aggregate in a highly ordered stagger to form fibrils. Much is known about the collagen's genes but fundamental questions remain unanswered, for example regarding the fibrils' length, organisation and assembly [20].

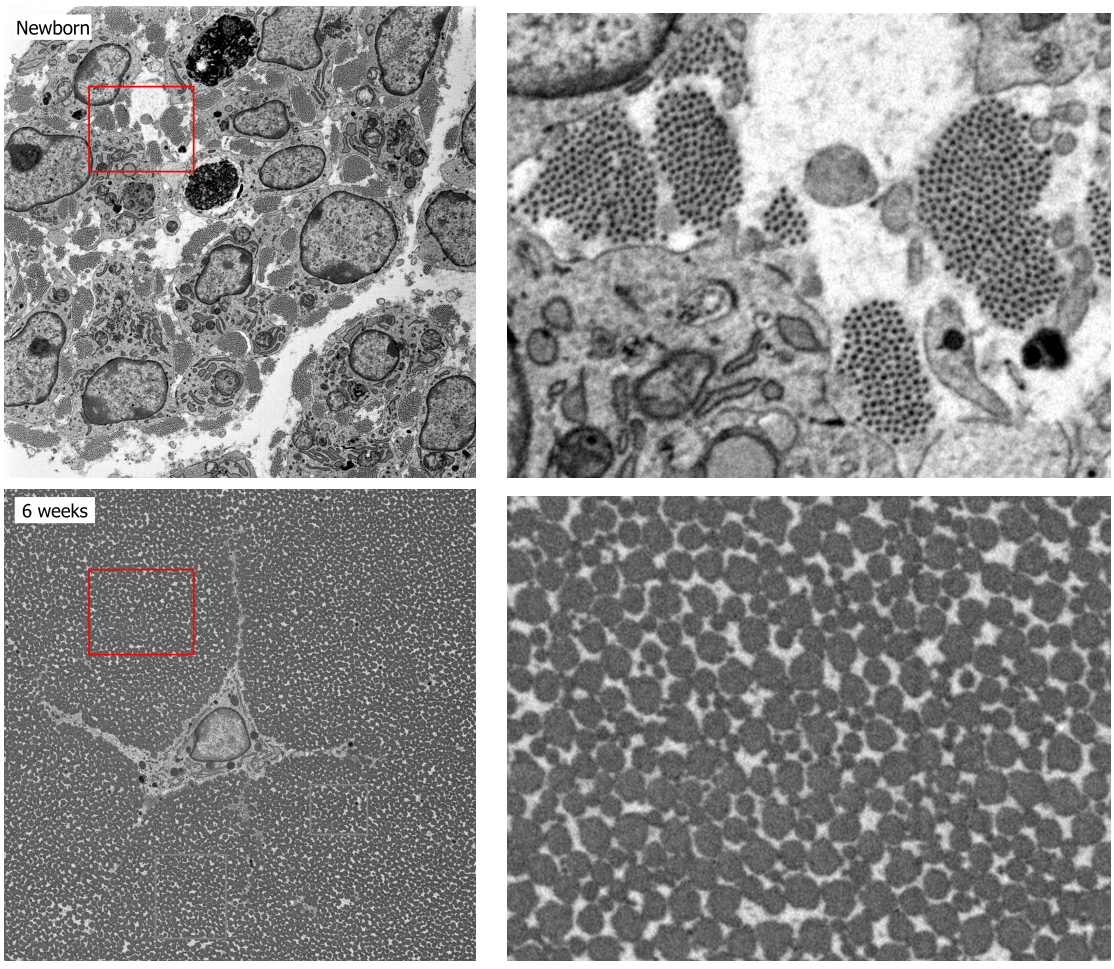


Figure 2.2: Images of mouse-tail tendon through development. The first row is for newborn where most fibres are grouped into distinct bundles. Second row shows the fibres after 6 weeks where fibres are larger and bundles have merged.

To learn more about tendon development, the exact positions and trajectories of these fibres need to be examined. During different stages of development

significant changes occur. For example, for mouse-tail tendons at embryonic day 15.5, fibres have a circular shape and are organised in bundles. At birth their diameter and radii have increased and their number increased by roughly four times. After six weeks fibres are larger and have an ellipsoid shape, and bundles are merged [10] (see Fig. 2.2).

Most fibres are grouped together, forming bundles. However, some of the fibres leave their bundle and join others, or enter other structures (see Fig. 2.3). The paths followed by such lone fibres are of interest to biologists. Lone fibres leave a bundle as a single fibre or as a small group of fibres. They might leave their bundle and merge with another, or they might return to the original bundle.

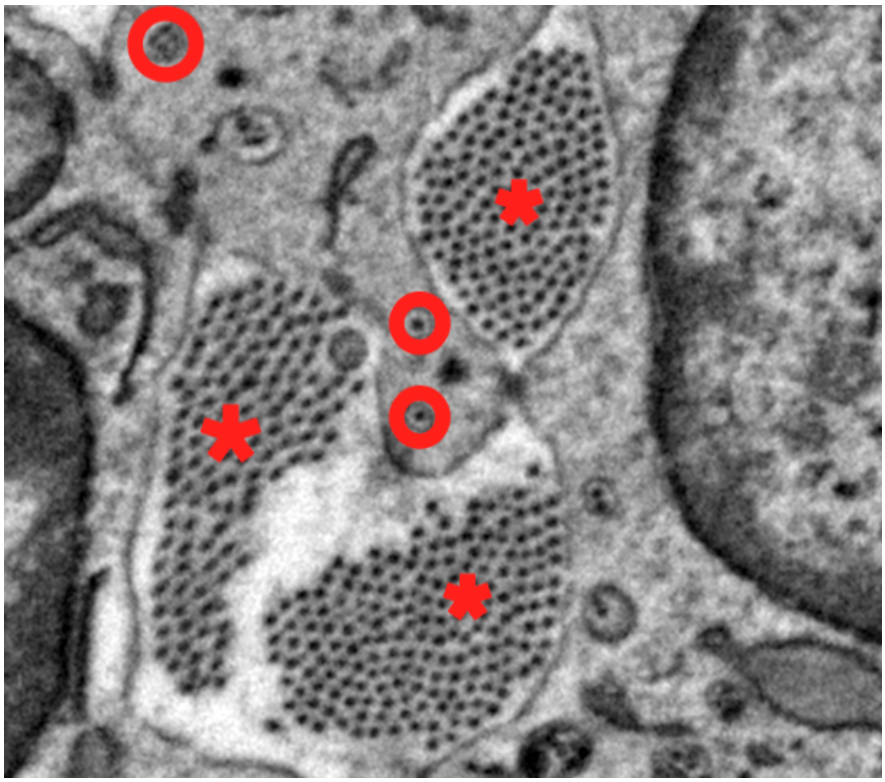


Figure 2.3: Stars represent bundles and circles show lone fibres.

Little is known about fibrils' length in a tissue and the interaction between fibrils. The length of the fibrils is difficult to measure because fibrils' ends or tips are rarely observed in cross-sections [21]. Moreover, fibrils may leave the boundaries of the image stack, which makes finding the longest length even harder. It is also interesting for researchers to study the interaction between fibrils and bundles. Bundles can merge with other bundles, forming a larger one. A bundle can also split into two smaller bundles.

2.3 Structure and Function of the Nucleus

The nucleus is the most prominent organelle of the cell and is located in its centre. There are several other structures inside the cell such as mitochondria and Golgi complex (see Fig. 2.4). The nucleus appears as a dense, roughly spherical or irregular organelle. The nucleus contains DNA arranged in chromosomes, which constitute most of the genetic material of the cell. The nucleus occupies approximately 10% of the cell volume and has an average diameter of about 6μm [17]. The nucleus is surrounded by the nuclear envelope (nuclear membrane), which is made up of two membranes, an inner and outer membrane, which separate the chromosomes from the cytoplasm.

The nuclear membrane has thousands of nuclear pores, which are circular hollow proteins with an outer diameter of roughly 100nm and an inner channel of about 40nm [21]. The nuclear pores work as a transport channel to exchange materials between the nucleus and the cytoplasm, and they link the inner and the outer membrane. Any cell goes through a life-cycle, which is a process of growth and division. The cell cycle is a series of events that occur in a cell resulting in its division, producing two new daughter cells. Mitosis is one part of the cell cycle, when replicated chromosomes are separated into two new nuclei.

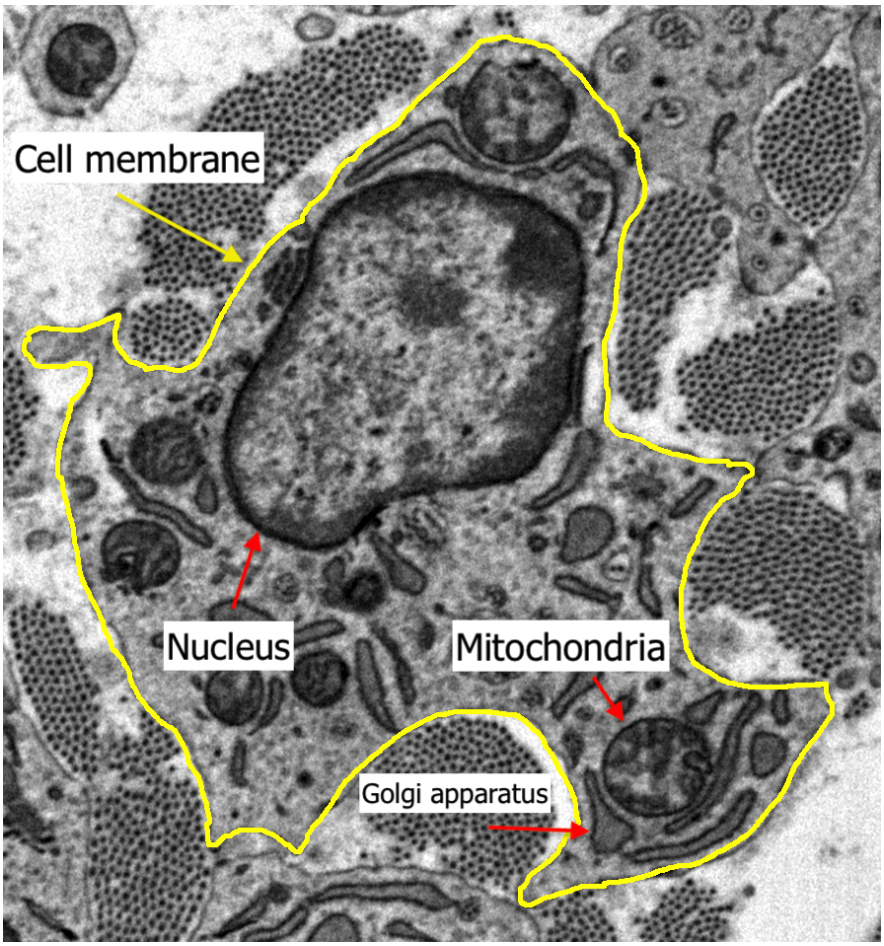


Figure 2.4: Example of one cell where the cell membrane is marked with a yellow line, and showing nucleus, mitochondria and Golgi complex structres

Our interest in nuclei is to reconstruct the 3D shape of each nucleus by finding the nucleus' membrane in each slice of the image stack. Biologists are interested in studying the nucleus' structure in 3D, quantifying nuclei and measuring their volume, radii and length. Researchers have found that there is a close relationship during embryogenesis between fibrils and cell membranes [10]. This has initiated investigations into the 3D cell shape to fully understand the interactions between fibrils and cell membranes and how they affect each other in their development.

2.4 Challenges

2.4.1 The Challenges for Fibre Detection and Reconstruction

A fully automatic system for locating and tracking fibres is challenging due to the complexity of the networks that the bundles form and the variation of shape of fibres within the image volume. Particular challenges include:

- The tortuous morphology of fibres.
- Fibres may disappear and appear again from the side of the image or due problems in sectioning.
- The large numbers of individual fibres - as many as 20,000 per slice.
- Bundles of fibres may split or merge.
- Some individual fibres may leave one bundle and join another, following significantly different paths to the majority of the fibres in the bundle - these are often the most interesting to biologists, but are the hardest to track.

2.4.2 The Challenges for Nuclei Detection and Reconstruction

A fully automatic system for segmenting and reconstructing nuclei is challenging due to the following:

- Structure with low contrast.
- Other cell compartment such as mitochondria and Golgi complex (see Fig. 2.4).
- Membranes of two nuceli may overlap.

2.5 Machine Learning Techniques

Machine learning methods have the ability to automatically learn from a set of examples to make a prediction or classify new instances. They are general-purpose approaches and can successfully classify unseen data. Machine learning algorithms are often categorised into two categories:

Supervised Learning: The training examples comprise input data (typically a vector x) and their target labels y (typically a number for class identification). The model aims to learn a function $y = f(x)$ from pairs of examples such as (x_1,y_1), (x_2,y_2),. . . ,(x_n,y_n). The typical example for this type of learning is classification. There are many examples of this type of algorithm, such as support vector machine, random forests, neural networks and, recently, deep learning. An example in medical imaging of supervised learning is the segmenting and classification of cells using a trained random forest classifier [22] or using decision forests to automatically segment high-grade gliomas in multi-channel MR [23], as well as more recent work using deep learning [24] to automatically classify human epithelial type-2 cells.

Unsupervised Learning: The training data comprise input data X (typically a vector) without any target label y. This type of learning aims to group similar instances using a distance measure. This can be applied to medical images; for example, the research presented in [25] incorporated a K-means clustering and watershed segmentation algorithm. To segment an image, the K-means algorithm is first applied to cluster the image; this produces a primary segmentation of the image before the application of the watershed algorithm.

2.5.1 Random Forest

Decision trees have been used for a long time [26] [27] but suffer from over-fitting and lack of generalisation. The Random forest was introduced by Breiman [28] and has become a popular technique in classification. A Random Forest is an ensemble of trees each trained using a different random subset of the trained set and features. The result of the forest is obtained by combining the results of each individual tree (either by averaging or by voting). This is an example of Bootstrap Aggregation (bagging), which has been shown to improve classification and reduce overfitting. The training approach encourages each tree to give an independent estimate of the result- combining these reduces the overall variance.

Random forests can handle large input data and they tend to avoid overfitting. There are several parameters that affect the random forest construction and prediction:

- The number of trees.
- The maximum depth of each tree.
- The proportion of features considered at each node.
- The choice of features.

Shotton et al. [29] have shown that the accuracy of random forest classifier increases significantly when forest size increases. Later studies have shown that learning deep trees could lead to over-fitting and how to maintain large amount of training examples [30]. Random forests are used widely to solve classification and regression problems.

2.5.2 Building a Random Forest for Classification

A Random Forest is an ensemble of T trees. Suppose we have a training set of n examples $X = \{(x_i, a_i)\}$ (data vector x_i with associated label a_i). Each tree is trained on a different set of data generated by randomly sampling (with replacement) n examples from the original training set.

Each tree contains a set of decision nodes, leading to a set of final leaf nodes. Each decision node evaluates one feature (for instance, one element of the feature vector x_i) and sends the sample to one of its two child nodes depending on whether the feature value is above or below a threshold. Thus the decision node splits the training set into two parts, X_1 and X_2. During training a random subset of possible features and associated thresholds are considered, and the pair that minimises a suitable cost function is chosen. When training a classifier, the cost function is usually chosen to ensure that the resulting data subsets are as "pure" as possible, by measuring their entropy.

Nodes are constructed recursively from the root, stopping either when a particular depth is reached or when the training samples reaching a node are sufficiently "pure". The final leaf nodes record the result to be returned by the tree (for instance, as a probability of one class or the other). See Fig. 2.5.

Earlier works have demonstrated the power of random forests in image classification [31] [32] [33]. Random forests have been applied to medical images successfully where the problem in most cases is treated as binary classification. Support Vector Machines were the most popular choice to solve a binary classification problem as they guarantee a maximum margin between two classes. However, applying Support Vector Machines to classify an input into multiple classes deteriorates its performance [34]. An empirical evaluation has been done for random forest with other binary classifiers [35] which showed that random

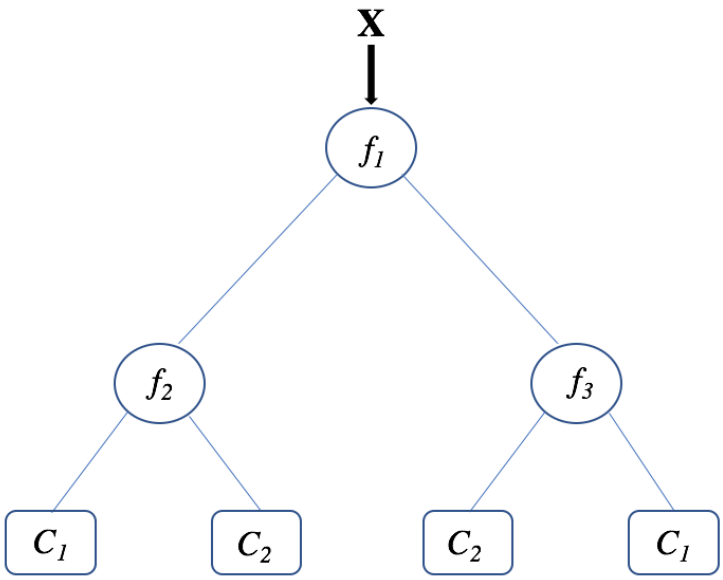


Figure 2.5: Example of a tree creates a split of feature space, and each leaf node contains the class prediction.

forests have overall best performance. It showed also that random forests can handle high dimensional problems. Random forests have shown robustness to significant overlaps between classes and noise in the training examples [28]. One of the challenges in processing medical images is segmentation due to low signal-to-noise ratio.

Lempitsky et al. [36] use random forest to discriminate automatically delineation of the myocardium in echo-cardiograms. They extracted a 3D patch and then trained random forests on these patches and experimented on different number of trees. This study showed that the performance deteriorated when the number of trees reduced from 10 to 3. Moreover, unbalanced distribution of classes may have a detrimental effect on training random forests. This occurs in some applications such as performing image segmentation to detect object from background. This problem could be mitigated by resampling training examples so they have roughly the same uniform training distributions.

2.5.3 Artificial Neural Networks (ANNs)

The Artificial Neural Networks (ANNs) approach was initially inspired by the biological neural networks in the brain, which consist of a collection of connected units or nodes (neurons). ANNs represent one of the better-known methods used in pattern recognition. ANNs have many advantages over conventional methods, since ANNs are self-adaptive and do not require manual adjustment; this makes it suitable for modelling complex problems and for estimating posterior probabilities [37].

Perceptron

In biological terms, neurons receive a signal using dendrites and the cell body processes the input by measuring the weight of the signal. Then, it fires a pulse with known strength and duration along the axon, and the axon sends a signal to other neurons. In the human brain there are billions of neurons interconnected with each other to form a network [38].

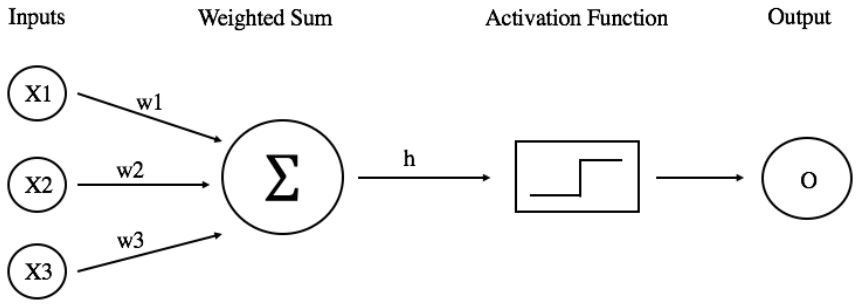


Figure 2.6: The mathematical representation of a neurone.

Rosenblatt [39] presented the Perceptron which is a probabilistic model of a neural unit Fig. 2.6 illustrates its structure. The model contains a set of weights **w** for the inputs, an adder that sums the input value **x** weighted by **w** and an activation function as a thresholding to produce the final output of the neurone. Formally,

1- Initialise the weights w_ij by small random numbers.
2- For T iterations or until the iteration error is less than a specified user threshold.
3- Calculate the activation using equation 2.5 for each neurons.
4- Update the weights for each neurons using equation 2.6

Table 2.1: Perceptron Algorithm summary

the model has n inputs $\mathbf{x}$, each input is associated with weight $\mathbf{w}$ and a bias b that shifts the decision boundary:

$$y = g(\mathbf{x} \cdot \mathbf{w} + b) = g(h + b), \tag{2.1}$$

where

$$h = \sum_{n=1}^{n} w_i x_i \tag{2.2}$$

The bias is put inside the vector $\mathbf{w}$ such that $w_0 = b$, and similarly we add $x_0 = 1$ inside the vector $\mathbf{x}$. This makes equation 2.1 become:

$$y = g(\mathbf{x} \cdot \mathbf{w}) \tag{2.3}$$

where the activation function g is given as:

$$g(x) = \begin{cases} 1, & \text{if x } \geq 0; \\ 0, & \text{otherwise.} \end{cases} \tag{2.4}$$

The perceptron is a learning algorithm that can classify its input vector $\mathbf{x}$ into two classes. The vector $\mathbf{w}$ is the separating hyper-plane such that $\mathbf{w} \cdot \mathbf{x} \geq 0$ the item is classified as class 1 and $\mathbf{w} \cdot \mathbf{x} < 0$ for class 2. Table 2.1 describes the steps of the algorithm used to train the weights of a perceptron.

$$y_j = g(\sum_{n=1}^{n} w_{ij}x_{ij}) = \begin{cases} 1, & \text{if } \sum_{n=1}^{n} w_{ij}x_{ij} > 0 \\ 0, & \text{if } \sum_{n=1}^{n} w_{ij}x_{ij} \leq 0 \end{cases} \tag{2.5}$$

$$w_{ij} \leftarrow w_{ij} - \eta(y_j - t_j) \cdot x_i \tag{2.6}$$

The learning algorithm uses equation 2.5 to compute the activation of the neurons before computing the error. A simple error function can be formulated to calculate the error between the output and the target. Given that y_i is the algorithm output and t_i is the actual target, an error function can be formulated as $(y_i - t_i)$.

The perceptron works with a classification problem. However, it works only with separable data. For example it fails to classify simple classification problem if the data is not separable such as XOR.

Multi Layer Perceptron

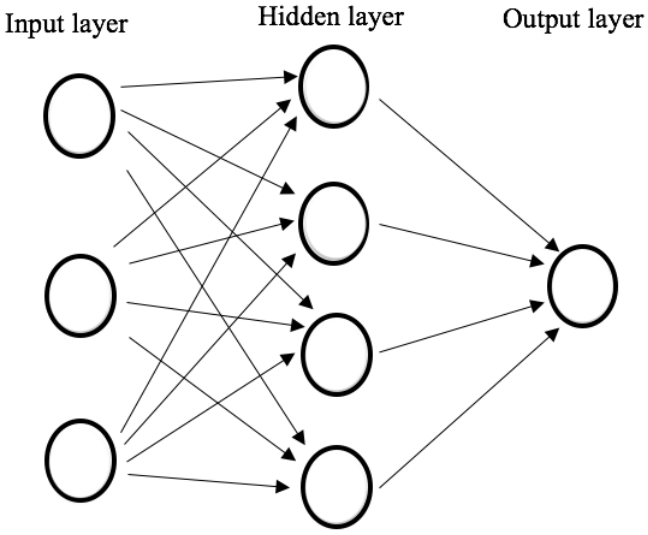


Figure 2.7: Fully connected ANNs showing one input layer, hidden layer and output layer.

There are several types of neural networks, but multilayer perceptrons (MLPs) are the most widely used. They can handle the case of separable data. MLPs consist of three-layers input, a hidden and an output layer (see Figure 2.7). The hidden layer may contain many layers. The weights of the network are derived from a back-propagation error algorithm [40].

In general, the data are fed into the network using the input layer, then the multiple hidden layers transform the data into a nonlinear form. All neurons in the hidden layer and in the output layer are connected to the previous layer. The weighted sum for each neuron is calculated from its inputs. Then, the output of each neuron is calculated using an activation function. One example of used activation function is rectifier linear unit (ReLU) [41]. It can learn faster compared to other functions such as sigmoid or tanh unit [42].

The number of hidden layers relates to the depth of the network, and width relates to the number of neurons in one of the hidden layers. With recent technological advances, especially in GPUs, it has become possible to train networks with a large number of hidden layers. ANNs with more than two layers are sometimes called deep networks.

Model representations are captured using the weight between neurons, learned from the input/output samples. The loss function needs to be minimised through the learning algorithm by comparing the model output to the target label. Given the non-convex and high-dimensional nature of the loss function, the minimisation procedure is challenging. The back-propagation algorithm was invented in the 1970s, but was only fully appreciated in the late 1980s after the publication of an influential paper by David Rumelhart et al. [40]. They describe ANNs where back-propagation performs faster than previous methods for learning. The goal

1- Initialise weights with small random values.
2- For each training sample in a batch:
a) Feed to network to calculate outputs at each layer.
b) Propagate final error back through network to
estimate gradient of cost function with respect to each weight.
3- Update weights using the estimates of the gradients.

Table 2.2: MLP Algorithm summary

of back-propagation is to compute the gradient of the cost function. During learning, all the edge weights are initially assigned a random number. For every sample input, the ANNs are activated to calculate the output; then, the output is compared to the target label. The error is then back-propagated to the previous layer in the network in order to calculate the gradient of the loss function and the weights are updated. Once the error falls below a certain threshold the procedure is terminated. Common used activation functions are:

$$g(x) = x, \text{ Linear} \tag{2.7}$$

$$g(x) = \frac{1}{1 + e^{-cx}}, c > 0, \text{ Sigmoid} \tag{2.8}$$

$$g(x) = \frac{1 - e^{-x}}{1 - e^{-x}}, \text{ Hyperbolic tangent} \tag{2.9}$$

$$g(x) = \begin{cases} 0, & \text{for x } < 0; \\ x, & \text{for x} \geq 0 \end{cases}, \text{Rectified Linear Unit (ReLU)} \tag{2.10}$$

To quantify the error the mean sum-of-square error function is used:

$$E(t, y) = \frac{1}{2} \sum_{j=1}^{n} (y_j - t_j)^2 \tag{2.11}$$

where y is the predicted value obtained from the neural network model and t is the actual value. Table 2.2 summarises the MLP algorithm.

Several algorithms are used for supervised learning, including gradient descent and gradient descent with momentum. The reported results have shown that ANNs with gradient descent with momentum perform best.

2.5.4 Convolutional Neural Networks (CNNs)

The Convolution Neural Network (CNN) share several features with MLP: it has neurons that have weights and bias, fully connected layers and a loss functions (see Fig. 2.8). However, CNN is different from MLP in two main aspects. First, it uses convolutions and fully connected layers. Secondly, it has a pooling layer (downsampling) to compress representations. CNN can be traced back to the 1990s, where it found some applications in deep learning and image segmentation. These early attempts were limited in terms of the training size and the depth of the networks [2].

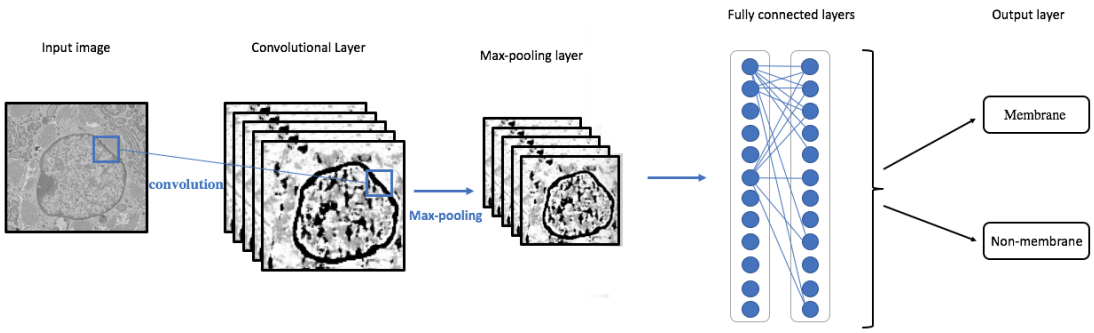


Figure 2.8: CNN architecture consist of convolutional, pooling layers and two fully connected layers. followed by one output layer. Each convolutional layer has several feature maps.

Since 2012, when the results of the ImageNet competition were released, CNNs have captured the interest of the computer vision community. Krizhevsky et al. [43] won the competition by a wide margin, developing a CNN architecture to classify 1.2 million high-resolution images into 1000 different classes. The

architecture consisted of a deep network with eight layers that was trained on the pixel intensity. Krizhevsky et al. reported that their network had 60 million parameters and 650,000 neurons, and the model achieved high accuracy compared to other methods. In recent years, several architectures have been proposed with larger and deeper networks, such as [2].

