High-Throughput Image Analysis of Zebrafish Models of Parkinson's Disease

The University Of Sheffield.

Bo Dong

Department of Electronic and Electrical Engineering

University of Sheffield

This dissertation is submitted for the degree of

Doctor of Philosophy

August 2017

To my parents and my wife

Declaration

I hereby declare that except where specific reference is made to the work of others, the contents of this thesis are original and have not been submitted in whole or in part for consideration for any other degree or qualification in this, or any other University. This thesis is the result of my own work and parts of it have been taken from certain published/to be published journal/conference papers. These publications are listed below:

1. **B. Dong**, L. Shao, A. F. Frangi, O. Bandmann, and M. Da Costa. "Three-Dimensional Deconvolution of Wide Field Microscopy with Sparse Priors: Application to Zebrafish Imagery." In 2014 IEEE 22nd International Conference on Pattern Recognition (ICPR), pp. 865-870. IEEE, 2014. (Chapter 2)

2. **B. Dong**, L. Shao, M. D. Costa, O. Bandmann, and A. F. Frangi. "Deep learning for automatic cell detection in wide-field microscopy zebrafish images." In 2015 IEEE 12th International Symposium on Biomedical Imaging (ISBI), pp. 772-776. IEEE, 2015. (Chapter 3)

3. L. Zhang, A. Gooya, **B. Dong**, R. Hua, S. E. Petersen , P. Medrano-Gracia, and A. F. Frangi. "Automated Quality Assessment of Cardiac MR Images Using

Convolutional Neural Networks." In International Workshop on Simulation and Synthesis in Medical Imaging, pp. 138-145. Springer International Publishing, 2016.

4. **B. Dong**, L. Shao, M. Da Costa, O. Bandmann, and A. F. Frangi. "Automatic Neuron Detection in Three-Dimensional Light Microscopy Zebrafish Images." The 4th Conference on Pattern Recognition (ACPR 2017) (Chapter 3)

5. **B. Dong**, L. Shao, L. Trollope, O. Bandmann, and A. F. Frangi. "Interactive Cell Counting with Sparse Bayesian Poisson Regression in Fluorescence Microscopy Images." The 4th Conference on Pattern Recognition (ACPR 2017) (Chapter 4)

Bo Dong

August 2017

Acknowledgements

First I would like to thank both of my co-supervisors Prof. Ling Shao and Prof. Alejandro F Frangi for their enthusiasm, guidance, and unrelenting support throughout this process. Prof. Ling Shao offered me a unique opportunity to study in Sheffield, and he has given me a lot of encouragement and useful suggestions. Most importantly, he has given me maximum independence to pursue my ideas. Prof. Alejandro F Frangi's positive outlook and confidence in my research inspired me and gave me confidence. He has routinely gone beyond his duty to instil great confidence in both my work and myself.

I would like to thank my colleagues: Dr. Xiantong Zhen, Dr. Simon Jones, Dr. Ruomei Yan, Dr. Di Wu, Dr. Li Liu, Dr. Fan Zhu, Dr. Feng Zheng, Dr. Mengyang Yu, Yang Long, Redzuan Bin Abdul Manap, Fehri Hamid, Rui Hua, Peng Peng, Zhang Le, YuanJun Lv, Yawen Huang, and Danyang Wang. It has been my most valuable memory and experience of working with them.

I would like to thank two academics: Prof. Oliver Bandman and Dr. Ali Gooya, who have been so helpful during my PhD study.

I specially thank Dr. Marc Da Costa and Dr. Lisa C Trollope for their help and kindness, and also thank their contribution of providing zebrafish datasets for my project.

Abstract

Light microscopy can be used to advance our understanding of the molecular and cellular biology related to human health and diseases. As a powerful new vertebrate model, zebrafish have been used in various research areas, particularly in cancer and Parkinson's disease research. Large-scale data extraction from microscopy is highly attractive because it enables unbiased multivariate analysis that could lead to systems medicine approaches. To obtain useful information from large-scale data, high-throughput image analysis methods and applications are desperately required.

In this thesis, we have explored methods and developed applications for high-throughput light microscopy zebrafish image analysis, including addressing the key problems related to three-dimensional (3D) deconvolution/deblurring, robust feature detection and description, and object counting. In biological image analysis, dealing with out-of-focus light noise, low image quality, large-scale dataset, illumination, overlapping, occlusion and insufficient prior knowledge remains challenging. Methods to address the following problems have been presented in this thesis.

The *low image quality of fluorescence microscopy images* is addressed in Chapter 3. Owing to the limitations of light microscopes, the whole imaging process can be considered as a convolution between the object and the point spread function.

Out-of-focus light and noise cause loss of detail in captured images. Deconvolution is an ill-posed inverse problem; thus, regularization methods are required to obtain better results. A maximum a posterior approach with a novel regularisation strategy to remove out-of-focus blur in 3D fluorescence microscopy images is introduced in Chapter 3.

The second problem is *dyed dopaminergic neurone detection in zebrafish RGB optical sections* (Chapter 4). Owing to the large-scale of the image, low image quality, irregular appearance of neurones and touching situations, manually counting individual neurones via the microscope can be labour-intensive, time-consuming, subjective, and error-prone. To solve this problem, this thesis explores different methods to detect individual neurones in 3D zebrafish RGB images, including using detectors with many different handcrafted features and features learned automatically from deep learning architectures. An additional class-imbalanced problem is discovered during the experiments involving the training of patch-based deep learning techniques using a large-scale dataset that contains a limited number of positive samples. To solve this problem, a dynamic cascade framework with deep learning architectures is designed.

The last problem is *cell counting in fluorescent microscopy images* (Chapter 5). Detecting individual cells in two-dimensional (2D) fluorescence microscopy images is difficult owing to overlap. Rather than counting-by-detection methods, a counting-by-regression method with an interactive interface for cell counting in a fluorescent image is proposed. Sparse Bayesian Poisson regression based on a Relevance Vector Machine framework is also proposed. The proposed framework enables accurate

counting of a discrete number of cells and leads to much sparser models, which results in faster performance and maintains a comparable generalisation error.

Table of contents

List of figures

List of tables

List of Algorithms

List of Acronyms

2D	two-dimensional
3D	three-dimensional
AIDI	adaptive image deconvolution algorithm
ANN	artificial neural networks
CCD	charge-coupled device
CNN	convolutional neural network
CPU	conventional central processing unit
CUDA	compute unified device architecture
FFT	fast Fourier transform
GPR	Gaussian process regression
GPU	graphics processing units
HL	hyper-Laplacian
HOG	histogram of gradient
IIS	image intensity standardisation
KFTV	total variation regularisation in the frequency domain
LBP	local binary pattern
MAE	mean-absolute-error
MAP	maximum a posteriori
MESA	maximum excess over subarrays
MLEM	maximum likelihood expectation maximisation
NMISE	normalized mean integrated squared error
OTF	optical transfer function
PD	Parkinson's disease
PS	probabilistic sampling
PSF	point spread function
PSNR	peak signal-to-noise ratio
RGB	red-green-blue
RMSE	root-mean-square-error
RVM	relevance vector machine
SBPR	sparse Bayesian Poisson regression
SIFT	scale-invariant feature transform
SMOTE	synthetic minority oversampling technique
SNR	signal-to-noise ratio
SVM	support vector machine
TH	tyrosine hydroxylase
TV	total variation
TM	Tikhonov-Miller
VRIGL	variance refractive index Gibson and Lanni
WF	widefield
WISH	whole-mount *in situ* hybridisation

Chapter 1

Introduction

1.1 Background

1.1.1 Zebrafish Model for Parkinson's Disease

Zebrafish have been used as an important model organism for research into many types of diseases, such as cancer and Parkinson's disease (PD). Using zebrafish as a research model has several advantages. For example, the zebrafish is a vertebrate and has an immune system that is very similar to that of humans. In addition, zebrafish embryos develop externally and are transparent, which provides an unparalleled opportunity to study the biology of the immune response. Furthermore, compared to image analysis of tissue samples from other species, such as mice or humans, where slides with clearly orientated tissue samples of well-defined standardised thickness typically form the basis of subsequent image acquisition and analysis.

A growing number of transgenic and mutant zebrafish lines have been developed for disease research to answer fundamental questions about disease pathogenesis. For example, in research into mechanisms linked to PD, the dopaminergic nerve cells, which produce dopamine to control muscle movement, gradually die. The discovery

of monogenically inherited PD genes has greatly enhanced understanding of the underlying mechanisms leading to PD but have not yet resulted in better treatments.

Recently, researchers focused on the death of dopaminergic neurones in a zebrafish stable mutant line with a loss-of-function mutation in PD gene PINK1. Autosomal recessively inherited PINK1 mutations are known to cause early-onset PD in humans [1]. Interestingly, this mutant zebrafish model showed an approximate 25% reduction of dopaminergic neurones as early as three days post fertilisation compared with wild-type zebrafish controls, as determined by counting tyrosine hydroxylase (TH) positive neurones after whole-mount *in situ* hybridisation (WISH). Here, TH is the rate-limiting enzyme in the synthesis of dopamine which is frequently used as a marker for dopaminergic neurones; the marked neurones are called TH-labelled neurones.

1.1.2 Light Microscopy: Imaging as a Convolution

Light microscopy, which has been fundamental in advancing understanding of the molecular and cellular biology of human health and disease, has remained a largely qualitative approach. Large-scale data extraction from light microscopy is highly attractive because it facilitates unbiased multivariate analysis that can lead to systems medicines approaches. This is a rapidly developing research area that has delivered significant advances in '-omic' biological sciences (e.g. genomic, proteomic, etc.). However, clinical medical imaging has been severely limited in light microscopy imaging due to the unavailability of appropriate analysis tools. High throughput image methods and software are required to reduce labour costs and speed up analysis processes.

A light microscope (or optical microscope) uses visible light to magnify objects. Three primary factors determine the lateral spatial resolution in light microscopy, i.e. the numerical aperture, the wavelength of the light and the direction of the optical

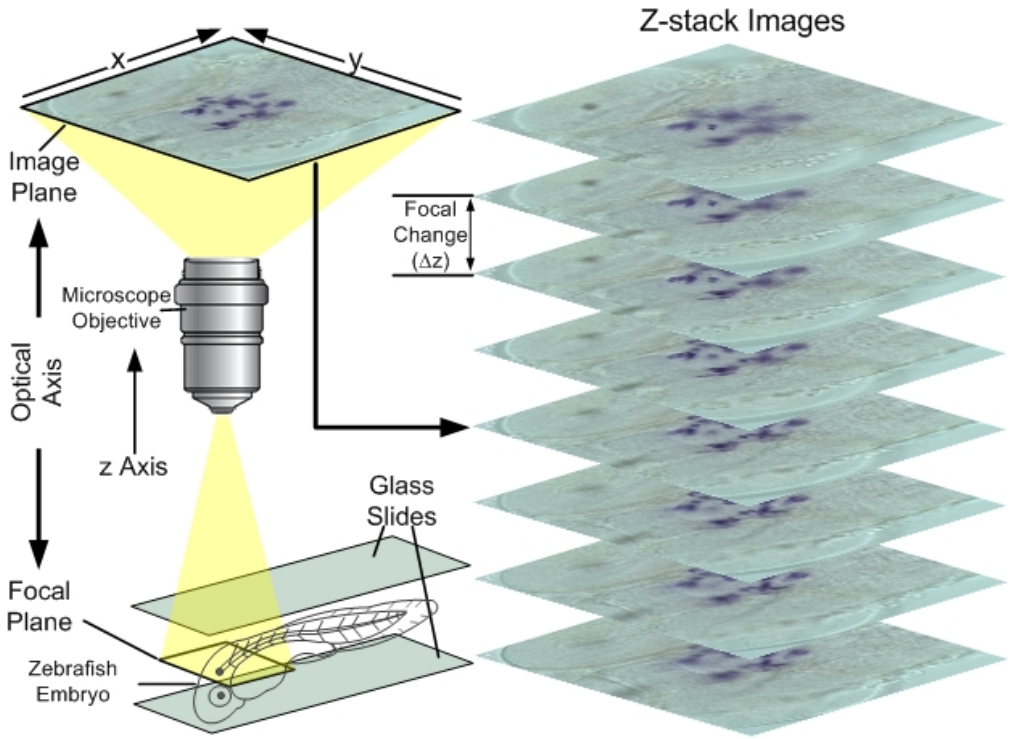


Fig. 1.1 Acquisition of optical sections (z-stack).

axis (depth of field) [2]. The former two factors are inherent limitations of optical microscopy, and the third factor, i.e. depth of field, contributes to 'out-of-focus' light during image processing.

With a certain depth of field, features within this distance of the focal plane contribute to the image, thereby swamping or even losing details. Due to the limitations of optical microscopes, the entire imaging process can be considered a convolution between the object and the point spread function (PSF). The PSF is simply an image of a single point that describes how each point of the object is blurred. Note that the PSF is wholly determined by the microscope. By knowing all optical properties of the optical microscope, the PSF can be measured or estimated.

Fig 1.1 shows the acquisition of optical sections, which are also known as z-stacks. By moving the objective once a unit in the z-direction (depth of field or Δz), the microscope can produce clear images of focal planes in a thick sample. Due to the loss of specimen information between adjacent focal planes and the effect of

out-of-focus light, a smaller depth of field results in better optical sectioning, which also generates more z-stack slices. Using computer techniques, z-stack images can be reconstructed or viewed as a three-dimensional (3D) volume or a single 3D image, which makes further image processing techniques, such as 3D deconvolution and 3D neurone detection, easier to perform.

Linearity and shift invariance are two conditions that describe an imaging process as a convolution. In this thesis, two types of optical microscopy are used, i.e. widefield (WF) and confocal microscopy. Both WF and confocal microscopy are linear and shift invariant processes to a good approximation [3].We denote the specimen as a function of position $f(\mathbf{x})$, the PSF as $h(\mathbf{x})$ and the resulting image as $g(\mathbf{x})$. Then, the image is the convolution $(*)$ of the object with the PSF:

$$g(\mathbf{x}) = f(\mathbf{x}) * h(\mathbf{x}). \tag{1.1}$$

During imaging, except out-of-focus light, some unexpected noise also exists:

$$g = f * h + n. \tag{1.2}$$

Unlike common additional noise in natural images, such as Gaussian or white noise, the dominant noise sources in the microscope are electronic noise or, in some cases, fluctuations in illumination intensity. For fluorescence microscopy deconvolution, it is often assumed that the noise follows a Poisson distribution due to the photon-counting process [4]. The imaging process can be written in short form as follows:

$$g = n(f * h), \tag{1.3}$$

where n is the Poisson noise function. From a mathematical perspective, convolution processes for WF and confocal microscopy do not differ. However, there are principle

differences between the two types of microscope. Confocal microscopy can eliminate most the out-of-focus light, which can increase the contrast of details and improve the in-plane resolution. For example, in fluorescent confocal microscopy, the image is scanned at one spot or at an array of spots at a time, and a charge-coupled device (CCD) camera can record up to 30% of the total light emitted by the specimen [2].

However, eliminating out-of-focus light means that part of the light emitted by the specimen will be excluded. In contrast, with conventional WF microscopy, all pixels are recorded simultaneously, including all out-of-focus light. In principle, the PSF of WF microscopy is wider than that of confocal microscopy. To remove out-of-focus light, a computer image processing technique, i.e. deconvolution/deblurring, can be used. With advances in computer hardware and parallel computation techniques, it is much more convenient and user friendly to use WF CCD microscopes with deconvolution software. Consequently, WF microscopy with deconvolution techniques is used more frequently.

1.2 Objectives

This thesis primarily focuses on light microscopy zebrafish image analysis. The objectives are (1) to explore methods for high-throughput visual information processing in large-scale light microscopy zebrafish datasets, (2) transfer expertise and experience from computer vision to light microscopy image analysis, (3) address key problems in light microscopy images, (4) develop image-based modelling of dynamic biological processes based on robust machine learning techniques and (5) implement open-source tools and applications for light microscopy zebrafish image analysis.

1.2.1 Definition of Problems

Three key problems are considered in this thesis.

Problem 1. Low image quality of widefield fluorescence microscopy

With the limitation of imaging options and the imaging systems for the beginning of this project, the quality of obtained WF fluorescence microscopy images is quite low. The imaging process can be viewed as a convolution process between the zebrafish specimen and the PSF, which makes the recorded image more blurry. More details about light microscopy can be seen in the previous section 1.1.2. The alternative microscopes can be used to remove out-of-focus light to improve the image quality, such as confocal fluorescence microscope and light sheet fluorescence microscope (LSFM). Due to the limitations of the equipments during the beginning of this project. Widefield fluorescence microscope with computer aided deconvolution methods are used in this thesis. 3D deconvolution attempts to remove out-of-focus light and additional noise in optical sections, particularly for WF fluorescence microscopy. However, the deconvolution is an ill-posed inverse problem, so regularization methods are needed for obtaining better results. In this thesis, a maximum a posterior approach with novel regularisation strategy to remove the out-of-focus blur in 3D fluorescence microscopy images is explored.

Problem 2. Cell Detection

In a 3D multichannel zebrafish dataset, specific cells must be detected individually. Thus, the following must be considered.

(a) ***Cell detection with handcrafted features***. Typical supervised analysis considers several different handcrafted features. Various handcrafted feature extraction techniques, such as histogram of gradient (HOG), local binary

pattern (LBP) and scale-invariant feature transform (SIFT), have been proposed. Important features can be selected with several experiments using many different handcrafted features. With the same typical detector, different features and combinations are tested.

(b) ***Automatic cell detection with learned high-level features***. It is not easy to find useful and perfect low-level features, as they are generally insufficient to directly represent the semantic meaning of desired object. Different combination of feature vectors will also make the detection process inefficient. There are two ways to extract features from an image, i.e. handcrafted feature extraction and automatic feature extraction using deep learning architectures. Automatic cell detection using deep architectures is investigated. With deep learning techniques, important high-level object features can be learned automatically. Among different deep architectures, convolutional neural network (CNN) is one of the best options. An automatic framework with CNN is explored and tested with our zebrafish dataset.

(c) ***High-dimensional feature learning***. Typical feature extraction methods are mainly explored in two-dimensional (2D) space. However, the cell detection task requires exploring high-dimensional features in optical sections (z-stack images). High-dimensional deep architectures are studied and evaluated with our multi-channel z-stack images for capturing high-dimensional features directly. A fully automatic cell detection framework with high-dimensional deep architectures is designed for capturing centre pixels of desired neurons.

(d) ***Class-imbalanced problem***. A dataset is considered 'imbalanced' if the number of samples in a single class is much smaller or larger than that of other classes [5]. More details about class-imbalanced problem can be found in Chapter 2.4. Typically, the minority class is of primary interest. However, standard learning algorithms trained using a dataset with an imbalanced

distribution of classes will produce biases toward the majority class, thereby resulting in misclassified and unsatisfactory results. The class-imbalanced problem will be encountered when the supervised learning method is used with our positive-label-only dataset. In our situation, the class-imbalanced problem is serious with the dataset containing only positive examples. In this thesis, a dynamic cascade framework with deep neural network is designed to address the class-imbalanced problem.

Problem 3. Interactive cell counting in the presence of cell overlap

In PD research, only cells in specific regions of the brain are counted for further analysis. Counting cells in a zebrafish fluorescence microscopy image involves the following problems. Although our designed deep learned detection methods for problem 2 can be applied in this problem, but the accuracy is reduced because of the severe overlapping, touching, and out-focus-light situations in 2D fluorescence microscopy. The compared results with CNN will be presented in Chapter 5. To achieve the requirement of comparing the number of dopaminergic neurons in certain regions, interactive application with regression method is designed.

(a) ***Specific region selection/segmentation.*** An interaction region selection method with user input is considered for specific region segmentation. With this interactive application, users can select specific regions they want.

(b) ***Counting in specific selected regions with overlapping conditions.*** In reality, there are nearly no overlapping cells in 3D space; however, in a two-dimensional (2D) projection image, overlapping is severe. A counting method in specific regions that does not require the detection of an individual cell is investigated.

1.2.2 Applications and Tools

A zebrafish light microscopy dataset provides opportunities to discover the unique structure inside the zebrafish brain and apply knowledge from the computer vision domain to medical image analysis to create practical and interactive applications and tools. In this thesis, several applications are developed for light microscopy zebrafish image analysis.

Microscopy Image Deconvolution Application

Due to the limitations of the imaging system, images captured using a WF fluorescence microscope are significantly influenced and degraded by the out-of-focus signal. Methods to increase image quality and restore details for 3D fluorescence microscopy are extremely useful for disease researchers. Applications for microscopy image deconvolution are developed for restoring details for 3D fluorescence microscopy images.

Automatic Cell Detection Application

The ability to detect cells in microscopy images can advance research, such as tracking, counting and neurone analysis. To detect certain cell types individually, such as mitosis and dopaminergic neurones, we aim to develop a robust machine learning-based detector that can be applied to 3D low-quality light microscopy images.

Interactive Cell Segmentation Application

In disease research, cells must be detected or counted in specific regions selected by a neuroscientist. Thus, an interactive cell segmentation application that takes user input is required.

Cell Counting Application

Cell counting is an important subset of cytometry. We aim to develop an application to count target cells in images where individual object detectors do not work properly due to overlapping, merging, crowding and irregular appearance.

1.3 Contributions

The main contributions of this thesis are described in the following.

• The maximum a posteriori (MAP) method with sparse priors is applied to 3D deconvolution for Poisson noise WF microscopy.

• Two priors with Hyper-Laplacian distribution and a local mask are used in the deconvolution framework for removing out-of-focus light and reducing ringing artefacts, respectively.

• To assist dopaminergic cell detection in a zebrafish model of PD using TH-labelled neurones in WF microscopy z-stack RGB images, high-throughput automatic neurone detection methods with 2D and 3D frameworks are proposed.

• A dynamic cascade framework with a deep neural network is designed to address the problem of training deep learning structures using a large-scale image dataset that contains a limited number of positive samples. To the best of our knowledge, the employed imbalanced dataset used for training is one of the largest to date.

• We successfully apply 2D and 3D deep learning architectures to cell detection in a high-dimensional dataset.

• An efficient GPU-based parallel framework is implemented for training.

• We contribute the first zebrafish database [1] for cell detection. It is expected that this database will serve as a comparative benchmark for future research.

• We introduce a counting-by-regression framework for cell counting in fluorescent

[1] http://www.cistib.org/cistib_shf/index.php/download/zebrafish-embryo-datasets

microscopy images. The proposed framework integrates sparse Poisson Bayesian regression and interactive superpixel-based graph cut segmentation.

• A sparse Bayesian Poisson regression (SBPR) based on a Relevance Vector Machine (RVM) framework that enables a discrete number of cells to be counted accurately with a sparse solution resulting significantly less computation cost, is proposed.

• An interactive interface is implemented with a superpixel-based graph cut for objective foreground segmentation.

• Integrating the sparse regression model with suitable features for cell counting in densely clustered regions.

1.4 Thesis Outline

The remainder of this thesis is organized as follows.

- In Chapter 2, previous work related to high-throughput image analysis, deep learning, neurone detection and neurone counting-by-regression is reviewed.

- In Chapter 3, the 3D deconvolution method for WF fluorescence microscopy zebrafish images is presented. The maximum a posteriori estimation method with sparse image priors, including the hyper-Laplacian prior and a local prior with a mask, is presented to solve non-blind and blind 3D deconvolution problems for WF fluorescence microscopy. Then, the proposed method is tested using synthetic microscopy data and real WF microscopy data from zebrafish. Compared to other regularisation methods, the unique contribution of the proposed method is combining the hyper-Laplacian model and a local mask in the MAP framework.