A typical CNN consists of a set of consecutive convolutional and pooling layers one or more fully connected layers and an output. The output can be a single class, i.e. a number or a probability map for classes that describe the image. Each convolutional layer generates one or more maps of neurons, called feature maps.

The input of each neuron within a feature map is only connected to a small neighbourhood among the feature map from the previous layer, called a receptive field. Each feature map shares parameters with all of its neurons. Since the feature map moves along the inputs sharing the same parameters, set of weights and bias, in each layer, different feature maps can detect different features such as edges in different orientations. Parameter sharing and local connectivity allow the number of parameters of the network to be reduced. Once the features are computed, the image is passed to a subsampling layer (pooling layer) to perform a local averaging of the weights.

The exact location and number of features has limited influence, in most applications, on the final prediction. Based on this assumption, the pooling layer calculates the average or the maximum of neighbours' neurons to summarise their activity. This smooths the feature representation. The same pooling layer repeats for smaller patches but shifts by n pixels. This effectively down-samples the input image. Thus, the model parameters are reduced significantly.

Fully Convolution Network (FCN)

The Fully Convolution Network (FCN) is trained end-to-end fashion (see Fig. 2.9). The key insight for FCN is that it accepts any input size and produces correspondingly-sized segmented output. There have been several attempts to convert typical CNN into FCN by converting the fully connected layers in the CNN into convolution filters [44][45][46][47]. The convolutional filter has a kernel size to cover the entire input region. Thus, this technique can process any input size and produces an output map. However, the output is subsampled because of the pooling layers involved. This technique is not applicable for semantic segmentation, the task of labelling each pixel in the image to a predefined class, as the output map is not the same size as the input as it requires more spatial information.

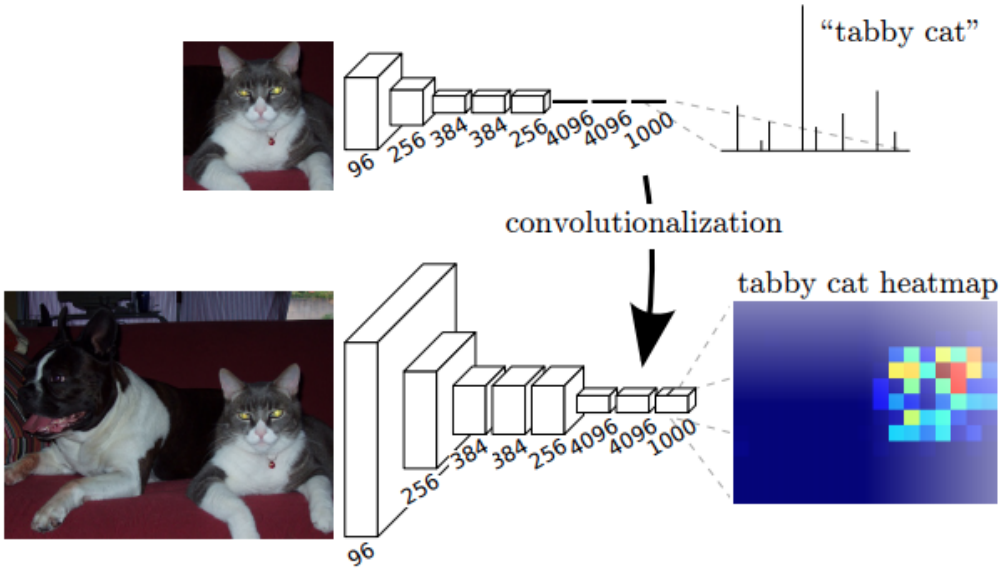


Figure 2.9: Trasforming CNN to perform classification task into Fully Convolutional Network (FCN) to make pixel-wise segmentation. Image is taken from [1]

The bounding-box (sliding window) technique that is used to perform semantic segmentation reduces the labelling process to annotated data. However, this

technique is not applicable for medical imaging since the segmented image needs to be accurate to each pixel. This technique can achieve high accuracy by training the network on square patches based on the image intensities centred on the pixel itself. Once the network has finished learning, it classifies each pixel in a sliding window fashion. For example, Ciresan et al [44] won the ISBI completion by a large margin using this approach. The limitation of this method is the classification process is slow since it must run over all the pixels and extract a square patch for every pixel. Moreover, any change in the patch size will affect the localisation accuracy since max pooling controls the amount of information to be processed.

Long et al. [1] proposed a method to convert existing CNN networks into FCN such as AlexNet [43] VGG nets [48] and the GoogLeNet4 [49]. The fully connected layer converted to a 1×1 convolution layer with two channels (background and foreground) to predict the class score, and 21 channel dimension to predict PASCAL classes. The output of each classifier is followed by a deconvolution (up-convolution) layer to perform bilinear upsampling of the coarse output to obtain pixel-dense output. Skips are introduced between layers to refine spatial accuracy and produce detailed segmentation. The skips combine the final layer with shallow layer appearance information which is learned in an end-to-end way (see Fig. 2.10).

In [1] the authors found that learning through upsampling combined with a skip layer is more effective and efficient. Interpolation can be applied to reconstruct lost signals in the sampling process and to smooth the data. Upsampling is convolution with a stride $1/f$ of some integral factor f. Backwards convolutional (up-convolutional) is the obvious way to perform upsampling with output stride of$f$. Therefore, the upsampling is learned in the network in an end-to-end by back-propagation from the pixel wise loss rather than a fixed upsampling.

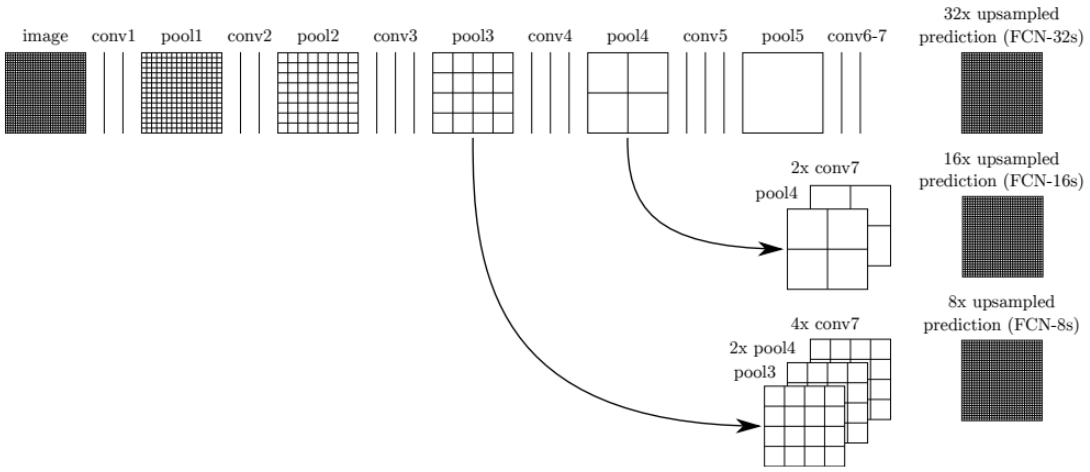


Figure 2.10: FCN skip architecture introduced by Long et al. [1]

The limitation of this work is that the network has a fixed receptive field size. This leads to mislabelling and separating the segmented object into small parts due to the objects are larger or smaller than the receptive field. Moreover, small objects may be classified as background [50].

U-Net

Ronneberger et al [2] proposed a CNN approach that won ISBI cell tracking challenge in 2015 by a large margin. The U-Net architecture consists of a contracting path which is similar to a typical CNN and an expansive path to create a high resolution segmentation map (see Fig. 2.11). The network consist of two convolutions with a filter size 3×3 followed by a ReLU and for downsampling they applied a max-pooling layer with filter size 2×2. They double the number of features at each down sampling step. The expansive path consists of upsampling of the feature map at every step following a convolution layer (up-convolution) with 2×2 kernels that reduces the number of feature channels by half. It also has a concatenation with high resolution feature from the contracting path and two convolution layers with 3×3 kernels each one followed by ReLU. The final

layer consists of 1×1 convolution to map the feature vector to the desired class number.

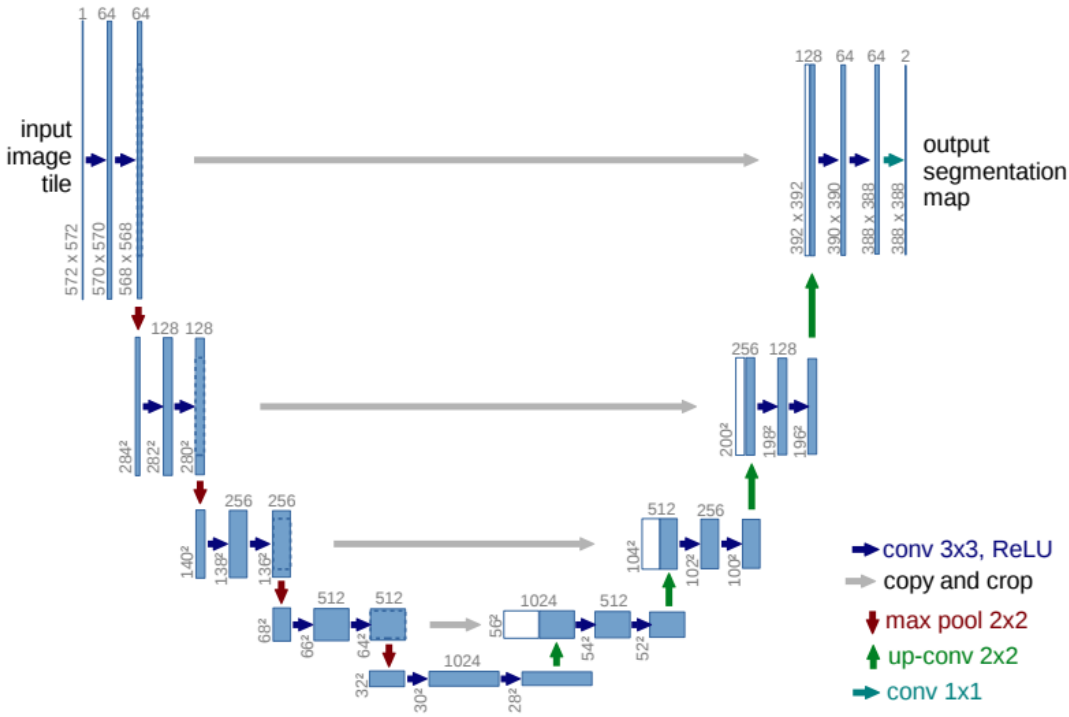


Figure 2.11: U-Net Architecture. Image is taken from [2]

The U-Net learns segmentation in an end-to-end setting which means a raw image is fed to the network and a segmented map is output. The architecture of U-Net allows the network to capture the image context in the contraction path where the expansion path allows precise localisation. The U-Net is a Fully Connected Network (FCN) but it differs from a typical FCN in terms of how the upsampling is achieved.

The recent advances in GPU and efficient implementation of 3D convolution have made it possible to extend FCN approaches to 3D and allow large amounts of annotated data to be processed. For example, 3D U-Net [51] has been proposed recently which is extension of the standard U-Net [2] but all the 2D operations

are replaced with their 3D counterpart.

2.6 Image Segmentation

Recent technological developments have led to the development of automatic 3D electron microscopes allowing the acquisition of large voxel volumes. These high-resolution images have become essential in several fields, such as connectomics, which aims to reconstruct structures comprehensively using these images. However, few software packages are available to perform reconstructions, and most systems require a user to assist in these reconstructions.

Several methods for segmenting objects from the background were developed in the literature. These methods, such as thresholding, ridge detection and watershed-based detection, have produced moderate results. For example, the simplest method for nuclei segmentation is to threshold the image to classify each pixel into two classes, background and foreground. The threshold value can be selected automatically by Otsu thresholding [52] which assumes the pixel values come from two classes and selects a value that minimises intra-class variance. Those methods pick a single value to threshold the entire image.

Mathematical morphology operators, such as erosion, dilation, opening and closing, have been applied to segment nuclei. Other basic morphological operators, like boundary extraction, hole filling, and skeletonization, can be generated from the previous four operators. For example, Yang et al. [53] presented a method to locate nucleus markers in fluorescence microscopy images using conditional erosion.

This section will review the recent work of image segmentation on EM images.

The segmentation methods can be categorised into semi-automated and automated segmentation.

2.6.1 Semi-Automated Segmentation Methods on EM Images

Great interest has been shown in automating reconstruction algorithms to produce an accurate reconstruction of an EM image stack without substantial human effort. However, no automated method can produce accurate reconstruction for different sub-cellular structures. Thus, several works focused on developing semi-automated methods to reduce human effort and assist manual reconstruction. This section summarises the semi-automated methods that were developed for reconstructing EM image stacks.

The concept of interactive segmentation is similar to that of active learning [54]. Incorporating sparse input from the user supplies additional constraints to the segmentation algorithm. Immediate feedback is given to the user after providing this additional data. The user interactions may take on several different forms, such as drawing a general boundary [55], drawing brush strokes [56][57][58] or outlining the boundary [59] of the object of input.

The interactive learning and segmentation toolkit (ilastik) was developed by Sommer et al. [22] to segment biological images. Their work allowed users with no knowledge of image processing to perform segmentation and classification. The user graphically provided the ground truth by manually labelling the image. Then, a random forest classifier was trained on these data to produce a pixel classifier. The user could provide additional data at each iteration of the pixel classifier if the result was not acceptable by including examples in which the classifier made mistakes. Andres et al. [60] similarly worked on a design to segment neural tissues automatically. However, ilastik was interactively designed

(semi-automatic) to produce a pixel classifier that could automatically segment images based on user interest.

Another interactive semi-automated reconstruction that used the same concept as ilastik was built by Straehle et al. [61]. In this reconstruction method, the training data was provided manually by the user to adjust a graph cut and watershed-based segmentation algorithms. The segmentation was defined, according to the work in [62], through labelling of a graph.

Jones et al. [63] presented an approach for sparse labelling, an alternative to iterative labelling. In this approach, the image was presented to the user with an overlaid grid, and the user marked the intersection of the grid with the membrane location. Although this method reduced the time that was required for annotation, its sensitivity to the penalty parameter might introduce noise. Therefore, there were several cases that the pixel classifier could not handle. Additionally, this method could not deal with multiple structures at once.

Carlbom et al. [64] developed an algorithm to segment neurons using active contours (snakes). In this method, the user needed to provide an estimate of the final contour. The user could also change the parameters of the snake algorithm to correct errors during the segmentation. However, this method is sensitive to noise and not applicable to perform automatically on medical images.

Semi-automated segmentation methods reduce the reconstruction time significantly compared to manual reconstruction. However, these methods are not scalable to large image volumes; a large scale would require years of manual efforts by experts to reconstruct just 1 mm^3.

2.6.2 Automated Segmentation Methods on EM Images

Poor segmentation results are achieved by general segmentation methods, which have been suggested for natural image datasets [56]. When dealing with EM datasets, the global probability boundary [65] and boosted edge learning [60] perform poorly, despite segmentation performance on natural images as demonstrated by Jain et al. [66]. Thus, optimisation of a successful method is needed for segmenting specific structures, such as nuclei or mitochondria, in EM images.

Fully automated segmentation and reconstruction methods of sub-cellular structures for EM images can be categorised into two general approaches. The first approach starts by segmenting the sub-cellular structures in 2D slices and then links consecutive slices to reconstruct the 3D shape. This method was suggested due to the present anisotropy of most EM image volumes, excluding FIBSEM. The second approach perform segmentation directly in 3D.

In several studies, the cross-sections were treated as nodes on a graph that link the cross-sections across the consecutive slices. Nodes in consecutive slices were connected by edges, and weight was assigned to each edge based on the consistency between pairs of cross-sections. These cross-sections were linked together by finding a set of minimum-cost paths through graphs [67][65] or by performing a hierarchical clustering [68].

Unsupervised Learning

Several unsupervised attempts were made in terms of 2D sub-cellular segmentation. Anisotropic directional filtering was applied to segment membranes and to enhance their continuity [69][67]. Although applying this filtering improved the membrane continuity, it could not segment the membrane with accuracy and performed poorly in removing the intra-cellular structures. Kumar et al. [70]

attempted to remove the intra-cellular structures using a multi-scale contextual model with radon-like features. However, this model only achieved moderate performance in suppressing membranes from different sub-cellular structures.

Fok et al. [71] developed a method which used snakes to segment fibres. Moreover, this method did not require any input from the user to initialise the snake parameters. It roughly estimated the axon centre using the Hough transform in which the boundaries of the axon were found using an active contour model. This method is sensitive to initialisation and could not be generalised. An earlier approach by Bertalmio et al. [72] was proposed for object segmentation and tracking in EM images. The algorithm used a 2D deformable curve model, and the tracking was achieved by solving partial differential equations.

A more recent study by Frangakis and Hegerl [73] was based on normalised cuts and extended the work of [62] on image stacks. The segmentation method is solved by using a generalised eigenvector. However, large numbers of objects and large image size can cause computational problems for the algorithm. An alternative method using graph cuts with texture regions was developed by Chang et al. [74]. This method assumed that objects of interest could be discriminated from the textured background. An energy function was defined according to curve evolution, and a graph was constructed and optimised using a graph cut. However, those methods have the same limitation; finding optimal initialisation parameters automatically was difficult. Many methods required input for initialisation.

A normalised cuts approach [73] was proposed to segment neurons in EM images. In [75], a level set was used to segment sub-cellar structures which were limited to segment one object and were sensitive to initialisation. An active contour base was applied to segment mitochondria since most were elliptical. However, this method missed most of the non-elliptical mitochondria, since the algorithm was

designed to detect elliptical blobs [76]. With the utilisation of a watershed-based supervoxel segmentation, only local information was taken into consideration by a convolutional neural network in [60]. Mitochondrial textural features were used to train a classifier in [77], but shape information was ignored.

Supervised Learning

However, membranes can be detected, and neurons can be segmented into 2D images with supervised learning methods, which have been proven to give high accuracy. Jain et al. [78] trained a binary pixel classifier using convolution neural networks to distinguish membranes; this achieved better results than traditional approaches. Mishchenko [79] used Hessian eigenspace features to build a single perceptron classifier to detect cell boundaries in EM images and achieved promising results. However, this method still needed further post-processing to close membrane gaps and remove intra-cellular structures.

Several studies by Jurrus et al. [80] [67] also tracked membranes. In methods proposed by these studies, multiple MLPs were connected sequentially to recognise the membrane. In the first phase, these networks received an input of the EM image; in later stages, the outputs of the previous phase were combined with the current stages image. By combining one phase with the next, each phase was given greater spatial context.

Recent work by Jurrus et al. [81] tracked multiple axons across large image volumes. The image stack was acquired using the SBFSEM technique, and the tracking algorithm was built on the Kalman-snakes framework proposed by [82]. The image volume was denoised, and an active contour model was used to detect axon boundaries. The initial contour was fitted into the first slice where the user defined the initial parameters or automatically using a watershed filter to detect

locations. In order to maintain a contour that fit the shape of the axon, a simple smoothing constraint was applied at every frame. Each subsequent contour was then tracked using a Kalman filter to predict each point's location in the next slice, which was specified by location and velocity. Once the Kalman filter finished tracking the axons, the user validated the result manually and reinitialised error prediction when necessary.

Andres et al. [60] proposed a method to segment neural tissue using a hierarchical classifier in three stages. In the first stage, a random forest classifier was trained on manually segmented samples to validate each voxel. The watershed algorithm was then used in the second phase to evaluate these classifications, producing an over-segmentation. Finally, the segments produced by the watershed algorithm were classified as correct or incorrect by another trained random forest classifier. To further enhance the segmentation results, other, more complicated methods were suggested. Closing membrane gaps was improved using a graph cut framework, which had perceptual grouping constraints [68]. To enhance 2D neural segmentation, Liu et al. [83] incorporated intersection information, using a merge forest structure. Vitaladevuni and Basri [84] considered co-clustering pairs of adjacent sections and expressing them as a quadratic optimisation problem for 3D linking.

Dan et al. [44] proposed a deep artificial neural network to separate pixels into two classes (membrane or non-membrane). The deep neural network consisted of convolution, max-pooling and fully connected layers. A classifier was trained on samples which consisted of square patches annotated manually. The patch centre was located on the membrane pixel. This method did not require any post-processing and could be applied on several EM images to segment objects.

The second approach is performing 2D segmentation and 3D reconstruction simultaneously. Supervoxels were merged into segmentations by a reinforcement learning framework, introduced by Jain et al. [85]. Incorporating both supervoxel face and boundary curve information for 3D supervoxel merging was proposed by Andres et al. [86] using a graphical model framework. Multiple 2D segmentation hypotheses were generated and the 3D segmentation fusion was expressed in a Markov random field framework [87]. Funke et al. [88], represented the 2D segmentation hypotheses using tree structures and solve an integer linear programming problem with constraints to perform segmentation and reconstruction simultaneously.

To perform segmentation directly in 3D would require the data to be isotropic to obtain acceptable performance. As of now, there are limitations in acquiring images with isotropic resolution even the emergence of new EM imaging techniques, such as FIBSEM image volume that can be generated is currently limited [89]. Therefore, the algorithms that directly segment in 3D may not be appropriate for most of the current data sets, as they are anisotropic.

2.7 Basic Methodology

2.7.1 Template Matching

The template matching technique is widely used for object detection It uses a predefined template for an object and computes the correlation between that template and the detected object. There are two types of template matching technique: a rigid and deformable template [90]. Exact matching rarely occurs due to noise, size and object orientation. A common method is to normalise the candidate region to the template size to obtain a more accurate estimate. The conventional method is to compute the correlation between the template for every pixel in the original image. Therefore, this process is time consuming; however,

the number of candidate pixels can be reduced using thresholding or alternative techniques.

Orientation does not cause a problem when finding fibrils in an image, because the fibrils have a roughly circular shape. Thus, a template can be produced based on average fibrils. The similarity between the template and image is calculated using a normalised cross correlation for an image f and template t of size N×N as follows [91]:

$$\gamma_{(u,v)} = \frac{\sum_{x,y}(f(u+x,v+y) - \bar{f}_{u,v})(t(x,y) - \bar{t})}{\sqrt{\sum_{x,y}(f(u+x,v+y) - \bar{f}_{u,v})^2 \sum_{x,y}(t(x,y) - \bar{t})^2}} \tag{2.12}$$

Where $f(x,y)$ is the pixel's value at the point (x,y). $\bar{f}_{u,v}$ is the mean of $f(x,y)$ within area of t shifted to (u,v) which is computed by the following equation:

$$\bar{f}_{u,v} = \frac{1}{N_x N_y} \sum_{x=u}^{u+N_x-1} \sum_{y=v}^{v+N_y-1} f(x,y) \tag{2.13}$$

Rigid objects can be localised robustly by the normalised cross correlation with the presence of noise and occlusion.

2.7.2 Kalman Filter

Kalman Filter (KF) is a recursive algorithm used to estimate process state by recursively updating and estimating. KF can be used in any system with uncertain information about any dynamic system. There are many applications for Kalman Filter, such as video surveillance, robotics and surgery. KF was first developed in 1960 and is described as a linear system as follows:

$$x_{k+1} = F_k x_k + w_k, \tag{2.14}$$

$$y_k = H_k x_k + v_k, \tag{2.15}$$

where x_{k+1} is the state vector (e.g. position, velocity) at time $k+1$ and y_k is the vector of measurements. F is the state transition matrix and H is the transformation matrix that maps the state vector parameters into the measurement domain. w_k and v_k are random variables with covariance Q_k and R_k, respectively, initialised by $\hat{x}_{1|0} = E(x_0)$ and $P_{1|0} = Cov(x_0)$,.

KF measurement and time updates together give a recursive solution to the problem. Measurement update is applied as follows:

$$\hat{x}_{k|k} = \hat{x}_{k|k-1} + K_k(y_k - H_k\hat{x}_{k|k-1}) \tag{2.16}$$

$$P_{k|k} = P_{k|k-1} - K_k H_k P_{k|k-1} \tag{2.17}$$

where

$$K_k = P_{k|k-1}H_k^T(H_k P_{k|k-1}H_k^T + R_k)^{-1} \tag{2.18}$$

and time update is given as follows:

$$\hat{x}_{k+1|k} = F_{k+1}\hat{x}_{k|k} \tag{2.19}$$

$$\hat{P}_{k+1|k} = Q_k + F_{k+1}P_{k|k}F_{k+1}^T \tag{2.20}$$

The measurement and time updates are then repeated.

2.8 Summary

There are many questions in biology that have not been answered yet, such as: How long are the fibres? What is the relationship between fibres and cells? Recent advances in EM have led to large amounts of images of biological samples. This has allowed biologists to capture unprecedented details of sub-cellular structures such as nuclei, collagen fibres and mitochondria. There are two different

techniques for accruing images using EM, TEM and SEM. TEM requires the sample to be sectioned prior to imaging, whereas SEM takes an image of the surface sample and then sections it. However, to analyse these structures, each object in every image needs to be segmented and linked to its corresponding object in the subsequent slice.

There is active research in the area of semi-automatic segmentation to assist biologists with performing segmentation and reconstruction of image volumes. However, when reconstructing a large volume, these approaches would require tedious work from an expert to initialise the algorithm. Moreover, an expert's time is too valuable to be spent on such a task. Therefore, there is a great demand for automatic segmentation and reconstruction algorithms in large-scale image volumes. The number of images in an image volume can be more than 1,000, and the number of objects can be more than 20,000 per image.

General automatic segmentation methods developed for natural images have achieved great accuracy. However, when they are applied to EM images, they perform poorly. This is due to the nature of images and the presence of low-contrast structures, and to the fact that different objects may share the same appearance, such as nucleus and mitochondria membranes. These algorithms need to be fine-tuned in order to obtain acceptable accuracy. We reviewed several automatic methods that have been applied to EM images. Most of these methods are designed for specific structures and cannot be generalised to handle multiple structures. We have not found any work for segmenting and reconstructing collagen fibres or nuclei that performed on a similar level to our images that were acquired by SBFSEM.

The project developed a fully automatic method based on template-matching and Random Forest classifiers to detect fibres. We then developed a tracking

algorithm to reconstruct the fibres and analyse their complex paths. We used a deep-learning approach to segment nucleus membranes and to reconstruct their 3D shapes.

Chapter 3

Fibre Detection and Reconstruction

In this work we propose a fibre detection and tracking approach. Image slices are roughly orthogonal to the fibres, so each fibre appears as a small disk or ellipsoid. A circle is sufficient to represent the fibre cross-section since biologists are more interested in the paths of the fibres than details of their cross-section. We demonstrate our method for segmenting fibres and tracking each individual fibre through the image stack. Finally we generate a detailed reconstruction of fibres in 3D.

It was found to be sufficient to use template matching to identify candidates for approximately circular fibres. However, fibres with polygonal shapes were located using a trained classifier. False template matches were eliminated using a Random Forest classifier. Candidates in each slice were then linked to identify extended fibres.

3.1 Fibre Representation

We focused on image volumes from SBFSEM, which were gathered so that the image planes are roughly orthogonal to the direction of the fibres of interest. In samples from early stage of development each fibre appears as a small disk or ellipsoid diameter in the range 2 to 25 pixels. During late development, fibres have a more polygonal shape in the range of 20 to 70 pixels.

A collagen fibril could be modelled as a sequence of points (see Fig. 3.1). For such a fibre track we use the notation $\mathbf{X}^{(p)} = \{\mathbf{x}_1^{(p)}, ..., \mathbf{x}_{n^{(p)}}^{(p)}\}$ where $p \in \{1, ..., N_f\}$ denotes the fibril number with N_f representing the total number of fibrils, and $n^{(p)}$ is the number of points that form the p^{th} fibril.

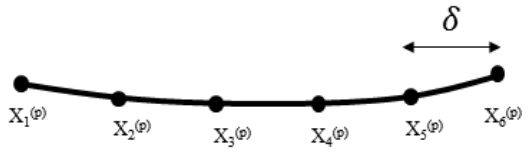


Figure 3.1: A sequence of points representing fibril $\mathbf{x}^{(p)}$. δ is the distance between any two adjacent image slices which is fixed to be a constant, and referred to as the step length.

3.2 Fibre Detection Algorithm

3.2.1 Template Matching Approach for Locating Fibres

Each fibre has a roughly circular shape which may vary from slice to slice due to sectioning. We developed our method based on template matching. We defined multiple models each trained to locate fibres at a particular radius. Each model focuses on locating fibres with radii within a narrow range. These are then used to generate samples (at a range of scales).

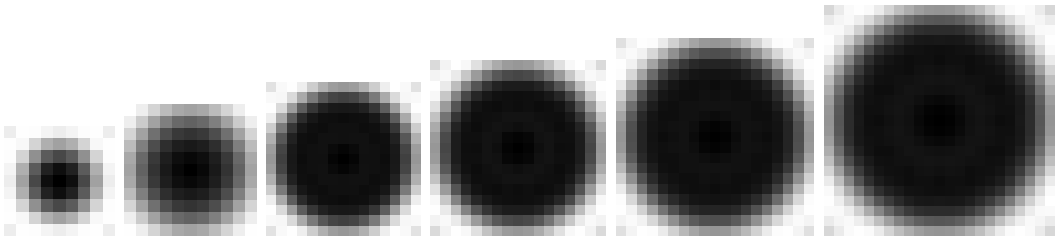
An image pyramid consists of a set of scaled (lower resolution) copies of the base. A typical case is where the copies are each a factor of two smaller than the next lower image in the pyramid. The biggest image is at level 0 and the next lower resolution image at level 1 and so on.

The assumption is that there is an optimal size (in pixels) for detecting particular objects, say 10 pixels radius. Thus a model trained for 10 radius can be used to detect objects at 10, 20, 40 etc. radius by running on different pyramid levels. Models looking for smaller than optimal size will only be run on the base level (highest resolution) image of a pyramid, such as when the radius is 3, 4, 5, 8. Table 3.1 shows an example of our model design.

Target Radius	Image Level	Model Radius
3	0	0
5	0	5
8	0	8
10	0	10
12	0	12
16	1	8
20	1	10
⋮	⋮	⋮
32	2	8

Table 3.1: Models used to search for fibrils at given radii.

Each individual template is constructed as the mean of a set of training examples centred on fibres with a particular range of radii (see Fig. 3.2). Table 3.2 shows the number of samples for each template. We manually annotate fibres in a training set, drawing a circle around each fibre of interest. We then divide up the training set by radius to train a range of templates. The template for a fibre of radius r is of size $(2r+3)$ x $(2r+3)$ - allowing a small border around each disk.



(a) (b) (c) (d) (e) (f)

Figure 3.2: Templates constructed with different sizes and mean radius such as (a) 9×9, with mean radius 3 (b) 11×11, with mean radius 5 (c) 15×15, with mean radius 7 (d) 17×17, with mean radius 8 (e) 19×19, with mean radius 9 (f) 23×23, with mean radius 11.

Template	Number of Samples
A	288
B	432
C	720
D	648
E	722
F	691

Table 3.2: Template statistics.

We use normalised cross-correlation (NCC) to search each image with the given template. As can be seen from Figure 3.4 (top), there are also many false positives, non-fibre patches. Since different sized templates will give different ranges of NCC output at the correct location, we apply a linear correction factor to the output,rq. The fit value returned for each result will be $f = (q - \hat{q})/\sigma$, where q is the normalised dot product between kernel and image patch, $\hat{q}$ is the median of q when measured at the correct position in a verification set and σ is the standard deviation of q over the verification set.

Orientation does not cause any problem when finding fibrils in an image, because the fibril has a roughly circular shape. The similarity between the template and image is calculated using a normalised cross correlation as explained in section 2.7.1.

By searching with templates at different scales, and choosing local maxima in the response, f, in both position and scale, we locate both the position and radius of good candidates - see for instance Figure 3.4 - which have responses which pass a threshold on f. Any false positives will be removed in the next phase using a Random Forest classifier.

3.2.2 Random Forests Approach for Locating Fibres

We extended the detection algorithm by training Random Forest to detect non-circular shapes. Eight Random Forest classifiers have been trained, each designed to deal with a different (narrow) range of radii of fibres, centred on $\hat{r}$. To obtain training examples we annotate fibres manually over a set of images by drawing circles around fibres. Thus, we have a set of circles defining the centre and radius of each fibre.

To increase the training data we rotated each patch four times with different scaling. False examples are generated automatically by displacing the circle around the centre of a fibre's patch at different scales (see Fig.3.3). Each Random Forest was then trained on image patches around all the samples which fell within its radii range.

Image patches of size $2\hat{r}+3$ were taken, centred on the centre of the circle defining the candidate. The features used to make decisions at each node of each tree in the forest were based on the difference between intensities in two randomly chosen pixels within the patch. We experimented with other features such as Haar-like features but did not obtain any significant improvement.

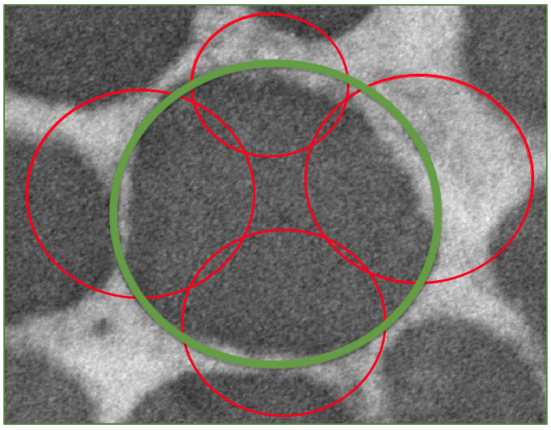


Figure 3.3: Fibre annotation to train classifier for detecting non-circular fibres : Positive example (green circle) and negative examples (red circles)

3.2.3 Random Forests for Eliminating False Matches

To eliminate the large number of false matches that are produced using NCC approach for detecting circular fibres, we trained another binary Random Forest classifier to distinguish fibres from non-fibre patches. Similarly we trained eight Random Forest classifiers each designed to deal with a different range of radii of fibres. However, the training samples were collected in a different way by running the NCC models (as described above) over a set of images with low threshold to minimise the number of false negatives (missed fibres). We then manually marked each circle as fibre or non-fibre.

To remove the false positives we evaluated all the candidates that were produced using the template matching approach. We extracted the relevant features for each sample and then the RF to make predictions whether is it fibre or not. Figure 3.4 shows the result of detecting fibres using template matching (Top), and after removing these false positives using RF classifiers (bottom).

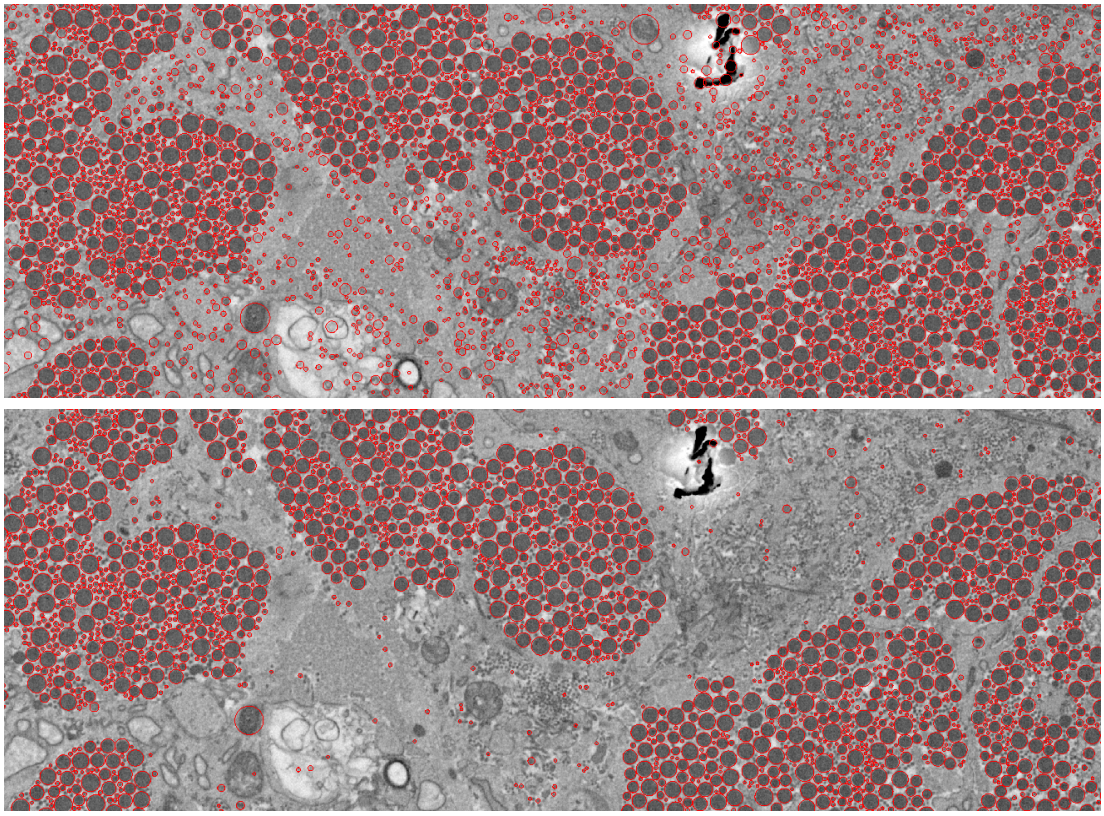


Figure 3.4: Fibre detection using normalised cross correlation result (top); and Fibre candidates after removing false positives using RF classifiers (bottom).

3.3 Fibre Tracking Algorithm

After running the detection stage on every frame we have a set of candidate disks, each of which is likely to be from a fibre. Let disk i in plane z have centre $\vec{p}_{i,z}$ and radius $r_{i,z}$, $i = 1..n_z$, n_z is the number of candidates in plane z. Each fibre appears as a sequence of candidate disks of similar radii in consecutive frames.

The simplest approach to tracking would be to extend each fibre with the nearest candidate disk of the correct radius in the next slice, if it was within a suitable threshold distance. In practise this does not work well for fibres in bundles, as the 'drift' of the bundle between frames can be larger than the separation between nearby fibres. This leads to ambiguity in matching the candidates to the fibres

and thus incorrect linking.

To reduce the chances of this we take account of the drift (the movement of the fibre bundle from one frame to the next) and assume each fibre only moves small amounts relative to its neighbours in the bundle when locating suitable candidates in the next frame.

We assume all candidate disks in the first frame are the start of a fibre, creating n_1 fibres each containing a single disk. We then process the subsequent frames one at a time, using the candidate disks to either extend an existing fibre, or to create new fibres if there is no match in the previous frame.

When processing a frame, the first step is to group the ends of fibres in the previous frame in order to identify bundles. This is done by a clustering algorithm, in which each fibre is added to an existing bundle if its disk at slice $(z-1)$ has a centre within a radius r_c of any other fibre centre in the cluster. The radius r_c was determined through extensive experiments across several datasets to find the optimal value. We evaluated different values of r_c between 3 to 30, and measure the tracking algorithm performance against our ground truth. We found that the optimal value for r_c is 9.

We then estimate the 'drift' of each bundle as the translation of all fibres which minimises the distance of their centres to the centres of candidates on the next slice. In particular, let $D_z(\mathbf{x})$ be the distance transform of the centres of disks in image z

$$D_z(\mathbf{x}) = min_i|\mathbf{p}_{i,z} - \mathbf{x}| \tag{3.1}$$

Let $\{\mathbf{x}_{b,j}\}$ be the centres of the n fibres identified in a bundle b on frame $(z-1)$.

The movement of the bundle is then the translation $\hat{\mathbf{t}}_b$ which minimises

$$S_b(\mathbf{t}) = \sum_j D_z(\mathbf{x}_{b,j} + \mathbf{t}) \tag{3.2}$$

Let $\mathbf{x}'_{b,j} = \mathbf{x}_{b,j} + \hat{\mathbf{t}}_b$ be the estimated centre of each fibre from frame $z-1$ projected onto frame z, and $r_{b,j}$ be the radius of the disk for that fibre in frame $(z-1)$. We now consider every candidate disk $\{\mathbf{p}_{i,z}, r_{i,z}\}$ in turn. If $|\mathbf{p}_{i,z} - \mathbf{x}'_{b,j}| < d_t$ and $|\log(r_{i,z}/r_{b,j})| < \log(1.25)$, then the disk (i,z) is used to extend fibre j in bundle b, otherwise the disk is used to start a new fibre. Fibres are assumed to have ended if no match is found in the next frame.

The constraint threshold d_t is the maximum amount of movement allowed for a particular fibre to move from one frame to another. To find the optimal value for this parameter we manually annotated 70 fibres across 50 slices. We analysed the movement of each fibre by finding the min, max and the average movement. We found that max movement for the smallest radii is 4 pixels (the smallest radius of fibre is 3). However, fibres with large radius can move more than 4 pixels. We evaluated several values to determine the movement of fibres with large radii between 0.40 to 0.80 $\times r_{i,z}$, and measure the tracking algorithm performance. Thus, we found that the optimal value for d_t to be 4 for fibres with small radii (between 3 and 7 radius) and $0.6 \times r_{i,z}$ for fibres with large radii (for larger than 7).

The other constraint we have is when to extend fibre from one frame to subsequent frame. The two disks should satisfy the constraint $|\log(r_{i,z}/r_{b,j})| < \log(1.25)$. This is because the fibre radius may differ slightly from one slice to another. To find the maximum change we use our annotated data and calculated the $|\log(r_{i,z}/r_{b,j})|$. We found that the proportion is less than $\log(1.25)$.

At the end of the process we have identified a large number of fibres, each running through some subset of the available frames. By comparing the disks at the ends of fibres we can identify and correct small gaps caused by detection failures. Short fibres (for instance, those visible in fewer than 10 frames) can be rejected as they are typically cause by false detections and are unlikely to be collagen fibres, which usually pass through many frames.

3.3.1 Filling Gaps in Fibres

Occasionally a candidate disk for a fibre is not detected in an image, either due to the failure of the detector, or because some image slices are corrupted by 'tearing' the surface of the block when the diamond knife cuts the slice. Such missing disks cause a long fibre to be split into two (or more) shorter fibres. To detect and correct small gaps caused by such detection failures we use a linear prediction of the fibre's location at frame z. We identify the end of every fibre at frame $z-1$ and estimate their center projected onto frame z. Similarly, we identify every fibre starting at frame $z+1$ and estimate their center projected onto frame z, then we link fibres that satisfy a constraint according to the following procedure:

Let disk i in plane z have centre $\vec{p}_{i,z}$ and radius $r_{i,z}$, $i = 1..n_z$, n_z is the number of candidates in plane z. For every fibre i ending at frame $z-1$ we estimate the centre of the fibre at frame z as

$$\vec{p}_{i,z} = \mathbf{p}_{i,z-1} + (\mathbf{p}_{i,z-1} - \mathbf{p}_{i,z-2}) \tag{3.3}$$

For every fibre j starting at frame $z+1$ we estimate

$$\vec{p}_{j,z} = \mathbf{p}_{j,z+1} - (\mathbf{p}_{j,z+2} - \mathbf{p}_{j,z+1}) \tag{3.4}$$

Then, we link fibre i to j if $|\mathbf{p}_{i,z} - \mathbf{p}_{j,z}| < (r_{i,z} + r_{j,z})/2$ This is to ensure that the

difference between the predicted fibre that ends at frame $z-1$ and the fibre that starts at frame $z+1$ is less than the mean radii of the two fibres. The second constraint is that the two fibres radii ratio should satisfy $0.7 \leq |r_{i,z}/r_{j,z}| \leq 1.5$. This is to ensure that the two fibres' radii have similar radius.

3.3.2 Correcting Misconnections

When two fibres approach closely or a fibre changes direction, the connection algorithm above can create mis-matches or lose track. This is particularly common for lone fibres (which leave one bundle and join another) or when two bundles split or merge. Because of their stiffness, fibres tend to be fairly straight and thus the position of their centre point in a slice varies smoothly from one slice to another. We use a Kalman Filter (KF) framework to predict the position of the fibre centre in sequential frames, and thus identify when a mismatch may have occurred.

Each fibre is associated with its own KF. The filter is initialised by the first five slices using a fibre's centre location. In subsequent slices we use the Kalman equations to predict the fibre's location in the next slice and compare it against that produced by the original linking algorithm. Where it finds a discrepancy, that position is recorded and the filter is re-initialised on the next 5 slices.

Each discrepancy is then re-visited, and corrected where necessary using **Algorithms 1** and 2. There are several cases to deal with, including gaps caused by missing detections, two fibres which have been incorrectly swapped when they are close and cases where one fibre has terminated.

The algorithm is run forward and then backward through the volume to locate as many such cases as possible.

Algorithm 1: Correcting Misconnections

```
Input: Fibre fibres_n; // list of fibres where n is number of
       fibres; each fibre consist of set of disks.
Disk disk1, disk2 // Disk is an object that has centre x,y and
       slice number z, all initialised by -1
for i = 1 ← to n do
    disk1 = FindDiscrepancyUsingKalman(fibre_i);
    if (disk1.z() != −1) then
        fibre1 = RetrieveFibre(disk1);// retrieve fibre that has
            disk1
        disk2 = FindDiscrepancyUsingKalman(fibre1);
        if (disk2.z() != −1) then
            fibre2 = RetrieveFibre(disk2);
            if (fibre_i == fibre2) then
                // kalman has agreed that there is a problem in
                    the two fibres at slice z
                BreakFibres(fibre_i, fibre1, disk1.z()); // break the
                    two fibres and swap them at slice z
```

Algorithm 2: Find Discrepancy Using Kalman

```
Input: Fibre fibre_n; // fibre where n is the length of fibre.
Disk disks_z // load all the disks for each slice where z is the
       slice number.
Disk KDisk // disk suggested by kalman, x,y,z initialised by -1
for i = 1 ← to n do
    if (i < 6) then
        // initialise kalman filter using fibre centre location
            of the first 5 slices
        UpdateKalmanMeasurement(fibre_i);
    else
        // predict next disk at slice fibre_i.z().
        disk = PredictNextDisk();
        if (isTwoDisksMatched(fibre_i, disk)) then
            UpdateKalmanMeasurement(fibre_i);
        else
            if (isDiskMatchedWithAnyFibre(disk, disks_{fibre_{i.z()}})) then
                KDisk = disk;
return KDisk
```

3.3.3 Lone Fibres

Most fibres are grouped together forming bundles. However, some of the fibres leave their bundle and join others, or enter other structures. Biologists are interested in the path of such lone fibres. Lone fibres leave a bundle as a single fibre or as a small group of fibres. They might leave their bundle and merge with another or they might return to the original bundle. Examples of such fibres are shown in Figure 3.5. The challeges in identifying lone fibres:

- Fibre start and exit from image boundary.
- Image (slice) corruption by 'tearing' the surface of the block.
- Number of neighbours for lone fibre might be more than one.

To identify such fibres we scan each fibre and count the number of neighbouring fibres within a range of r_L within each slice. We set the r_L radius to 20 pixels, and the number of neighbouring fibres within that radius should be at most 3 for a lone fibre.

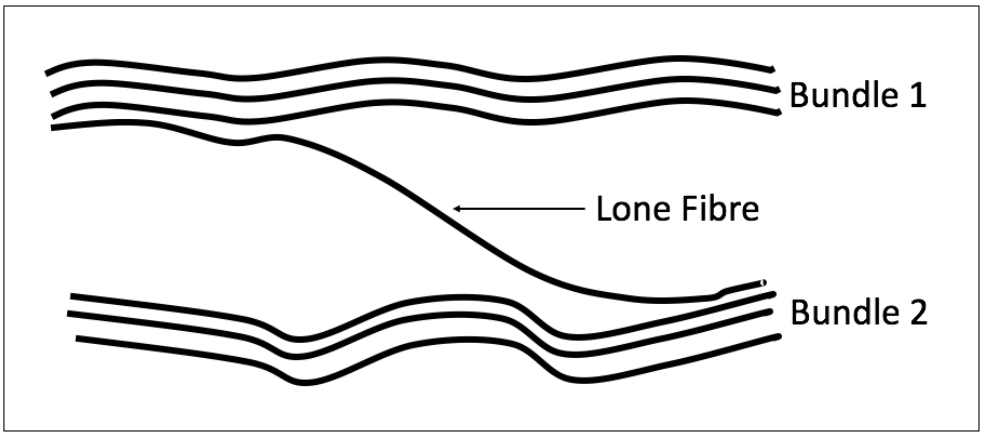


Figure 3.5: Lone fibre moving from bundle 1 to bundle 2.

Any fibre which has more than two slices satisfying our constraints with no other neighbours (excluding those close to the image boundaries) is labelled as a lone fibre. These constraints are set by biologists to identify lone fibres that are moving a small distance from their bundles of far away. The procedure for identifying lone fibres can be seen in **Algorithm 3**.

Algorithm 3: Lone Fibres Identification

Input: Fibre $fibres_{nm}$; `// list of fibres where` n `is number of fibres and` m `is the length of fibre; each fibre consist of set of disks.`

Define a circle $c_{x,y}$ with radius r_L

`// the radius is set to 20`

for $i = 1 \leftarrow$ **to** n **do**

  for $j = 1 \leftarrow$ **to** m **do**

    $c_{x,y} \leftarrow fibers_{ij(x,y)}$ `// assign the center of fibre` f_i `at slice` j `to the center of circle` c

    numberOfNeighbours $\leftarrow$ countNumberOfFibres($fibres_{nm}$,$c_{x,y}$)

      `// function to count number of fibres that center falls inside the circle` $c_{x,y}$ `with the radii of` r_L `at slice j`

    if $(numberOfNeighbours < T)$ **then**

      `// T is a fixed threshold set by biologist to 3`

      loneFibre.add(f_i)

3.4 Experimental Results

3.4.1 Data

We use images from three different datasets; (i) an embryonic 16.5 day wild type mouse tail sample used as a control for an MT1 knock out protease that cleaves collagen molecules (among other things), (ii) an embryonic 17.5 day wild type mouse tail sample used as a control for a collagen mutation that protects the fibrils from cleavage, (iii) an embryonic wild type mouse close to 17.5 day used as a control for a collagen receptor knock-down mouse. The other datasets were collected from wild type mouse tail in different timinig and ages (seet Table 3.3).

The datasets were collected by a SBFSEM system [5] under the brand name 3View. The images were created by collecting the back scattered electrons before an in-chamber ultramicrotome removes a section [7] with the assistance of Gatan DigitalMicrograph software.

Dataset	Number of Images	Resolution
1	182	4096×4096
2	510	1000×1000
3	500	966×876
4	500	4096×4096
5	695	4096×4096
6	700	4096×4096
7	1400	4096×4096
8	700	4096×4096
9	500	4096×4096
10	520	4096×4096
11	510	4096×4096
12	500	4096×4096
13	500	4096×4096
14	510	4096×4096
15	800	4096×4096
16	876	4096×4096

Table 3.3: Our data consists of 16 different datasets.

Annotated Data To quantitatively assess the performance of the tracking algorithm we manually annotated random fibres across several slices. We chose the first slice randomly from the image stack. The location of the centre of each fibre was recorded on each image in a sequence.

Several fibres may exit the image boundary or simply end before we reach the last slice. Thus, not all fibres are visible in all the slices (see Table 3.4).

Dataset	# of Annotated fibres	# of Images	# of Points
1	209	102	21019
2	64	30	1920
3	83	30	2447
4	87	20	1740
5	54	30	1620
6	103	30	2725
7	59	30	1770
8	83	30	2223
9	48	30	1440
10	60	30	1611
11	72	30	1896
12	62	30	1734
13	44	30	1320
14	47	30	1410
15	51	30	1530
16	48	30	1440

Table 3.4: The ground truth for each dataset.

3.4.2 Reconstruction Evaluation Measure

To quantitatively assess the performance of the algorithm we defined the following metric. Let $m_j(z)$ define the position of the marker for jth fibre at the zth slice.

The tracking for fibre i at slice z is defined as correct if $m_i(z)$ falls inside the circle that is identified by the detection and tracking algorithm for the outline

of that fibre. We also report mismatched errors that occur when two fibres are swapped as they pass close to each other. Then we consider whether the fibre is correctly tracked after the error correction algorithm (see Figure 3.6).

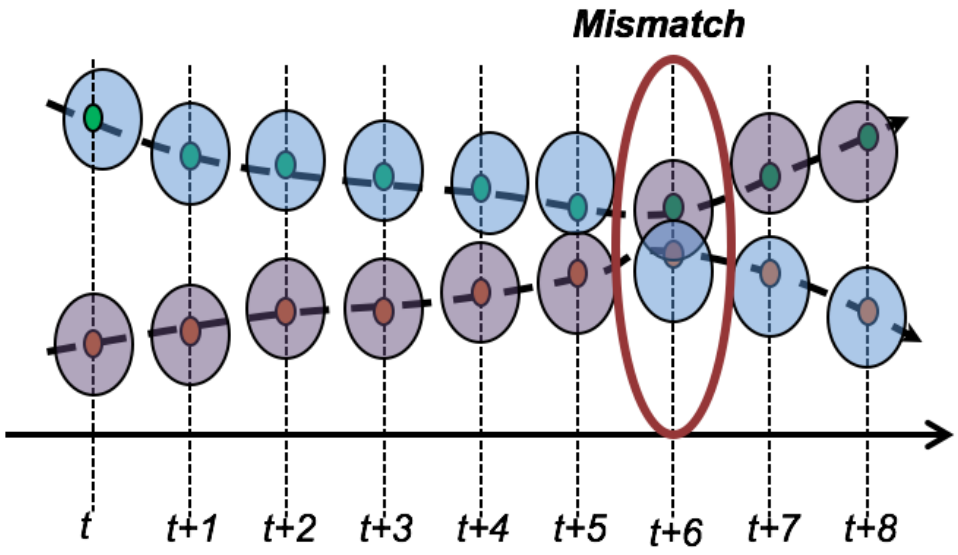


Figure 3.6: Evaluation measure criteria.

3.4.3 Evaluating 3D Reconstruction

In these experiments we only report the true positive rate since there are over 20000 fibres in the image - we cannot identify false positives (other positives may correspond to correct but unannotated fibres). The algorithm tested against the ground truth in Table 3.4. Table 3.5 summarises the algorithm reconstruction performance for each dataset after applying gap filling and correcting misconnection. The average accuracy score over all of the datasets is 94%.

Dataset	Accuracy
1	99%
2	96%
3	99%
4	88%
5	95%
6	94%
7	91%
8	92%
9	87%
10	90%
11	97%
12	98%
13	93%
14	96%
15	95%
16	96%

Table 3.5: The algorithm performance on each dataset

3.4.4 Evaluating Gap Filling and Kalman-Filter

The tracking algorithm result tested against the ground truth. We excluded the false positives since there are over 20000 fibres in the image in several datasets. The gap filling algorithm aims to correct missed fibres where the Kalman-Filter aims to correct mismatched as describe in section 3.4.2. We selected five datasets randomly to show the two algorithms' performance. Tables 3.6 - 3.11 shows the tracking performance, with gap filling and with Kalman-Filter based correction.

The gap filling algorithm corrects over 60% on Dataset 1 as shown in Table 3.6, where it corrects over 79% of the miss errors on Dataset 3 as shown in Table 3.7. The gap filling performance on other datasets were not significant because the failures at detection stage were high compared to those in Dataset 1 and 3.

On the other hand, the Kalman-Filter based method corrects over 70% of the matching errors on Dataset 1 as shown in Table 3.6 and between 63% to 69% on

the other datasets. However, on Dataset 4 we have not identified any mismatches as shown in Table 3.8 since fibres in this dataset are collected from an adult mouse. Thus, fibres become more straight at this stage unlike fibres in embryonic or newborns where they are more curvy.

	Method		
	Basic system	with Gap Filling	with Kalman Filter
Correctly Tracked	20835	20949	20951
Miss	184	70	68
Mismatched	289	289	82

Table 3.6: Improvements given by Gap Filling and Kalman-Filter based correction on dataset 1.

	Method		
	Basic system	with Gap Filling	with Kalman Filter
Correctly Tracked	2404	2438	2442
Miss	43	9	5
Mismatched	62	62	23

Table 3.7: Improvements given by Gap Filling and Kalman-Filter based correction on dataset 3.

	Method		
	Basic system	with Gap Filling	with Kalman Filter
Correctly Tracked	1450	1532	1537
Miss	290	208	203
Mismatched	0	0	0

Table 3.8: Improvements given by Gap Filling and Kalman-Filter based correction on dataset 4.

We have performed two experiments on two data sets to show the tracking performance with and without gap filling on the full depth of the block. Figure 3.7

	Method		
	Basic system	with Gap Filling	with Kalman Filter
Correctly Tracked	2490	2532	2584
Miss	235	193	141
Mismatched	42	42	13

Table 3.9: Improvements given by Gap Filling and Kalman-Filter based correction on dataset 6.

	Method		
	Basic system	with Gap Filling	with Kalman Filter
Correctly Tracked	1094	1142	1231
Miss	226	178	89
Mismatched	93	93	31

Table 3.10: Improvements given by Gap Filling and Kalman-Filter based correction on dataset 13.

	Method		
	Basic system	with Gap Filling	with Kalman Filter
Correctly Tracked	1291	1328	1388
Miss	149	112	52
Mismatched	39	39	14

Table 3.11: Improvements given by Gap Filling and Kalman-Filter based correction on dataset 16.

shows a histogram of the number of fibres with particular lengths (number of consecutive frames in which they are located) when analysing a block of images.

As can be seen from the figures many more longer fibres are tracked when gap filling is applied. The first experiment shows the number of fibres with a length 183 (the full depth of the block) was about 200. However, after applying gap filling the number increased to 4800. In the second experiment the number of fibres at the length of 150 and 200 were very small number about 10, but after applying gap filling the number of fibres increased to 200.