- In Chapter 4, individual cell detection in 3D light microscopy zebrafish images with three RGB channels is discussed. First, experiments with different hand-

crafted features are performed to identify the main feature. Then, a supervised max-pooling Convolutional Neural Network (CNN) is trained to detect cell pixels in regions preselected by a Support Vector Machine (SVM) classifier. Finally, a cascade framework with deep learning architectures to improve detection accuracy and optimise the training procedure is investigated.

- Chapter 5 a robust counting-by-regression method for fluorescent microscopy images. In the framework, cell regions are first segmented using graph cut approach based on hierarchical superpixels interactively segment cell regions, and then the discrete counts in each segmented region will be predicted using SBPR method with extracted local features. Then, SBPR is applied to predict discrete counts in each segmented region based on the extracted local features. Sparse regression leads to much sparser models, resulting in faster performance with test data while maintaining a comparable generalisation error. The proposed counting method is evaluated using two different datasets and is compared to different state-of-the-art techniques.

- Conclusions and suggestions for future work are presented in Chapter 6.

Chapter 2

Literature Review

2.1 High-throughput Zebrafish Image Analysis

Microscopy image analysis is currently being applied to various diverse fields, such as medicine, biological research, drug testing and metallurgy. High-throughput microscopy with computer-assisted intervention makes it possible for biologists to capture high-resolution fluorescent or colour specimen images rapidly. However, manually analysing large-scale data is difficult and expensive. Thus, as the amount of available data increases, automatic high-throughput microscopy image analysis is urgently required. However, extracting interpretable information and knowledge from raw images is difficult and developing a high-throughput image analysis system to qualify biological processes with few or no human interactions is challenging [6].

Recently, 3D imaging techniques that can be applied to light microscopes have been developed, such as mosaic/optical sectioning confocal laser scanning and mosaic/optical sectioning WF microscopy [7]. Zebrafish are a perfect model to use as specimens in such 3D imaging techniques, and several researchers have focused on zebrafish image analysis. For example, studies that focus on deconvolution or deblurring in zebrafish embryo images [8], segmentation and characterisation of zebrafish retinal horizontal neurones [9], counting zebrafish neurone somata [7], and

high-throughput analysis for region detection and quantification of zebrafish embryo images [10, 11] have been conducted. The 3D imaging techniques combined with suitable neuronal stains have created new possibilities for neurone detection and counting within entire volumes.

Several microscopy applications for high-throughput biological image analysis have been proposed, such as ImageJ/Fiji [12, 13], Ilastik [14], ICY [15], and Vaa3D [16]. We will evaluate some of those existing applications and tools for different problems compared with our proposed methods and applications.

2.2 Deep learning in Biomedical Imaging

Deep learning methods attempt to represent high-level abstractions in data with multiple-layer architectures [17–21]. Some deep learning algorithms, such as artificial neural networks (ANN), CNNs [22], Deep Belief Networks [23], and stacked auto-encoders [24], have been developed for different difficult tasks. Rather than adopting problem-specific handcrafted features, such as HOG and SIFT, deep learning algorithms, e.g. CNNs, can automatically learn high-level abstractions from training data.

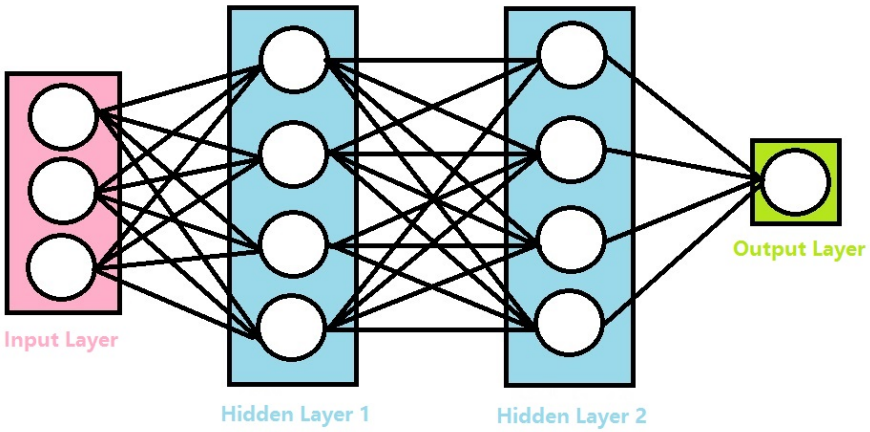


Fig. 2.1 A regular three-layer neural network.

Fig 2.1 shows a regular 3-layer ANN, i.e. a multi-layer perceptron (MLP), that includes an input layer, two hidden layers and one output layer. Formally, a two-hidden-layer ANN can be represented as the function $f : R^D \rightarrow R^L$, where D is the size of the input vector $\mathbf{x}$ and L is the size of the output vector $f(\mathbf{x})$. Then, the matrix notation will become:

$$f(\mathbf{x}) = G\left\{b^{(3)} + W^{(3)}\left[s\left(b^{(2)} + W^{(2)}(s(b^{(1)} + W^{(1)}\mathbf{x}))\right)\right]\right\}, \tag{2.1}$$

where $b^{(1)}$, $b^{(2)}$ and $b^{(3)}$ are bias vectors; weight matrices $W^{(1)}$, $W^{(2)}$, and $W^{(3)}$; activation functions G and s. Typically, s can be a *tanh* or *sigmoid* function, and G can be a *softmax* function. To train this network, all parameters $\theta = \{W^{(1)}, W^{(2)}, W^{(3)}, b^{(1)}, b^{(2)}, b^{(3)}\}$ can be learned using Stochastic Gradient Descent methods [25]. Parameters are updated according to the gradients with back-propagation algorithm [26].

CNNs, which are considered one of the most effective deep learning techniques, were introduced in [27] and have been further developed [28, 22, 18]. Among popular deep learning models, CNNs are one of the most successful architectures, which have been applied in image classification [22], face recognition [29], natural language processing [30] and mitosis detection in histology images [31]. Compared to standard feedforward neural networks, CNNs have significantly fewer connections and learnable parameters because they share the same basis function across different image locations [18]. Thus, CNNs are relatively easy to train.

Instead of adopting hand-crafted features, such as Histogram of Oriented Gradients (HOG) and Scale-Invariant Feature Transform (SIFT), deep learning can learn high-level features from training data automatically. Furthermore, CNN is able to reduce learnable parameters significantly by sharing the same basis function across different image locations [18].

CNNs are biologically inspired variants of regular ANNs with a sequence of layers, typically convolutional, pooling and fully-connected layers. Each layer type has unique properties. The convolutional layer includes a parameter sharing scheme. The pooling layer reduces the number of parameters and controls overfitting, and high-level reasoning is implemented in the fully-connected layer.

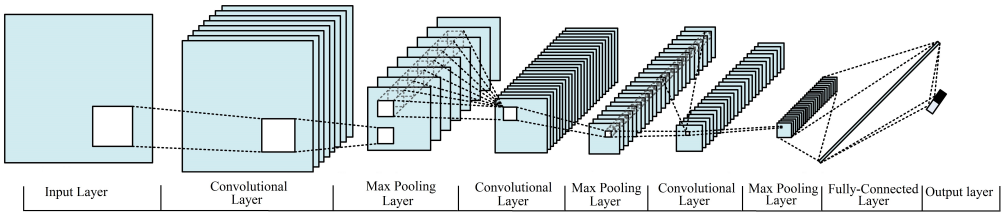


Fig. 2.2 CNN framework

A standard CNN framework is shown in Fig 2.2. The input can be an entire image or a small patch extracted from an image, and each square represents a single feature map in the layer. There are multiple feature maps in each hidden layer to generate a richer representation of the input image. We denote the k_{th} feature map at a given layer as m^k, the filter in the convolutional layer as W^k and bias as b_k. Then, the feature map m^k (for *tanh* non-linearities) is obtained as follows:

$$m_{ij}^k = tanh((W^k * x)_{ij} + b_k). \tag{2.2}$$

By inputting an image array into the CNN structure, the image will be passed through a series of convoluional, pooling, and fully connected layers, and then an output image will be generated. The first layer is called convolutional layer, in which filters are convolved with every unique location on the input image and a unique number is generated in the hidden layer. By sliding the filter over all the locations, the outpput after convolution is called activation map or feature map. By using more filters, we can get more feature maps in the convolutional layer. In the pooling layer, we can undersampling the image to reduce the number of parameters and control

overfitting. By attaching a fully connected layer to the end of the structure, the high level features will be detected and connected to different classes. In 2D CNNs, convolutions can only compute 2D features from the spatial dimensional only. The 3D CNN is first designed and introduced in paper [108] for human action recognition in videos, and it is desirable to capture the motion information in multiple contiguous frames. In our situation, we need to capture the vertical information of the cells in multiple slices in the 3D stack image. The pixel value at position (x,y,z) on j_{th} feature map in i_{th} layer is given by

$$v_{ij}^{xyz} = tanh\left(b_{ij} + \sum_{m}\sum_{p=0}^{P_i-1}\sum_{q=0}^{Q_i-1}\sum_{r=0}^{R_i-1} w_{ijm}^{pqr} v_{(i-1)m}^{(x+p)(y+q)(z+r)}\right), \tag{2.3}$$

where b is the bias and w is the weight.

2.3 Deconvolution in Biomedical Imaging

Due to the limitation of the imaging system, biomedical images are degraded during acquisition. Deconvolution is an inverse problem that can be performed if the imaging process is assumed to be linear and shift invariant. However, inverse problems can be very sensitive to the inputs. As with most of inverse problems, even if the PSF is known, image deconvolution is difficult because some information is lost during the imaging process. Deconvolution without prior knowledge of the PSF, is called 'blind deconvolution'.

As mentioned previously, image deconvolution is an ill-posed problem, i.e. a problem that has *no solution*, *many solutions* or an *unstable* solution [32]. No single mathematical approach can stabilise this type of inherently unstable inverse problem [33]. The most common approach of solving deconvolution is to attempt to solve linear equation (1.2), which, in Fourier space, becomes:

$$G(\mathbf{x}) = F(\mathbf{x})H(\mathbf{x}) \Rightarrow F(\mathbf{x}) = G(\mathbf{x})/H(\mathbf{x}), \tag{2.4}$$

where $G(\mathbf{x})$, $F(\mathbf{x})$ and $H(\mathbf{x})$ are the Fourier transforms of $g(\mathbf{x})$,$f(\mathbf{x})$ and $h(\mathbf{x})$, respectively. $H(\mathbf{x})$, which is Fourier transforms of the PSF, is typically referred to as the optical transfer function (OTF). The image transform is simply the product of the object transform and the OTF, and the latent image is simply the inverse Fourier transform result. However, due to noise and because parts of the OTF become very small or even zero, satisfactory results cannot be generated.

2.3.1 2D Methods

One of the most powerful methods to solve this type of inverse problem is to add some additional 'constraints'. Most inverse problems are ill-posed, and the deconvolution problem is no exception [32]. The observed image (g) can be explained using infinite pairs of the PSF (h) and ground truth image (f) [34]. Therefore, to obtain the best latent image from a blurred image, extra prior knowledge about the known image structure must be added to regularise the deconvolution process. Traditional 2D methods, such as inverse filtering [35], Tikhonov filtering [36], Weiner filtering [37], the Maximum Likelihood (ML) approach, and the Richardson-Lucy algorithm [38, 39], were proposed several decades ago for inverse image restoration problems. Some traditional methods can generate better results with additional constraints, such as the MAP approach, which is an ML approach with priors.

Several methods using MAP estimation with sparse image priors to solve the 2D deconvolution problem have been proposed [34, 40–43]. The MAP approach, as a nonlinear iteration method, has been proved very efficient and accurate in the recovery of a latent image or the true blur kernel of a 2D natural image with blur due to object movement or camera ego-motion [34].

The sparse derivative distribution, as an image prior, has been used to solve many ill-posed 2D problems, such as motion deblurring [43], denoising [44], transparency separation [45], and super resolution [46] problems.

The most well-known regularisation method for the 2D deconvolution problem is called Total Variation (TV), which has been proved very successful for preserving edges and removing noise. The hyper-Laplacian distribution has been applied to 2D non-blind deconvolution [47]. In [43], the hyper-Laplacian distribution was modified using a Gaussian noise model, and a lookup table was developed to improve computation time.

Shan et al. [42] presented an algorithm that uses an MAP to remove motion blur from a single 2D image with two different priors, i.e. global and local priors. To fit the logarithmic gradient distribution, they introduced two piecewise functions, $y = -k|x|$ and $y = -(ax^2 + b)$, which represent the sharp peak and heavy tails of the distribution, respectively. For the local prior, they found that errors in the smooth area between a blurred image gradient and an unblurred image gradient follow a Gaussian distribution. They used a Gaussian distribution with zero mean to formulate the local prior to suppress ringing artefacts. Our local prior part is similar to a previously proposed method [42].

Compared with the gradient distribution using TV regularization, using hyper-Laplacian distribution can be more precise to describe the heavy tails of the logarithmic gradient distribution. However, only using one global prior, ringing artifacts can be produced during deconvolution due to the errors in the estimated PSF function and inaccurately modeled image noise. Based on the literature, using another local prior to reduce the ringing artifacts can be helpful.

2.3.2 3D Methods

Due to success in solving the 2D deconvolution/deblurring problem, several methods based on the MAP approach have been proposed for 3D deconvolution of microscopy data. The iterative and nonlinear maximum likelihood expectation maximisation (MLEM) method, which is similar to the MAP approach, has been designed to solve both 3D blind and non-blind Poisson image deconvolution problems [48–51]. A regularising OTF method using MLEM has been proposed recently [50]. A machine learning method that models the PSF based on MLEM have been introduced for blind deconvolution without any prior to regularise a latent image during restoration [51].

State-of-the-art estimation strategies to demonstrate the validity of the sparse prior model have been proposed for confocal microscopy, including the Tikhonov-Miller (TM) regularisation [52, 53] and TV regularisation [49, 50], which are discussed for comparison. In [53], the TM regularisation method was first used to solve the deconvolution problem for confocal microscopy. This regularisation term can remove noise in Poisson noise images but tends to smooth details, which may result in important information being lost [54].

In [49], the Richardson-Lucy algorithm with TV regularisation was proposed for confocal microscopy. In [50], the TV regularisation term was presented for blind iterative deconvolution by updating both the latent image function and the PSF simultaneously. The TV regularisation method can remove noise and preserve edges effectively; however it may smooth texture and corner details [54].

Different penalty functions have been introduced for the gradient filter, such as $f_1(x) = |\bigtriangledown x|$ for TV regularisation and $f_1(x) = \|\bigtriangledown x\|^2$ for TM regularisation [49]. An adaptive image deconvolution algorithm (AIDA) [55] for multi-frame and 3D data has been proposed. A quadratic regularisation method [56] has been used to preserve edges and other sharp object features.

There are several freely-available software applications for 3D deconvolution problems, such as Iterative 3D [57], FFTJ and DeconvolutionJ [58], Parallel spectral Deconvolution 3D [59], DeconvolutionLab [60], and one commercial program i.e. Huygens (SVI, Hilversum, The Netherlands). These programs require the theoretical PSF or the PSF parameters based on the imaging system.

2.3.3 Blind Deconvolution

Blind deconvolution is essential if some parameters related to the PSF are unknown. It requires additional assumption on the latent image or the PSF due to the ill-posed feature of this problem. The blind deconvolution is heavily influenced by the initial guess of the PSF. If an inaccurate PSF is used, a processing ringing artefacts will occur around objects with sharp edges. The noise in the image may also be amplified during deconvolution process.

Blind deconvolution works started in early 1970s in astronomical imaging and medical imaging. Most of the works can be performed iteratively to improve the estimation of PSF and the latent image or signal. The most famous iterative methods are maximum a posteriori estimation and expectation-maximization algorithms. A good PSF estimation is very helpful to make the iterative process quicker to converge.

Usually, the blind deconvolution methods use regularization methods to reduce this kind of amplification. For example, [50] uses TV regularisation in the frequency domain (KFTV) of the PSF to preserve edges in the data. In [51], all parameters related to the PSF are known except for depth aberration d and refractive index n_p of the imaged sample. Many sets of PSFs are obtained by changing depth aberration d based on a PSF model [61]; from this, the shape of the PSF is learned as a prior based on principle components analysis [62] or kernel principal component analysis [63]. Even many works achieved tremendous progress, the results are still far from perfect.

2.4 Neurone Detection and the Class-imbalanced Problem

Neurone detection in microscopy images is a significant preliminary step for further experiments and analysis. Some detection algorithms have been published in computer vision and biomedical image analysis fields for cell counting/detection purposes [12, 64]. Automated and accurate detection of neurones from microscopic images has been extensively studied [65, 64]. However, most automated 3D cell detection methods are not suitable for manual cell detection [64].

In the past few decades, three main types of cell detection algorithms have emerged. The first class is segmentation- or thresholding-based detection [66]. The second class is model- or feature-based detection and counting [67, 68], and the third class is learning-based cell detection [69, 70, 10]. With the development of powerful computational equipment and continuously improved multi-layer algorithms, deep learning techniques have achieved significant success in many different fields.

The class-imbalanced problem affects most standard learning algorithms that minimise a global error function without taking the imbalanced data distribution into consideration. The performance of standard learning algorithms may be reduced by irrelevant features and high-dimensional data, particularly in class-imbalanced data [71]. Although a supervised deep neural network can learn important features automatically, performance is highly limited by the input data.

Methods for handling class-imbalanced problems have been reviewed in literature [5, 71]. These methods can primarily be divided into two categories. The first category is based on sampling methods [72], such as undersampling, oversampling and probabilistic sampling. In [73], a single deep neural network was trained with random sampling to segment neuronal membranes in electron microscopy images. In [72, 31], probabilistic sampling was used to train two deep neural networks to

improve the final detection accuracy. Probabilistic sampling is a unique sampling method that trains two probabilistic classifiers. The first deep neural network is trained with a class-balanced subset randomly or uniformly selected from the original dataset. With the first pre-trained network, the second training dataset is sampled by assigning a probability in the original dataset. In [31], the author tested 2D electronic microscopy data using two CNNs with probabilistic sampling, which performed better than a single CNN that does not address the class-imbalanced problem.

The second category is an ensemble-based method [74], such as cost-sensitive learning [75], boosting-based methods [76] and cascade classifiers [77]. Viola and Jones [78] introduced a cascade of classifiers for rapid face detection with increasing numbers of features in three stages, which achieved a high detection rate and reduced computation time significantly. Inspired by Viola and Jones' method, Wang and Hunter [77] introduced a cascade framework with diverse models for robust pose recognition. In our zebrafish dataset, only dozens of central voxels were labelled by experts as positive examples in each 3D stack for the neurone detection task, which makes neurone examples under-represented and causes a serious class-imbalanced problem when training supervised learning algorithms.

The current problem is that most previous methods are only adaptable to a small dataset with a small imbalance ratio (less than 100) [5, 71]. Such methods are impractical for large high-dimensional datasets with extraordinarily large imbalance ratios. For example, synthetic minority oversampling technique (SMOTE) [79], one of the most famous oversampling methods, causes overfitting if minority-class examples are generated. Note that it is very difficult to determine misclassification costs in cost-sensitive learning methods [75]. Furthermore, it is very impractical to use a boosting method to boost a simple classifier to a stronger one within hundreds or thousands of iterations using millions of training examples [71]. Moreover, it is very difficult to find appropriate features to classify different classes correctly, and

it takes significant time to calculate many different features, especially with a large high-dimensional dataset.

To solve our problem with the extremely large imbalance ratio with deep learning architectures, only using one strategy is not enough. Different strategies can be combined, including sampling, probabilistic sampling, and cascade framework, to reduce the imbalance ratio, selecting useful training data and improve the detection accuracy at the same time.

In the following, we will review the basic support vector machine method.

SVM was first introduced in 1992 [80], and then become popular because of its success in hand written digit recognition. Let $\mathbf{x}_i, y_i$ be the training examples, $i = 1, ..., l$, where $\mathbf{x}_i \in \mathbb{R}^d$, and $y_i \in \{1, -1\}$. Given a hyperplane $(\precapprox, b)$, a function, which can classify the data, is given by

$$f(x) = sign(\mathbf{w} \cdot \mathbf{x} + b). \tag{2.5}$$

This hyperplane should satisfy:

$$y_i(\mathbf{x}_i \cdot \mathbf{w} + b) \geq 1, \quad \forall i \tag{2.6}$$

To obtain the geometric distance from the hyperplane to the data points, we must normalize by the magnitude of $\mathbf{w}$. Therefore, the distance is

$$d\left((\mathbf{w}, b), \mathbf{x}_i\right) = \frac{y_i(\mathbf{x}_i \cdot \mathbf{w} + b)}{|\mathbf{w}|} \geq \frac{1}{|\mathbf{w}|} \tag{2.7}$$

The hyperplane will be found by minimizing $|\mathbf{w}|$, which can be done by using Lagrange multipliers [81].

2.5 Neurone Counting by Density Regression

The counting problem is to estimate the number of desired objects in videos or still images. Object counting occurs in many real-world scenarios and applications, including surveillance, security inspection, crowd counting and cell counting. The counting accuracy of fully unsupervised approaches is limited; thus, other methods based on supervised learning have been considered. Several methods solve counting problems by performing individual segmentation based on intensity or shape modelling.

Based on the literature, object counting methods can be divided into two main categories, i.e. counting by regression and counting by detection. Counting by detection is required to train a visual detector, which can localise desired objects individually in images, such as the methods described in the previous section 2.4. In computer vision, state-of-the-art methods for crowd counting are primarily based on detection or regression. Rather than detecting desired objects one by one, counting by regression can estimate the number/density in global or local regions. Crowd counting by regression avoids explicit object detection or clustering [82, 83]. Several recent studies have introduced global or local crowd counting methods based on regression.

For cell counting in microscopy images, recent studies have used a supervised learning process with regression models to estimate the number of cells [84, 14, 85–87]. In [87], they proposed counting-by-regression framework with maximum excess over subarrays (MESA) distance to accurately estimate the count in still images. Christoph et al. [14] introduced an open-source software for density estimation in biomedical images, and Xie et al. [85] applied a deep neural regression network to cell density estimation in fluorescence microscopy images. In [84], the authors combined an interactive method for car counting in satellite images. To avoid solving the hard detection problem, counting by regression methods attempt to capture the

relationship between features and the number of cells in the image. Typical regression methods used for counting objects include support vector regression, Gaussian process regression [82] (GPR), RVM regression [88], random forest regression [14] and ridge regression [84].

For example, [82] presented a privacy-preserving system to count the number of inhomogeneous crowds travelling in different directions without using individual object segmentation, detection or tracking. The video is first segmented into two groups of foreground crowd region groups using a mixture of dynamic textures based on the moving direction. For each crowd segment group, they extracted different local features and then a perspective map was applied to weight each image location based on the real scene. Finally, Gaussian process regression was used to estimate the number of people in each region. They combined the linear and squared-exponential (RBF) kernels to capture general linearity and local nonlinearity trends simultaneously.

[83] presented a counting-by-regression framework for a crowd in videos using a single regression model. This model can estimate the number of people in different localised regions without training many regressors. The proposed model can automatically learn the relationship between multi-dimensional structured outputs and the extracted local features. A multi-output regression model using multi-variant ridge regression was trained to estimate the number of people in each local region.

In the following, some relevant regression models are reviewed, including Poisson Regression and RVM Regression.

2.5.1 Poisson Regression

A random variable Y (observed count) is assumed to follow a Poisson distribution [89] with parameter λ, and the log-mean parameter is a linear function of the input vector $x \in \mathbb{R}^d$.

$$f(x) = x^T \beta, \;\; \lambda = e^{f(x)}, \;\; y \sim Poisson(\lambda(x)) \tag{2.8}$$

For $y = 0, 1, 2, ...$, the probability is as follows:

$$Pr\{y|x, \beta\} = \frac{e^{-\lambda} \lambda^y}{y!}. \tag{2.9}$$

The Poisson distribution is discrete, similar to binomial distribution, but has only a single parameter λ. The mean, variance and mode of this distribution are as follows.

$$E(Y) = var(Y) = \lambda, \;\; mode(y) = \lfloor \lambda(x) \rfloor \tag{2.10}$$

Given a matrix of input vectors $X = [x_1 ... x_N]$ and output vectors $Y = [y_1 ... y_N]^T$, the likelihood function in terms of β is as follows:

$$L(\beta|X, Y) = \prod_{i=1}^{N} \frac{e^{-\lambda_i} \lambda_i^{y_i}}{y_i!} \tag{2.11}$$

The weight vector β can be obtained by maximizing the log-likelihood as follows:

$$\ln \ell(\beta|X, Y) = \log p(y|X, \beta) = \sum_{i}^{N} (y_i \log \lambda_i - \lambda_i). \tag{2.12}$$

To find a maximum, the equation $\frac{\partial \ell(y|X,\beta)}{\partial \theta} = 0$ has no closed form solution; however, we can solve the negative log-likelihood $-\ell(\theta|X, Y)$. This can be solved using convex optimization methods (e.g. gradient decent).

2.5.2 Relevance Vector Machine Regression

In 2001, Tipping introduced the RVM, which is a general Bayesian framework for obtaining sparse solutions for regression or classification [88]. There are many

advantages to use an RVM for regression, including probabilistic prediction, which dramatically reduces computational complexity with sparse kernel representation. In this section, we briefly review sparse Bayesian learning for a regression model:

For a regression problem, given a set of example input-target pairs $\{\mathbf{x}_n, t_n\}_{n=1}^N$, the standard regression model with additive noise between input $\mathbf{x}$ and scalar target t can be described as follows:

$$t_n = y(\mathbf{x}_n) + \varepsilon_n, \tag{2.13}$$

where ε_n is additional noise, which is assumed to be zero-mean Gaussian with variance σ^2. Then, we obtain the following

$$p(t_n, \mathbf{x}) = \mathcal{N}(t_n | y(\mathbf{x}_n), \sigma^2). \tag{2.14}$$

The function $y(\mathbf{x}_n)$ is defined over the input space:

$$y(\mathbf{x}_n; w) = \sum_{i=1}^{M} w_i \phi_i(\mathbf{x}_n) = \mathbf{w}^T \phi(\mathbf{x}_n), \tag{2.15}$$

where $\phi(\mathbf{x}_n) = (\phi_1(\mathbf{x}_n), ..., \phi_M(\mathbf{x}_n))^T$, which is the non-linear and fixed basis function. If there is a bias, then $M = N + 1$ and $\phi_1(x_n) = 1$. The objective of this function is to estimate the parameter (weight) $\mathbf{w} = (w_0, w_2, ..., w_N)^T$. Then, the likelihood function is given by:

$$p(t | \mathbf{X}, \mathbf{w}, \beta) = \prod_{n=1}^{N} p(t_n | \mathbf{x}_n, \mathbf{w}, \beta^{-1}), \tag{2.16}$$

where $\beta = \sigma^{-2}$ is the noise precision. The RVM in [88] introduced a set of hyperparameters $\alpha = (\alpha_1, ..., \alpha_M)^T$, and the weight prior takes the following form:

$$p(\mathbf{w} | \alpha) = \prod_{i=1}^{M} \mathcal{N}(w_i | 0, \alpha_i^{-1}). \tag{2.17}$$

Based on the Bayes' rule, the posteriori distribution over the weights is given by:

$$\begin{aligned} p(\mathbf{w}|\mathbf{t},\mathbf{w},\beta) &= \frac{p(\mathbf{t}|\mathbf{w},\beta)p(\mathbf{w}|\alpha)}{p(\mathbf{t}|\alpha,\beta)} \\ &= (2\pi)^{\frac{N+1}{2}}|\Sigma|^{-\frac{1}{2}}e^{-\frac{1}{2}(\mathbf{w}-\mu)^T\Sigma^{-1}(\mathbf{w}-\mu)}, \end{aligned} \tag{2.18}$$

where the covariance and mean are respectively:

$$\Sigma = (A+\beta\Phi^T\Phi)^{-1} \tag{2.19}$$

$$\mu = \beta\Sigma\Phi^T\mathbf{t} \tag{2.20}$$

Here, Φ is the design matrix with $N\times M$ elements, $\Phi_{ni}=\phi_i(\mathbf{x}_n)$ and $A=diag(\alpha_i)$. To obtain the best values for α and β, the type-2 maximum likelihood is used with a marginal likelihood function:

$$p(\mathbf{t}|\mathbf{X},\alpha,\beta) = \int p(\mathbf{t}|\mathbf{X},\mathbf{w},\beta)p(\mathbf{w}|\alpha)\mathrm{d}\mathbf{w}. \tag{2.21}$$

Then, we can obtain the following:

$$\begin{aligned} \mathrm{In}p(\mathbf{t}|\mathbf{X},\alpha,\beta) &= \mathrm{In}\mathcal{N}(\mathbf{t}|0,\mathbf{C}) \\ &= -\frac{1}{2}\{N\mathrm{In}(2\pi)+\mathrm{In}|\mathbf{C}|+\mathbf{t}^T\mathbf{C}^{-1}\mathbf{t}\}, \end{aligned} \tag{2.22}$$

where

$$\mathbf{C} = \beta^{-1}\mathbf{I}+\Phi\mathbf{A}^{-1}\Phi^T. \tag{2.23}$$

To maximize equation (2.22) with respect to α and β, we can obtain re-estimated equations by setting the derivatives of the marginal likelihood to zero:

$$\alpha_i^{new} = \frac{\gamma_i}{m_i^2} \tag{2.24}$$

$$(\beta^{new})^{-1} = \frac{\|\mathbf{t} - \Phi\mathbf{m}\|^2}{N - \Sigma_i \gamma_i} \tag{2.25}$$

where $\gamma_i = 1 - \alpha_i \Sigma_{ii}$ in which Σ_{ii} is the i^{th} diagonal component of Σ in equation (2.19), and m_i is the i^{th} element of $\mathbf{m}$ in equation (2.20). With the optimized hyper-parameters α^* and β^*, the predictive distribution given a new input $\mathbf{x}$ will be:

$$\begin{aligned} p(t|\mathbf{x}, \mathbf{X}, \mathbf{t}, \alpha^*, \beta^*) &= \int p(t|\mathbf{x}, \mathbf{w}, \beta^*) p(\mathbf{w}|\mathbf{X}, \mathbf{t}, \alpha, \beta) \mathrm{d}\mathbf{w} \\ &= \mathcal{N}(t|\mathbf{m}^T \phi(x), \sigma^2(x)). \end{aligned} \tag{2.26}$$

The variance is given as follows:

$$\sigma^2(x) = (\beta^*)^{-1} + \phi(\mathbf{x})^T \Sigma \phi(\mathbf{x}), \tag{2.27}$$

where Σ is given by equation (2.19) with α and β equal to their optimized values α^* and β^*.

2.6 Conclusion

In this chapter, the literature review in different aspects is presented. For high-throughput zebrafish image analysis and deep learning for biomedical imaging, background and previous works are introduced. Recent related works are reviewed and evaluated based on the problems described in Chapter 1, including deconvolution, detection and counting by regression methods. By reviewing the existing knowledge in computer vision, machine learning and medical image analysis areas, flaws and gaps are identified for solving our specific problems. By improving the existing

methods, we can develop our own methods and contributions in this thesis. For examples, in the problem of 3D deconvolution in fluorescence microscopy, the idea of using the MAP method with novel regularization methods can improve the problems using traditional regularization methods, such as TV and TM regularization. By borrowing methods from computer vision, deep learning methods can improve the performance of the detection methods dramatically. Furthermore, inspired by the existing crowd counting methods in computer vision, interactive methods can be developed to solve our cell counting problem. In the flowing chapters, several novel methods and applications will be introduced based on the existing knowledge and the previous methods in the literature review.

Chapter 3

Three-dimensional Deconvolution for Light Microscopy Images

As a perfect experimental model organism, zebrafish has been frequently used in a variety of biomedical research studies. It is often convenient and efficient to adopt the WF fluorescence microscopy for recording, observing, and analysing labelled transparent features in zebrafish images; however, the large stack of acquired images at multiple focus depths (or z-stack) is intrinsically blurred upon acquisition, thus making proper deconvolution critical for further quantitative analysis.

3D deconvolution is an ill-posed inverse problem; therefore, additional priors are required to regularise it. In this chapter, we propose a Bayesian MAP method with sparse images prior to solve the 3D deconvolution problem in WF fluorescence microscopy. Our proposed sparse image priors include two parts, a global hyper-Laplacian prior and a local background region mask. These two priors are designed to preserve sharp edges and suppress ringing artefacts, respectively. Two situations, involving deconvolution with known and unknown PSFs, are considered in this chapter. Furthermore, both synthetic and real WF fluorescent microscopy images are used for evaluation. Our experimental results demonstrate the potential applicability of

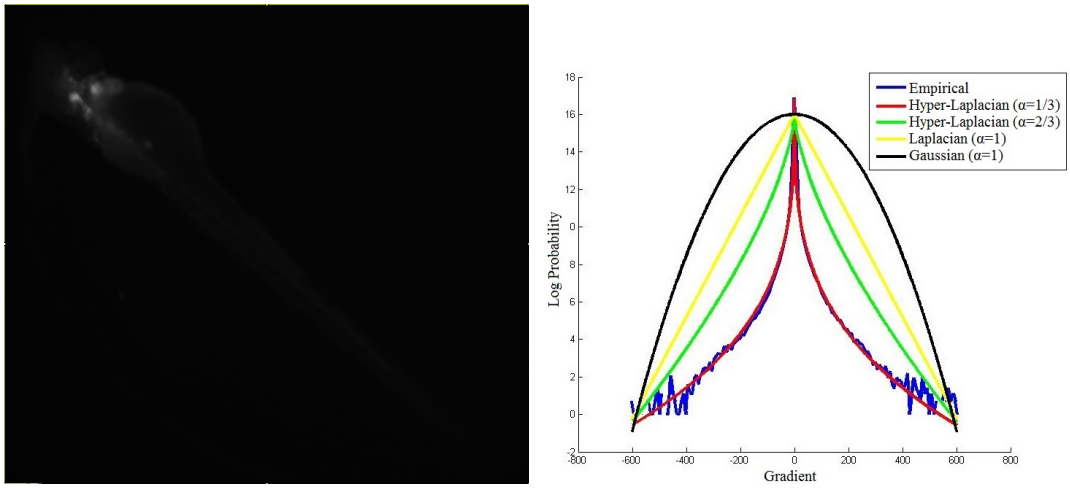
our proposed method for 3D fluorescence microscopy and its performance, compared to state-of-the-art 3D deconvolution algorithms and software.

3.1 Introduction

Light microscopy has been fundamental in advancing our understanding of molecular and cellular biology in health and disease research. In particular, WF fluorescence microscopy is a proven approach which is widely used for observing, analysing and imaging specific labelled features of small living specimens. Compared to confocal microscopy, WF microscopy is much faster and more convenient; further, WF microscopy can also record information regarding the entire specimen and achieve high throughput imaging, thus enabling large-scale image analysis.

During light illumination, however, both focus and out-of-focus light is recorded by a CCD camera, thereby caussing the original fluorescent information in the living specimen to either be lost or visualised with reduced contrast due to the low-resolution out-of-focus light. According to Shaw [90], there are two approaches to removing this out-of-focus light. One approach is to use confocal microscopy for biological imaging to eliminate much of the out-of-focus light from detection. The other approach is to record a stack of images at a series of focal planes via a WF microscope. Then, deconvolution techniques are used to reduce the effect of the out-of-focus light for further research, such as feature detection, cell tracking, and cell counting.

According to Young [91], the imaging process using WF or confocal fluorescence microscopy is assumed to be linear and shift invariant. The entire imaging process can be mathematically explained as a 3D convolution process [90]. The inverse procedure, *i.e.* removing out-of-focus light from z-stack images, is known as 3D deconvolution. The convolution process can be formulated as follows:



(a) One frame of the stack (b) log-gradient distribution

Fig. 3.1 (a) is one frame of a zebrafish embryo z-stack. (b): The blue curve describes the gradient distribution ($\bigtriangledown_x f$) of the image. The Hyper-Laplacian with $\alpha = 1/3$ is a more appropriate model to describe the log-gradient distribution (blue), compared with the Gaussian (black), Laplacian (yellow) or Hyper-Laplacian with $\alpha = 2/3$ (green) log-distribution.

$$g(\mathbf{x}) = n \sum_{\mathbf{k} \in \Omega} h(\mathbf{k}) f(\mathbf{x} - \mathbf{k}), \tag{3.1}$$

where

g : observed, blurred image,

$h : \mathbb{R}^3 \to \mathbb{R}$–point spread function of the imaging system,

$f : \mathbb{R}^3 \to \mathbb{R}$–latent image or ground truth image function,

n : voxelwise noise function,

Ω : $\Omega \subset \mathbb{R}^3$ is the support in the specimen domain recorded by the imaging system,

$\mathbf{x}$: 3-tuple of discrete spatial coordinates.

equation (3.1) can be written in a short form:

$$g = n(h * f), \tag{3.2}$$

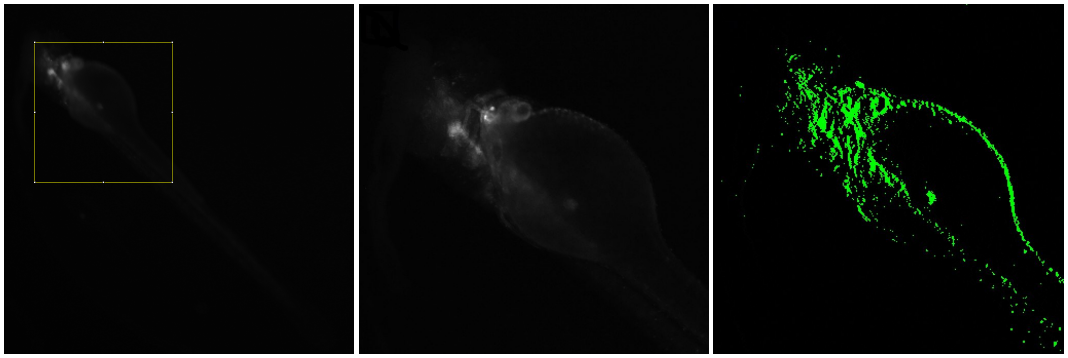
where $*$ is the 3D convolution operator. Convolution kernel h is the PSF, which describes the blurring shape of one point of light through the light microscope [90].

Deconvolution is the reverse problem in which we cannot exactly obtain original image h and PSF f only from observation g. The common method for 2D image deblurring is to use prior statistical knowledge of the given 2D natural images to preserve edges; such statistical knowledge includes their sparse derivative distribution [34, 40, 42, 45, 54], as the image gradient contains edge information. In 3D z-stack images, we can also calculate the derivative distribution. In Fig. 3.1, the blue curve represents the log-gradient distribution of a z-stack image. From the figure, we observe that most values in the blue curve are zero, so it is called a sparse derivative distribution and can be used as a sparse image prior to obtain satisfactory result. In this chapter, we apply this kind of sparse prior to 3D deconvolution via a Bayesian MAP framework.

Several parameters of the imaging system define the shape and intensity of the PSF. Two scenarios are considered in realistic cases. In the first scenario, all PSF parameters related to the imaging system are recorded during imaging. After estimating the PSF using the existing model, the rest of the process resembles a non-blind deconvolution problem. In the second scenario, the parameters are unknown; therefore, the PSF and latent image are estimated simultaneously via an iteration process which therefore translated to a blind deconvolution problem. Different models are used for estimating the PSF in these two scenarios.

During the imaging process, the fluorescent signal is influenced by photon noise. Compared to additive noise, photon noise largely depends on the original signal and is more difficult to be removed [51]. The process can be described as a Poisson distribution [4], *i.e.*

$$p(k) = \frac{\lambda^k e^{-\lambda}}{k!}, k = 0, 1, 2, ... \tag{3.3}$$



(a) Original frame. (b) Zoom-in of the yellow box in (a). (c) The green textue region and background regions Ψ.

Fig. 3.2 (a): one frame of an original zebrafish z-stack; (b): zoom-in of the yellow box in (a); (c): after applying the triangle threshold to image g_x; here, the green colour indicates texture regions of the image, while remaining background region is Ψ.

3.2 Non-blind Deconvolution Framework

In this section, a MAP algorithm with sparse priors is presented for estimating best latent image f given blurred image g and PSF h. According to the Bayes Rule, *i.e.*

$$p(f|g,h) = \frac{p(g|f,h)\,p(f)}{p(g)} \propto p(g|f,h)\,p(f), \tag{3.4}$$

where $p(f)$ is a prior describing f, $p(g|f,h)$ can be approximated as a Poisson distribution [93]. Here every point in the blurred image follows an independent Poisson process, and equation (3.3) can be rewritten as:

$$p(g(x)|f,h) = \frac{(f*h)(x)^{g(x)} e^{-(f*h)(x)}}{g(x)!}. \tag{3.5}$$

The sparse image priors will be discussed in the following subsections.

3.2.1 Sparse Image Priors

In this chapter, to simultaneously address the ill-posed nature of the problem and reduce ringing artifacts when restoring latent image f, prior $p(f)$ is decomposed

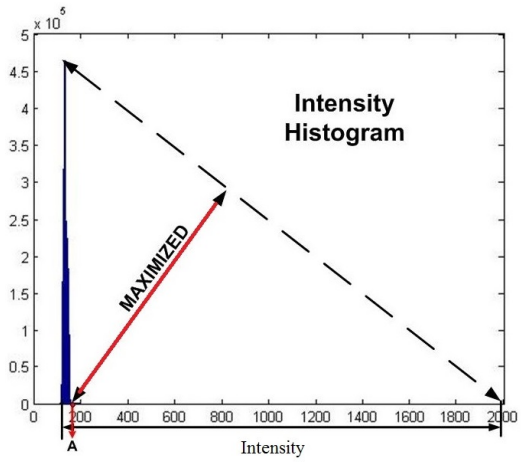


Fig. 3.3 An illustration of the triangle threshold algorithm [92] in which point A is selected by maximising the distance between the intensity distribution and the dotted line drawn between the highest point in the figure and the largest value on the horizontal axis; next, a fixed offset is added for locating the precise threshold point.

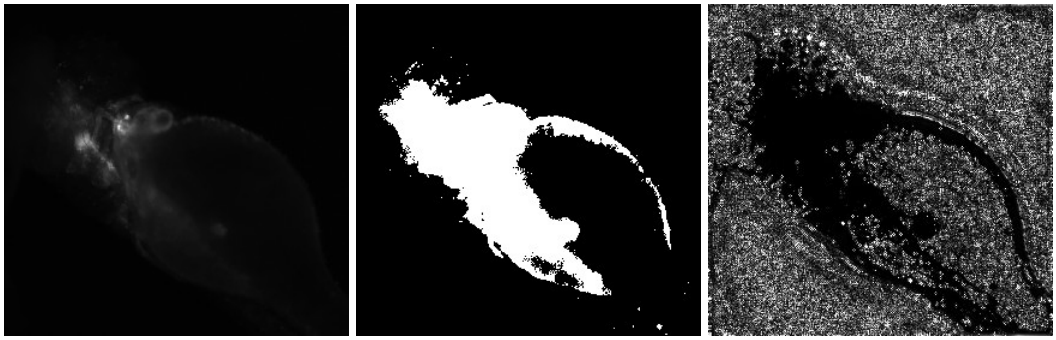
into two components, including a global prior $p_g(f)$ [43] and a local prior $p_l(f)$ [42]:

$$p(f) = p_g(f)\, p_l(f)\,. \tag{3.6}$$

Global Prior $p_g(f)$

Recent research focused on 2D natural image deconvolution [34, 40–43, 49, 46], applies log-gradient distributions (see Fig. 3.1) as a prior of the latent image is applied for preserving edges. There are several different kinds of expressions to approximate log-gradient distributions, including Gaussian and Laplacian distributions. In this chapter, we apply the hyper-Laplacian to the 3D deconvolution problem for WF fluorescent images.

The heavy-tailed distribution of the gradient in the WF fluorescence microscopy z-stack is well explained by the hyper-Laplacian distribution [43]. In Fig. 3.1, we observe that the best approximation for the logarithmic image gradient distribution is



(a) Original frame (b) Texture area (c) Ringing map

Fig. 3.4 Applying the triangle threshold algorithm to a WF fluorescent zebrafish image: (a) original frame; (b) binary mask produced by the triangle threshold algorithm. The white area indicates the texture region and (c) the ringing map is visualised in intensity range [0, 50] by computing the difference between the result using the local prior and the result without using the local prior.

the hyper-Laplacian distribution with $\alpha = 1/3$. Here, global prior $p_g(f)$ is defined as follows:

$$p_g(f) \propto \prod_{i \in \Omega \subset \mathbb{R}^3} e^{-\tau |\bigtriangledown f_i|^\alpha}, (0 < \alpha < 1), \tag{3.7}$$

where τ is the positive rate parameter and $\bigtriangledown$ is the 3D gradient filter.

Local Prior $p_l(f)$

Aside from the global prior, a local prior with a mask is used to reduce ringing artefacts, as first introduced in motion deblurring [42]. In this chapter, we present our design for a similar local prior for 3D deconvolution; the difference here is that we use a triangle threshold method [92] to automatically determine background regions in the z-stack images. The majority part of the fluorescence microscopy images is dark background region, and the triangle threshold algorithm works well for this kind of image.

From [42], we conclude that the background region in the image should also be background after restoration, and this local prior can effectively suppress ringing

artefacts in both the background regions and the texture regions. Due to an inaccurately estimated PSF, some ringing artefacts appear near the boundary and edges in the frequency domain after the restoration. Similar to [42], we define the error of the gradient between the estimated image and the blurred image in the background area to follow a normal distribution with zero mean, *i.e.*

$$p_l(f) = \prod_{i \in \Phi} N(\nabla f_i - \nabla g_i | 0, \sigma), \tag{3.8}$$

where Φ denotes the background regions in each frame of the z-stack images; note that background regions Φ are shown in Fig. 3.2. Even though this local prior is only defined in the background regions, it can globally reduce ringing artefacts due to the effect of the PSF during restoration [42].

Mask Function M

We define mask function, M, to describe the background regions, Ψ, in the z-stack images. Here, we apply the triangle threshold algorithm [92] to automatically determine the threshold with Ψ indicating the regions where $M(x) = 1$, *i.e.*

$$M(x) = \begin{cases} 0, & g(x) \geq \text{Threshold} \\ 1, & else. \end{cases} \tag{3.9}$$

From the intensity histogram shown in Fig. 3.3, we observe that most of the intensity values are less than 200, indicating that most of the points are in the background region. To segment the texture region from the background, threshold point A should be precisely selected. We use the triangle threshold algorithm [92], as illustrated in Fig. 3.3. This algorithm can cleanly segment the texture region from the WF fluorescent zebrafish image. In Fig. 3.4(c), the difference between the deconvolution result after 50 iterations using a local prior is obvious, thus indicating that ringing artefacts and noise can be reduced using the local prior.