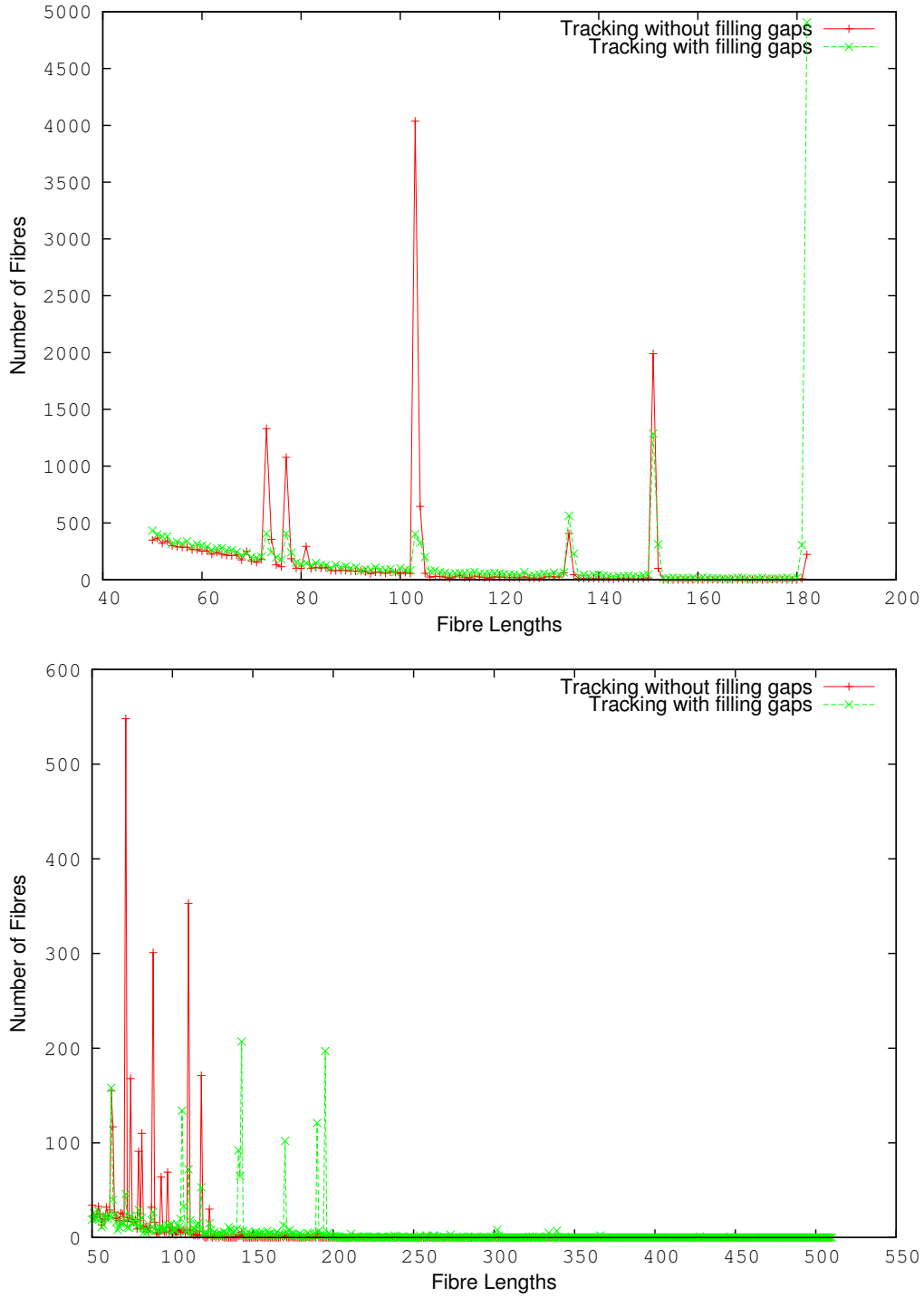


Figure 3.7: Histograms of the number of fibres with particular lengths on dataset 1 (top) and dataset 2 (bottom), with and without gap-filling. It demonstrates that the gap filling significantly increases the number of longer fibres detected.

Similarly we have performed three experiments on three datasets to show the tracking performance with gap filling after applying Kalman-Filter based correction. Figure 3.8 shows a histogram of the number of fibres with particular lengths (number of consecutive frames in which they are located) when analysing a block of images.

As can be seen from all the figures, many more longer fibres are tracked when the Kalman Filter is applied. The first experiment shows the number of fibres with a length 183 (the full depth of the block) were about 5000. However, after applying the Kalman based error correction the number increased to 6600. On the second experiment the number of fibres at length 220 and 290 the number of fibres increased about 100, and at the length 240 the number of fibres increased about 200. The third experiment showed that the number of fibres have increased after applying Kalman at length 500 from 8300 to 13500.

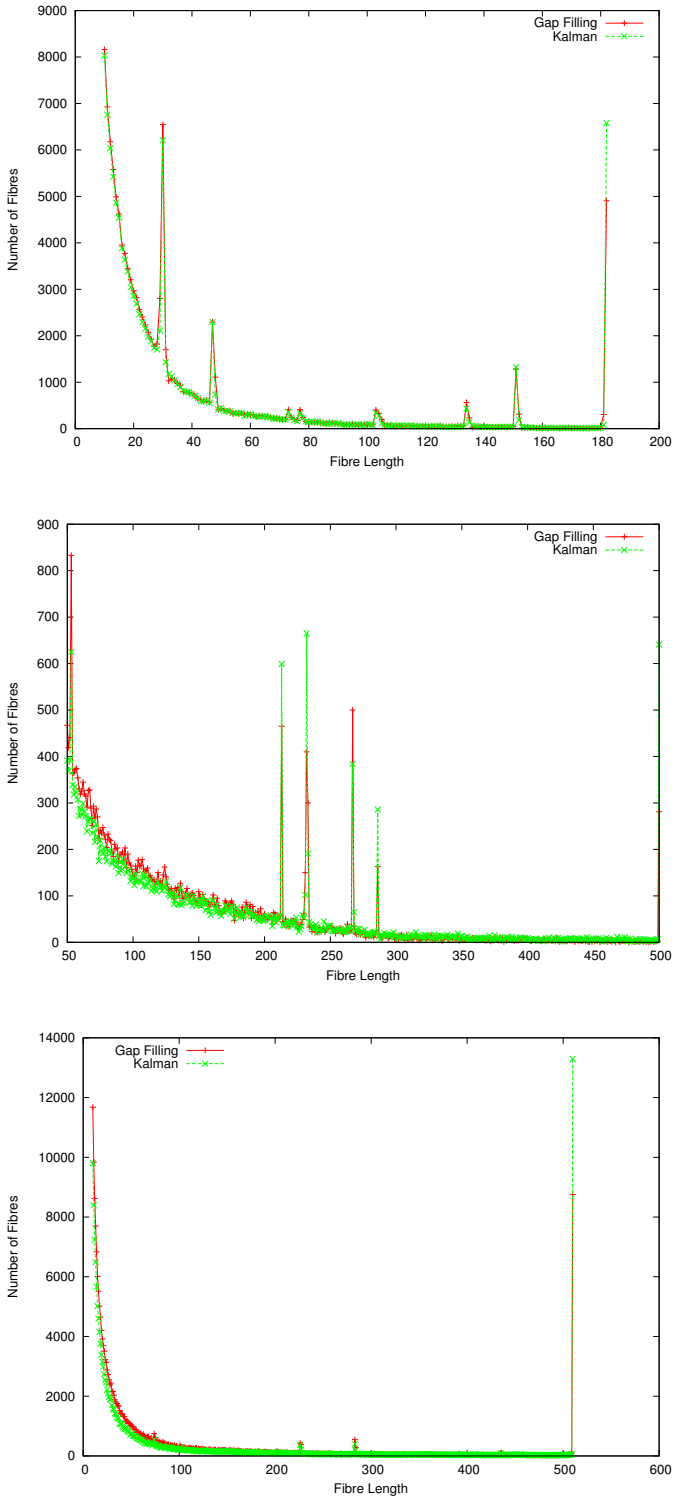


Figure 3.8: Histograms of the number of fibres with particular lengths on three data sets, with gap-filling and with Kalman correction, performed on Dataset 1 (above), Dataset 2 (middle) and Dataset 3 (bottom). It demonstrates that Kalman significantly increases the number of longer fibres detected.

3.4.5 Evaluating Random Forest for Eliminating False Matches

We performed experiments to evaluate how well the random forest classifiers could discriminate between fibres and non-fibre candidates found by the NCC-based models. Manual annotation of the output of the NCC models on a set of images gave 53480 true fibre candidates and 47022 non-fibre candidates, which were used for training and testing the classifier. We randomly split the examples into two sets where 70% of the examples are used for training and the remainder for testing.

We trained eight RF classifiers, each consisting of 20 trees, on samples with mean radii of 4, 5, 7, 11, 13, 15, 17 and 19 pixels. Figure 3.9 shows classifiers performance for each radius.

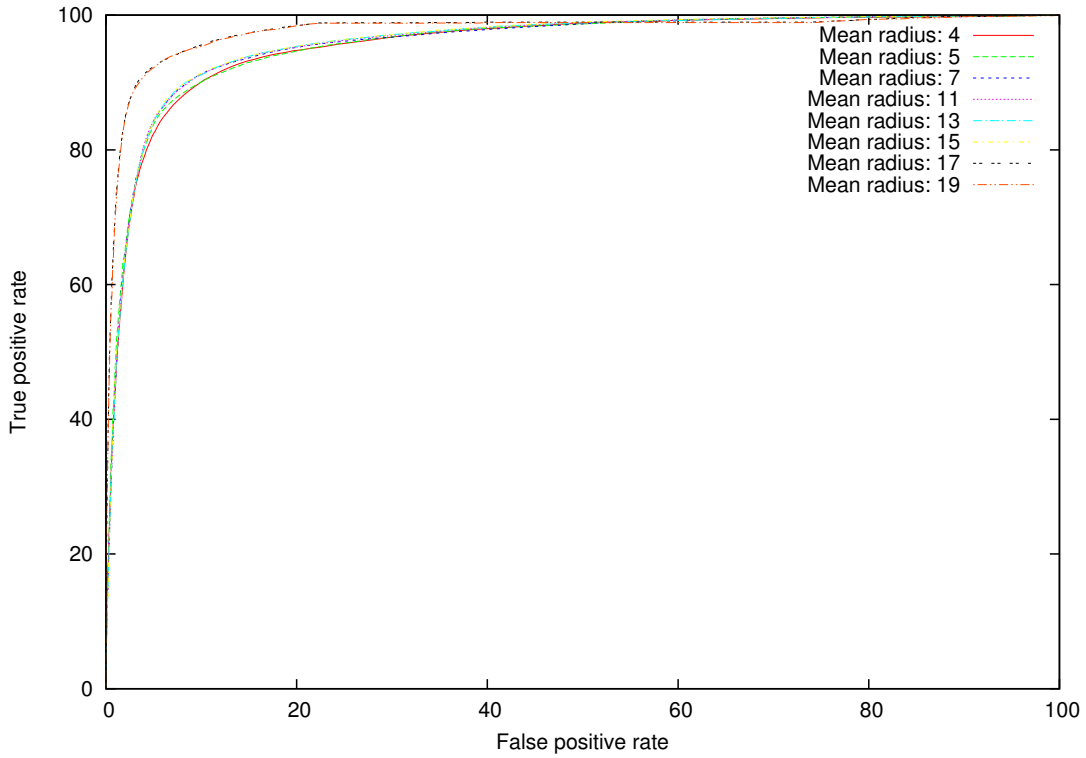


Figure 3.9: ROC for Random Forest classification performance for models of different radii.

3.4.6 Evaluating Random Forest for Fibre Detection

We performed experiments to evaluate how well the random forest classifiers could discriminate between fibres and non-fibre patches. Manual annotation on a set of images gave 952 true fibre candidates and 3808 nearby non-fibre candidates, which were used for training and testing the classifier. The testing data consists of 363 true fibres and 1452 non-fibres. To increase the training and testing data we performed rotation and scaling on the data.

We train eight RF classifiers, each consisting of 10 trees, with mean radii of 4, 5, 7, 9, 11, 13, 15 and 17 pixels. Figure 3.10 shows the Random Forest performance for each radius.

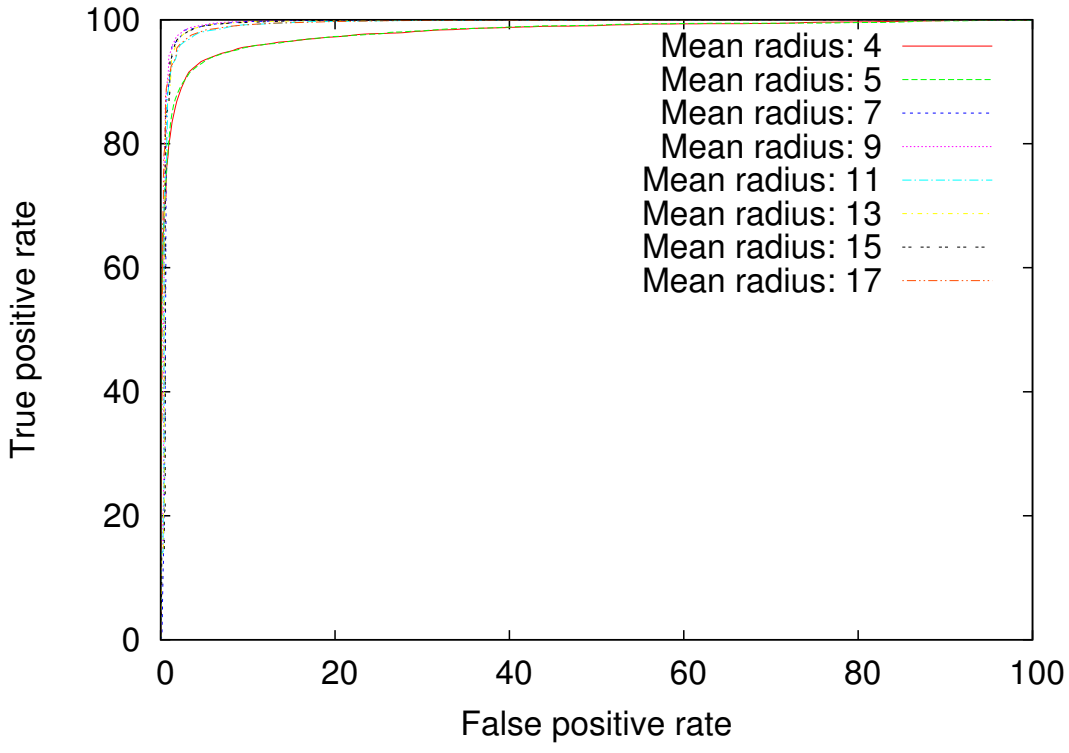


Figure 3.10: ROC for Random Forest classification performance for models of different radii.

To compare our detection algorithm using RF with [92] (based on NCC). we manually annotated 531 fibres across four images from Dataset 4. Fibres at this dataset have a more polygonal shape rather than a circular shape.

We performed an experiment to compare Random Forest classifier for locating fibres with the NCC based template search. We compute the precision and recall, where we define recall = TP/(TP+FN) and precision = TP/(TP+FP), where TP is the number of true positives, FN is the number of false negatives and FP is the number of false positives. We summarise the precision-recall in single number using the F1 score:

$$F1 = \frac{2 \cdot Recall \cdot Precision}{Recall + Precision} \tag{3.5}$$

The F1 score for finding fibres using template matching achieved 62% where the Random Forest classifier achieved 96%.

3.5 Fibre Analysis

We performed several quantitative analyses of collagen fibres on each dataset. Biologists are interested in quantifying the fibre's length, radius, deviation and curvature, as these measurements will assist them in studying and analysing the structures and behaviours of fibres. As may be seen in Figs. 3.7 and 3.8 in Section 3.4.4, a large number of fibres start from the top of the image stack and continue to the end of the volume. The short fibres end because the fibres may reach their end or exit the image boundaries or because an error occurs in the detection or tracking phase. These data require analysis by an expert in the field in order to compare the fibres' lengths across samples collected at various times. We also quantify the distribution of the fibres' radii across each dataset (see Fig. 3.11). This will allow experts to study the growth of the fibres and the contribution of small and large fibres to maintaining the tissue's strength. For example, the fibres' radii in datasets 1 and 2 are almost identical, while in datasets 3 and 4 the fibres' radii range over three distinct categories.

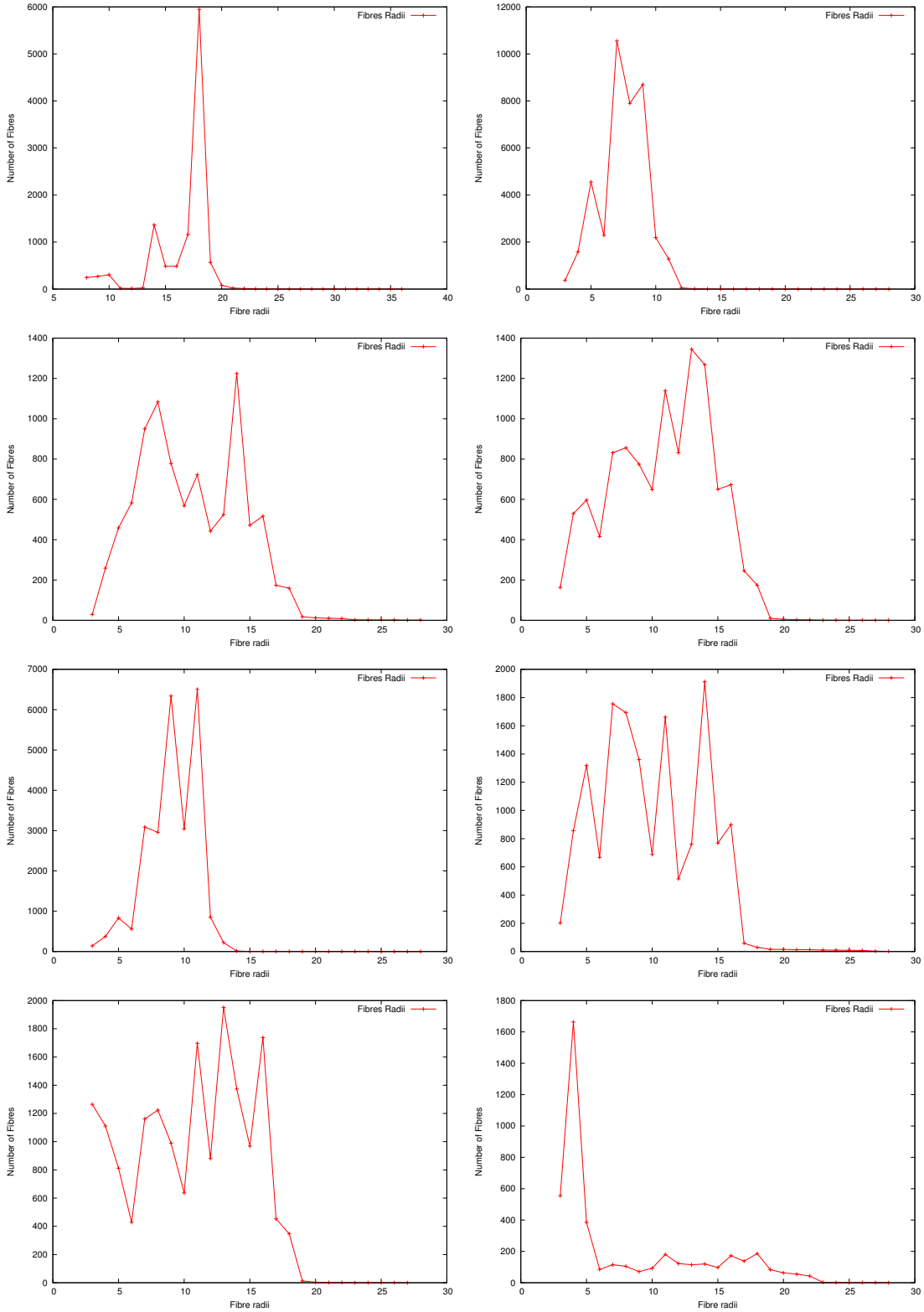


Figure 3.11: Fibres radius distribution of Dataset 4,6,7,10,13,14,15 and 16.

3.5.1 Large Scale Curvature

To measure the curvature for each cluster, we divide fibres into 3 clusters based on their radii. Suppose that a fibre runs from slice z_0 to slice z_1, with position $p_0 = (x_0, y_0)$ at z_0 and $p_1 = (x_1, y_1)$ at slice z_1. If we draw a straight line between the ends, then that passes through slice z at position:

$$L(z) = (1 - a) \times p_0 + a \times p_1, \text{ where } a = (z - z_0)/(z_1 - z_0) \tag{3.6}$$

If the fibre has a disk centred at $p(z)$ in slice z, then the deviation from the line is given by:

$$d(z) = |p(z) - L(z)|/(z_1 - z_0) \tag{3.7}$$

We calculate the statistics of $d(z)$ over the length of the fibre - its mean and max. We create a histogram of the number of fibres with each deviation (using 100 bins), then normalise to create a PDF (so that the area under the curve is 1).

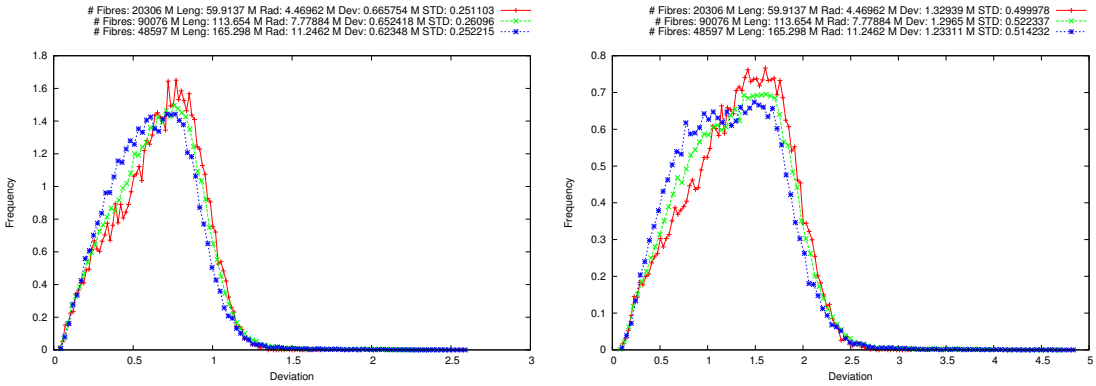


Figure 3.12: Mean (left) and max (right) fibres large scale deviation of Dataset 1.

Figure 3.12 and 3.13 shows the mean and the max deviation for dataset 1 and 7. The data shows that the fibres with small radii have slightly higher deviation though the difference is not big (fibres with small radii are not as long as fibres

with large radii). However, in dataset 16 the difference between the three fibres' radii is more obvious (see Fig. 3.14).

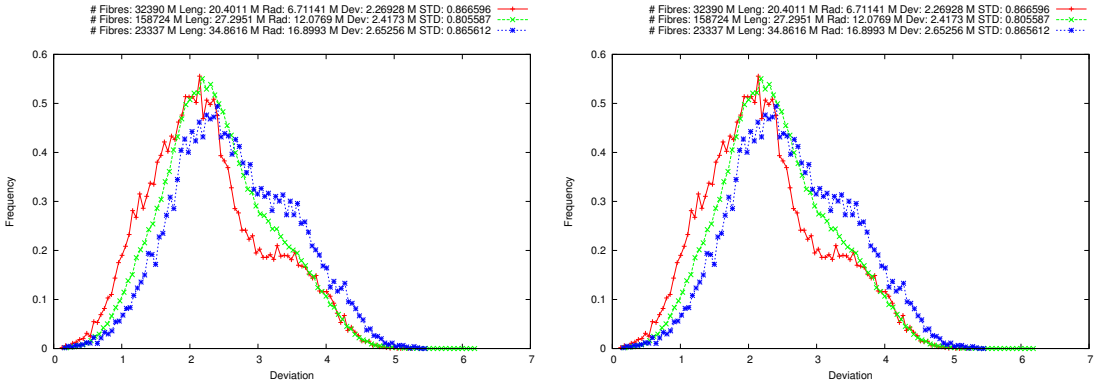


Figure 3.13: Mean (left) and max (right) fibres large scale deviation of Dataset 7.

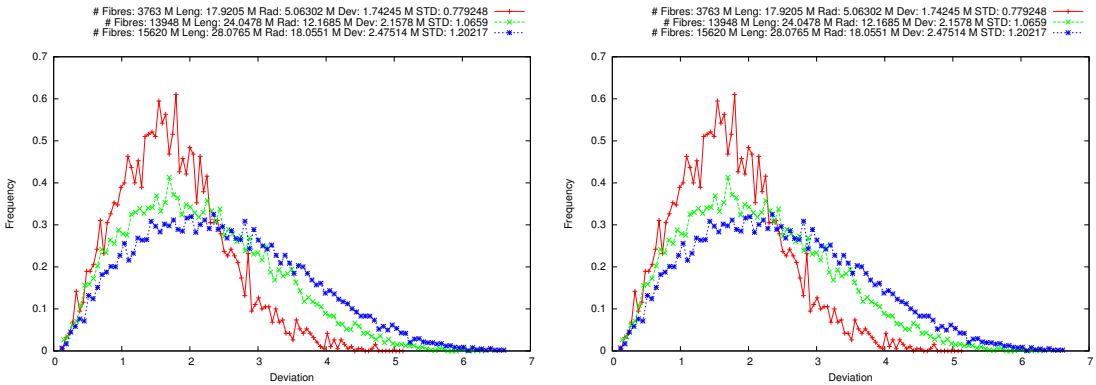


Figure 3.14: Mean (left) and max (right) fibres large scale deviation of Dataset 16.

3.5.2 Fibre Curvuture

Each fibre consists of a set of disks and each disk has centre $p(i)$ on slice i and radius r. We divide fibres into K clusters based on their radii so biologists can compare the curvature of fibres with different radii. Then we compute the mean curvature for every fibre at slice i:

$$Curv_i = \frac{|p_i - \frac{1}{2} \times (p_{i-S} + p_{i+S})|}{S} \tag{3.8}$$

$$\overline{Curv_i} = \sum_{s}^{N-S} \frac{Curv_i}{N - 2 \times S} \tag{3.9}$$

where S is the step size and N is the fibre length. We create a histogram of the number of fibres using step size $S = 15$ with each curvature (using 50 bins), then normalise to create a PDF (so that the area under the curve is 1). Figure 3.15 shows the curvature of different datasets where we divide the fibres into 3 clusters with different radii. It shows that more of the fibres with small radii have large mean curvature, compared to those with large radii, though the difference is not big.

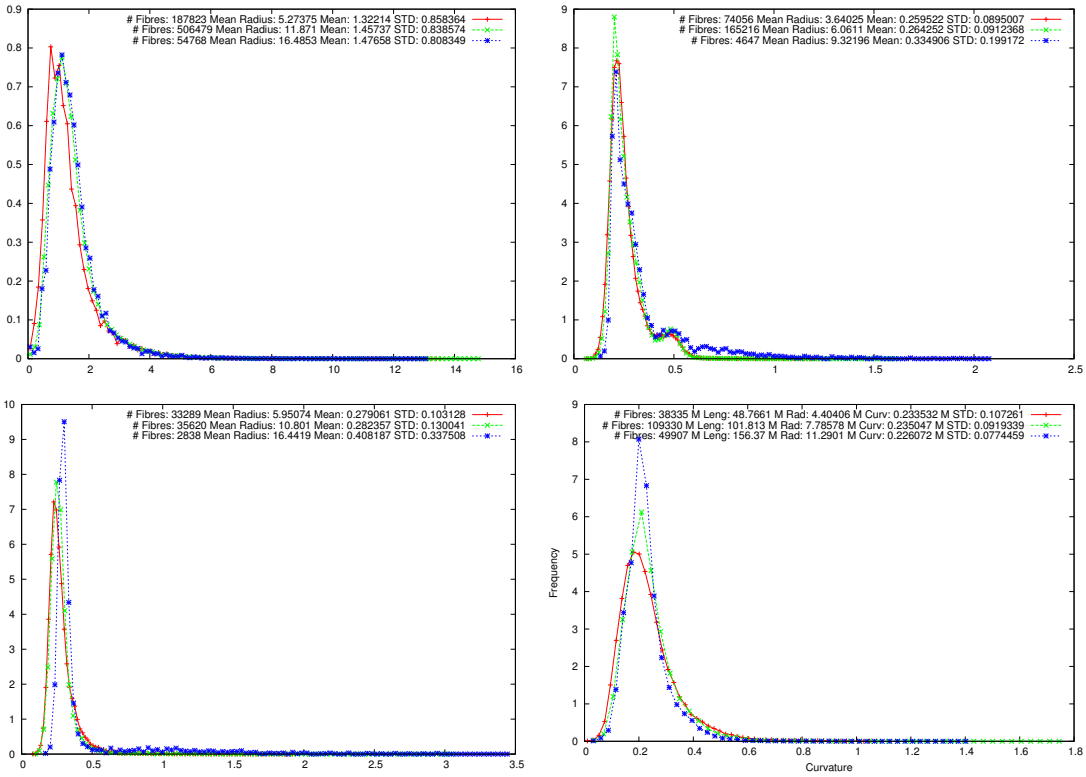


Figure 3.15: Fibres curvature of Dataset 8,12,14 and 16.