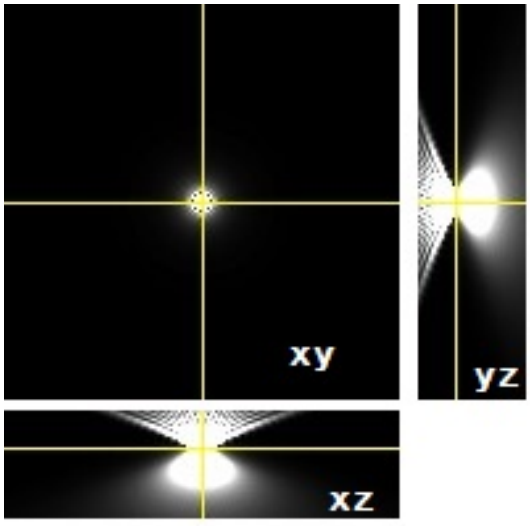


Fig. 3.5 Orthogonal views of the generated PSF.

3.2.2 Non-blind Three-dimensional Deconvolution

After defining the two priors, equation (3.4) becomes

$$p(f|g,h) \propto p(g|f,h)\, p_g(f)\, p_l(f)\,. \tag{3.10}$$

Instead of maximising the objective function shown in equation (3.10), negative log-likelihood function, $J(f)$, is minimised as

$$\begin{aligned} J(f) &= -\log\left(p(f|g,h)\right) \\ &\propto -\log\left(p(g|f,h)\right) - \log\left(p(f)\right) \\ &= -\log\left(p(g|f,h)\right) - \log\left(p_g(f)\right) - \log\left(p_l(f)\right) \\ &= J_{MLEM}(f) + \int_x \lambda_g |\nabla f|^{\alpha} dx \\ &\quad + \int_{\Phi} \lambda_l \|\nabla f(x) - \nabla g(x)\|^2 dx, \end{aligned} \tag{3.11}$$

where $J_{MLEM}(f)$ is redefined as follows:

$$\begin{aligned} J_{MLEM}(f) &= \int_x (f*h)(x) - g(x)\log\left(f*h\right)(x) \\ &\quad + \log\left(g(x)!\right)dx \\ &\propto \int_x (f*h)(x) - g(x)\log\left(f*h\right)(x)dx. \end{aligned} \tag{3.12}$$

The derivative of J with respect to f is calculated and set to zero. Next, we obtain a regularised version of the MLEM updating scheme, *i.e.*

$$\begin{aligned} &f_{k+1}(x) = \\ &\frac{f_k(x)}{1-\lambda_g\Psi_g - \lambda_l\Psi_l} \cdot [h(-x) * \frac{g(x)}{(f_k * h)(x)}], \end{aligned} \tag{3.13}$$

where $\Psi_g = div(\frac{\bigtriangledown f_k}{|\bigtriangledown f_k|^{2-\alpha}})$ and $\Psi_l = div((\bigtriangledown f_k - \bigtriangledown g)M(x))$ are the regularisation terms, λ_g, λ_l are regularisation parameters, and div is the divergence, *i.e.* $div(f) = \bigtriangledown \cdot f$.

To determine a stop criterion, we calculate the root mean squared error (RMSE) [50, 51] at each iteration between the updated latent image and ground truth. However, in real-world situation, there is no ground truth image, so we calculate the variance changes of the results between two iterations. Here, repeated iterations will be stopped when changes to the RMSE value are smaller than constant value ε. More specifically, we have

$$RMSE = \left(\frac{1}{N}\sum_{i=1}^{N}\left(f(i) - c\cdot\hat{f(i)}\right)\right)^{\frac{1}{2}}, \tag{3.14}$$

where N is the number of voxels, $\hat{f(i)}$ is the estimated latent image and $c = \frac{\sum_{i=1}^{N} f(i)}{\sum_{i=1}^{N}\hat{f(i)}}$. The non-blind 3D deconvolution procedure is summarised in Algorithm 1. The iteration process is implemented in the frequency domain via a fast Fourier transform (FFT) which greatly improves computational speed.

Algorithm 1 Non-blind deconvolution algorithm.

Input: 3D WF stack g, PSF h obtained using the VRIGL model
Initialize: $k = 0, f_0 = g$ $\varepsilon = 0.1$
while $|RMSE_{k+1} - RMSE_k| > \varepsilon$ **do**
Update latent image f
$f_{k+1}(x) = f_k(x)[h(-x) * \frac{g(x)}{(f_k * h)(x)}]$
$M(x) = \begin{cases} 0, & g(x) \geq T \\ 1, & else. \end{cases}$
T = Threshold using triangle threshold method
$\Psi_g = div(\frac{\bigtriangledown f_k}{|\bigtriangledown f_k|^{2-\alpha}})$
$\Psi_l = div((\bigtriangledown f_k - \bigtriangledown g)M(x))$
$f_{k+1}(x) = \frac{f_{k+1}(x)}{1-\lambda_g \Psi_g - \lambda_l \Psi_l}$
$k = k+1$
end while
Output: f

3.2.3 PSF Modelling with Microscopy Parameters

In the updating scheme of equation (3.13), the PSF (h) is known and produced by the variance refractive index Gibson and Lanni (VRIGL) model [94], which is a modified version of the well-known Gibson and Lanni PSF model [95]. Compared with 2D deblurring, solving the 3D deconvolution problem is more complex and time-consuming. To obtain better results, the precise PSF can be used when available.

There are three key approaches for estimating the PSF, namely experimental, analytical, and computational methods [96]. Using an experimental method to obtain the PSF usually results in a rather poor signal-to-noise ratio (SNR) [35]. Using a computational method such as blind deconvolution, both the PSF and latent image are calculated based on a blurred image; therefore, the optimal result is substantially biased by the initial guess of the PSF.

Recently, several analytical models for approximating the PSF of WF fluorescence microscopy systems have been proposed [94, 97, 95, 98–100]. In [97], a simple 3D Gaussian function is used for estimating the PSF of the confocal microscopy system; however, the WF PSF is not easily approximated using only a

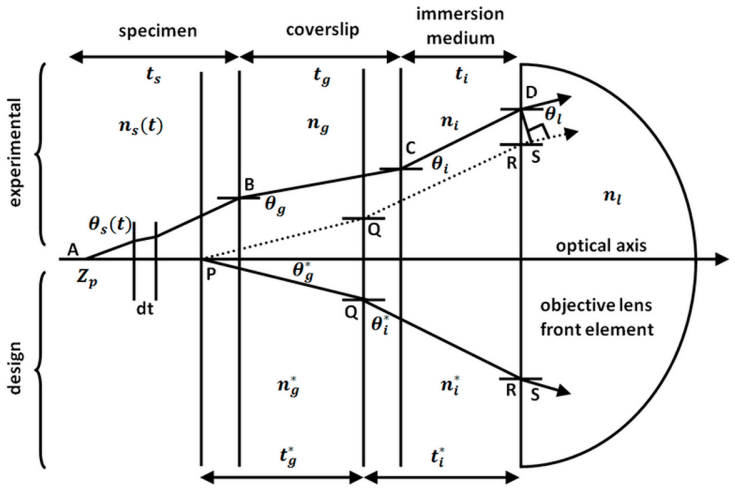


Fig. 3.6 Illustrating the underlying principles of the PSF model.

simple model, and all parameters can influence the shape and intensity of the PSF, a factor that should be considered in the approximation model. In [94], the VRIGL model is designed to estimate the PSF of the WF fluorescence microscopy system.

The WF PSF depends on several parameters of the imaging system itself, such as the numerical aperture (NA) of the microscope, the refractive index of the immersion medium (n_i), the wavelength of the light (λ), the spatial resolution (Δr), and the Axial resolution (Δz) [96].

In this chapter, we use the VRIGL model [94] for modelling the PSF as an analytical method. In Fig. 3.5, orthogonal views of the PSF generated using VRIGL is shown, and principles behind the VRIGL method are illustrated in Fig. 3.6.

The VRIGL model is described as follows:

$$h(\mathbf{x};\mathbf{x}_p,\tau) = \left| C \int_0^1 J_0\left(kr\mathrm{NA}\rho\right) e^{i\phi\left(\rho,z;z_p,\tau\right)} \rho d\rho \right|^2 \tag{3.15}$$

where

$$\begin{aligned}\phi(\rho,z;z_p,\tau) =& k[\int_0^{z_p} \sqrt{n_s^2(t) - \mathrm{NA}^2\rho^2}dt + t_i\sqrt{n_i^2 - \mathrm{NA}^2\rho^2} \\ & - t_i^*\sqrt{n_i^{*2} - \mathrm{NA}^2\rho^2} + t_g\sqrt{n_g^2 - \mathrm{NA}^2\rho^2} \\ & - t_g\sqrt{n_g^{*2} - \mathrm{NA}^2\rho^2}]\end{aligned}$$

where $\mathbf{x} = (x, y, z)$: The coordinate of a point in the image plane.

$\mathbf{x}_p = (x_p; y_p; z_p)$: The coordinate of the point source.

$\tau = (\text{NA}; n; t)$: The parameter set of the imaging system; NA: The numerical aperture; n: The refractive index; t: The thickness.

$r^2 = (x - x_p)^2 + (y - y_p)^2$: The point at which the ray intersects the exit pupil.

J_0: The Bessel function of the first kind of order zero.

$n_s; n_g; n_i; n_g^*; n_i^*; t_g; t_i; t_g^*; t_i^*$ can be found in Fig. 3.6.

The VRIGL model is designed to obtain shift varying PSF for a thick specimen. The VRIGL model improves the standard Gibson and Lanni PSF model by including refractive index variation along the depth of the specimen. The aberration caused by the variation of refractive index within the thick specimen is captured by this model. Shift varying nature of PSF in axial direction is also captured only by three predefined functions: linear Logarithmic, and Exponential. Compared with standard Gibson and Lanni PSF model, the VRIGL model is significantly better. The optimization procedure is very complex and it is beyond the scope of this thesis, so we only defined the final equation of this model.

3.3 Extension to Blind Deconvolution

3.3.1 Blind Deconvolution Framework

The goal of the blind deconvolution is to simutaneously estimate the latent image function f and point spread function h by maximising the objective function

$$p(f, h|g) = \frac{p(g|f, h)\, p(f)\, p(h)}{p(g)}. \tag{3.16}$$

Given this, the negative log-likelihood function is calculated as

$$\begin{aligned} J(f,h) &= -\log\left(p\left(f|g,h\right)\right) \\ &= J_{MLEM}(f,h) - \log(p(f)) - \log(p(h)), \end{aligned} \tag{3.17}$$

where $p(f)$ and $p(h)$ are the prior probability functions for latent image function f and PSF h. Further, the prior $p(f)$ is the same as the prior used in the non-blind deconvolution framework. In prior $p(h)$, we use TV regularisation to preserve edges of the PSF. By minimising equation (3.17), we present the blind deconvolution updating scheme as follows:

$$f_{k+1}(x) = \frac{f_k(x)}{1-\lambda_g\Psi_g - \lambda_l\Psi_l} \cdot [h_k(-x) * \frac{g(x)}{(f_k * h_k)(x)}], \tag{3.18}$$

$$h_{k+1}(x) = \frac{h_k(x)}{\int_\Omega f_k(y)dy - \lambda_h\Psi_h} \cdot [f_k(-x) * \frac{g(x)}{(f_k * h_k)(x)}], \tag{3.19}$$

where λ_h is the regularisation parameter and Ψ_h is the regularisation term for updating the PSF. In this chapter, we select different PSF regularisation terms from state-of-the-art methods to evaluate our proposed blind deconvolution method. In particular, we use three different PSF regularisation methods for comparison. The blind deconvolution method is illustrated in Algorithm 2.

3.3.2 PSF Modelling without Microscopy Parameters

The parameters of the PSF based on the imaging system are unknown to the blind deconvolution framework. However, the result depends on the initial guess of the PSF, so we use a simple diffraction-limited PSF Gaussian model [100] with far fewer free variables to initialise the PSF in the blind deconvolution framework. We then compare deconvolution results using three different PSFs for blind deconvolution experiment section. The generated PSF is symmetric around the central xy section.

We modify the Gaussian model to obtain

$$\begin{aligned} h(x,y,z) &= \frac{1}{\sigma(z-C_z)^2} e^{-\frac{(x-C_x)^2+(y-C_y)^2}{2\sigma(z-C_z)^2}}, \\ \sigma(z-C_z) &= \frac{|z-C_z|}{C_z} * (\sigma_2 - \sigma_1) + \sigma_1, \end{aligned} \tag{3.20}$$

where C_x, C_y and C_z are the coordinates of the central point of the PSF; Further, σ changes linearly with z between σ_1 and σ_2; here, σ_1 is the variance value in the central xy section, and σ_2 is the variance value in the first or last xy section. Given the above, we next need an approximate shape for the initial PSF; therefore, σ_1 and σ_2 are defined as one and ten, respectively. The values of those two parameters are selected by testing different values. The combination of 1 and 10 can produce the best result and make Gaussian model very close to the synthetic VRIGL model with microscopy parameters.

Algorithm 2 Blind deconvolution algorithm.

Input: 3D WF stack g
Initialize: counter $k = 0$, latent image $f_0 = g$, PSF initialized using Gaussian model ($\sigma_1 = 1$, $\sigma_2 = 10$).
while $|RMSE_{k+1} - RMSE_k| > \varepsilon$ **do**
  Update latent image f
  $f_{k+1}(x) = f_k(x)[h(-x) * \frac{g(x)}{(f_k * h)(x)}]$
  $M(x) = \begin{cases} 0, & g(x) \geq T \\ 1, & else. \end{cases}$
  T = Threshold using Triangle Threshold method
  $\Psi_g = div(\frac{\bigtriangledown f_k}{|\bigtriangledown f_k|^{2-\alpha}})$
  $\Psi_l = div((\bigtriangledown f_k - \bigtriangledown g)M(x))$
  $f_{k+1}(x) = \frac{f_{k+1}(x)}{1-\lambda_g \Psi_g - \lambda_l \Psi_l}$
  Update PSF h
  $h_{k+1}(x) = h_k(x) \cdot [f_k(-x) * \frac{g(x)}{(f_k * h_k)(x)}]$
  $\Psi_h = div(\frac{\bigtriangledown h_k}{|\bigtriangledown h_k|})$
  $h_{k+1} = \frac{h_{k+1}(x)}{\int_\Omega f_k(y)dy - \lambda_h \Psi_h}$
  $k = k + 1$
end while
Output: f

3.4 Experiments and Results of Non-Blind Deconvolution

To demonstrate the effectiveness of our proposed approach, we conducted experiments using four different datasets, including three synthetic datasets and one real dataset, the latter being the WF fluorescence microscopy zebrafish embryo dataset. We calculated a variety of performance metrics, including the RMSE [50, 51], peak signal-to-noise ratio (PSNR), and normalized mean integrated squared error (NMISE). The MAP with global hyper-Laplacian (HL) and the MAP with hyper-Laplacian and a local mask (HL+Mask) were tested and compared with TV and TM regularisation methods. Note that the deconvolution process stops when the change in the RMSE between two adjacent iterations is smaller than small constant value ε.

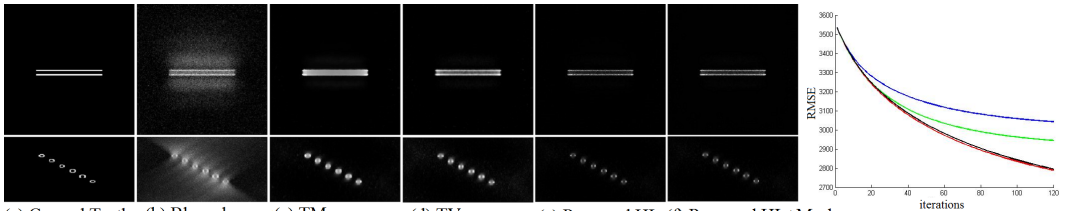
3.4.1 Synthetic Light Microscopy Data

Three groups of synthetic data are used for testing, including the Hollow Bar, the Hela Cell Nucleus dataset and data from the 3D deconvolution challenge which was part of the IEEE International Symposium on Biomedical Imaging 2014. For the TV and TM regularisation methods, the RMSE values converged after approximately 120 iterations.

Synthetic Hollow bar Dataset

The synthetic hollow bar data was obtained from the website [1]. The ground truth data in hollow bar dataset were blurred using a theoretical WF microscopic PSF, then corrupted by the introduction of both Gaussian noise and Poisson noise with a SNR of 15dB. Note that voxel volumes of this dataset are $256 \times 256 \times 128$. Parameters of the synthetic PSF include numerical aperture NA= 1.4, spherical aberration W =

[1] http://bigwww.epfl.ch/deconvolution/?p=bars



(a) Ground Truth (b) Blurred RMSE: 3602.2 (c) TM RMSE: 3042.4 (d) TV RMSE: 2944.2 (e) Proposed HL RMSE:2794.8 (f) Proposed HL+Mask RMSE: 2788.2 (g) RMSE

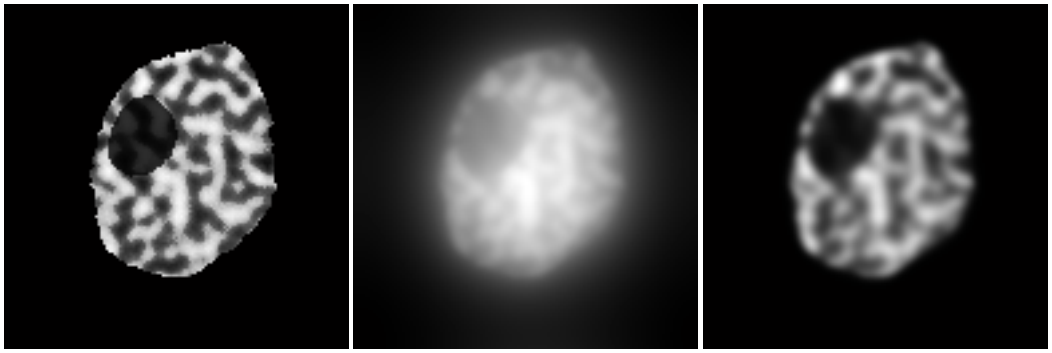
Fig. 3.7 Deconvolution results of four different methods with the top row indicating the central xy-section of the 3D image data and the bottom row showing the central xz-section; (a) the ground truth data. (b) the blurred images. (c) results with TM regularisation after 120 iterations. (d) results with TV regularisation. (e) results with the proposed HL. (f) results with HL+Mask. (g) the trend of the RMSE with blue indicating TM, green indicating TV, black indicating HL and red indicating HL+Mask).

0, wavelength λ = 500nm, spatial resolution Δr = 100nm and axial resolution Δz = 200nm. We achieved the best results for TM and TV regularisations and for our proposed methods using regularisation parameters $\lambda_{TM} = 3 * 10^{-7}$, $\lambda_{TV} = 0.001$, $\lambda_g = 0.05$, $\lambda_l = 10^{-8}$ and $\alpha = 1/3$.

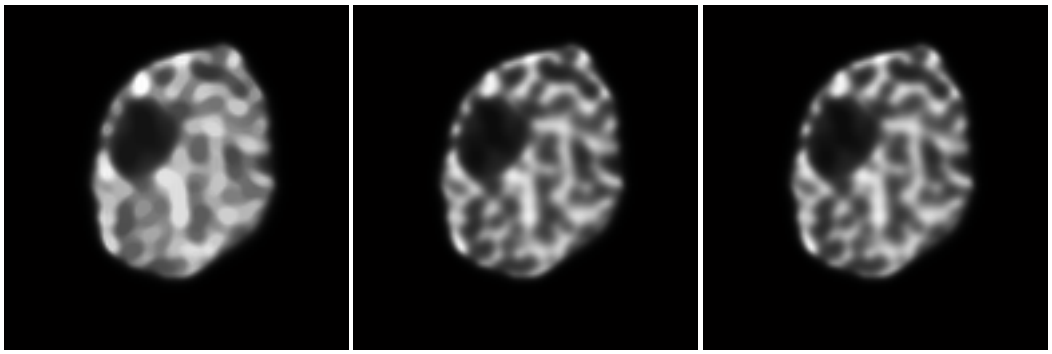
Results using the aforementioned methods after 120 iterations are shown in Fig. 3.7, illustrating that our two proposed methods produce improved results over those of TM and TV. Results using the TM regularisation become over-smooth after 100 iterations, and the regularisation term became increasingly through each iteration. Further, results of HL+Mask was a little sharper than that of HL. Fig. 3.7(g) shows the tendencies of the RMSE values using the four different methods. The TV and TM methods converged after 100 iterations, whereas the RMSE values of our two proposed methods converged after only 50 iterations, much more quickly than those of the other methods.

HeLa Cell Nucleus

The HeLa Cell Nucleus dataset was generated using an online simulation tool [101]. The data were corrupted by a generated WF PSF without aberration, then Poisson noise was added. The PSF was generated with numerical aperture NA=



(a) Ground Truth (b) Blurred RMSE:7885.9 (c) TM RMSE:3392.1



(d) TV RMSE:3062.4 (e) Proposed HL RMSE:2710.1 (f) Proposed HL+Mask RMSE:2689.0

Fig. 3.8 Results of the HeLa cell Nucleus dataset using four distinct methods.