3.6 Discussion and Conclusions

We have demonstrated a fully automatic system for detecting fibres and tracking them across image volumes. The detection system involves finding candidates by using template matching and then discarding false matches by using a random forest classifier. We constructed multiple models, each trained to locate fibres of particular radii (see Fig. 3.16). The training samples collected to train the RF classifier were gathered after locating candidate fibres by using a template-matching approach. The training samples of true fibres and false fibres numbered in the thousands and were equally balanced.

During the late development of fibres, their shape becomes more polygonal, and, when we used the previous model, it failed to locate fibres and produced a large number of false positives. To improve the accuracy, we designed a new model for this specific shape by training a RF classifier to locate those fibres. We annotated the fibres by drawing a circle around the fibres and, to address false matching, we drew circles by displacing the true circle around the fibre. We then defined multiples models, each trained to locate fibres of particular radii (see Fig. 3.17).

The tracking algorithm, which takes account of the movement of fibre bundles, was found to be effective for linking the detected disks together into extended fibres. We applied a gap-filling algorithm to correct small gaps, as there were candidate disks for a fibre that was not detected due to failures in the detection stage or because some image slices were corrupted by tearing. However, when two fibres move toward each other from different directions, the algorithm may swap them or even do that when a fibre changes its direction suddenly. We used a Kalman filter to predict the location of a fibre in the next slice, based on the previous information about the fibres location. Each discrepancy between the tracker and the Kalman filter was re-visited and corrected when necessary. The gap-filling algorithm was designed to fix broken links, whereas the Kalman filter

was designed to deal with mismatched fibres. The two algorithms were able to increase the fibres lengths significantly.

The algorithm is able to track thousands of fibres across hundreds of slices (see Fig. 3.16). Figure 3.19 shows that the lone fibres follow complex paths through the tissue as compared to the fibres in bundles (shown in Fig. 3.20). We also visualise all fibres from datasets 4 and 11, where the number of fibres is more than 30,000 (see Fig. 3.21).

We evaluated the random forest classifier for eliminating false positives and for locating fibres. The experiments showed that the RF classifier was able to remove false positive fibres and locate fibres with high accuracy.

We evaluated our tracking algorithm against our ground truth, and we also evaluated the tracker after applying gap-filling and Kalman-filterbased correction. We showed the number of fixed broken links using gap filling and the number of corrected mismatched fibres using the Kalman filter. We showed histograms of a number of fibres of particular lengths on the full depth of the block of our tracker and compared the lengths after applying gap-filing and Kalman-based correction. From these experiments, we demonstrated how the tracker improved and how the fibres' lengths increased.

There are two main limitations of the current approach for reconstructing fibres. The first occurs in the detection phase, when the fibres are closely packed and overlapped. Moreover, the fibres took irregular shapes in some cases, due to errors that occurred when acquiring the image volume (see the first and second row in Fig. 3.18). These failures dramatically affect the tracking algorithms performance. The second limitation results from the fact that some fibres move faster than most of the fibres in the consecutive slices because they are perpendicular to

the surface of the sample. Those fibres leave their bundles for a certain time, and our algorithm assumed that all fibres in a bundle move in roughly the same way. However, in some cases, the whole bundle may move faster than is normal because it becomes oriented far from the normal to the surface (see the last row in Fig. 3.18).

In comparing our results to studies in general, a comparison of different approaches to reconstructing ultra-structures from EM images presents challenges for several reasons. These fall into three categories: tissue staining, imaging method and tissue type (we discuss these issues in detail in Chapter 5). However, we have not found any work or benchmark that is similar to our work in order to make a comparison.

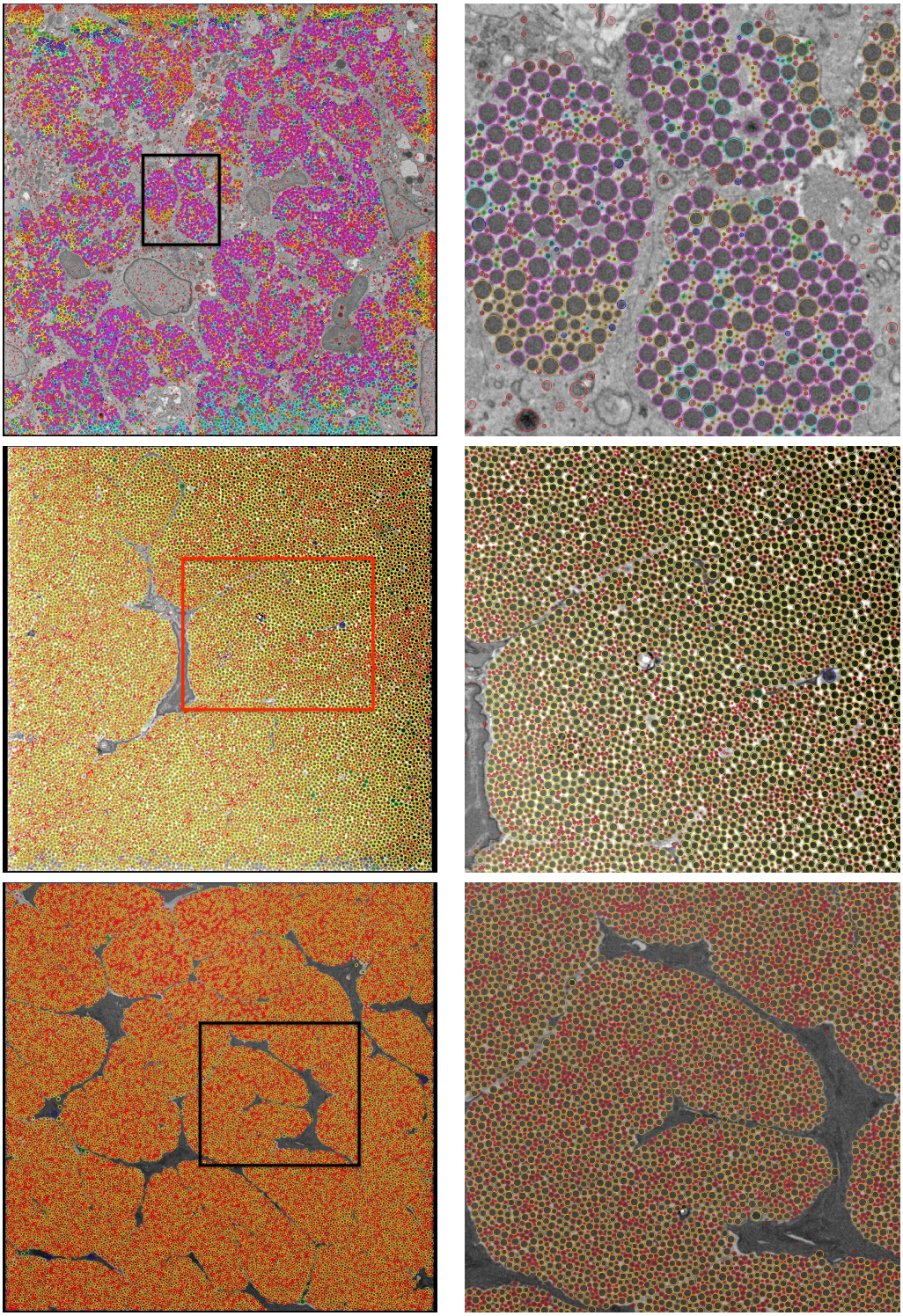


Figure 3.16: Images of tracked fibres from dataset 2, 11 and 15 (colours are given to fibres based on their radius).

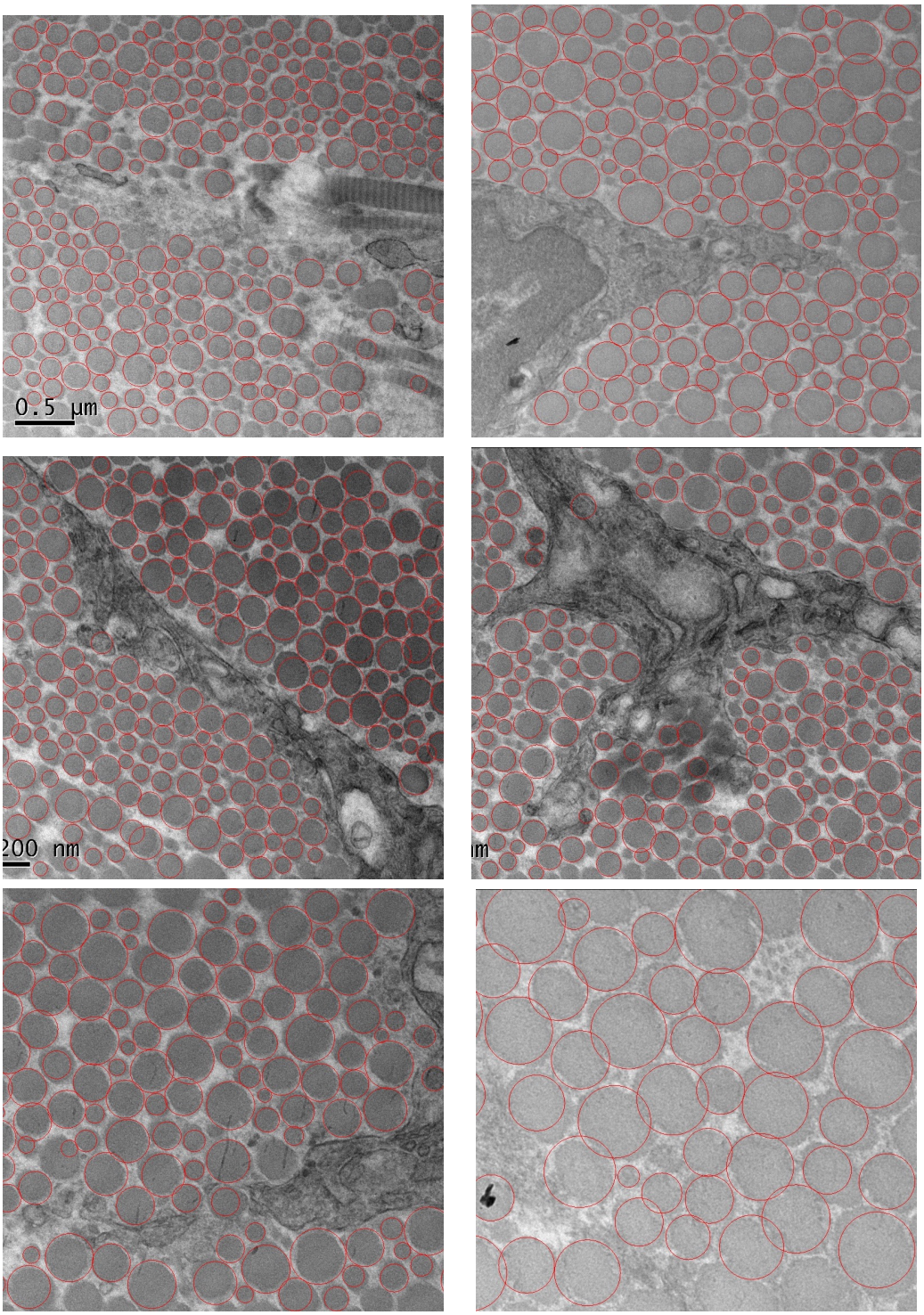


Figure 3.17: Example of Random Forest detection on dataset 3, 5 and 7.

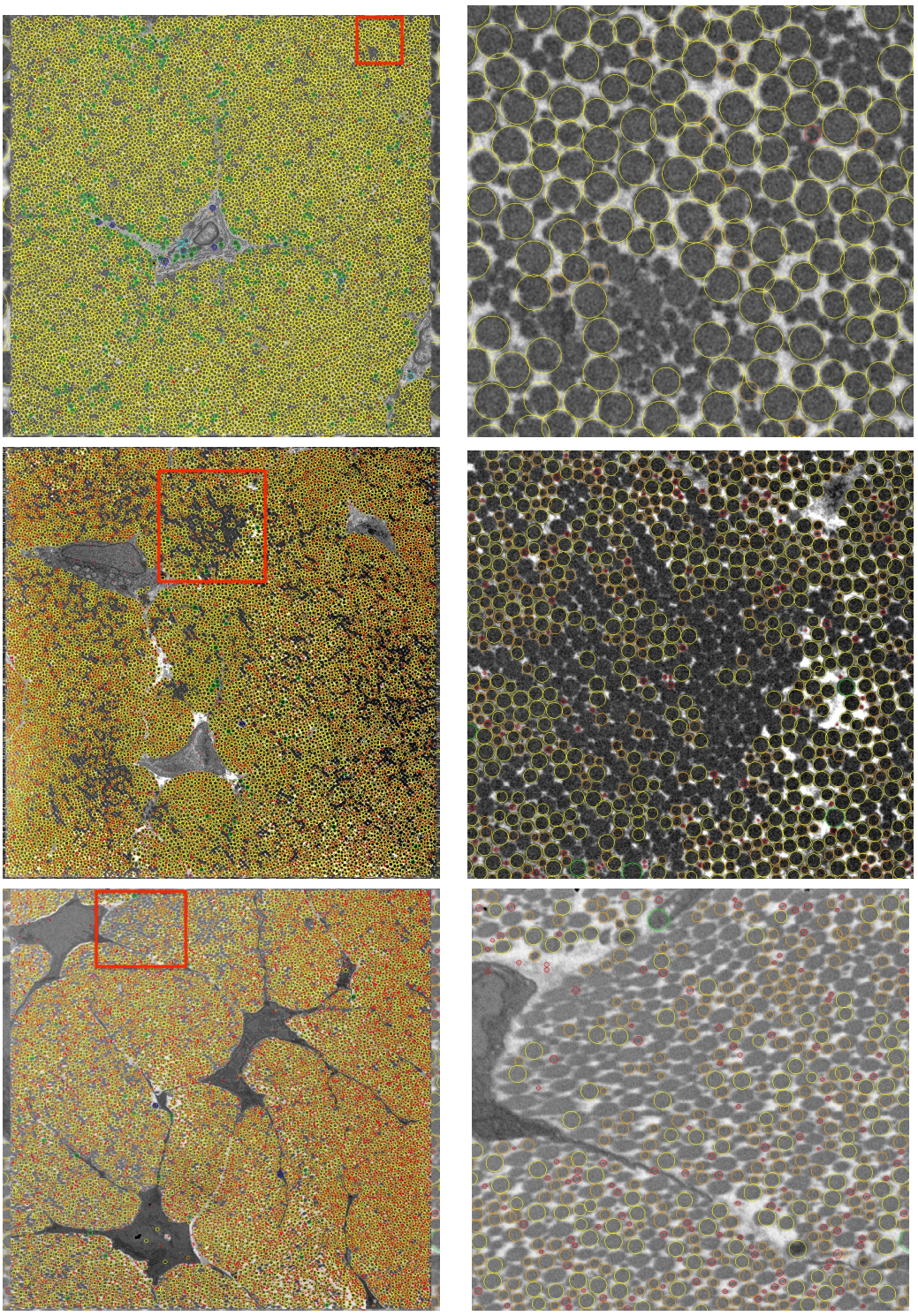


Figure 3.18: Images of failed detection of fibres in dataset 4, 8 and 10.

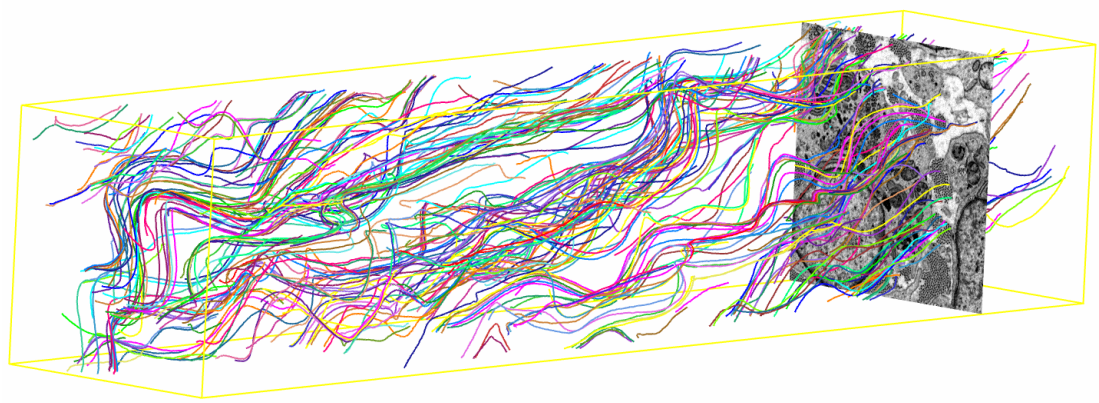


Figure 3.19: Visualisation of lone fibres found by the tracking algorithm on Dataset 1.

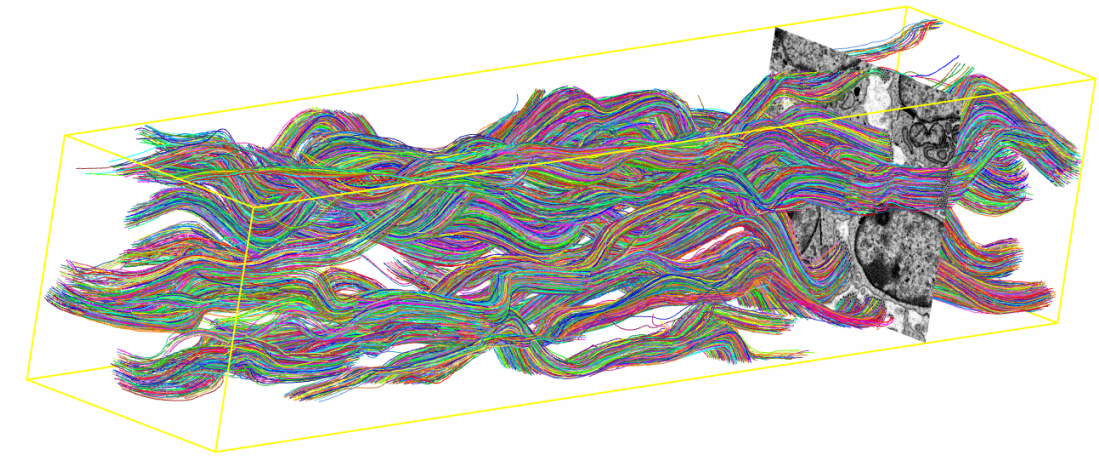


Figure 3.20: Visualisation of all fibres found by the tracking algorithm on Dataset 1.

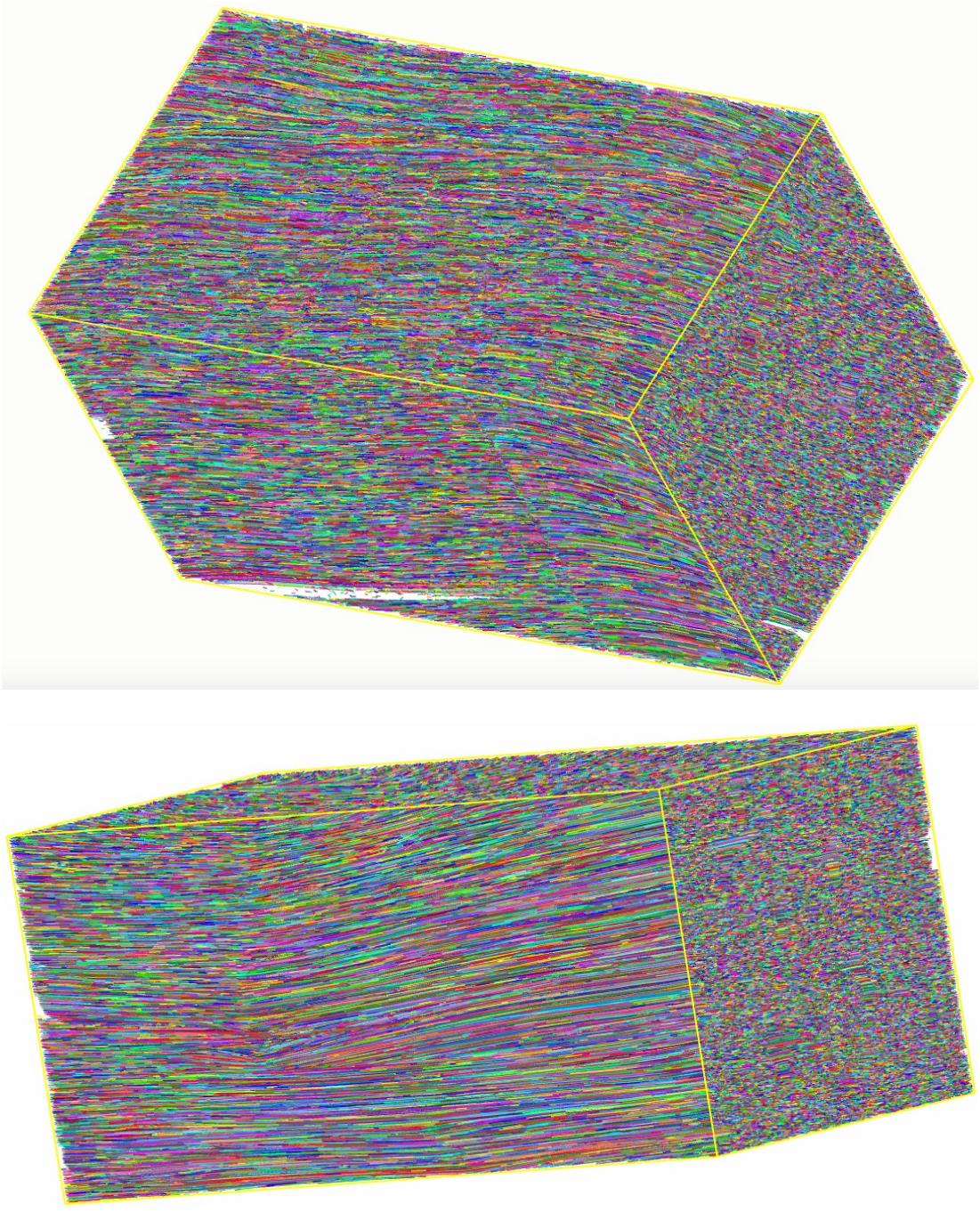


Figure 3.21: Visualisation of all fibres found by the tracking algorithm on Dataset 4 and 11.

Chapter 4

Nuclei Segmentation and Reconstruction

The reliable segmentation of nuclei from microscopic images is an important task in many biological studies. Studies on nuclei analysis can be dated back six decades [93]. The segmentation of cells has been analysed in a large number of research papers covering several domains, such as cell counting, cell types and phases, and cell migration and interaction. In general, there is a significant complexity and variability in cell image data, which makes segmenting cells from other objects a problem. Thus, different approaches are often used with different data sets. Most methods that are used in the literature for cell segmentation use a pipeline of segmentation based on filtering steps and mathematical operations combined with a few basic segmentation techniques, such as intensity thresholding, feature detection, morphological filtering, watershed transformation, or deformable model fitting [93].

The motivation for this work arose from certain processes within the field of biomedical imaging, such as detecting and tracking nuclei and mitochondria in a complex subcellular environment. This chapter introduces our approach for segmenting nuclei and reconstructing them from large image stacks that were

acquired using the SBFSEM technique.

4.1 Introduction

In recent years, automatic methods have achieved great success in nucleus segmentation. In the literature, several works on nucleus segmentation have been published [94]. It should be noted, however, that although these approaches might work efficiently on the authors' data sets, their clinical application is not yet viable. In [95], smoothing is applied to strengthen the nucleus' border and maintain the low contrast between background and cytoplasm. The edges are then enhanced by applying a gradient direction edge enhancement operator to obtain a better performance. Finally, the nuclear membrane is refined using a nucleus intensity threshold. Their approach was designed for isolated cells, however, not multiple cells.

The traditional approaches deal with only a single nucleus. It is more challenging to address more than one nucleus due to the uneven illumination. Thus, it is essential to develop a new automatic tool for segmenting nuclei for biologists to study nuclei shapes in 3D. This would allow them to perform measurements on the nuclei, such as their length, radii, and volume, and to study their structure.

For nucleus detection, most of the existing work in literature on EM images was performed to detect nuclei centroids. Xu et al. [96] used stacked sparse auto-encoder on high-resolution histopathological images for nuclei detection by learning high level features of nuclei using pixel intensity. Recently a structural regression CNN [97] has been proposed which learn a proximity map of nucleus. They modify the typical CNN by replacing the last layer with structured regression layer to encode topological information.

We use a multistage approach to segment nuclei in a complex environment in microscopic image data. The method processes each slice (as a 2D image) to find the boundaries of the nuclei, then tracks them and links each nucleus across the slices. In the first step, we train a classifier that can classify each pixel according to whether it belongs to a nucleus membrane or not. Once we classify every pixel in the image, a binary image is produced showing the nuclei membranes and the boundaries of other objects, such as mitochondria and Golgi apparatus. These objects' membranes share some features of nuclei membranes and need to be eliminated. The next process identifies the nuclei membranes and eliminates the other objects' membranes using post-processing methods that involve analysing every connected component and performing morphological operations, such as erosion and dilation.

In the final stage, we reconstruct every nucleus through the image stack. The nuclei radii in SBFSEM range from 200 pixels to 900 pixels. Finding the centre of gravity for each nucleus in the first frame and their matches in the next frame is sufficient to link them. This approach is easy to implement, relatively fast, and efficient at segmenting, quantifying, and tracking nuclei. Experiments on manually annotated data were also performed to measure the accuracy of the approach in terms of segmentation and the creation of a 3D structure.

4.2 Method

The aim of this chapter is to automate the process of identifying nuclei on each slice and reconstructing their 3D shape, allowing the nuclei structures to be measured and visualised in 3D. Biologists are interested in the shape, volume, length and radii of nuclei. The approach we take is to first detect candidates for nuclei in every image, then link the candidates across neighbouring images to form extended nuclei. The detection stage can make mistakes, which we attempt to

identify and correct in the linking stage.

Our solution for segmenting nuclei is to use deep learning. We used two approaches; (1) a pixel classification approach using a CNN scanned over the image with a sliding window, and (2) a U-Net which estimates the boundary pixels in one pass. The probability of a pixel being a nuclear membrane is computed by the two appraoches. Our models were trained on a different stack that shared similar characteristics, and the nuclear membranes were manually annotated.

4.2.1 Convolutional Neural Network Architecture

Our segmentation algorithm is based on training a CNN to perform pixel binary classification. The architecture of the network consists of six layers, four convolutional layers, and two fully connected layers, as shown in Figure 4.1. The output of the network is the probability that a pixel belongs to a nuclear membrane. To train the CNN, we extracted a local window N×N (we choose N = 64 as this provides a sufficient trade-off between the local information to be extracted and memory usage) and fed it into the network.

Several structures were tested using an input image (16×16) and (32×32) but the performance for each one was visually poor. However when we used (64×64) with different structures the performance was acceptable. To find the optimal structures we evaluated each structure against 15 images from our ground truth (Table 4.1 shows the accuracy obtained from each structure).

Network	layer1	layer2	layer3	layer4	layer5	layer6	layer7	layer8	layer9	layer10	layer11	Accuracy
1	conv (7x7)	maxp (2x2)	conv (5x5)	maxp (2x2)	conv (3x3)	maxp (2x2)	FC	FC	output			91%
1	conv (5x5)	maxp (2x2)	conv (5x5)	maxp (2x2)	FC	output						87%
2	conv (7x7)	maxp (2x2)	conv (5x5)	maxp (2x2)	conv (4x4)	maxp (2x2)	conv (3x3)	maxp (2x2)	FC	FC	output	98%
4	conv (5x5)	maxp (2x2)	conv (4x4)	maxp (2x2)	conv (3x3)	conv (3x3)	max (2x2)	conv (3x3)	maxp (2x2)	FC	output	94%

Table 4.1: Accuracy obtained from different structures.

Our architecture is similar to [44], but our network includes a dropout layer to prevent overfitting [98]. Moreover, the kernel size and number of output maps is larger in the first convolutional layers to capture more information about variability in the low-level features. The output of the last fully connected layer is fed into a binary softmax to produce the probability of class label.

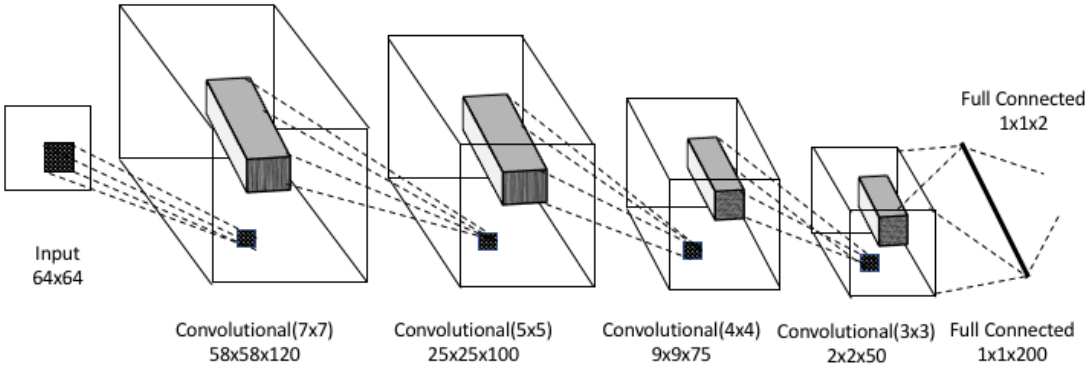


Figure 4.1: CNN architecture consisting of four convolutional layers and two fully connected layers.

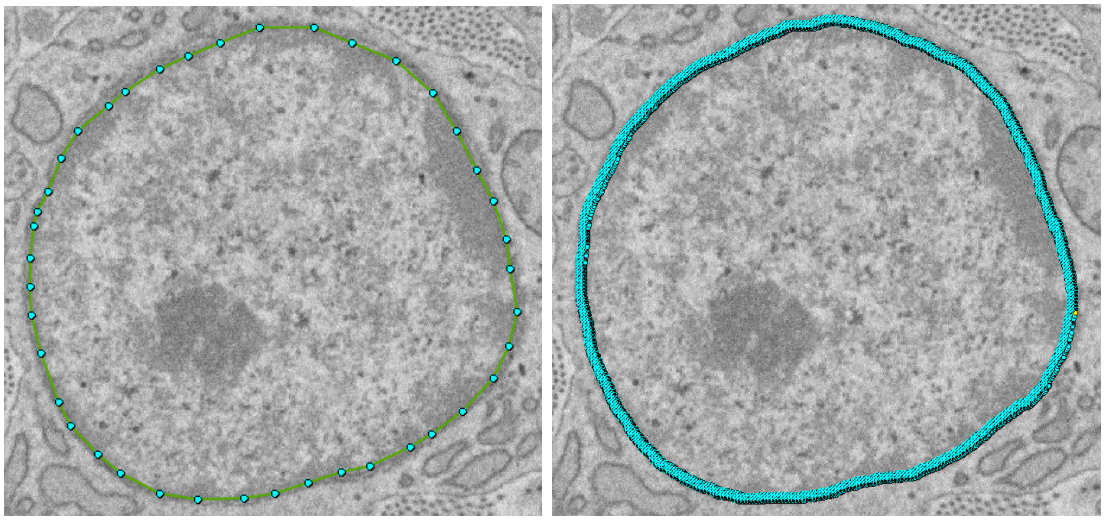
Each layer of convolution is followed by Rectified Linear Units (ReLU). Max pooling is applied to the output of each convolutional layer to prevent overfitting and down-samples by a factor of 2. The second-to-last fully connected layer is followed by a drop out layer.

The first convolutional layer takes the input image (64×64) and filters it with 125 kernels of size 7×7 with a stride of 1 pixel. The output of the first convolutional layer is fed into a max-pooling layer that down-samples the data by a factor of 2. The max-pooling layer output is then fed to the second convolutional layer to filter it with 100 kernels of size 5×5. The third convolutional layer has 75 kernels of size 4×4 , and filters the output of max pooling. The final convolutional layer has 50 kernels of size 3×3.

Training Examples

The training examples for nuclei segmentation were collected from a stack of images consisting of 157 slices with a resolution of 966×876. We chose 16 images randomly, and annotated the nuclear membranes by marking them with a curve. Other objects' membranes, such as Golgi and mitochondrial cytoplasm membranes, were annotated in the same way (see Figure 4.2a). As the convolutional network requires a large number of training examples, we generated points based on the annotated curve to increase the number of examples, as shown in Figure 4.2b. We then extracted patches with a size of 64×64 centred on each annotated point.

We employed data augmentation to increase our sample by applying rotation, scaling and reflection. We used eight different rotations and three different scaling for each sample. As we trained a binary classifier, we have two classes, namely, membrane or not-membrane. The total number of examples for each class after applying data augmentation was 600,000 samples, and 1.2 million training examples in total.



(a) Manual annotation. (b) generated points.

Figure 4.2: (a) Manual annotation around the nuclear membrane. (b) generated points based on the manual annotation

4.2.2 U-Net

We used U-Net (described in section 2.5.3) which is a multi-scale convolutional network. To train the networks we use the pixel-wise cross-entropy as a loss function with Stochastic gradient descent optimiser where the initial learning rate is set to 0.0001. The networks were trained for 30 epochs with batch size of 4. The input images and their corresponding segmentation map size were 512×512.

We train the networks on 24 images, each image annotated manually as can be seen in Figure 4.3. The images were selected randomly from 4 different datasets. The output mask was a pixel-by-pixel mask where each pixel either belonged to a nucleus' membrane or the background.

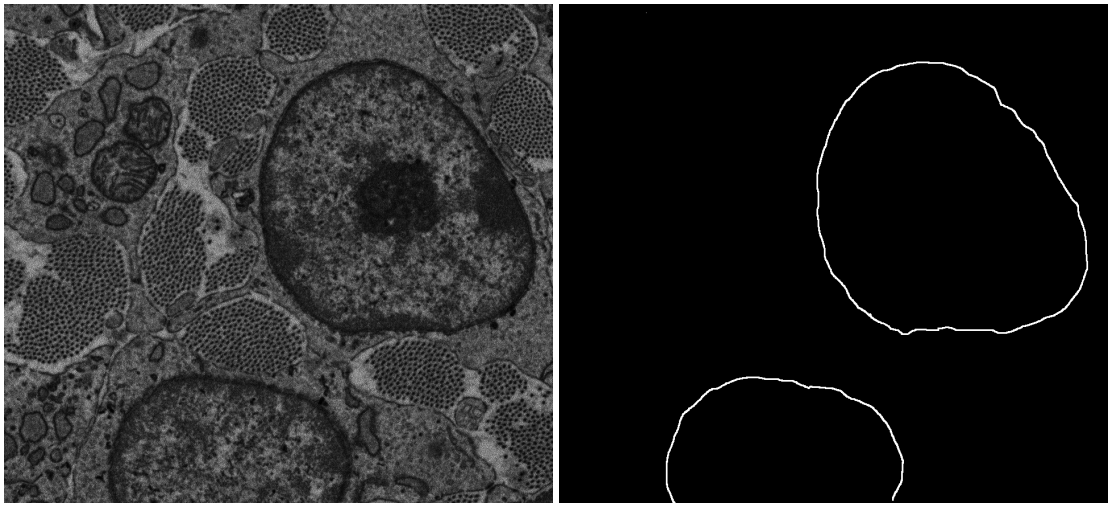


Figure 4.3: Original image (left). Segmentation map (right).

4.2.3 Post Processing

The segmentation results that we obtained from applying our classifier to classify each pixel as either foreground or background contained noise (see Figure 4.4b)

that would affect our nuclei segmentation and tracking in the next phase. Therefore, a post-processing stage was needed to eliminate the noise in the image. It is essential to separate the nuclear membranes from other objects' membranes, especially mitochondrial membranes as their membranes share some visual features with nuclear membranes. We applied post-processing on the two approaches (CNN results and U-Net results).

To achieve our goal of removing the noise and separating the nuclear membranes from other objects, we measured the width of the nuclear membranes in pixels to perform a series of iterations of erosion. The erosion process is performed by convolving the image with a kernel in order to isolate individual elements. We found that the average resulting membrane width was about 15 pixels. Therefore, we performed erosion three times (see Figure 4.4c).

Once we removed the noise, we analysed each connected component label to remove the remaining noise. The connected components labelling algorithm scans a binary image and groups its pixels into objects based on their connectivity. First, we measured the smallest nucleus size in a different image stack, and found that the smallest one had a radius of about 75 pixels. Therefore, we analysed each connected component and removed any connected component that was smaller than 75 pixels in height or width. The resulting image after removing the small connected components contained only the nuclear membranes. Figure 4.4d shows the nucleus after removing small connected components. The next step is to find the contour of the nucleus.

4.3 Nuclei Detection and Reconstruction

By this stage, only the nuclear membranes are present in the image. To find the nuclear membrane we just find the contour of every connected component.

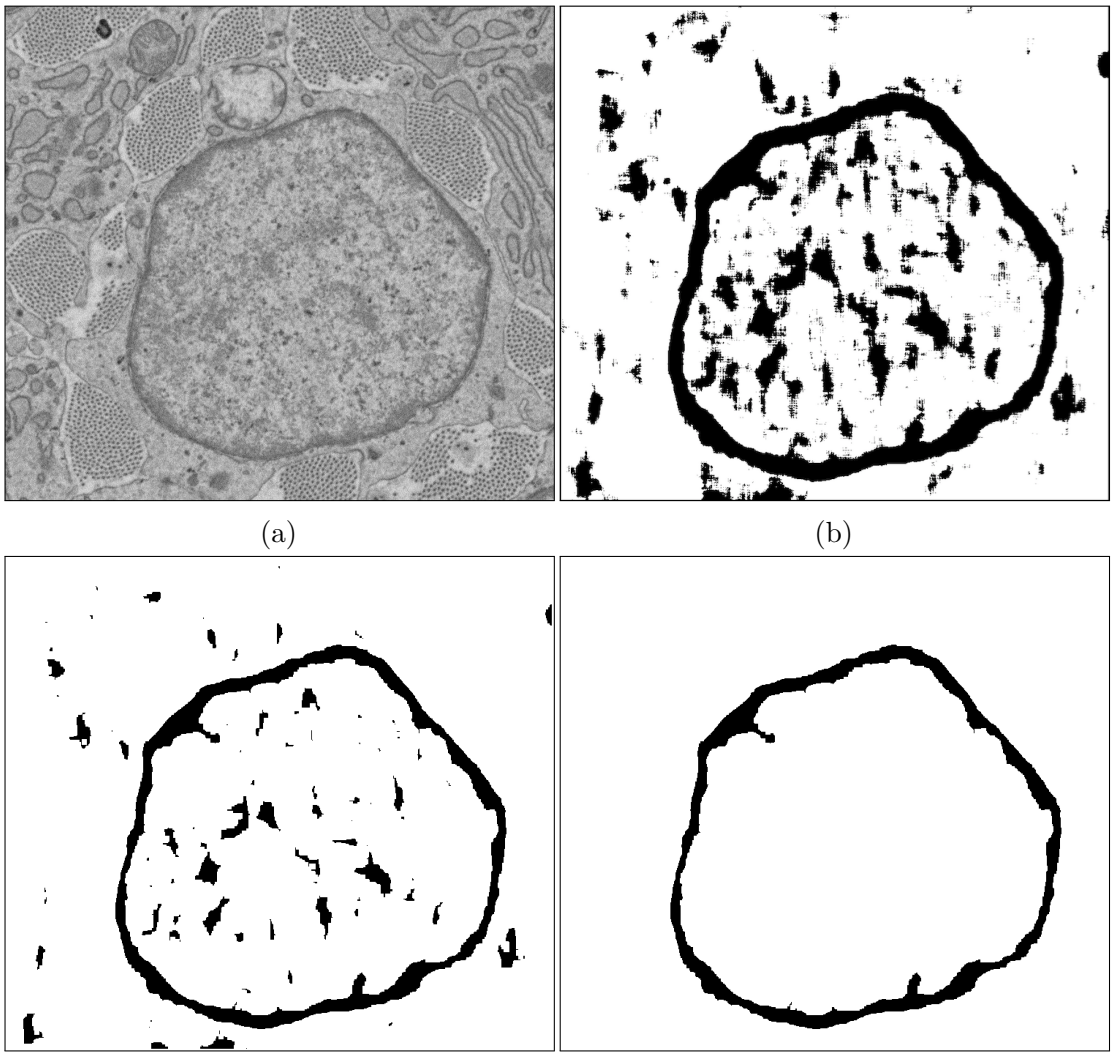
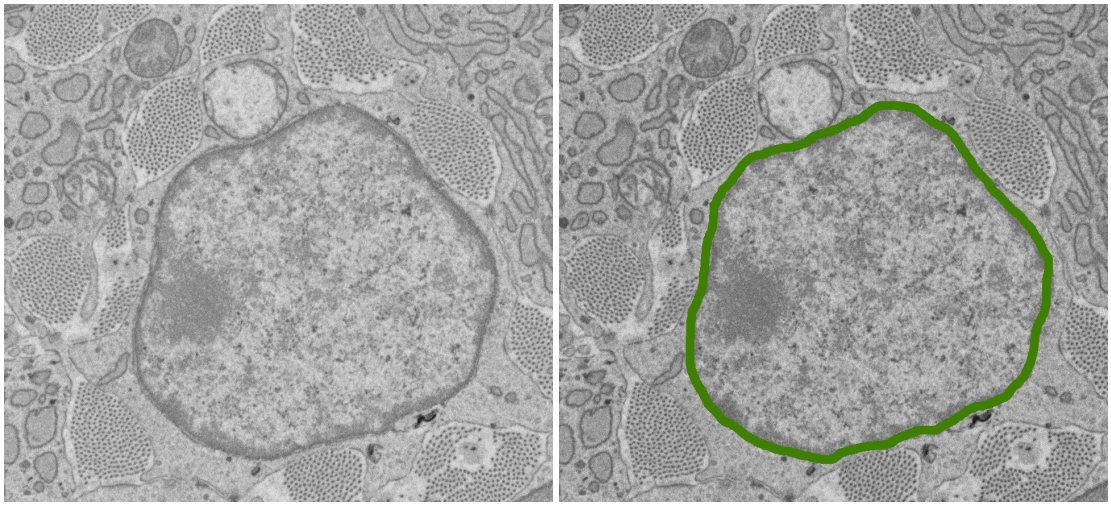


Figure 4.4: (a) Original image. (b) Nucleus segmentation result. (c) Resulting image after applying erosion. (d) Resulting image after eliminating the small connected components.

Figure 4.5 shows the original image with a contour around a nuclear membrane. Not all nuclear membranes contours are continuous, however, and some have broken links between them. This issue occurs for two reasons: firstly, during tissue preparation errors can occur, especially when slicing the sample. Secondly, the classifier can fail to detect the nuclear membranes during the segmentation process. To address this issue, we merged three neighbouring slices together by averaging them as the movement of the nuclei between slices is relatively low.

Thus, if a nuclear membrane is broken, it will be fixed by merging the previous and subsequent slice (see Figure 4.6). We then found the external contour for each nucleus.



(a) (b)

Figure 4.5: (a) Original Image it has only one nucleus. (b) Nucleus with contour on its membrane.

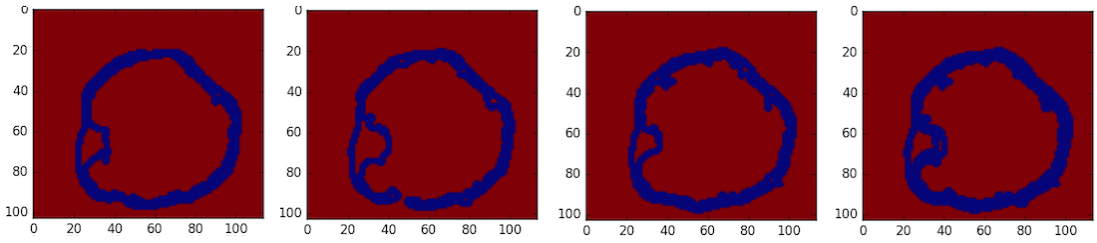


Figure 4.6: A sequence of three slices (z-1), (z), (z+1), and all combined.

As each nucleus in our datasets exhibited slight drifting between slices and the distance between nuclei was relatively large, we used a simple approach that tracks nuclei and links each nucleus to its match in the next slice. We fitted a disk according to the set of points obtained from the nucleus contours, similar to the method in [99]. In the linking of each nucleus to its match in the succeeding slice, we matched the disk centres from the previous frame and determined whether each previous disk centre was within the radius of any of the disks in the next

frame.

The reconstruction algorithm starts from the first slice and assumes all candidate disks to be the start of a nucleus. Then, it creates n nuclei, each containing a single disk. Subsequent slices are processed one by one, with each candidate disk compared with an existing one. If the compared disks' radii overlap, then the nucleus will be extended; if no match is found in the previous frame, then a new nucleus will be created.

4.4 Experimental Results for Nuclei Segmentation

We applied our model to four different datasets, each consisting of hundreds of images that were acquired using the SBFSEM technique (see Table 4.2). The images contained nuclei, mitochondria, and Golgi apparatus. Most of the nuclei had roughly circular or elliptical cross-sections, whereas the mitochondria had a mostly circular shape. Golgi and mitochondrial membranes share several features and they look the same, but the mitochondrial inner body has parallel lines due to the folding of the inner membrane. The outer membrane of mitochondria, however, sometimes appear as thick as those of nuclei, and share the same appearance as nuclei. This led our system to classify mitochondrial outer membranes as nuclei.

4.4.1 Dataset

We ran our model for segmenting and tracking nuclei on four datasets. Each dataset consisted of hundreds of images with different resolutions, meaning that the nuclei differed in appearance across these datasets. All of our datasets were collected using the SBFSEM technique. Table 4.2 shows the number of images and the resolution size of all our datasets.

Dataset	Number of Images	Resolution
Dataset 1	1000	3000×3000
Dataset 2	700	4096×4096
Dataset 3	603	4096×4096
Dataset 4	157	966×876

Table 4.2: Our data consists of four different datasets.

4.4.2 Evaluation Criteria

For quantitative assessment, the ground truth and segmenting results are given by filled closed contours. The foreground objects (nuclei) are shown as white pixels and the background is shown as black pixels. To evaluate a binary segmentation, we selected the Precision, Recall, Accuracy, and Dice coefficients [100] as performance measures. These metrics are calculated as follows:

$$Precision = \frac{TP}{TP + FP} \tag{4.1}$$

$$Recall = \frac{TP}{TP + FN} \tag{4.2}$$

$$Accuracy = \frac{TP + TN}{TP + FP + FN + TN} \tag{4.3}$$

$$Dice = \frac{2 \cdot TP}{2 \cdot TP + FP + FN} \tag{4.4}$$

Where (TP) is true positives, (TN) is true negatives, (FP) is false positives, and (FN) is false negatives of the classified pixels. Table 4.3 summarise the output of comparing the actual result against the ground truth. A perfect segmentation is obtained when Precision and Recall are 1. When Precision is low, this indicates over-segmentation, and when Recall is low, it indicates under-segmentation.

GT \ Result	0	1
0	TN	FP
1	FN	TP

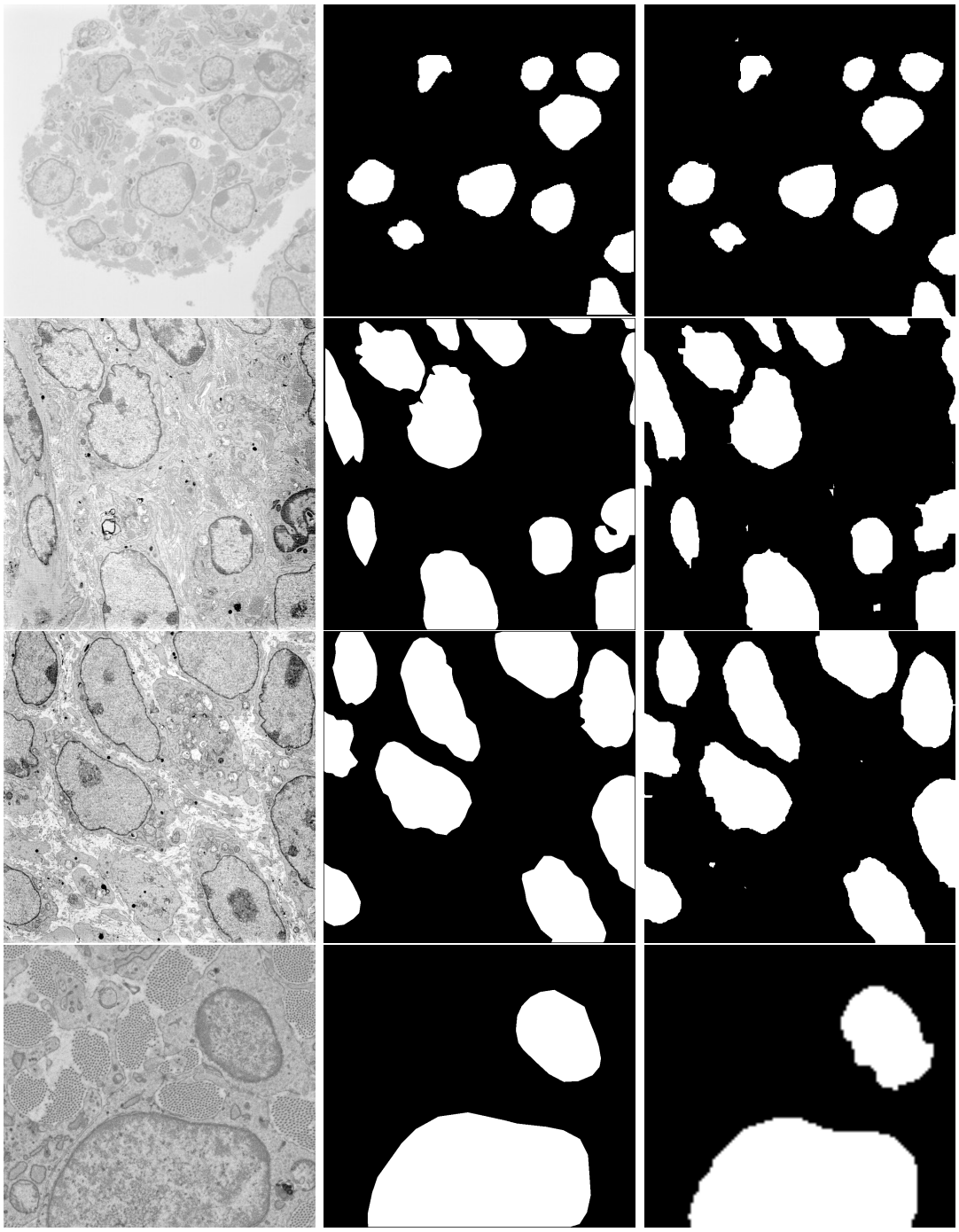
Table 4.3: Representation of confusion matrix.

Accuracy is the proportion of the true positives and true negatives among the total number of pixels. To assess the accuracy of image segmentation, the Dice coefficient is used to measure the spatial overlap. The Dice coefficient value ranges from 0 to 1, where 0 indicates no spatial overlap between two binary images and 1 indicates a complete overlap [100].

Selected qualitative segmentation results are shown in Figure 4.7 from the four datasets obtained by CNN. Figure 4.8 shows the segmentation results obtained by UNet. These figures show the original image, the manually annotated nuclei, and the segmentation results that we obtained from our model.

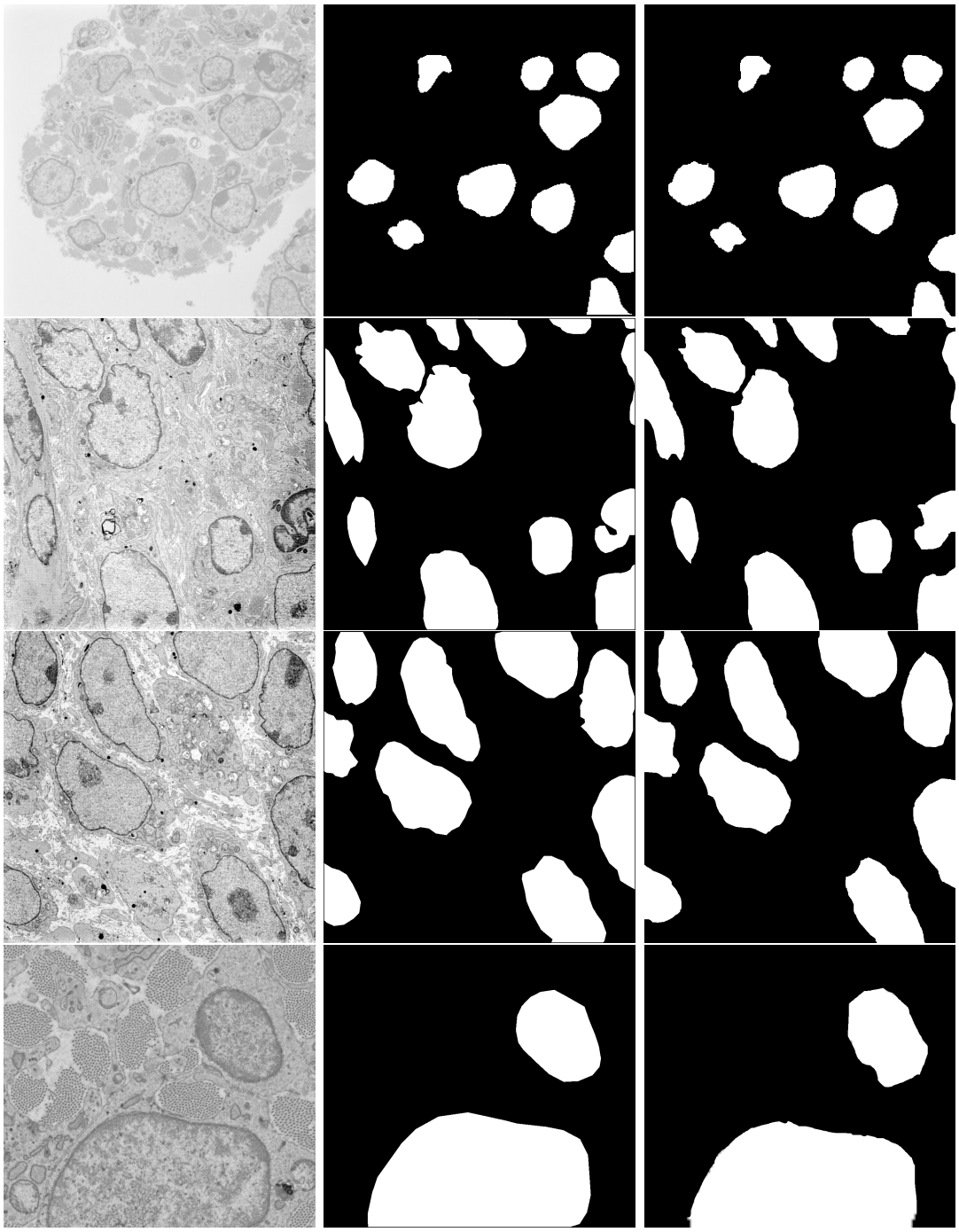
To quantitatively assess our algorithm, we calculated the Precision, Recall, Dice, and Accuracy coefficients for the nuclear membrane position comparing the segmentation results and the ground truth. We annotated 120 images manually, Table 4.4 shows the number of annotated images per dataset. In the first three datasets each image contains 10 nuclei, the last one contains only two nuclei. The results show the Dice, Recall, Precision and Accuracy performance of our method in the four datasets per image.

The Dice, Recall, Precision, and Accuracy performance of CNN method in the four datasets per image can be seen Figure 4.9. Similarly Figure 4.10 shows the performance of U-Net. The two methods achieved high accuracy in finding the nuclei membranes. However, U-Net performed better and produces fewer false



(a) Original image. (b) Ground truth. (c) CNN segmentation.

Figure 4.7: Random image were selected from each dataset. First row from Dataset 1, second row from Dataset 2, thrid row from Dataset 3 and fourth row from Dataset 4.



(a) Original image. (b) Ground truth. (c) U-Net segmentation.