1.4, wavelength λ = 500nm, spatial resolution Δr = 100nm and axial resolution Δz = 200nm. Voxel volumes of the dataset were $129 \times 129 \times 96$. For this dataset, we selected $\lambda_{TM} = 10^{-5}$, $\lambda_{TV} = 0.01$, $\lambda_g = 0.01$ with $\alpha = 1/3$, $\lambda_l = 10^{-7}$ with $\alpha = 1/3$ to test TM, TV, proposed HL and proposed HL+Mask methods, respectively. RMSE results covering 550 iterations are shown in Fig. 3.8 and are based on four different methods; from the figure, we observe that more out-of-focus light was reduced using our proposed methods. Further, the TV regularisation method was able to preserve image edges, but the result was an image that was over-smooth. Different methods converged at different iterations, but all methods converged before approximately 550 iterations. Therefore, we compare the results at 550 iterations. Fig. 3.9 illustrates that the RMSE results using our proposed methods were lower than those of the TV and TM methods. Table 3.1 shows scores for our proposed methods using two other

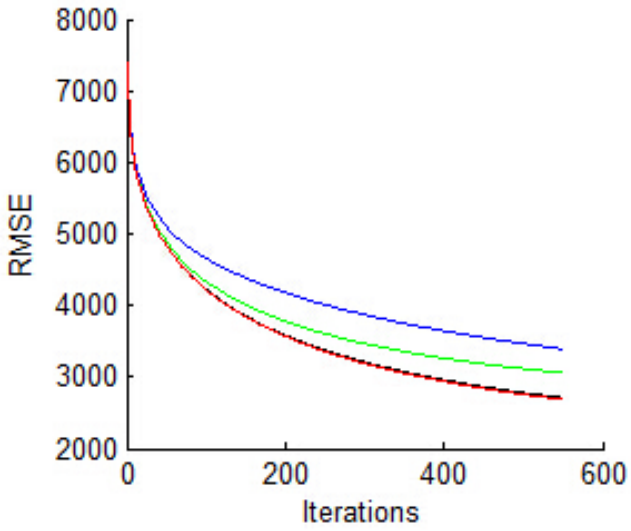


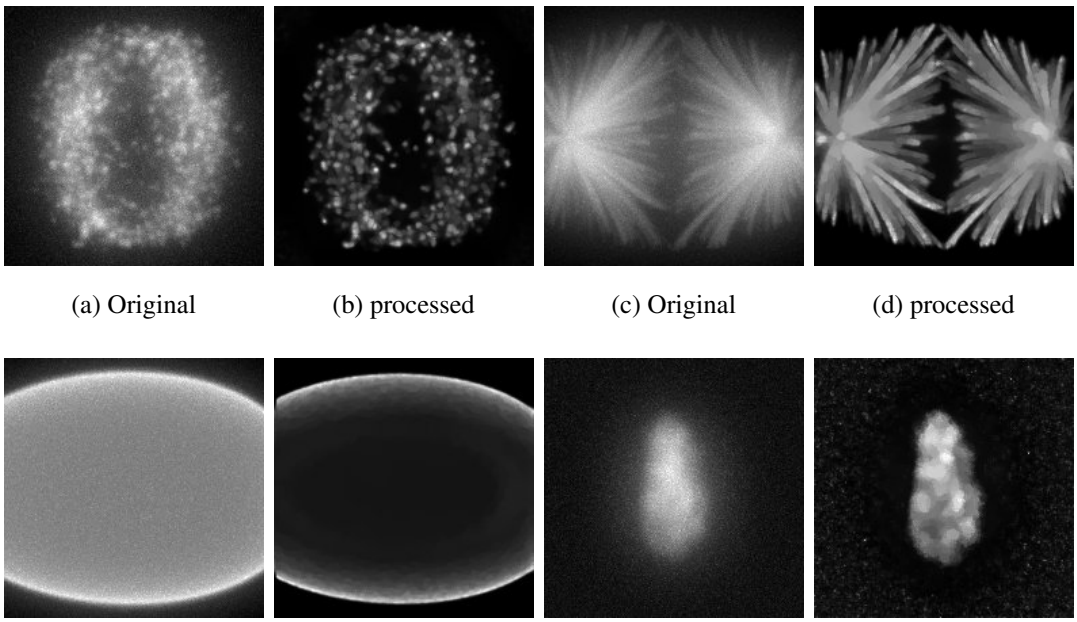
Fig. 3.9 RMSE results for the synthetic Hela Cell dataset with blue indicating TM, green indicating TV, black indicating our proposed HL method and red indicating our proposed HL+Mask method

performance metrics, *i.e.* PSNR and NMISE from the table, we observe that scores for our proposed method were also the best as compared with scores for the TV and TM regularisation methods.

3.4.2 Synthetic Data from 3D Deconvolution Challenge

This third dataset originates from the Second International Challenge on Three-Dimensional Deconvolution Microscopy competition and can be download from the corresponding website [2]. The data comprises four channels which correspond to different structures (*i.e.* point sources, filaments, membranes and dense objects). The competition did not publish corresponding ground truth data, as the organisers aimed to reproduce the conditions of a real deconvolution problem. Therefore, performance metrics cannot be calculated. For this dataset, we selected $\lambda_g = 0.01$ and $\lambda_l = 10^{-7}$ with $\alpha = 2/3$ for our testing. Results are shown in Fig. 3.10. And in Fig. 3.11, we present results of our testing the TM, TV and proposed HL+Mask methods on one of the z-stack images in the dataset.

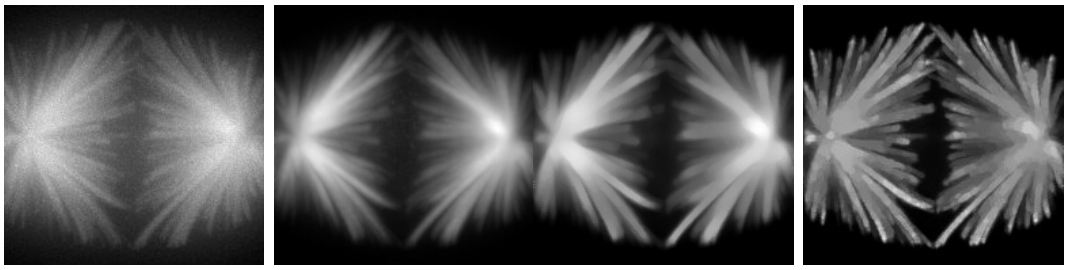
[2] http://bigwww.epfl.ch/deconvolution/challenge/index.html



(a) Original (b) processed (c) Original (d) processed

(e) Original (f) processed (g) Original (h) processed

Fig. 3.10 Results of testing our HL+Mask method on the dataset from the Second International Challenge on 3D Deconvolution Microscopy; maximum-intensity projections of the data (a), (c), (e), (f) and corresponding results are shown in parts (b), (d), (f), (h); note that all the results are displayed as the maximum intensity value projection along the vertical z direction.



(a) Original (b) TM (c) TV (d) HL+Mask

Fig. 3.11 Results of testing the TM, TV and HL+Mask methods on the dataset from the Second International Challenge on 3D Deconvolution Microscopy; all the results are displayed as the maximum intensity value projection along the vertical z direction.

3.4.3 Zebrafish Embryo Data

Zebrafish are a popular model organism for human disease research, including research involving neurological diseases, such as PD. Zebrafish are vertebrates with good genetic conservation in relation to humans. They are cheap to house compared with mammalian models, and due to their high fecundity externally developing and their transparent embryos, zebrafish are ideal for use in drug and toxin screens. Typically, image analysis of tissue samples from other species, such as mice or humans, is carried out on slides with clearly orientated tissue samples of well-defined, standardised thickness and would typically form the basis of subsequent image acquisition and analysis. Whole mount *in situ* staining, immunohistochemistry and confocal microscopy of zebrafish embryos result in the challenge of having to undertake image analysis in tissue samples of varying dimensions and orientations.

We collected fluorescent light microscopy images of three-day post fertilisation (3dpf) embryos that have fluorescently labelled monoaminergic neurones. The WF fluorescent zebrafish embryo dataset was recorded by the In Cell Analyzer 2000; further the microscopy type is fluorescence with numerical aperture NA $= 0.1$, wavelength $\lambda = 490nm$, spatial resolution $\Delta r = 3.7\mu$m, axial resolution $\Delta z = 20\mu$m, magnification =2.0, and focal length =100μm. The stack size was $2048\times2048\times41$. Parameters for this dataset were $\lambda_{TM} = 5\times10^{-5}$, $\lambda_{TV} = 0.01$, $\lambda_g = 0.05$ with $\alpha = 1/3$, and $\lambda_l = 10^{-6}$ with $\alpha = 1/3$.

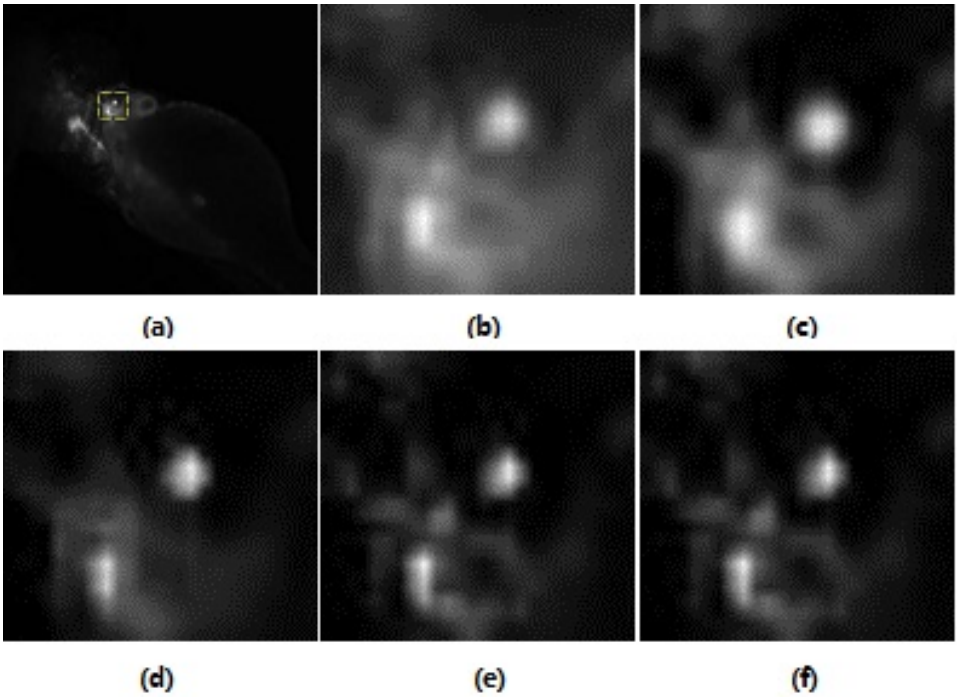


Fig. 3.12 To observe our results in detail, we magnified the results shown in Fig. 3.13; here, we show (a) the central xy-section with a yellow square; (b) a zoom-in of the yellow square region; (c) TM; (d) TV. (e) our Proposed HL and (f) our proposed HL+Mask methods.

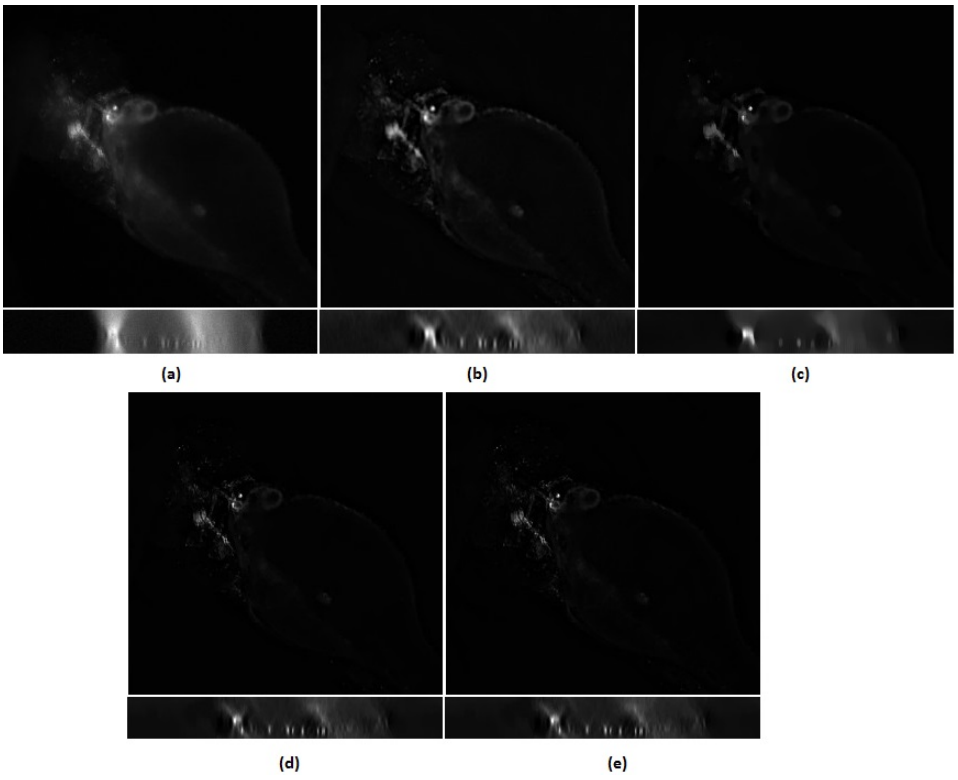
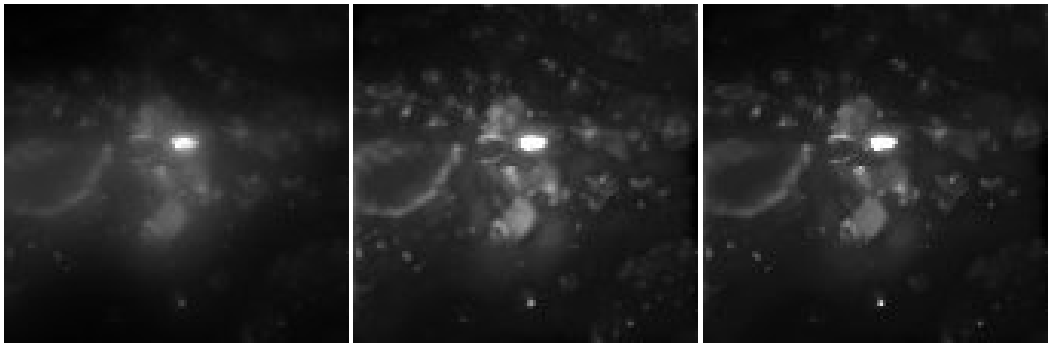
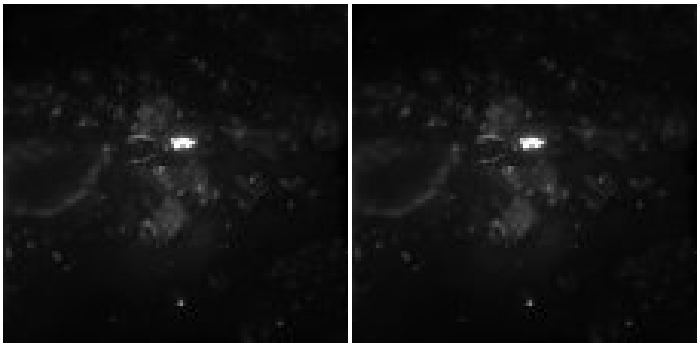


Fig. 3.13 The zebrafish data results using four distinct methods. The central xy-section and central xz-section of the z-stack images are shown in each block; (a) central frame of the recorded images; (b) TM; (c) TV; (d) our proposed HL method; (e) our proposed HL+Mask method.

Given that there were no corresponding ground truth data for the recorded zebrafish images, we did not calculate quantitative scores using performance metrics of the final results. Our qualitative results are shown in Fig. 3.12, Figs. 3.13 and 3.14. In Fig. 3.12, we zoom in on a specific region of the results to identify more details. Compared with our other results, we note that clearer information is shown in the texture regions of the results processed by our HL and HL+Mask methods.



(a) Blurred (b) TM (c) TV



(d) HL (e) HL+Mask

Fig. 3.14 Zebrafish dataset results displayed as the maximum value projection along the z direction the given four distinct methods; here we show (a) central frame of the recorded images; (b) TM; (c) TV; (d) our proposed HL method and (e) our proposed HL+Mask method.

3.5 Experiments and Results of Blind Deconvolution

We used three datasets to test our blind deconvolution methods; these dataset were the Hollow Bar dataset used in our non-blind deconvolution experiments, a new

Table 3.1 Comparison of results with different performance metrics

HeLa	Blurred	TM	TV	HL	HL+Mask
RMSE	5907.4	4153.0	4081.7	4059.9	**4052.0**
PSNR	19.149	21.550	21.667	21.751	**21.752**
NMISE	8439.0	4966.7	4926.7	4878.7	**4875.8**
Bar	Blurred	TM	TV	HL	HL+Mask
RMSE	5907.4	4153.0	4081.7	4059.9	**4052.0**
PSNR	25.4601	26.9703	27.4010	**27.8959**	**27.8921**
NMISE	3062.2	3042.4	2944.2	2794.8	**2788.2**

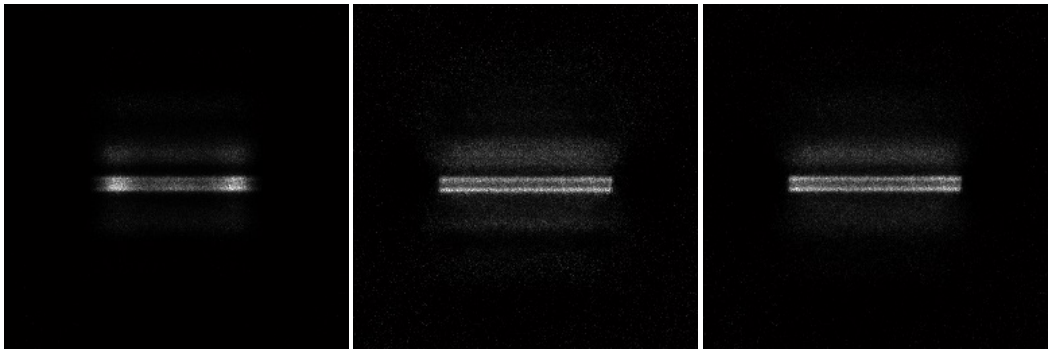
HeLa Cell dataset and the zebrafish embryo dataset. We assume that the parameters related to the PSF and imaging system of the datasets are all unknown.

3.5.1 Synthetic Light Microscopy Data

To evaluate our proposed regularisation terms, *i.e.* HL and HL+mask methods, in the blind deconvolution, we used four regularisation methods (*i.e.* TM, TV, HL and HL+Mask) to constrain the latent image, and three different PSF regularisation terms (*i.e.* KFTV without OTF mask [50], TM, TV) to update the PSF during each iteration. In total, we tested twelve combinations of these regularisation methods. Further, we used two different synthetic datasets to evaluate the blind deconvolution methods. As note above, these datasets were the Hollow Bar dataset and the HeLa cell dataset. When evaluating the RMSE for the different combinations of the regularisation terms, the iterations will not stop until the RMSE value converges. In our experiments, we chose the best results using different combinations to compare the RMSE values.

Hollow Bar

The resulting computational speed largely depends on the size of the dataset. Therefore, we resized the original synthetic dataset to improve computational speed, noting that this has no influence on selecting the best method with the smallest RMSE value. The original size of the clean Hollow Bar dataset is $256 \times 256 \times 128$, whereas the



(a) One (b) Gaussian (c) VRIGL

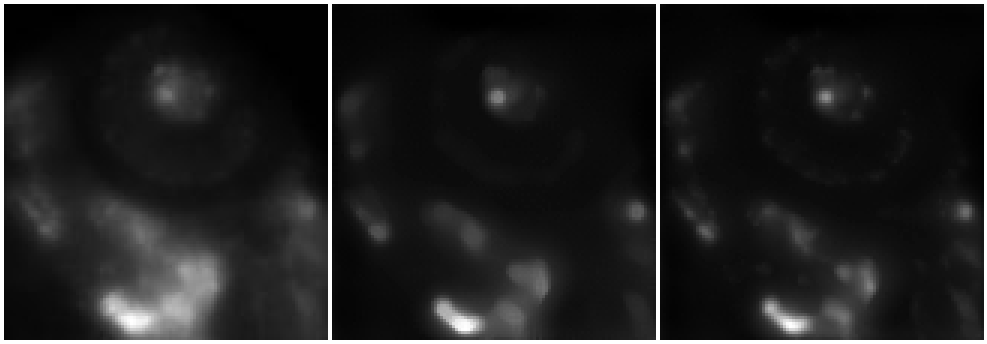
Fig. 3.15 The blind deconvolution results applied to the hollow Bar dataset.

reduced size becomes $64 \times 64 \times 32$. Next, we corrupted the data using a WF PSF, which has the same parameters as the original ones used in the previous section. The RMSE value of the resized blurred dataset was 2965.5 (Table 3.2). In Fig. 3.15, we compared the different deconvolution results using three different PSFs, including Gaussian, VRIGL and the PSF with all elements equal to one.

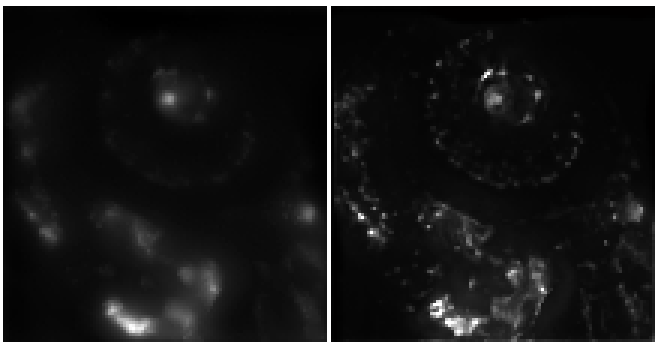
Table 3.2 Comparison of results for hollow bar dataset

Hollow Bar	KFTV $\lambda_h = 0.1$	TM (PSF) $\lambda_h = 10^{-7}$	TV (PSF) $\lambda_h = 0.05$
TM $\lambda_{tm} = 10^{-6}$	2802.6	2750.0	2714.5
TV $\lambda_{tv} = 10^{-2}$	2808.0	2431.7	2305.8
HL $\lambda_g = 10^{-2}$	**2795.1**	**2408.9**	**2225.0**
HL+Mask $\lambda_l = 10^{-7}$	**2795.0**	**2407.2**	**2221.8**

In Table 3.2, the RMSE values are calculated using twelve combinations between of the latent image regularisation methods and the PSF regularisation terms. The trend of RMSE values in Table 3.2 demonstrates that our proposed method was better than the other two regularisation methods (TM and TV) combining with four different PSF updating strategies. In addition, the best method is to use our HL+Mask method with TV regularisation.



(a) Blurred (b) TM+TV(PSF) Blind deconvolution (c) TV+TV(PSF) Blind deconvolution

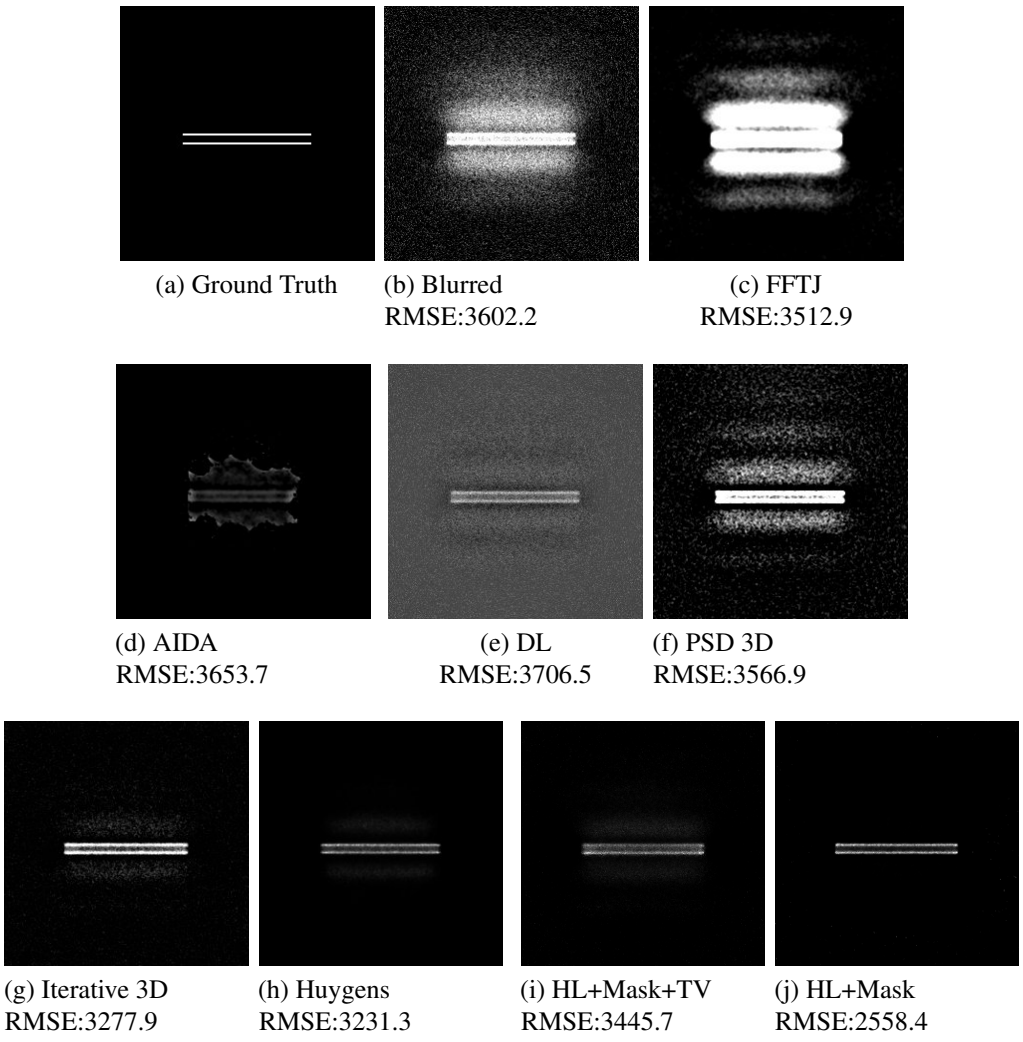


(d) HL+Mask+TV(PSF) Blind deconvolution (e) HL+Mask Non-Blind deconvolution

Fig. 3.16 Results using blind deconvolution methods with different combinations of regularisation terms, including TM+TV(PSF), TV+TV(PSF), proposed HL+Mask+TV(PSF) blind deconvolution method and HL+Mask non-blind deconvolution method; all results are displayed as the maximum value projection along the z direction.

HeLa Cell Nucleus

We also tested a new HeLa cell dataset to evaluate the blind deconvolution algorithm. Voxel volumes of this dataset were $328 \times 328 \times 100$. Assuming the parameters were unknown, we calculated RMSE values for the twelve different combinations noted above. In Table 3.3, our numbers indicate that the proposed methods (*i.e.* HL and HL+Mask) have potential for blind deconvolution.



(a) Ground Truth (b) Blurred RMSE:3602.2 (c) FFTJ RMSE:3512.9

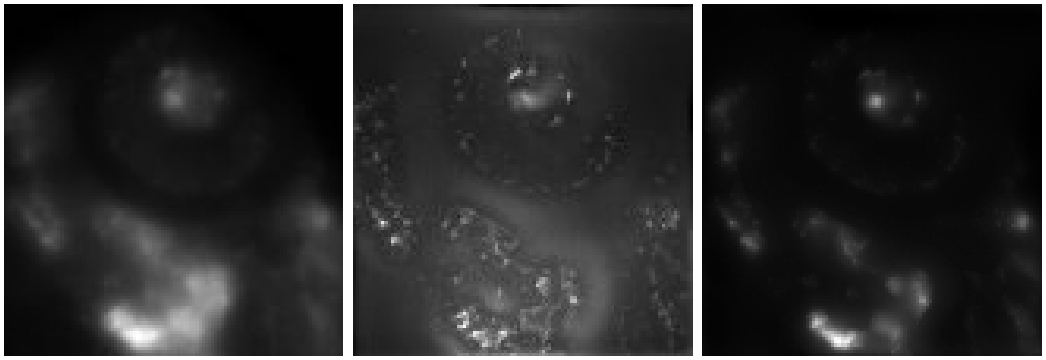
(d) AIDA RMSE:3653.7 (e) DL RMSE:3706.5 (f) PSD 3D RMSE:3566.9

(g) Iterative 3D RMSE:3277.9 (h) Huygens RMSE:3231.3 (i) HL+Mask+TV RMSE:3445.7 (j) HL+Mask RMSE:2558.4

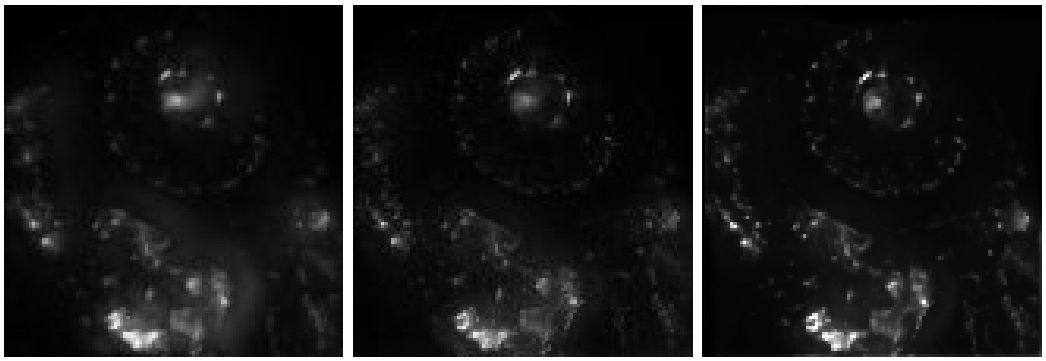
Fig. 3.17 Comparison of results with different software for the hollow bar dataset.

3.5.2 Widefield Fluorescence Data

We tested the WF fluorescence microscopy zebrafish images using the blind deconvolution methods. More specifically, we used three different regularisation methods (*i.e.* TM, TV and HL+Mask) to update both the latent image and TV regularisation term. Our results are shown in Fig. 3.16.



(a) Blurred (b) DconvolutionLab (c) HL+Mask+TV(PSF)



(d) Iterative 3D (e) Huygens (f) HL+Mask

Fig. 3.18 Deconvolution results with different software for the zebrafish data.

3.6 Comparison with State-of-the-art Methods

We implemented the proposed methods using MATLAB 2011(b), and tested this implementation on a PC with an Intel(R) Core(TM) i5-3470 CPU, 8GB of memory and a 64-bit operating system. We compared our proposed algorithm with state-of-the-art methods and some existing software using the hollow bar dataset; the state-of-the-art methods included Iterative 3D [57], FFTJ and DeconvolutionJ (FFTJ) [58], parallel spectral deconvolution 3D (PSD 3D)' [59], DeconvolutionLab (DL) [60], AIDA [56] and the trial version of one commercial package called Huygens [102]. Note that DL, Iterative 3D and Huygens are based on iterative frameworks; further, all methods require the true PSF or the parameters of the PSF based on the imaging system.

Table 3.3 Comparison of results for HeLa cell dataset

HeLa Cell Nucleus	KFTV $\lambda_h = 0.1$	TM (PSF) $\lambda_h = 10^{-5}$	TV (PSF) $\lambda_h = 10^{-2}$
TM $\lambda_{tm} = 10^{-6}$	4777.5	4809.7	4831.6
TV $\lambda_{tv} = 10^{-2}$	4780.3	4811.3	4831.0
HL $\lambda_g = 10^{-3}$	**4740.0**	**4769.2**	**4791.0**
HL+Mask $\lambda_l = 10^{-8}$	**4727.1**	**4765.9**	**4786.5**

We used our proposed HL+Mask algorithm and the HL+Mask+TV(PSF) blind deconvolution method for comparison. We tested all of the above software programs on the Holly Bar dataset (Fig. 3.17) and evaluated programs based on the iterative methods on the zebrafish dataset (Fig. 3.18). We tested all methods using the same VRIGL PSF model and collected the best result of each method. For the iterative methods, we collected results when they converged. Different iterative methods have different convergence criteria, so we did not compare the number iterations corresponding to those iterative methods. From the two figures, we observe that results using our proposed HL+Mask were much clearer and contained more details in the texture region as compared with results using other 3D deconvolution algorithms and existing software.

3.7 Conclusion

we proposed a deconvolution method for 3D WF microscopy data using the MAP approach with sparse priors. The novelty of the proposed method is that we use hyper-Laplacian distribution as a sparse prior to describe the log-gradient distribution. In addition, a local mask is applied to 3D deconvolution to reduce ringing artefacts. The hyper-Laplacian distribution is much closer to the log-gradient distribution than other models, particularly for WF z-stack images. The proposed method can be used in both non-blind and blind deconvolution. Results obtained with synthetic data and real WF fluorescence data from zebrafish embryos demonstrate that the proposed

methods are more applicable to 3D WF microscopy images compared to other state-of-the-art methods. The limitations of the proposed methods are that they require hundreds of iterations to obtain satisfactory results, which is time consuming, and the parameters are not adaptive. In future, we will address how to determine suitable parameters and improve computation speed for high-throughput image analysis.

Chapter 4

Automatic Neurone Detection in Three-Dimensional Light Microscopy Zebrafish Images

4.1 Introduction

Currently, this counting process of tyrosine hydroxylase labelled (TH-labelled) neurones is a slow process to distinguish individual neurones from the background. Neuroscientists must spend substantial amount of time to count these individual neurones through a light microscope, which is a manual, labour-intensive, subjective, and error-prone process. Indeed, it would be fair for biomedical researchers to complain that they spend more than half of their research time manually labelling or counting cells in the specimen through a microscope. High-throughput methods and applications are desperately needed to facilitate the use of this new model system for PD for high-throughput drug screens.

As noted previously, zebrafish is an excellent system model for PD research. In particular, it is easy to visualise dopaminergic neurones in the zebrafish brain by using the aforementioned WISH with a probe against messenger ribonucleic acid (mRNA)

for TH. TH is the rate-limiting enzyme that synthesises L-dihydroxyphenylalanine, the precursor of dopamine. Some of the genetic zebrafish models of PD showed an approximate 25% reduction in their dopaminergic neurones, as assessed by WISH for TH, as early as three days post-fertilisation [103].

Once WISH has been performed on the larvae, they are mounted on microscope slides, and a series of images are taken, capturing each larva in slices called z-stack images. When these z-stack images are combined, they can be rendered as a 3D image showing the dopaminergic neurones in the larva's head. Using this 3D image, it is possible to detect or count the individual neurones. Currently, this is a manual time-consuming process that is therefore subjective and prone to error. The expectation here is that automatic algorithms can free neuroscientists from having to do this detection and counting processes manually.

Our main challenge is to detect individual neurones in z-stack images, as shown in Fig. 1.1. As noted above, this would facilitate the use of this new model system for PD for high-throughput drug screens. To automatically count the number of cells in the zebrafish using computers automatically instead of manually counting, we must first learn the cell feature pattern. Machine learning techniques can be used to achieve this task, including feature extraction, feature description, feature detection and deep learning techniques.

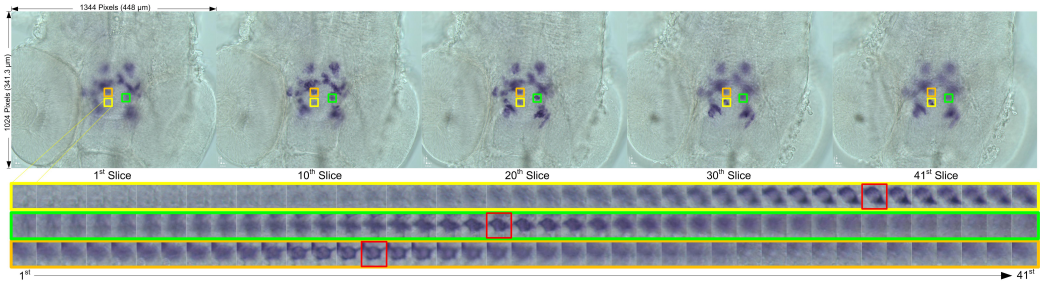


Fig. 4.1 An example of z-stack images containing 41 slices with five slices enlarged (*i.e.* the 1^{st}, 10^{th}, 20^{th}, 30^{th} and 41^{st} slices); the bottom three rows represent three different examples of TH-labelled neurones located within the original z-stack images with three different colours; note that the red square in each row represents the clearest frame of the TH-labelled neurone.

4.2 Major Challenges

4.2.1 Large-scale Image

To process a dataset with large-scale images, high-throughput methods are required to achieve both efficiency and efficacy at a low cost. To obtain our zebrafish dataset, each embryo was visually recorded in several (*i.e.* ranging from 30 to 50) optical sections (*i.e.* z-stack images) by moving the focus plane of the light microscope (See Fig. 1.1). Further, each slice contains three channels in the red-green-blue (RGB) colour space with a fixed size of $1344 * 1024 * 3$. Currently, several high-throughput microscopy applications have been published for biological high-throughput image analysis, such as ImageJ/Fiji [13], CellProfiler [104], Ilastik [14], ICY [15], and Vaa3D [16]; however, most existing image analysis software is limited in scope, capacity and ability to detect TH-labelled neurones in large-scale datasets. With the ongoing development and advances in computational equipment and technology, it has become increasingly feasible to use personal devices for scientific applications. Due to their much higher computational power versus that of conventional central processing units (CPUs), graphics processing units (GPU) are gradually evolved into highly parallel processing units with relatively low cost. Therefore, we used a GPU device to accelerate the computational speed of automatically detecting dopaminergic neurones in large-scale light microscopy images.

4.2.2 Low image quality

There are problems inherent in the 3D multi-channel images recorded through a light microscope which make the detection process much more challenging. One of the major issues here is that z-stack images contain much out-of-focus light because of limitations of the light microscope [2]. During the imaging process,

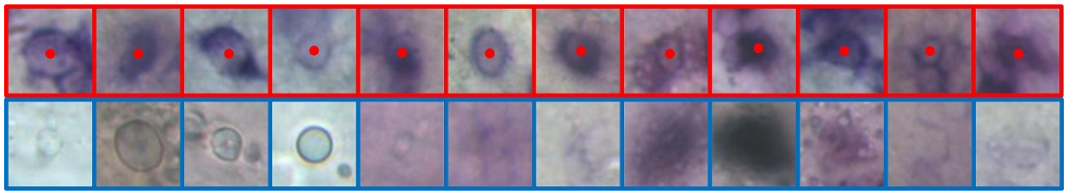


Fig. 4.2 A dozen positive examples (*i.e.* clearest 2D slices) atop a dozen typical negative examples shown in red and blue boxes, respectively; further, red dots identify the central voxels of the positive examples.

image information from the in-focus specimen plane is mixed with out-of-focus light arising from regions outside the focal plane.

Another issue is that not all of the 3D information is recorded by moving the focus plane in a fixed distance at each time, hence the information between two neighbouring slices is lost. An alternative approach to obtain higher-resolution images is to apply deconvolution techniques to remove the out-of-focus light [105, 8]; however, due to the lack of information regarding the PSF and corresponding ground truth images, this inverse problem cannot be perfectly restored [8]. Moreover, the computational cost of current deblurring techniques is extremely high due to the large dataset. In this section, instead of applying deconvolution or restoration algorithms, supervised deep learning algorithms are trained to directly capture unique blurry features.

4.2.3 Irregular appearance

Figs. 4.1 and 4.2 illustrate the particular appearance of TH-labelled dopaminergic neurones in the z-stack images recorded through a light microscope. The dopaminergic neurones appear as a distinguishable colour from the background after the visualisation step. Different neurones may focus on different slices. In a single neurone, the cell membrane is usually stained so heavily than the rest of the cytoplasm, causing the neurone to appear as an irregular ellipse in a single slice. Due to the out-of-focus light, the further the neurone is situated away from the focused frame,

the blurrier it becomes, making it rather difficult to segment and split the touching neurones using simple segmentation algorithms, such as thresholding and clustering. Moreover, because of the individual differences of zebrafish embryos and operation error in the visualisation step, not all z-stacks are stained the same colour or at the same depth.

Detecting TH-labelled neurones in light microscopy images is difficult, because different neurones have highly variable appearances, and the TH-labelled cells are often clustered together, thereby making individual cells that much more difficult to detect. In addition, different imaging conditions, such as exposure time, light source intensity and the transparency of the specimen, can cause variations in image intensity in each of the three RGB channels, thus resulting in cells appearing with different colours in different specimens.

Further, when imaging z-stacks of the zebrafish larvae using light microscopy, there is a slightly out-of-focus light that appears in the recorded z-stack images. We purposefully do not apply any deblurring or deconvolution [8] to the z-stack images, thereby making the task more challenging and our proposed method applicable to realistic scenarios in which the deconvolution parameters are difficult or impossible to obtain. Finally, Fig. 4.1 shows that the father away the cell is from the central frame, the blurrier the cell appears.

There are some detection-based algorithms published in computer vision and biomedical image analysis [65, 7, 66, 9]; however, as mentioned in Chapter 2.2, most of the automated 3D cell detection methods are an unsuitable replacement for manual cell detection [64]. With the development of machine learning techniques, deep learning, *e.g.* using deep neuronal networks, has the potential to achieve automatic detection [31, 10]. Instead of using handcrafted features to distinguish neurones from the background, deep learning-based methods can automatically learn high-level abstractions from the sub-images extracted from the original data. Among popular deep learning models, CNN is one of the most successful architectures and applied

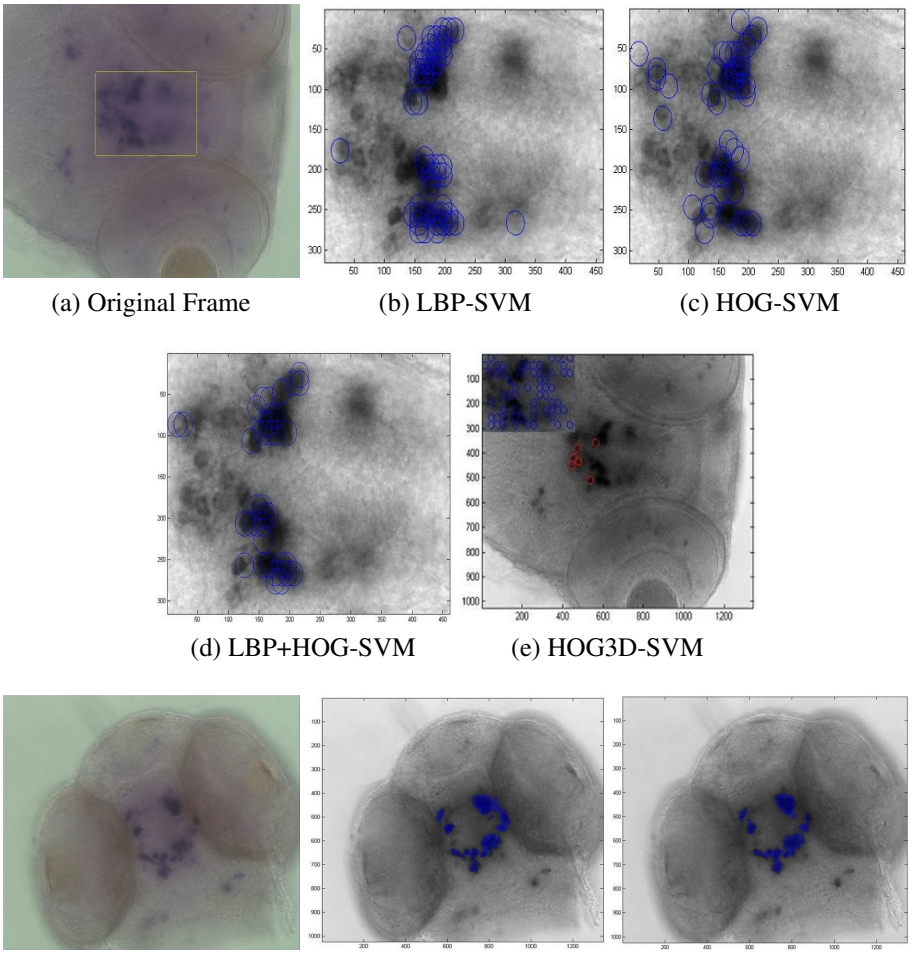
in different areas. Compared to standard feed-forward neural networks, CNNs have much fewer connections and learnable parameters by sharing the same basis function across different image locations [18], which makes CNNs easy to train.

4.2.4 Class-imbalanced problem

In our dataset, only a few positive voxels are labelled in each of the large 3D multi-channel stacks, thus creating a serious class-imbalanced problem for pixel-wise classification using supervised learning algorithms. Here, the minority-class is of primary interest; however, supervised learning algorithms trained using a dataset with an imbalanced distribution of classes will produce a bias towards the majority class, thereby resulting in misclassified and unsatisfactory results. Relevant methods about class-imbalanced problem is reviewed in Chapter 2.4. However, only using one method to solve a class-imbalanced problem with a large imbalance ratio (IR) will not obtain satisfactory results. Therefore, we combine several different techniques for solving the class-imbalanced problem, including oversampling [79], probability sampling [31], boosting [71] and the cascade framework method [78].

4.2.5 Common issues of training deep architectures

Aside from the class-imbalanced problem, there are three other common issues inherent to training a deep learning architecture using a large-scale dataset: these issues are the overfitting problem, impractical training process and no golden standard structures. In this section, we implement the seed expansion and drop-out techniques [106] to address overfitting. We used a GPU based on the compute unified device architecture (CUDA) to dramatically reduce the computational time. There is no golden standard structures for all situation using deep learning architecture. To select the best structure, several different structures with different input image sizes, filter sizes and numbers of layers are tested.



(a) Original Frame (b) LBP-SVM (c) HOG-SVM

(d) LBP+HOG-SVM (e) HOG3D-SVM

(f) Original Frame (g) ColorHist-SVM (h) ColorSift-SVM

Fig. 4.3 Comparison of detection methods that utilise different features

4.3 Cell Counting by Detection with Handcrafted Features

To identify useful features for describing the unique pattern of the desired dopaminergic neurones, we first trained a simple 2D detector with different handcrafted features.

Using this approach, we selected positive cell centre points using ImageJ with the Point Picker plugin [107]. Based on the coordinates of labelled points, we extracted fixed-size patches, each patch containing a positive neurone example. We randomly selected the rest of the unlabelled points, and extracted patches to form

negative inputs. Overall, we used 14 stacks for training and 6 stacks for testing. After obtaining the patches, useful features were extracted from these patches to distinguish desired cells from the non-cell background. In Fig. 4.3, we tested several existing feature extracting methods in computer vision, including local binary pattern (LBP), histogram of gradient (HOG), 3D-HOG, scale-invariant feature transform (SIFT), colour histogram (colorHist), colour SIFT (colorSIFT), edge descriptor and colour descriptor methods.

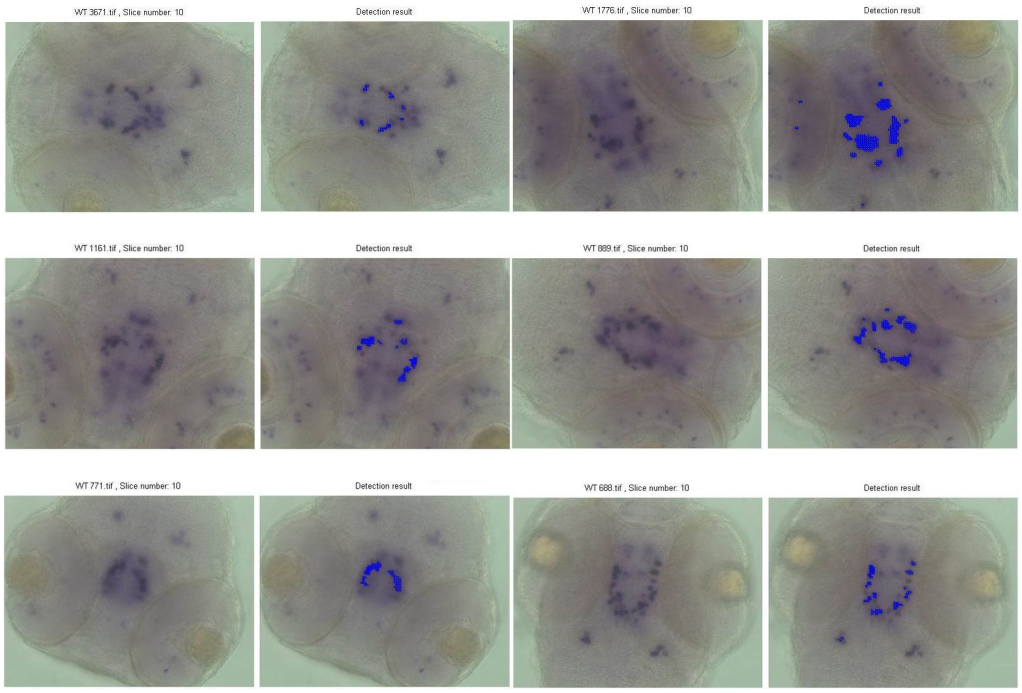


Fig. 4.4 Detection results with manual selected features

There were several cell regions in the original images. In each of these cell regions, there were likewise several cells gathered together, thus making it difficult to distinguish individual cells. We attempted different experiments using several feature extraction methods to train a basic SVM classifier to check the potential applicability of selected features. Compared with several features identified via the SVM, methods with ColorHist and ColorSIFT features performed well, *i.e.* with a

much lower proportion of false negatives. In the sections that follow, we compare results of our proposed methods and traditional approaches.