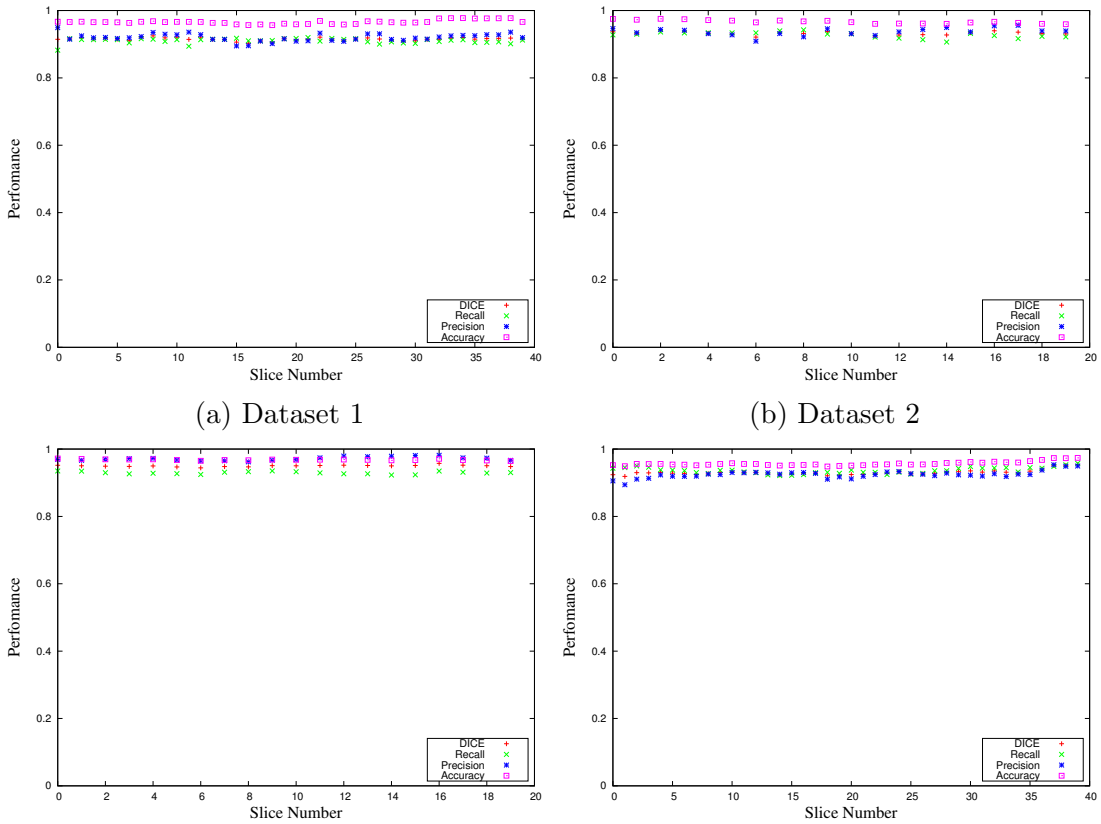
Figure 4.8: Results of U-Net segmentation on example images. First row from Dataset 1, second row from Dataset 2, thrid row from Dataset 3 and fourth row from Dataset 4.

Dataset	Number of Annotated Images
Dataset 1	40
Dataset 2	20
Dataset 3	20
Dataset 4	40

Table 4.4: Number of annotated images per dataset.

positives.

The average accuracy of CNN is above 96% for dataset 1, 96.5% for dataset 2, 97% for dataset 3, and 95.5% for dataset 4. On the other hand, the average accuracy of U-Net is 96.5% for dataset 1, 97.3% for dataset 2, 97.2% for dataset 3, and 96.7% for dataset 4.



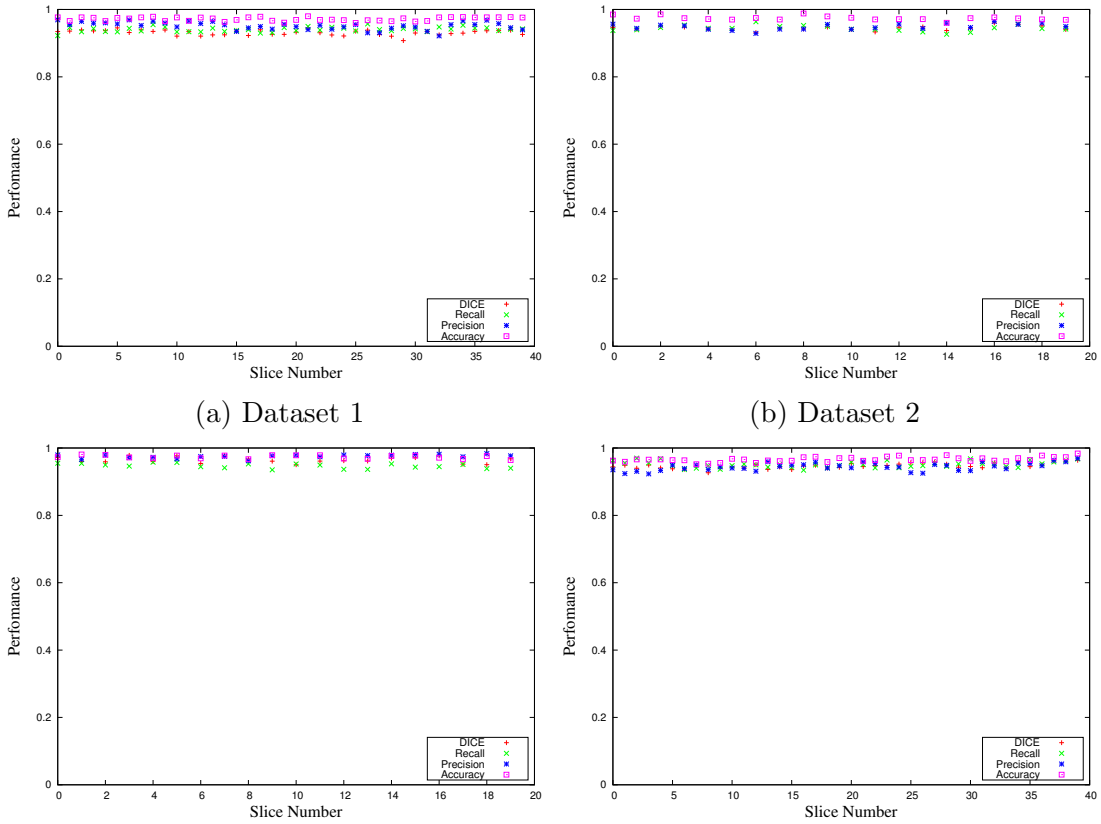
(a) Dataset 1

(b) Dataset 2

(c) Dataset 3

(d) Dataset 4

Figure 4.9: Quantitative nuclei segmentation results for the four datasets using CNN.



(a) Dataset 1 (b) Dataset 2

(c) Dataset 3 (d) Dataset 4

Figure 4.10: Quantitative nuclei segmentation results for the four datasets using U-Net.

4.5 Discussion and Conclusions

We demonstrated a method that can segment and reconstruct nuclei from large image volumes, and our images were collected using the SBFSEM technique. We trained a pixel classifier using a CNN to classify each pixel to a nucleus membrane or non-membrane. CNN have recently achieved considerable success owing to their capability to learn hierarchically abstracted features. However, an accurate model was needed for handling overfitting because of the limited training samples. We thus employed data augmentation to increase the number of samples. We adopted common methods, including rotation, scaling and flipping.

The pixel classifier scanned each image and produced a probability map of each pixel stating whether the pixel was below a membrane or not. Then, we applied post-processing to the resulting image using such techniques as erosion, dilation and removal of small connected components to extract the nucleus membranes. This step was necessary since the nucleus and mitochondrial membranes looked similar. The only difference was that the mitochondrial membrane was not as thick or wide as the nucleus membrane. Therefore, performing erosion with a large kernel could remove only the mitochondrial membrane.

To reconstruct nuclei in 3D, we linked each nucleus to its correspondence in the subsequent slice. We fitted a disk on each nucleus and then matched the disk centre from the previous frame. If the previous disk centre was within the radius of any of the disks in the next frame, then we extended that nucleus. Disks with no matches were used to create new nuclei. This approach was found to be effective in reconstructing nuclei since the nucleus radius was large and the movement of each nucleus between slices was small.

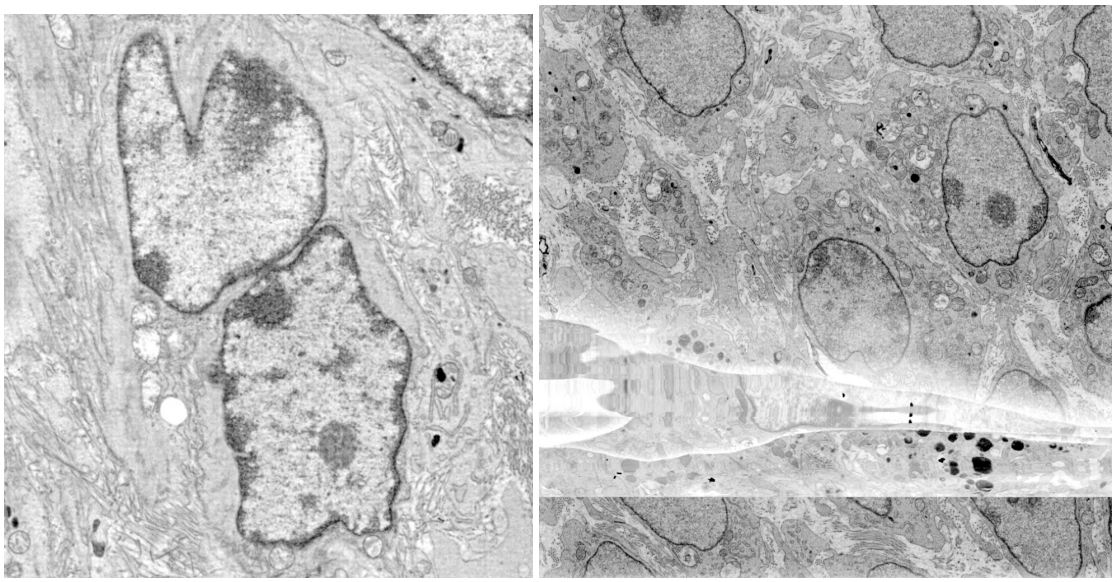
We evaluated our segmentation method against our ground truth. We selected random images from four datasets, and we manually marked the nucleus membrane. The total number of images was 120, the average number of nuclei in three datasets was about 8 and one dataset contained only 2 nuclei. The experiments showed that the average accuracy of both the CNN and Unet exceeded 95% for all the datasets. However, Unet performed better and produced less false positives than did the typical CNN.

Biologists are interested in quantifying nuclei on the basis of their lengths, radii and volumes. We therefore measured the length, average radius and volume of each nucleus for each dataset. Figure 4.12 shows a histogram of nuclei with a particular length on the four datasets. The nucleus length is the number of slices.

Figure 4.13 shows a histogram of nuclei with a particular radius. To measure the radius of each nucleus, we fitted a circle according to the nucleus membrane contour points for each slice and then calculated the mean radius. Figure 4.14 shows a histogram of nuclei with a particular volume. The volume is the total number of pixels inside the nucleus membrane for each slice. The software IMOD was used to perform 3D visualisations of the four datasets, and random colour was given to each nucleus. Figures 4.15 4.16 4.17 4.18 show the 3D reconstruction of each nucleus.

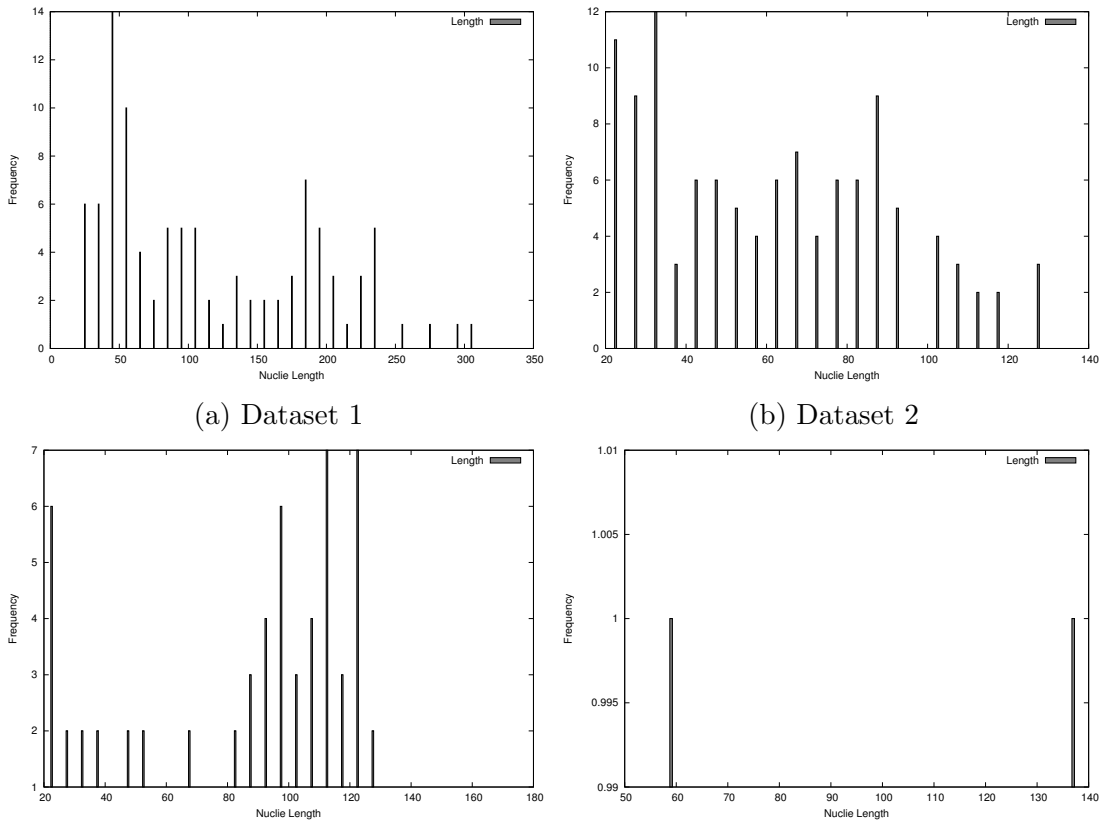
The main limitation of the current approach occurs when two nucleus membranes overlap (see Fig. 4.11a). This affects the linking algorithm in that the two nuclei are merged as one nucleus for several slices and subsequently split (as the two nucleus membranes split in the original image). This can be solved by either identification of the overlapped nuclei using 2D information or incorporation of the prior and subsequent slices. However, another failure occurs when the image volume contains slices corrupted by tearing on the block surface (see Fig. 4.11b).

A comparison of our work with previous studies reveals that there is no existing dataset that is similar to ours. There are several datasets for cell segmentation, but they differ from ours in terms of tissue staining, imaging method and tissue type (see discussion in Chapter 5.).



(a) (b)

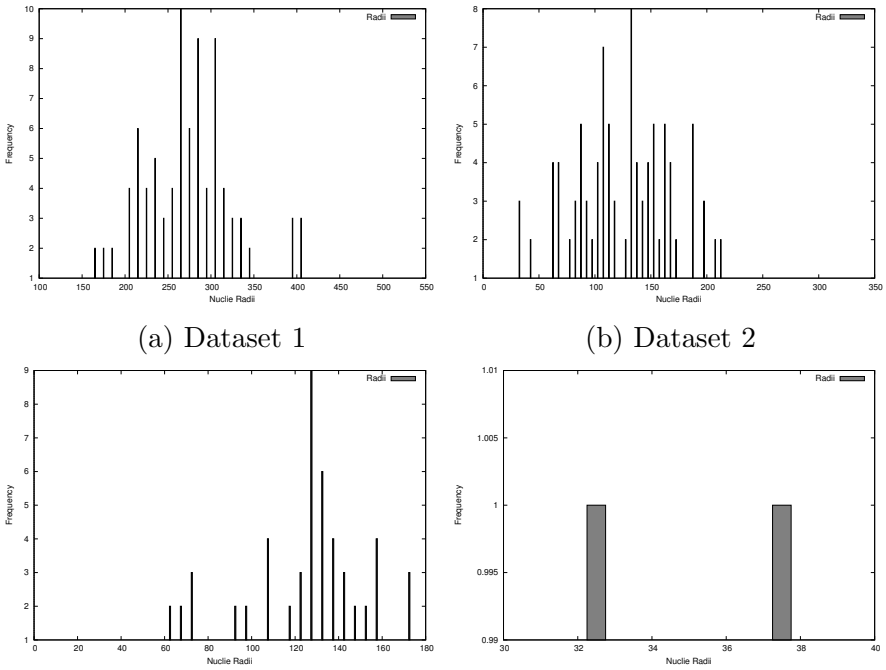
Figure 4.11: (a) Two nuclei membranes overlapped. (b) corrupted slice caused by tearing the surface of the block.



(a) Dataset 1 (b) Dataset 2

(c) Dataset 3 (d) Dataset 4

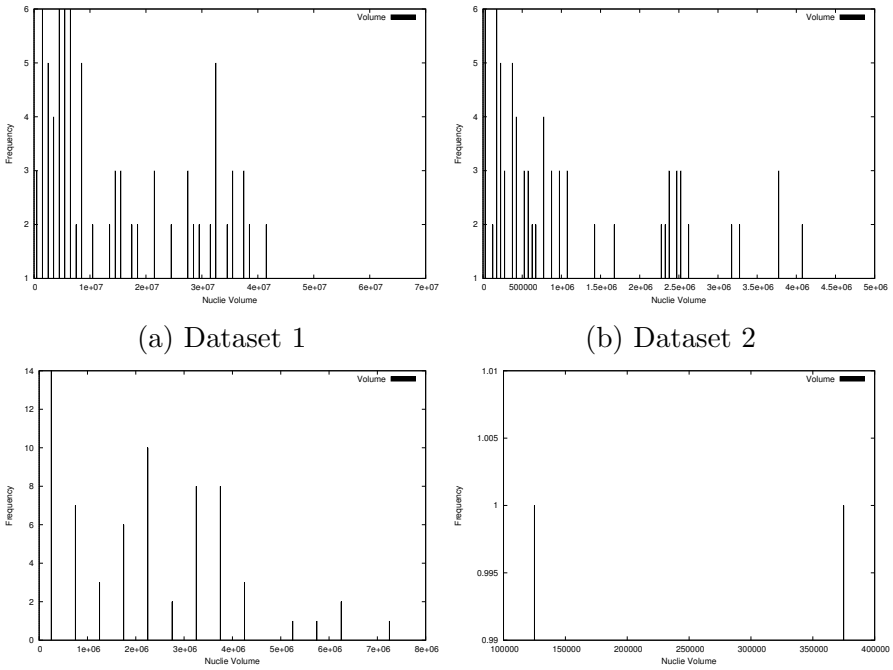
Figure 4.12: Quantitative analysis of nuclei length for the four datasets.



(a) Dataset 1 (b) Dataset 2

(c) Dataset 3 (d) Dataset 4

Figure 4.13: Quantitative analysis of nuclei radii for the four datasets.



(a) Dataset 1 (b) Dataset 2

(c) Dataset 3 (d) Dataset 4

Figure 4.14: Quantitative analysis of nuclei volume for the four datasets.

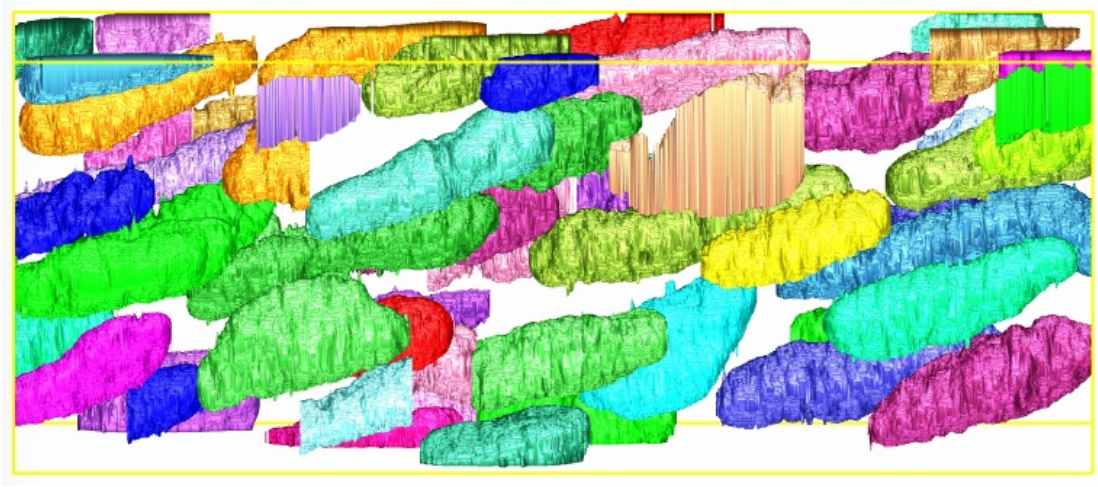


Figure 4.15: Visualisation of all nuclei of Dataset 1 found by the algorithm.

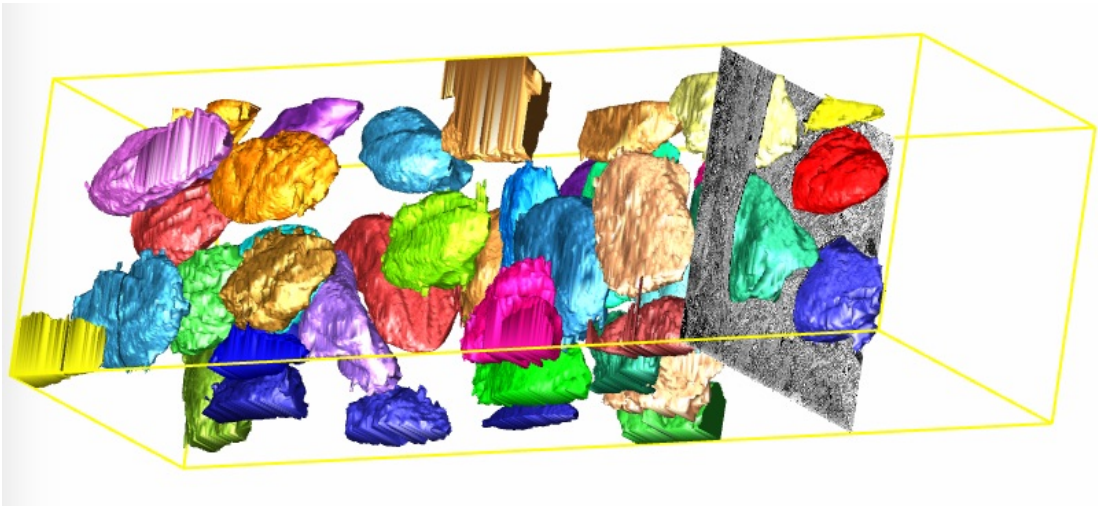


Figure 4.16: Visualisation of all nuclei of Dataset 2 found by the algorithm.

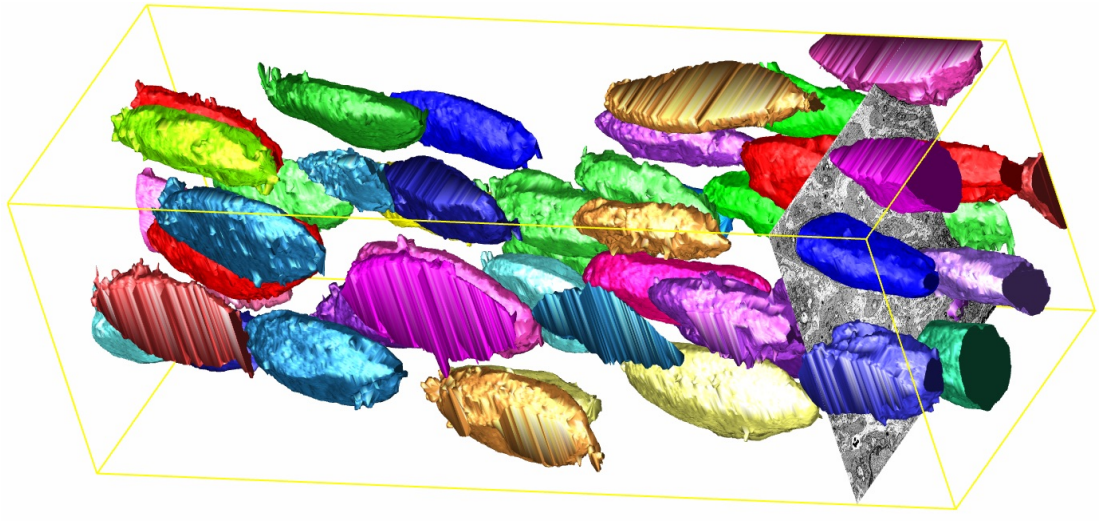


Figure 4.17: Visualisation of all nuclei of Dataset 3 found by the algorithm.

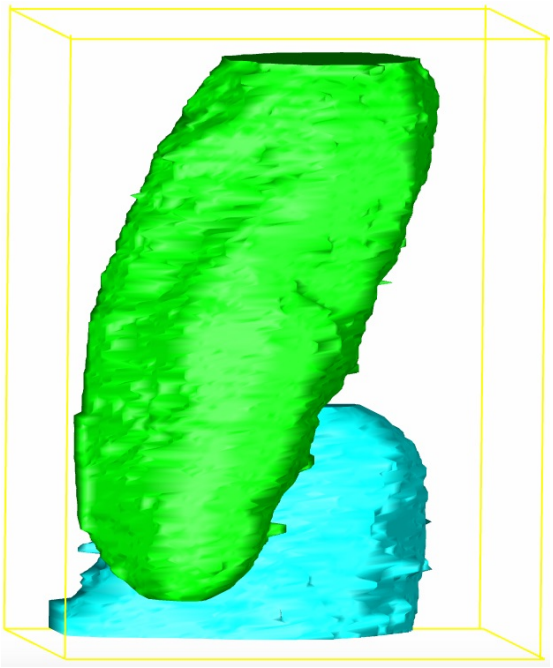


Figure 4.18: Visualisation of all nuclei of Dataset 4 found by the algorithm.

Chapter 5

Discussion and Future Work

This project has demonstrated a fully automatic system that detects and reconstructs collagen fibres across an image volume. The detection system involves finding candidates by using template matching to detect fibres that have a circular shape. In the case of fibres with a polygonal shape, we train a random forest classifier to locate them. Both techniques produce a large number of false positives. We discard these false matches using a random forest.

The linking algorithm accounts for the movement of fibre bundles between slices and was found to be effective in linking the detected disks together into extended fibres. Occasionally, a candidate disk for a fibre is not detected in an image, either because of a failure of the detector or because some image slices are corrupted by tearing the surface of the block. Such missing disks cause a long fibre to be split into two (or more) shorter fibres. To find and correct the small gaps caused by such detection failures, we use a linear prediction for gap filling. Moreover, some errors occur when two fibres almost touch. To identify such errors, we use a Kalman filter to look for inconsistencies. Each discrepancy is then revisited and corrected where necessary. The algorithm is able to track thousands of fibres across hundreds of slices, showing that fibres follow complex paths through the tissue.

The system is able to reconstruct the collagen fibres in 3D, allowing biologists to study and measure the network that they form. Another interest for biologists is finding lone fibres those fibres that leave their bundle to analyse them and identify their role. The most interesting measurements for biologists are the length, radii and curvature of fibres. The experiment demonstrated that our approach achieved high accuracy in detecting and tracking fibres through the image volume.

Another contribution of this project is a multistage approach for segmenting nuclei in a complex environment in image data. The proposed method processes each slice (as a 2D image) to find the boundaries of the nuclei and then tracks them and links each nucleus across the slices. We train a CNN to classify each pixel according to whether or not it belongs to a nucleus membrane. A further post-processing step is performed to eliminate other objects, such as mitochondria and Golgi apparatus, as they share some of the nuclei membranes features.

In the final stage, we link every nucleus with its match in the next slice through the image volume. We found that the nucleis radii range from 200 to 900 pixels in SBFSEM images. Using this assumption, we found that determining the centre of gravity of each nucleus in one slice and matching it with the nucleus in the next slice is sufficient to link them. The system is able to segment, reconstruct and quantify nuclei across hundreds of slices. For any given dataset, the system is automatically able to reconstruct nuclei, visualise them in a 3D viewer and quantify each nucleus based on its volume, radius and length. Experiments have been done to evaluate our approach in various datasets, and we obtained high accuracy in segmenting nuclei.

5.1 Comparing Results across Different Studies

Comparing different studies on the techniques used for reconstructing cellular structures from EM images can be challenging for a variety of reasons. Certain problems, including the selection of similarity measurements and the density of the reconstruction, can create even more difficulties when one is attempting to compare the different methods within one study.

5.1.1 Tissue staining

There are two principal kinds of staining that are used for generating EM images of connective tissue, the first being the traditional intracellular stain. This approach stains all the interior and exterior cellular membranes with heavy metals, rendering them darker in the resulting images. This method of staining facilitates the observation of intracellular structures, such as the vesicles and post-synaptic densities, that distinguish the synaptic linkages among membranes. Nevertheless, as both the exterior and interior membranes are stained identically, it can be challenging to determine one from the other, while the existence of intracellular clutter creates additional challenges when one attempts to divide membrane cross-sections reliably.

The second method is extracellular staining, which involves staining only the outside membrane. In this method, the lack of intracellular clutter facilitates the process of segmentation. In the majority of neural tissues, however, the absence of intracellular staining of the membrane increases the difficulties of clearly identifying the linkages among membranes. It is probable that most segmentation methods will exhibit improved performance on datasets that utilise the extracellular stain. Hence, an effective comparison of the reconstruction techniques demonstrated in various studies is fundamentally impracticable.