Although our detection results were not entirely satisfactory, we were able to identity some useful features which can be used in our further experiments. As shwon in Fig. 4.4, the SVM detector with the colorHist feature was able to detect most of the desired neurones, but with many false positives, which is a very good preliminary step in finding regions containing desired neurones.

4.4 Automatic Neurone Detection with Two-Dimensional Convolutional Neural Networks

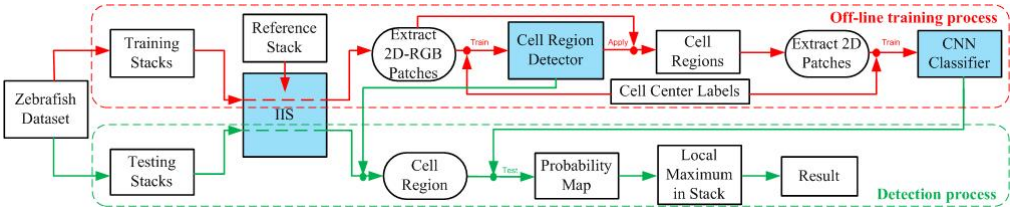


Fig. 4.5 The workflow of the TH-labelled cell detection framework; the off-line training and detection processes are illustrated within the red and green dotted rectangles, respectively.

In this section, we propose an automatic deep learning cell detection framework for 3D z-stack microscopy zebrafish images. In our framework, to improve the efficiency and accuracy of training a CNN from large-scale images, we apply an SVM classifier to detect cell regions and form the CNN training set. Our major contributions herein include our application of deep learning technique, our development of a supervised max-pooling CNN, and results showing the potential capabilities for automatic cell detection in z-stack images. Moreover, we establish and populate an accessible light zebrafish microscopy database with TH-labelled neurones which includes manual annotations by experts on neurone localisation; our database and annotations can be used for evaluating and benchmarking other competing methods.

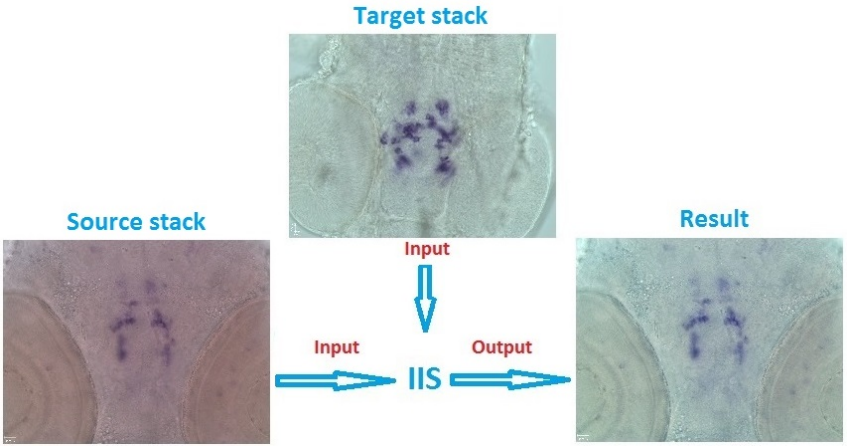


Fig. 4.6 Colour normalisation with image intensity standardisation.

4.4.1 Zebrafish Dataset

Our zebrafish dataset contains 35 stacks scanned under $20X$ magnification through a light microscope. Of the 35 stacks, 25 were selected for the off-line training and 10 for testing. Overall, there were approximately 1000 TH-labelled cells. The volume in voxels of each stack was $1024 * 1344 * z$, where different stacks have different values of z. The spatial resolution was $3\mu m$ and the axial resolution was $1.5\mu m$. The magnification of the objective was $20X$, and numerical aperture NA was 0.7. We made this dataset publicly available for other researchers to develop and evaluate their own TH-labelled cell detection or cell counting algorithms. For each stack in our dataset, a professional observer labelled all centres of TH-labelled cells as ground truth using Point Picker [107], a plugin of the open source Java programme, ImageJ [12].

4.4.2 Pre-processing: Colour Normalisation

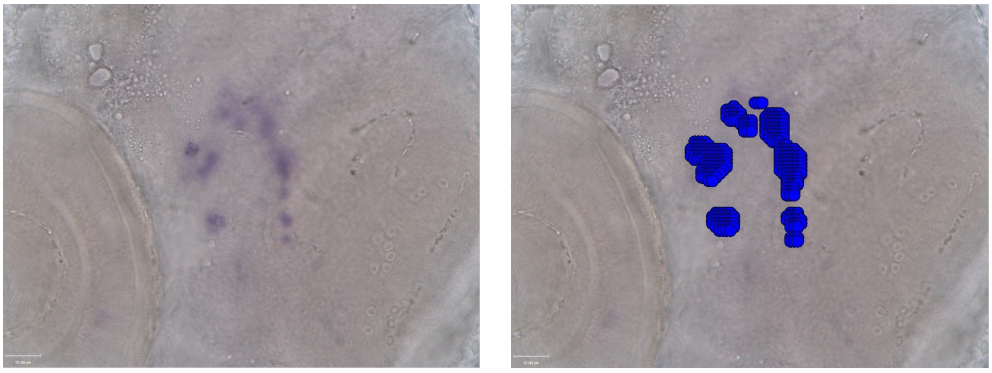
Zebrafish larvae were recorded in several sessions spanning several days to completing populate the dataset. Exposure times were not guaranteed to be the same for each recording session involving the light microscope, so the colour of each stack may differ. We applied image intensity standardisation (IIS), a technique first introduced

in [108] for intensity normalisation 2D grayscale scale images. More recently, in [70], Bogunović *et al.* modified the IIS algorithm to normalise the intensity of the 3D greyscale rotational angiography. In our work, we applied the original IIS as a colour normalisation method for 3D light microscopy images. First, we calculated three histograms of the three channels of the entire RGB stack. Next, we aligned the histogram of each channel to the corresponding reference based on the nonlinear registration method described in [108]. An example of this process is shown in Fig. 4.6.

4.4.3 Cell Region Detection

We determined the cell regions R to subsequently discard irrelevant background regions. Selecting background training patches was also important for training our CNN. Therefore, we detected cell regions efficiently and roughly using an SVM classifier, after which the cell and background training patches were collected from R instead of the entire stack.

When recording the zebrafish larvae, all the TH-labelled cells were located in the centre of the image. We observed that all TH-labelled cells had distinctive colours from the background, and the largest part of the image area was covered by background pixels. To collect supportive training patches for training our CNN detector, we needed to collect patches near the TH-labeled cells. All of the stacks had a similar RGB histogram after IIS colour normalisation was applied; therefore, the colour histogram was the most useful and reliable feature in distinguishing the cell and non-cell regions. The binary SVM classifier based on RGB histogram features (*i.e.* SVM-RGB Histogram) [109] was used as a rough and fast cell region detector. Using this SVM-RGB Histogram detector also guaranteed that all TH-labelled cells were detected within cell regions. Another reason we detected cell regions first was that although CNNs are a powerful tool, training sets must be carefully selected.



(a) Original Frame (b) Cell region R

Fig. 4.7 Cell region detection.

Based on the size of the TH-labeled cell, we extracted patches of size $41*41$ (*i.e.* $123\mu m*123\mu m$). Further, to increase the number of cell samples, for each labelled ground-truth central-pixel $C(x,y,z)$, we drew a cube with spatial radius $r_s = 3$ pixels (*i.e.* $9\mu m$) and axial radius $r_a = 2$ pixels (*i.e.* $3\mu m$) along the z direction. All pixels within the cube were considered as cell-sample centres C_p, and all remaining pixels in the stacks belonged to background-sample centres C_n. We extracted all cell patches from C_p in 20 training stacks, then extracted the same number of background patches randomly from C_n. To train the SVM-RGB Histogram detector, approximately 0.1 million cell patches and background patches were extracted.

Given the above, we used the SVM detector to detect the cell region and thereby collect CNN training patches, that are able to remove most of the background pixels. This stage was more like a feature selection pre-processing stage. Using this approach, the CNN could also be more accurate in detecting cell samples within the cell region. Similarly, in the test stage, for accuracy, we also first applied the SVM detector to detect those regions. We compared our proposed method in training the CNN without this cell region detection stage. For training the conventional CNN, cell samples were the same, but the non-cell samples were randomly selected from the background.

SVM is more like a training data pre-selection step. Due to the feature divergence and complexity, we need more data to train the deep learning architectures directly without the pre-selection step. Another problem is the data is class-imbalanced. We do not have too many positive training examples. Therefore, the SVM pre-selection step is helpful to reduce the feature complexity and balance two classes of the training data.

4.4.4 Training Set Pre-processing

After the cell region R was detected by the SVM-RGB Histogram detector, we extracted cell and background patches in region R from all test stacks to train CNN with the same patches sizes and neighbourhood setting described in Section 4.4.3. Given that pixels within cell region R had similar colours, colour features in the cell region were not a reliable means of distinguishing cell and background patches. To reduce training time, we transferred all RGB patches into the YUV colour space, using only the Y-channel patches. For each Y-channel cell patch, we rotateed 0, 90, 180 and 270 degrees to make the detector rotation invariant and increase the number of cell samples. The cell and background patches can have overlapping pixels which is beneficial for increasing the probability of detecting true cells. Approximately 0.5 million cell patches were extracted from all training stacks, similarly, approximately 0.5 million background patches were extracted from cell region R.

4.4.5 Cell Pixel Detection

After the max-pooling CNN was trained, we tested it on the test stacks. Cell region R was first detected by the SVM-RGB Histogram detector in every frame of each stack in our test dataset. Then, we used pre-trained CNN to detect cells by scanning every pixel in R, assigning probability value P_c to each pixel. This strategy required less than 300 seconds to process each slice. As a result, we obtained 3D probability

map M_p for each stack, and used a Gaussian filter to smooth this probability map M_p. Finally, we can find all 3D local maxima in the smoothed M_p using method described in [31], which requires a self-tuning threshold.

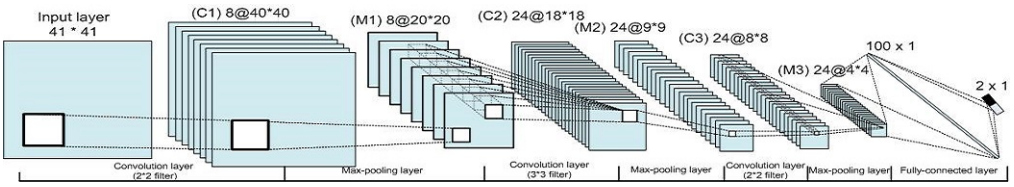


Fig. 4.8 The outline of the convolutional neural network architecture.

4.4.6 CNN Training

We trained the CNN network shown in Fig. 4.8. Training this CNN required three days of computation using a MATLAB R2011b implementation with C++ library [110] on a personal computer with an i5-3470 CPU clocked at 3.4GHz, 14GB memory and 64-bit operating system. Less than 15 epochs were required to reach an error of less than 10% with a total of 1 million training patches. We tested different architectures, including different layers, filter sizes and patch sizes. The structure in Fig. **??** works the best.

4.4.7 Results

We evaluated our detection results according to cells labelled by human observers in the test set. Detected cells closer than 10 pixels ($30\mu m$) in the 2D slice, and five pixels ($7.5\mu m$) in the vertical direction from a ground-truth centroid were defined as true positives. The distance is decided based on the closest distance of two neurones in the 3D stack. In addition to number of true positives N_{TP}, we also counted the number of false positives N_{FP} and false negatives N_{FN}. Next, we calculated performance measures, including recall (*i.e.* $recall = N_{TP}/(N_{TP}+N_{FN})$, precision (*i.e.* $precision = N_{TP}/(N_{TP}+N_{FP})$, and F_1 (*i.e.* $F_1 = 2PR/(P+R)$).

We called our proposed method here the Refined CNN, and we compared it with a CNN without the above cell region detection stage (CNN); we also compared our Refined CNN with an SVM method with different features, including colorHist, colorSIFT, and colorSIFT+HOG. Table 4.1 shows that our proposed method performed significantly better than the other approaches. In addition, we observe from the example results shown in Fig. 4.9 that our detection accuracy was good if cells were not clustered together. Therefore, we also conclude here that our proposed method requires improvements in the terms of cell cluttering.

Table 4.1 Comparison of results on test stacks.

Method	Recall	Precision	F1
Refined CNN (Proposed)	**0.7692**	**0.6452**	**0.7018**
CNN	0.6071	0.3542	0.4474
colorSift+HOG	0.5600	0.2593	0.3544
colorHist	0.4839	0.1136	0.1840
colorSift	0.3704	0.0962	0.1527

4.5 Three-Dimensional Framework for Automatic Neurone Detection

As noted in previous sections, automatic detection of neurones in zebrafish images is required to be fast, robust and accurate for disease research. In this section, we present a high-throughput automatic detection method using a dynamic cascade framework for progressively locating desired neurones. Our dynamic cascade framework is designed to address the problem of training the aforementioned deep learning architectures from a severely class-imbalanced dataset with labelling only available at the central voxel of desired objects.

In the framework, to progressively capture the unique high-dimensional structure of desired neurones, we first trained support vector machine, a 2D CNN and a 3D multi-channel CNN. The entire detection framework was implemented with acceler-

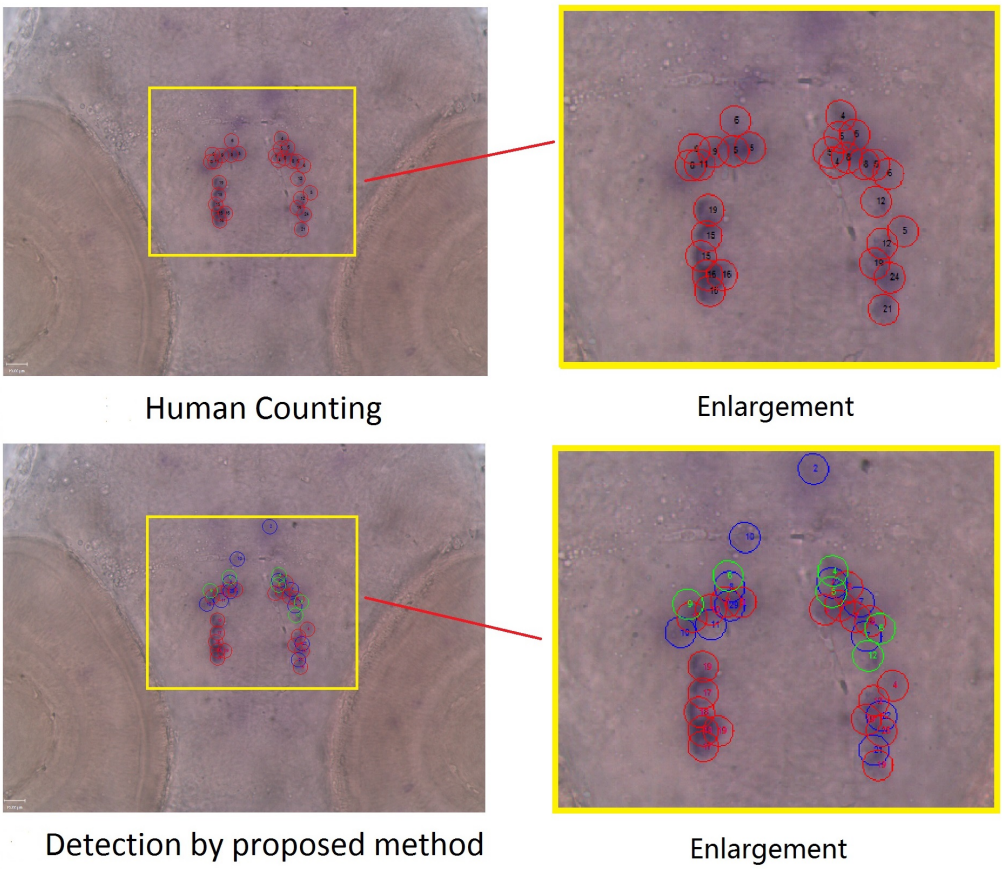


Fig. 4.9 Detection result for two stacks; here, human observers identified 28 TH-labelled cells; in results produced by our proposed method, there were 20 red, 11 blue and six green circles in the stack, where red, blue and green circles represent true positives, false positives and false negatives, respectively; note that numbers in circles indicate the slice location of each cell.

ation via a GPU. We thereby illustrate our proposed method to detect dopaminergic cell in a zebrafish model of PD using TH-labelled neurones in light microscopy images. Further, we publish our new zebrafish image dataset, so that further research in this area has a common benchmark for comparison. Our results show that our proposed deep learning framework outperformed some state-of-the-art techniques, thereby demonstrating its potential for high-throughput automatic cell detection in light microscopy images.

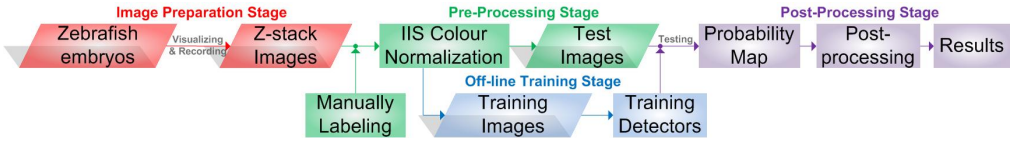


Fig. 4.10 Outline of our framework, including four main stages: image preparation, pre-processing, offline training and post-processing.

As illustrated in Fig. 4.10, our framework consists of the following procedures: (1) image preparation; (2) pre-processing (3) offline training and (4) testing stage. In this section, we present details of each of these steps.

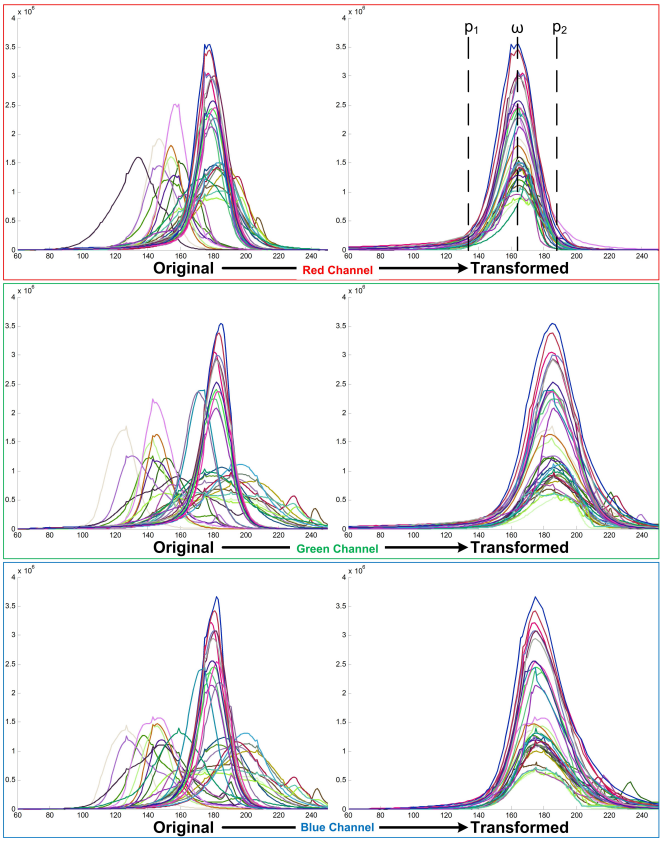


Fig. 4.11 Image histograms (3 channels) of 35 stacks before and after Image Intensity Standardisation transform. Each colour represents one stack of images. ω, p_1 and p_2 are three control points (landmarks) of the transform.

4.5.1 Zebrafish Embryo Image Preparation

Neurone Visualisation

Neurone visualisation is a critically important step in distinguishing dopaminergic neurones from the background. Dopaminergic neurones in the zebrafish brain were visualised using WISH with a probe against mRNA for TH. WISH is a vital tool used to characterise the phenotype of genetic mutations and chemical genetic perturbations.

Z-Stack Images Recording

After 3 days post fertilisation, zebrafish embryos were collected and ready for imaging. Once WISH was performed on the zebrafish embryos, they were mounted on microscope slides with their heads facing upwards. In doing so, zebrafish embryos should be fixed in place; otherwise, any movements will cause motion blurring during the recording with a CCD camera. An alternative way of preventing motion blur is to cut the head off before the imaging process. To capture all TH-labelled neurones in each zebrafish embryo, the entire head of the zebrafish embryo must be recorded, thus narrowing the choice of microscope and objective lens. To ensure higher image quality and less imaging time, we used a conventional light microscope and objective lens with 20-times magnification to image the entire head of the zebrafish embryos.

4.5.2 Pre-Processing

Labelling Z-Stack Images for Supervised Training

To train a supervised learning algorithm, TH-labelled neurone (*i.e.* positive) examples and non-neurone (*i.e.* negative) examples should be collected from a range of z-stack microscopy images. In this thesis, the central voxel of each TH-labelled neurone was labelled by experts, thereby showing the location of a positive example. In

Fig. 4.13(a), the red point indicates the labelled voxel for recording 3D coordinate information of a dopaminergic neurone in 3D z-stack images. We used Point picker V.27.09.2003, a Plug-in for ImageJ [107], to label the locations of TH-labelled neurones. Neuroscientists had to find the clearest frame in the z-stack images and label the central voxels of all possible TH-labelled neurones.

Image Intensity Standardisation

The entire dataset was completed by recording many zebrafish embryos several times. In general, we cannot guarantee that imaging conditions will be the same each time zebrafish embryos are recorded. Different exposure times, the transparency level of the specimen, lighting conditions and the degree of cleanliness of the objective target can affect image intensity values in each channel of the RGB colour space. Further, the intensity range of dopaminergic neurones is also varies.

In this thesis, we achieved colour normalisation based on non-linear IIS which was first described in [108] for 2D magnetic resonance (MR) image. We extended the IIS approach to attain a robust colour normalisation method for 3D light microscopy images. Following IIS transform, we normalised the intensity histogram by establishing the correspondence between two histograms calculated from generic z-stack and reference z-stack images. This registration was initialised with a transformation composed of three control points aligning the peak (ω) and two shoulders (p_1, p_2) of the unimodal histogram. Formula and calculation details of the IIS transform can be found in [108]. Further, Fig. 4.11 illustrates histograms from 35 stacks before and after the IIS transform.