5.1.2 Imaging Method

There are two elements of imaging that can affect the performance of segmentation techniques: (i) whether the slicing happens before or after imaging (serial block-face microscopy or serial section) and (ii) whether the imaging process uses transmitted or reflected electrons (scanning microscopy or transmission). Imaging techniques that extract a section of the sample before it is imaged (serial-section imaging) are constrained in terms of the minimum z-resolution that it is possible to achieve, as the section removed must be sufficiently thick to be handled. This generally restricts the z-resolution to approximately 50 nm. However, the cell membranes can also experience significant blurring in the neurites that run at an angle to the plane of the segment in transmission imaging technique.

5.1.3 Tissue Type

Collagen fibres have different appearances in tissues from different locations. For example, tissues from normal skin follow specific patterns in which bundles are aligned in different directions to resist forces from multiple axes (see the top left image in Fig. 5.1). On the other hand, tissues from the cornea have different patterns, in which the fibres are mostly in orthogonal arrays to allow for maximum transparency (see the top right area in Fig. 5.1).

The normal collagen fibres in connective tissues are uniform in terms of size and spacing. However, fibres from patients with diseases or disorders can change their shapes, sizes and distribution (see the bottom left image in Fig. 5.1). Mao and Bristow [3] have shown how fibres are affected by certain diseases in their shapes and distributions. For example, in patients with dermatosparaxis, the fibre no longer has a circular shape, and its morphology undergoes dramatic alterations (see the bottom right image in Fig. 5.1). This disease results in severe effects on the tensile strength of connective tissues.

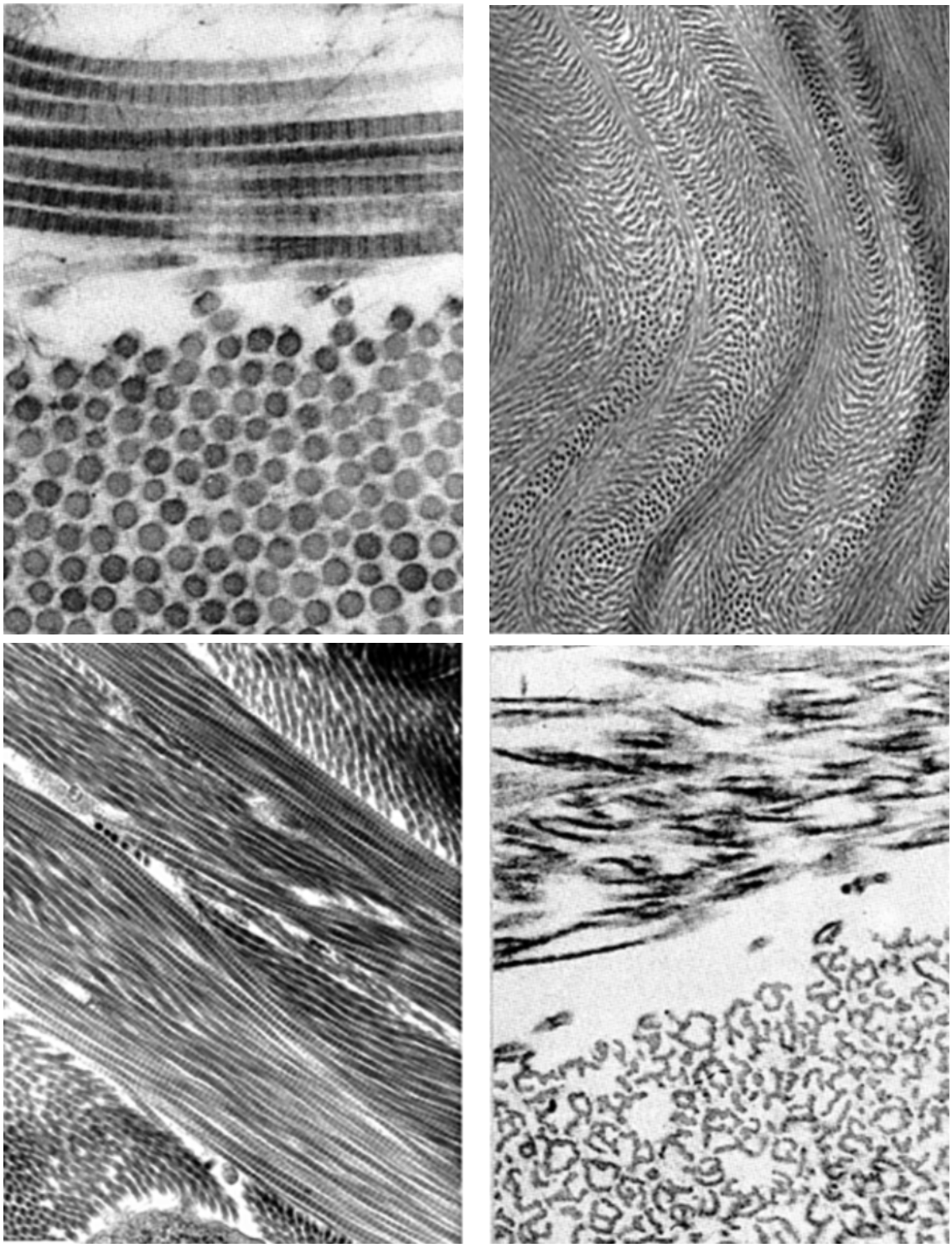


Figure 5.1: Normal collagen fibres (top left image). Fibres in cornea (top right image). Fibre in normal skin follow specific patterns (bottom left image). A dramatic alterations in fibres caused by dermatosparaxis disease (bottom right). images are taken from [3].

5.2 Future Work

- During the acquisition using SBFSEM several artifacts may produced in the image volume. The most notable are discontinuities along the section axis. This is due to variation in section thickness, sample derogation and error in sectioning. These artifacts need to be corrected to obtain more robust reconstruction.

- The current approach for tracking fibres assumes that fibres are in a bundle. Further work needs to be carried out to examine and enhance the algorithm to correctly track those fibres. Potentially this can be corrected by a multi-pass approach.

- Bundles can split and then merge with another which makes the algorithm mis-track the fibres, since the fibres change. This could be solved by identifying the split and merging of bundles that occurs in the image volume and perform a second pass to correct such errors.

- Nuclei in the image volume can become close to each other in some slice. Thus, their membrane can be touching and makes the segmentation very challenging. Further work is needed to identify the case and split them.

- Mitochondria and Golgi membranes share several features of the nucleus membranes. In order to segment and reconstruct them different features need to be extracted to train pixel-wise classifier.

- Automatic segmentation of other structures, such as mitochondria and Golgi complexes.

Bibliography

[1] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 07-12-June, pages 3431–3440, 2015.

[2] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, volume 9351, pages 234–241, 2015.

[3] Jau Ren Mao and James Bristow. The Ehlers-Danlos syndrome: On beyond collagens. *Journal of Clinical Investigation*, 107(9):1063–1069, 2001.

[4] Karl E. Kadler. Fell Muir Lecture: Collagen fibril formation in vitro and in vivo. 98(1):4–16, 2017.

[5] Winfried Denk and Heinz Horstmann. Serial block-face scanning electron microscopy to reconstruct three-dimensional tissue nanostructure. *PLoS Biology*, 2(11), 2004.

[6] Moritz Helmstaedter, Kevin L. Briggman, and Winfried Denk. High-accuracy neurite reconstruction for high-throughput neuroanatomy. *Nature Neuroscience*, 14(8):1081–1088, 2011.

[7] Tobias Starborg, Nicholas S. Kalson, Yinhui Lu, Aleksandr Mironov, Timothy F. Cootes, David F. Holmes, and Karl E. Kadler. Using transmission electron microscopy and 3View to determine collagen fibril size and three-dimensional organization. *Nature Protocols*, 8(7):1433–1448, 2013.

[8] D. A.D. Parry, G. R.G. Barnes, and A. S. Craig. A comparison of the size distribution of collagen fibrils in connective tissues as a function of age and a possible relation between fibril size distribution and mechanical properties. *Proceedings of the Royal Society of London - Biological Sciences*, 203(1152):305–321, 1978.

[9] Bruce Alberts, Alexander Johnson, Julian Lewis, Martin Raff, Keith Roberts, and Peter Walter. 2014 ASCB/IFCB Meeting abstracts. *Molecular Biology of the Cell*, 25(25):3987–3987, 2014.

[10] Nicholas S. Kalson, Yinhui Lu, Susan H. Taylor, David F. Holmes, and Karl E. Kadler. A structure-based extracellular matrix expansion mechanism of fibrous tissue growth. *eLife*, 4(MAY):1–22, 2015.

[11] Jaspreet K Kular, Shouvik Basu, and Ram I Sharma. The extracellular matrix: Structure, composition, age-related differences, tools for analysis and applications for tissue engineering. *Journal of Tissue Engineering*, 5:204173141455711, 2014.

[12] Holger Schwarz, Günter Giese, Holger Müller, Michael Koenen, and Veit Witzemann. Different functions of fetal and adult AChR subtypes for the formation and maintenance of neuromuscular synapses revealed in ϵ-subunit-deficient mice. *European Journal of Neuroscience*, 12(9):3107–3116, 2000.

[13] M L Watson. Staining of Tissue Sections for Electron Microscopy with Heavy Metals. *The Journal of Cell Biology*, 4(4):475–478, 1958.

[14] Christopher J. Peddie and Lucy M. Collinson. Exploring the third dimension: Volume electron microscopy comes of age. 61:9–19, 2014.

[15] E R Macagno, C Levinthal, and I Sobel. Three-Dimensional Computer Reconstruction of Neurons and Neuronal Assemblies. *Annual Review of Biophysics and Bioengineering*, 8(1):323–351, 1979.

[16] John K Stevens, Thomas L Davis, Neil Friedman, and Peter Sterling. A systematic approach to reconstructing microcircuitry by electron microscopy of serial sections. 2(1-3):265–293, 1980.

[17] Robert L. Trelstad and Kimiko Hayashi. Tendon collagen fibrillogenesis: Intracellular subassemblies and cell surface changes associated with fibril growth. *Developmental Biology*, 71(2):228–242, 1979.

[18] Jean Yves Exposito, Ulrich Valcourt, Caroline Cluzel, and Claire Lethias. The fibrillar collagen family. *International Journal of Molecular Sciences*, 11(2):407–426, 2010.

[19] Ervin H. Epstein. (Alpha1 (3)) 3 human skin collagen. Release by pepsin digestion and preponderance in fetal life. *The Journal of biological chemistry*, 249:3225–3231, 1974.

[20] Rene B. Svensson, Andreas Herchenhan, Tobias Starborg, Michael Larsen, Karl E. Kadler, Klaus Qvortrup, and S. Peter Magnusson. Evidence of structurally continuous collagen fibrils in tendons. *Acta Biomaterialia*, 50:293–301, 2017.

[21] Martin W. Hetzer. The nuclear envelope. 2(3), 2010.

[22] Christoph Sommer, Christoph Straehle, Ullrich Kothe, and Fred a Hamprecht. Ilastik: Interactive learning and segmentation toolkit. In *Proceedings - International Symposium on Biomedical Imaging*, pages 230–233, 2011.

[23] Darko Zikic, Ben Glocker, Ender Konukoglu, Antonio Criminisi, C. Demiralp, J. Shotton, O. M. Thomas, T. Das, R. Jena, and S. J. Price. Decision Forests for Tissue-Specific Segmentation of High-Grade Gliomas in Multi-channel MR. In *Medical Image Computing and Computer-Assisted Intervention (MICCAI)*, pages 369–376, 2012.

[24] Hongwei Li, Huang Hao, Wei Shi Zheng, Xie Xiaohua, and Jianguo Zhang. HEp-2 specimen classification via deep CNNs and pattern histogram. In *Proceedings - International Conference on Pattern Recognition*, pages 2145–2149, 2017.

[25] H.P. Ng, S.H. Ong, K.W.C. Foong, P.S. Goh, and W.L. Nowinski. Medical Image Segmentation Using K-Means Clustering and Improved Watershed Algorithm. In *2006 IEEE Southwest Symposium on Image Analysis and Interpretation*, pages 61–65, 2006.

[26] J Ross Quinlan. *C4.5: Programs for Machine Learning*, volume 16. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1993.

[27] L Breiman, J H Friedman, R A Olshen, and C J Stone. Classification and Regression Trees. *The Wadsworth statisticsprobability series*, 19:368, 1984.

[28] Leo Breiman. Random forests. *Machine Learning*, 45(1):5–32, 2001.

[29] Jamie Shotton, Matthew Johnson, and Roberto Cipolla. Semantic texton forests for image categorization and segmentation. In *26th IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, 2008.

[30] Jamie Shotton, Andrew Fitzgibbon, Mat Cook, Toby Sharp, Mark Finocchio, Richard Moore, Alex Kipman, and Andrew Blake. Real-time human pose recognition in parts from single depth images. In *CVPR 2011*, pages 1297–1304, 2011.

[31] Anna Bosch, Andrew Zisserman, and Xavier Munoz. Image Classification using Random Forests and Ferns. In *2007 IEEE 11th International Conference on Computer Vision*, pages 1–8, 2007.

[32] Vincent Lepetit, Pascal Lagger, and Pascal Fua. Randomized trees for real-time keypoint recognition. In *Proceedings - 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, CVPR 2005*, volume II, pages 775–781, 2005.

[33] Raphaél Marée, Pierre Geurts, Justus Piater, and Louis Wehenkel. Random subwindows for robust image classification. In *Proceedings - 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, CVPR 2005*, volume I, pages 34–41, 2005.

[34] Antonio Torralba, Kevin P. Murphy, and William T. Freeman. Sharing visual features for multiclass and multiview object detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(5):854–869, 2007.

[35] Rich Caruana, Nikos Karampatziakis, and Ainur Yessenalina. An empirical evaluation of supervised learning in high dimensions. In *Proceedings of the 25th international conference on Machine learning - ICML '08*, pages 96–103, 2008.

[36] Victor Lempitsky, Michael Verhoek, J. Alison Noble, and Andrew Blake. Random forest classification for automatic delineation of myocardium in real-time 3D echocardiography. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, volume 5528, pages 447–456, 2009.

[37] Guoqiang Peter Zhang. Neural networks for classification: A survey. *IEEE Transactions on Systems, Man and Cybernetics Part C: Applications and Reviews*, 30(4):451–462, 2000.

[38] Suzana Herculano-Houzel. The human brain in numbers: a linearly scaled-up primate brain. *Frontiers in Human Neuroscience*, 3:64, 2009.

[39] F. Rosenblatt. The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological Review*, 65(6):386–408, 1958.

[40] David E. Rumelhart, Geoffrey E. Hinton, and Ronald J. Williams. Learning representations by back-propagating errors. *Nature*, 323(6088):533–536, 1986.

[41] Richard H.R. Hahnioser, Rahul Sarpeshkar, Misha A. Mahowald, Rodney J. Douglas, and H. Sebastian Seung. Digital selection and analogue amplification coexist in a cortex- inspired silicon circuit. *Nature*, 405(6789):947–951, 2000.

[42] Xavier Glorot, Antoine Bordes, and Yoshua Bengio. Deep sparse rectifier neural networks. In *AISTATS '11: Proceedings of the 14th International Conference on Artificial Intelligence and Statistics*, page 34, 2011.

[43] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. ImageNet classification with deep convolutional neural networks. *Advances in Neural Information and Processing Systems (NIPS)*, 60(6):84–90, 2012.

[44] Dan C. Ciresan, Alessandro Giusti, Luca M. Gambardella, and Jurgen Schmidhuber. Deep Neural Networks Segment Neuronal Membranes in Electron Microscopy Images. *Nips*, pages 1–9, 2012.

[45] Pho Pinheiro and Ronan Collobert. Recurrent convolutional neural networks for scene labeling. In *Proceedings of The 31st International Conference ...*, page 34, 2014.

[46] Saurabh Gupta, Ross Girshick, Pablo Arbeláez, and Jitendra Malik. Learning rich features from RGB-D images for object detection and segmentation.

In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, volume 8695 LNCS, pages 345–360, 2014.

[47] Bharath Hariharan, Pablo Arbeláez, Ross Girshick, and Jitendra Malik. Simultaneous Detection and Segmentation. *arXiv preprint arXiv ...*, 44(3):75, 2014.

[48] Karen Simonyan and Andrew Zisserman. Very Deep Convolutional Networks for Large-Scale Image Recognition. *International Conference on Learning Representations (ICRL)*, 34(6):54, 2015.

[49] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 07-12-June, pages 1–9, 2015.

[50] Hyeonwoo Noh, Seunghoon Hong, and Bohyung Han. Learning deconvolution network for semantic segmentation. In *Proceedings of the IEEE International Conference on Computer Vision*, volume 2015 Inter, pages 1520–1528, 2015.

[51] Özgün Çiçek, Ahmed Abdulkadir, Soeren S. Lienkamp, Thomas Brox, and Olaf Ronneberger. 3D U-net: Learning dense volumetric segmentation from sparse annotation. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, volume 9901 LNCS, pages 424–432, 2016.

[52] Nobuyuki Otsu. A Threshold Selection Method from Gray-Level Histograms. *IEEE Transactions on Systems, Man, and Cybernetics*, 9(1):62–66, 1979.

[53] Xiaodong Yang, Houqiang Li, and Xiaobo Zhou. Nuclei segmentation using marker-controlled watershed, tracking using mean-shift, and Kalman filter in time-lapse microscopy. *IEEE Transactions on Circuits and Systems I: Regular Papers*, 53(11):2405–2414, 2006.

[54] Burr Settles. Active Learning Literature Survey. *Machine Learning*, 15(2):201–221, 2010.

[55] Carsten Rother, Vladimir Kolmogorov, and Andrew Blake. "GrabCut": interactive foreground extraction using iterated graph cuts. *ACM Transactions on Graphics*, 23(3):309, 2004.

[56] Jakob Santner, Thomas Pock, and Horst Bischof. Interactive multi-label segmentation. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, volume 6492 LNCS, pages 397–410, 2011.

[57] Markus Unger, Thomas Pock, Werner Trobin, Daniel Cremers, and Horst Bischof. TVSeg - Interactive Total Variation Based Image Segmentation. In *Procedings of the British Machine Vision Conference 2008*, pages 40.1–40.10, 2008.

[58] Vladimir Vezhnevets and Vadim Konouchine. GrowCut - Interactive Multi-Label N-D Image Segmentation By Cellular Automata. *Graphicon*, 2:50–156, 2005.

[59] Eric N. Mortensen and William A. Barrett. Intelligent scissors for image composition. In *Proceedings of the 22nd annual conference on Computer graphics and interactive techniques - SIGGRAPH '95*, pages 191–198, 1995.

[60] Björn Andres, Ullrich Köthe, Moritz Helmstaedter, Winfried Denk, and

Fred A. Hamprecht. Segmentation of SBFSEM volume data of neural tissue by hierarchical classification. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, volume 5096 LNCS, pages 142–152, 2008.

[61] C. N. Straehle, U. Köthe, G. Knott, and F. A. Hamprecht. Carving: Scalable interactive segmentation of neural volume electron microscopy images. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, volume 6891 LNCS, pages 653–660, 2011.

[62] Jianbo Shi and Jitendra Malik. Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):888–905, 2000.

[63] Cory Jones, Ting Liu, Mark Ellisman, and Tolga Tasdizen. Semi-automatic neuron segmentation in electron microscopy images via sparse labeling. In *Proceedings - International Symposium on Biomedical Imaging*, pages 1304–1307, 2013.

[64] Ingrid Carlbom, Demetri Terzopoulos, and Kristen M. Harris. Computer-Assisted Registration, Segmentation, and 3D Reconstruction from Images of Neuronal Tissue Sections. *IEEE Transactions on Medical Imaging*, 13(2):351–362, 1994.

[65] Elizabeth Jurrus, Shigeki Watanabe, Richard J. Giuly, Antonio R.C. Paiva, Mark H. Ellisman, Erik M. Jorgensen, and Tolga Tasdizen. Semi-automated neuron boundary detection and nonbranching process segmentation in electron microscopy images. *Neuroinformatics*, 11(1):5–29, 2013.

[66] Viren Jain, Benjamin Bollmann, Mark Richardson, Daniel R. Berger, Moritz N. Helmstaedter, Kevin L. Briggman, Winfried Denk, Jared B.

Bowden, John M. Mendenhall, Wickliffe C. Abraham, Kristen M. Harris, Narayanan Kasthuri, Ken J. Hayworth, Richard Schalek, Juan Carlos Tapia, Jeff W. Lichtman, and H. Sebastian Seung. Boundary learning by optimization with topological constraints. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 2488–2495, 2010.

[67] Elizabeth Jurrus, Ross Whitaker, Bryan W. Jones, Robert Marc, and Tolga Tasdizen. An optimal-path approach for neural circuit reconstruction. In *2008 5th IEEE International Symposium on Biomedical Imaging: From Nano to Macro, Proceedings, ISBI*, pages 1609–1612, 2008.

[68] Verena Kaynig, Thomas J. Fuchs, and Joachim M. Buhmann. Geometrical consistent 3D tracing of neuronal processes in ssTEM data. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, volume 6362 LNCS, pages 209–216, 2010.

[69] Tolga Tasdizen, Ross Whitaker, Robert Marc, and Bryan Jones. Enhancement of cell boundaries in transmission electron microscopy images. In *Proceedings - International Conference on Image Processing, ICIP*, volume 2, pages 129–132, 2005.

[70] Ritwik Kumar, Amelio Vázquez-Reina, and Hanspeter Pfister. Radon-like features and their application to connectomics. In *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition - Workshops, CVPRW 2010*, pages 186–193, 2010.

[71] Ying-Lun Fok, J.C.K. Chan, and R.T. Chin. Automated analysis of nerve-cell images using active contour models. *IEEE Transactions on Medical Imaging*, 15(3):353–368, 1996.

[72] M. Bertalmio, G. Sapiro, and G. Randall. Morphing active contours: a geometric approach to topology-independent image segmentation and tracking. *Image Processing, 1998. ICIP 98. Proceedings. 1998 International Conference on*, 3:318 – 322, 1998.

[73] Achilleas S. Frangakis and Reiner Hegerl. Segmentation of biomedical images with eigenvectors. In *Proceedings - International Symposium on Biomedical Imaging*, volume 2002-Janua, pages 90–93, 2002.

[74] Chang Hang, Yang Qing, Manfred Auer, and Bahram Parvin. Modeling of front evolution with graph cut optimization. In *Proceedings - International Conference on Image Processing, ICIP*, volume 1, pages 45–52, 2006.

[75] Amelio Vazquez-Reina, Eric Miller, and Hanspeter Pfister. Multiphase geometric couplings for the segmentation of neural processes. In *2009 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops, CVPR Workshops 2009*, volume 2009 IEEE, pages 2020–2027, 2009.

[76] Philippe Thevenaz, Ricard Delgado-Gonzalo, and Michael Unser. The ovuscule. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(2):382–393, 2011.

[77] Rajesh Narasimha, Hua Ouyang, Alexander Gray, Steven W. McLaughlin, and Sriram Subramaniam. Automatic joint classification and segmentation of whole cell 3D images. *Pattern Recognition*, 42(6):1067–1079, 2009.

[78] Viren Jain, Joseph F. Murray, Fabian Roth, Srinivas Turaga, Valentin Zhigulin, Kevin L. Briggman, Moritz N. Helmstaedter, Winfried Denk, and H. Sebastian Seung. Supervised Learning of Image Restoration with Convolutional Networks. In *2007 IEEE 11th International Conference on Computer Vision*, pages 1–8, 2007.

[79] Yuriy Mishchenko. Automation of 3D reconstruction of neural tissue from large volume of conventional serial section transmission electron micrographs. *Journal of Neuroscience Methods*, 176(2):276–289, 2009.

[80] Elizabeth Jurrus, Antonio R.C. Paiva, Shigeki Watanabe, James R. Anderson, Bryan W. Jones, Ross T. Whitaker, Erik M. Jorgensen, Robert E. Marc, and Tolga Tasdizen. Detection of neuron membranes in electron microscopy images using a serial neural network architecture. *Medical Image Analysis*, 14(6):770–783, 2010.

[81] Elizabeth Jurrus, Melissa Hardy, Tolga Tasdizen, P. Thomas Fletcher, Pavel Koshevoy, Chi Bin Chien, Winfried Denk, and Ross Whitaker. Axon tracking in serial block-face scanning electron microscopy. *Medical Image Analysis*, 13(1):180–188, 2009.

[82] John Aloimonos, Isaac Weiss, and Amit Bandyopadhyay. Active vision. *International Journal of Computer Vision*, 1(4):333–356, 1988.

[83] Ting Liu, Mojtaba Seyedhosseini, Mark Ellisman, and Tolga Tasdizen. Watershed merge forest classification for electron microscopy image stack segmentation. In *2013 IEEE International Conference on Image Processing, ICIP 2013 - Proceedings*, pages 4069–4073, 2013.

[84] Shiv N. Vitaladevuni and Ronen Basri. Co-clustering of image segments using convex optimization applied to EM neuronal reconstruction. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 2203–2210, 2010.

[85] Viren Jain, Srinivas C Turaga, K Briggman, Moritz N Helmstaedter, Winfried Denk, and H Sebastian Seung. Learning to Agglomerate Superpixel Hierarchies. *Advances in Neural Information Processing Systems*, pages 648–656, 2011.

[86] Bjoern Andres, Ullrich Koethe, Thorben Kroeger, Moritz Helmstaedter, Kevin L. Briggman, Winfried Denk, and Fred A. Hamprecht. 3D segmentation of SBFSEM images of neuropil by a graphical model over supervoxel boundaries. *Medical Image Analysis*, 16(4):796–805, 2012.

[87] Amelio Vázquez-Reina, Michael Gelbart, Daniel Huang, Jeff Lichtman, Eric Miller, and Hanspeter Pfister. Segmentation fusion for connectomics. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 177–184, 2011.

[88] Jan Funke, Bjoern Andres, Fred A. Hamprecht, Albert Cardona, and Matthew Cook. Efficient automatic 3D-reconstruction of branching neurons from em data. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 1004–1011, 2012.

[89] Graham Knott, Herschel Marchman, David Wall, and Ben Lich. Serial Section Scanning Electron Microscopy of Adult Brain Tissue Using Focused Ion Beam Milling. *Journal of Neuroscience*, 28(12):2959–2964, 2008.

[90] A.K. Jain, Yu Zhong, and Sridhar Lakshmanan. Object matching using deformable templates. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18(3):267–278, 1996.

[91] Kai Briechle and Uwe D. Hanebeck. Template matching using fast normalized cross correlation. In *Proceedings of SPIE*, pages 95–102, 2001.

[92] Yassar Almutairi, Timothy F. Cootes, and Karl Kadler. Tracking collagen fibres through image volumes from sbfsem. In Tryphon Lambrou and Xujiong Ye, editors, *MIUA*, pages 40–45. BMVA, 2015.

[93] Erik Meijering. Cell segmentation: 50 Years down the road. *IEEE Signal Processing Magazine*, 29(5):140–145, 2012.

[94] Humayun Irshad, Antoine Veillard, Ludovic Roux, and Daniel Racoceanu. Methods for nuclei detection, segmentation, and classification in digital histopathology: A review-current status and future potential. *IEEE Reviews in Biomedical Engineering*, 7:97–114, 2014.

[95] Shys Fan Yang-Mao, Yung Kuan Chan, and Yen Ping Chu. Edge enhancement nucleus and cytoplast contour detector of cervical smear images. *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, 38(2):353–366, 2008.

[96] Jun Xu, Lei Xiang, Qingshan Liu, Hannah Gilmore, Jianzhong Wu, Jinghai Tang, and Anant Madabhushi. Stacked sparse autoencoder (SSAE) for nuclei detection on breast cancer histopathology images. *IEEE Transactions on Medical Imaging*, 35(1):119–130, 2016.

[97] Yuanpu Xie, Fuyong Xing, Xiangfei Kong, Hai Su, and Lin Yang. Beyond classification: Structured regression for robust cell detection using convolutional neural network. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, volume 9351, pages 358–365, 2015.

[98] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: A Simple Way to Prevent Neural Networks from Overfitting. *Journal of Machine Learning Research*, 15:929–1958, 2014.

[99] Emo Welzl. Smallest enclosing disks (balls and ellipsoids). In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, volume 555 LNCS, pages 359–370, 1991.

[100] Kelly H Zou, Simon K Warfield, Aditya Bharatha, Clare M.C. Tempany, Michael R Kaus, Steven J Haker, William M Wells, Ferenc A Jolesz, and

Ron Kikinis. Statistical Validation of Image Segmentation Quality Based on a Spatial Overlap Index. *Academic Radiology*, 11(2):178–189, 2004.