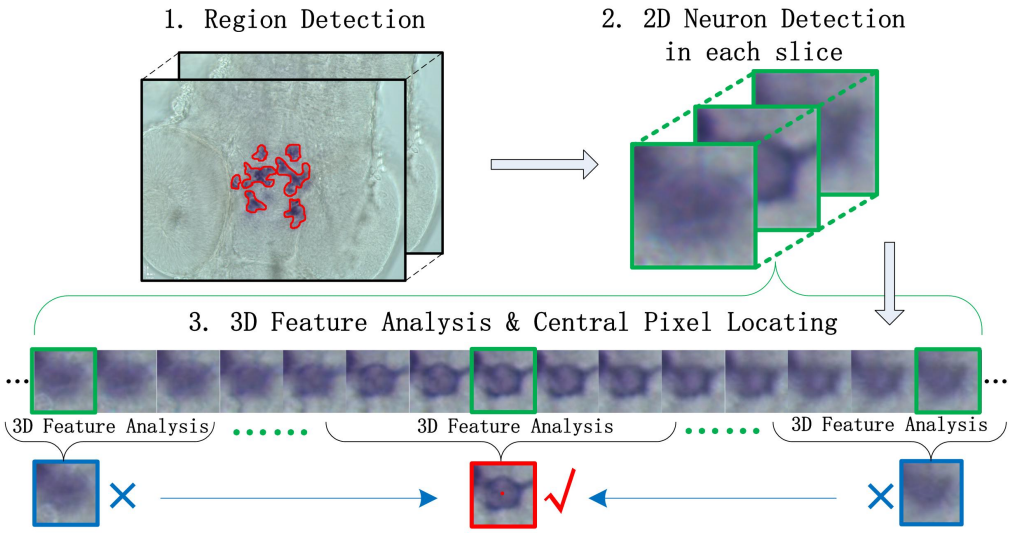


Fig. 4.12 Illustrating the progressive detection process in which the TH-labelled neurone region is first detected based on its distinguishable colour; next, all possible 2D neurones are detected in each slice; finally, the 3D feature of each possible neurone is analysed and the central voxel in the clearest frame is identified.

4.5.3 Off-line Training

Progressive Detection Design

It is rather difficult to directly locate the precise central voxel of a 3D neurone; even experts must follow a progressive process. Therefore, the general location of TH-labelled neurones is first visually detected. Next, experts focus on one possible neurone, then search through all slices for its clearest appearance. Finally, the centre voxel of the neurone will be labelled in its clearest frame. A similar progressive detection process is designed and described in Fig. 4.12; however, the class-imbalanced problem must first be solved to train a supervised learning algorithm.

Class-Imbalanced Dataset

Given training data with only a few positively labelled voxels, effective and efficient methods must be explored for training a deep learning architecture from such extremely imbalanced data. This detection problem can be translated into a classi-

fication problem, as each voxel should be classified into two classes, *i.e.* positive (cell) and negative (non-cell) classes. In this section, we describe the final goal of locating the centres of all TH-labelled neurones in z-stacks, such that the positive class refers to the region of interest. If all labelled voxels in one training stack are treated as positive samples, the remaining voxels will be negative samples. This class-imbalanced problem is so serious that the aforementioned IR for training is extremely large ($IR = 1344 * 1024 * 40/30 > 1.8M$).

Based on the literature, there is no clear definitive best method for handling such serious class-imbalanced problem. Due to the extraordinarily large IR, techniques are required primarily to adjust the distribution or reduce the IR of the data distribution before training. Since training with all majority examples is impractical and requires too much time, our designed cascade framework with a deep learning architectures focuses on hard examples in the majority class. In this section, the class-imbalanced problem is addressed using a dynamic cascade learning framework which uses seeds expansion to learn from entire images with positional information of TH-labelled neurones.

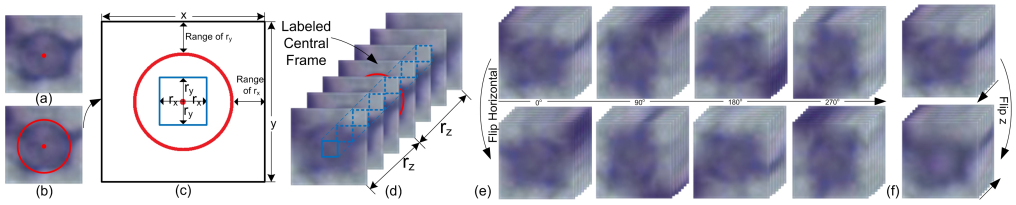


Fig. 4.13 Seed expansion process: (a) Seed (red point); (b) modelling shape with a circle; (c) neighbour voxels within $[r_x, r_y]$ are treated as positive voxels; (d) a 3D training patch extracted based on the seed with all voxels within $[r_x, r_y, r_z]$ (blue cuboid) treated as positive voxels; (e) the upper four 0°, 90°, 180° and 270° rotations of the original 3D patch and lower four rotations of the flipped 3D patch (*i.e.* we flip each frame horizontally); (f) flipping of the original 3D patch along the z direction.

Seeds Expansion

In this section, we use seed expansion to oversample the minority-class voxels based on the seed position in z-stack images. Resampling methods can adjust the

distribution of original classes to improve the accuracy of classifiers. Oversampling and undersampling are two widely used resampling techniques; however, there are problems when applying random oversampling and undersampling, including loss of information with undersampling and overfitting with random oversampling. In our case, if each labelled voxel was treated as a seed, the minority-class could be enlarged. Therefore, instead of generating synthetic data based on the K-nearest neighbours of features [79], the nearest neighbours in the original z-stack images based on the seed location are collected as positive voxels. To train patch-based deep learning architectures, positive training patches are extracted based on all positive voxels using this seeds expansion technique.

Fig. 4.13 illustrates the process of seed expansion. All labelled positive voxels treated as seeds are extended based on their 3D coordinates. The nearest neighbour voxels within three different dimensions are then selected as positive examples. The radius in each dimension, *i.e.* r_x, r_y, r_z, is chosen based on the true size of the TH-labelled neurones, and voxels within the radii are treated as positive voxels. We can then generate one 3D patch from each positive voxel. The principle behind selecting the range of radii (r_x, r_y, r_z) is that voxels belonging to the TH-labelled neurone should be extracted within positive training patches.

Since the supervised learning method described below is based on small patches extracted from original images, the features are limited by the size and position of each training patch. To enforce rotation and mirroring invariance, patches extracted from positive-class voxels are rotated and mirrored to produce more training samples; however, producing too many similar patches from one single voxel may cause overfitting problems [79]. In this section, only four degrees (*i.e.* 0°, 90°, 180° and 270°) are used for patch rotation, as it is easy to implement without any additional padding processes.

Dynamic Cascade framework

We designed the dynamic cascade framework to select harder negative examples to train a stronger classifier step by step. Our dynamic cascade framework is inspired by the cascade and boosting method. Viola and Jones introduced a cascade framework to achieve real-time face detection [78]. In their approach, at each of the cascade stage, a simple classifier was boosted to become a stronger one. By increasing the number of features, the boosted classifier can have the ability to classify hard examples.

The advantage of this framework is that each stage reduces the false positive rate as it simultaneously retains the high detection rate. One disadvantage is that training process takes too much effort to calculate different low-level features and is very time-consuming to boost a weak classifier to become a stronger one, requiring hundreds or thousands iterations. Another difficulty is that those selected low-level features may not have the ability to classify the hardest examples. Therefore, we build on the advantages and overcome the disadvantages of the cascade and boosting methods. In this section, we describe our framework which entails several strong classifiers which are cascaded to form a dynamic cascade framework for rapid small cell detection in large-scale images with only positively labelled information.

Fig. 4.14 illustrates our entire learning and detecting framework. In each training model, the primary purpose is to reduce the false positive rate and simultaneously retain the high detection rate. How to efficiently and accurately detect and remove the majority of negative voxels in large-scale images is the major challenge here. As the negative class is the majority class, once the voxel is rejected in any model, it is considered as a non-neurone voxel and treated as background.

Implementation details of our approach are described in Algorithm 3. Due to the unique structure and small number of TH-labelled neurones, we use seed expansion to oversample the small number of examples in the minority class (*i.e.* the desired

Algorithm 3 Dynamic cascade framework

Input: Training images and corresponding labels;
Probabilistic Training Model $\mathscr{M}^n, n = 1,2,...,N$;
Initialize: $\mathscr{S}_1 = \mathscr{S}, n = 1, t = 1$;
While $n <= N$
1. There are m_t^0 voxels of the majority class $\mathscr{S}_t^0$, and m_t^1 voxels of minority-class S_t^1 forming a training dataset $S_t = \mathscr{S}_t^0 \cup \mathscr{S}_t^1$;
2. Treat all m_t^1 voxels in minority-class $\mathscr{S}_t^1$ as seeds and do seed expansion to form temporal minority-class training set $\mathscr{S}_t^{1*}$ with label $y^1 = 1$, and remove overlapping ones $\mathscr{S}_t^{1*} \cap \mathscr{S}_t^0$ from $\mathscr{S}_t^0$;
3. Uniform undersampling majority class forming temporal majority class training set S_t^{0*} with label $y^0 = 0$ to balance the temporary training dataset with two classes $\mathscr{S}_t^* = \mathscr{S}_t^{0*} \cup \mathscr{S}_t^{1*}$;
4. Using training model $\mathscr{M}^n$ to train a probabilistic classifier $\mathscr{C}_t$ with patches extracted from voxels in $\mathscr{S}_t^*$, get a trained classifier $\mathscr{P}_t \leftarrow C_t(X)$;
5. Get the probabilities for voxels in both classes:
$\mathscr{P}_t^0 \leftarrow \mathscr{C}_t(\mathscr{S}_t^0(j)), j = 1,...,m_t^0$,
$\mathscr{P}_t^1 \leftarrow \mathscr{C}_t(\mathscr{S}_t^1(k)), k = 1,...,m_t^1$;
6. Calculate adaptive threshold
$th_t = min(\mathscr{P}_t^1(j)), j = 1,...,m_t^1$;
7. Remove example $\mathscr{S}_t^0(j)$ from $\mathscr{S}_t^0$, where $\mathscr{P}_t^0(j) < th_t$, to form a new temporal majority class training set $\mathscr{S}_{t+1}^0$;
8. $n = n+1$;
9. New training set with two classes $\mathscr{S}_{t+1} = \mathscr{S}_{t+1}^0 \cup \mathscr{S}_{t+1}^1$, where $S_{t+1}^1 = S_t^1$.
10. $t = t+1$;
end while
Output Cascade classifier $\mathscr{C}_k$ with threshold th_k, where $k = 1,2,...,t$.

positive examples). To train the probabilistic classifier with one model, we uniformly undersample examples in the majority class (*i.e.* negative examples) to rebalance the temporary training set. The trained probabilistic classifier therefore gives the probabilities for examples in both classes. Next, some of the examples in the majority class will be rejected if the probability values of those examples are lower than a given threshold.

This threshold is set to the lowest probability value in the minority class. By setting this adaptive threshold, we ensure a high detection rate and reduce the false positive rate. The remaining examples in the majority class will then be used in the next stage in the framework. Each model can be trained several iterations with the

same positive samples. When the model is unable to classify the remaining hard examples, the system will jump to the next model. This process will eventually be terminated after all models are involved.

In this section, we use three different models to classify each voxel in the image. The classification ability of these three models is progressive. Model 1 is a colour region detector using a linear classifier with one simple colour histogram feature. Model 2 is a 2D neurone detector with a 2D CNN. Model 3 is the 3D neurone detector with a 3D multi-channel CNN.

Model 1: Model 1 is a cell region detector. After the colour normalisation process is applied during the pre-processing stage, intensity values in each channel are normalised and the TH-labelled neurones appear in a similar colour. Colour is the main feature that distinguishes a TH-labelled neurone region from the background. Because of out-of-focus light, the candidate colour region is much larger than the true size of the TH-labelled neurones. Therefore, the false positive rate will be high, and the detection rate is also unsatisfactory if only the colour feature is used. Instead of adding more features in the next stage as is done in [78], we use a 2D CNN to automatically learn important 2D features. Model 1 can thereby rapidly shrink the candidate regions, leaving the harder examples to be detected by Model 2.

Model 2: Model 2 is a 2D neurone detector. CNNs are multi-layer models combined with different types of layers, including convolutional layers, sub-sampling layers, and fully connected layers. CNNs are one of the most successful deep learning models for supervised learning and which also achieve high performance in different applications [111]. In our 2D CNN structure, we have a convolutional layer, a max-pooling layer, and a fully connected layer.

Model 3: Model 3 is a 3D neurone detector. Note that 3D CNN were first introduced in [112] for human action recognition in videos. In 2D CNNs, convolutions are applied to 2D feature maps to compute features from only a single frame. When applied to three-dimensional problems, it is desirable to capture the axial information

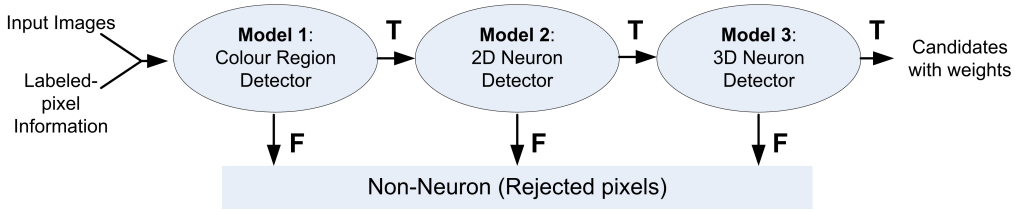


Fig. 4.14 Illustrating our dynamic cascade framework with different models which learn from large-scale images only containing positively labelled information. (a) Model 1 is a colour region detector using one colour feature and an SVM classifier; (b) Model 2 is a 2D neurone detector using 2D CNN with few feature maps and(c) Model 3 is 3D neurone detector using 3D multi-channel CNN with many feature maps.

of the third dimension encoded in multiple frames. We present a 3D multi-channel CNN which captures the unique 3D structure of the TH-labelled neurones in light microscopy z-stack images.

Three-Dimensional Multi-channel Convolutional Neural Network

To capture the unique three-dimensional features of TH-labelled neurones and find the specific locations of those neurones in the clearest frame, we designed a 3D multi-channel CNN, as shown in Fig. 4.15. After training the previous two models, we capture the colour feature and 2D features encoded in the TH-labelled neurones. In each channel, the value of the unit at position (x,y,z) of the i_{th} layer on the j_{th} feature map in the c_{th} channel is defined as follows:

$$v_{cij}^{xyz} = f\left(b_{cij} + \sum_{m}\sum_{p=0}^{P_i-1}\sum_{q=0}^{Q_i-1}\sum_{r=0}^{R_i-1} w_{cijm}^{pqr} v_{c(i-1)m}^{(x+p)(y+q)(z+r)}\right), \tag{4.1}$$

where $f(.)$ is the activation function, b is the bias and w is the weight.

At the image preparation stage, we used Volocity V6.3 [113] to manually and consistently adjust the imaging conditions for recording each stack of images. A function called Voxel Spy can guarantee exposures at the same level such that TH-labelled neurones appear neither too dark nor too bright in recorded images. TH-labelled neurones appear with the same structure in three different channels but

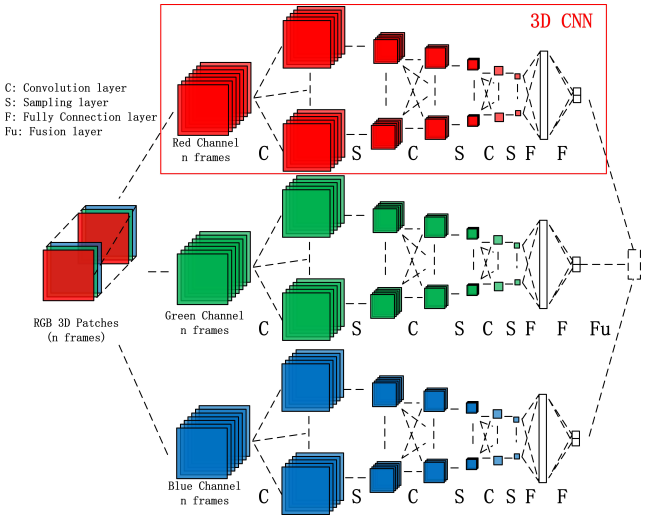


Fig. 4.15 3D multi-channel CNN.

appear with different contrasts. The colour map of TH-labelled neurones in the RGB space is approximately $[0.2, 0.2196, 0.3333]$ after the colour normalisation stage; more specifically, it appears in a unique dark purple colour.

After being processed by Model 1, the remaining voxels have similar colour features. Inspired by the method presented in [114], instead of using the RGB image to train one 3D CNN structure, we use three columns in Model 3 to separately process the information in three channels to improve the performance and also save on memory use. Since each TH-labelled neurone has a similar structure in three channels, we used three columns to train three different 3D CNNs with data collected from three channels of the RGB microscopy images, respectively. In each CNN structure, we applied the invariant back-propagation algorithm [115] to update the weights such that the CNNs are robust to variations and noise. In the final fusion step, instead of averaging the predicted results of each network p_c, we multiply them to obtain final probability value p, *i.e.*

$$p = \prod_{c=1}^{3} p_c, \tag{4.2}$$

where p_c is the probabilistic value obtained from column c, and $p_c \in [0, 1]$.

In each channel, the TH-labelled neurones appear almost with the same structure, but with a different range of intensity values. The multiplied probabilistic value map helps to ensure that the centre of the TH-labelled neurone is more obvious. We used rectified linear units (ReLu) as the activation function. Further, we used expansion and drop-out methods to prevent overfitting.

We treated the RGB image separately using multi-channel CNN include several reasons. Firstly, the parameters in the un-trained CNN model is randomly initialized, and based on the literature, training several networks can produce better result than only using one network. Secondly, the calculation cost of using 3D-MC-CNN is much lower than the combined CNN model. Finally, the 3D-MC-CNN can also learn the dependencies between different channels, which is in the fully-connected layer. We tested both of the methods and the result shows using 3D-MC-CNN can produce better result compared with the method using one combined CNN.

4.5.4 Post-Processing

In several cell detection methods [31, 10], the post-processing step finds local maxima and uses them to identify specific locations of desired objects. One disadvantage here is that due to noise, several local maxima may be selected around one true object. Therefore, the threshold among selected local maxima should be manually decided. Another disadvantage is that because of the irregular appearance of the TH-labelled neurone, the local minimum may not be the centre of the neurone. In this chapter, the post-processing step addresses these two problems. We apply an auto-thresholding method to multiply probabilistic value map M_{pv} to obtain a binary mask M_b which represents the most likely cell regions.

We tested several different auto-thresholding methods using ImageJ, and the best method selected is the Otsu thresholding method [116]. To ensure there is only one possible cell in each connected region of mask M_b, we apply a watershed

algorithm. We only need to find extremal regions in the probability map produced by 3D-MC-CNN for representing the locations of desired cells. Although watershed algorithm may over-segment the image and producing many small isolated region, we can still find a large isolated region to represent each location of the cell. Then, the small isolated region will be removed based on the size of the region.

Next, very small regions of M_b are removed, and we analyse the sensitivity of the size of the removed region in terms of the final result. In each isolated 3D connected region R^I, weighted centroid $\mathbf{c}^w$ of the isolated region is calculated using equation (4.3). Finally, the weighted centroids are detected as locations of TH-labelled neurones.

$$\mathbf{c}_i^w = \frac{\sum_{\mathbf{x}\in R_i^I} M_{pv}(\mathbf{x}) \cdot \mathbf{x}}{\sum_{\mathbf{x}\in R_i^I} M_{pv}(\mathbf{x})}, i = 1,2,3,...,n. \tag{4.3}$$

The relationship between the size of the removed small regions and the final result is illustrated in Fig. 4.16 in which the red line (*i.e.* weighted centroid) and green line (*i.e.* local maxima) are averaged over all training stacks. Evaluation score F_1 weighted centroid was much higher than that of the local maxima. To obtain the best evaluation score, we selected the size of the removed region when the red line reached its peak. Using this approach, the best choice for the size of the removed small regions was 360, meaning that any isolated region containing less than 360 voxels will be removed in the final stage.

4.5.5 Experiments and Results

Zebrafish Dataset

The dataset contains 60 stacks scanned under $20X$ magnification through a light microscope. Forty of these stacks were selected for the offline training, whereas the

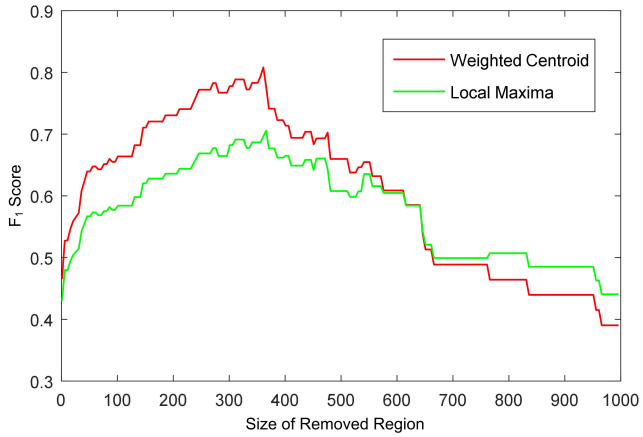


Fig. 4.16 The relationship between the size of the removed small regions and final evaluation score F_1 from equation (4.6); here, the red line represents the weighted centroid, and the green line represents the 'local maxima'.

Table 4.2 How much faster with GPU!

Train/Test	# patches	CPU(s)	GPU(s)	# times faster
Test	10	0.2132	0.2208	0.97
	1000	22.1687	0.3161	70.13
	10000	221.4367	1.3647	162.26
Train	200	12.4101	0.3334	37.22
	2000	128.0468	0.7867	132.76
	20000	1241.4702	5.0791	244.43

remaining 20 were used for testing. The number of TH-labelled neurones in one z-stack of images is around 20 to 40.

The volume (in voxels) of each stack was $1024 * 1344 * 3 * z$, where different stacks had different values of z. The spatial resolution was $3\mu m$ and the axial resolution is $1.5\mu m$. The magnification of the objective was $20X$, and the numerical aperture (NA) was 0.7. For each stack in this dataset, an experienced human observer labelled all centres of TH-labelled neurones as ground truths using Point Picker [107]. We make this dataset publicly available such that other researchers can develop and evaluate their own TH-labelled neurone detection or counting algorithms.

Implementation

The entire cascade framework was accelerated using a GPU, *i.e.* 2GB NVIDIA GeForce GTX 760 Ti with a 16GB memory on a 64-bit personal computer; the

entire framework was written on CUDA in C++ and MATLAB. In Table 4.2, it illustrates how much faster using GPU acceleration than only using CPU for training and testing the model 3 with 41*41*7-size patches. Our CNN structure was modified based on the ConvNet library created by Demyanov [117].

To choose the best structure for the 2D CNN and 3D multi-channel CNN structures for processing the zebrafish dataset, we tested the sensitivity and effectiveness of different structures, including different patch sizes and filter sizes. In the dynamic framework, we observed a unique property of our zebrafish dataset by which candidate regions containing TH were less than 10% of the entire z-stack images. Therefore, we used two iterations of Model 1 to quickly and precisely detect the unique purple colour regions.

As a result, we pre-trained two iterations of Model 1 and one iteration of both Model 2 and Model 3. In Model 1, the size of each patch was $41 * 41$, and we used a linear SVM and RGB-Histogram to detect candidate regions. Finally, the bin of the RGB-Histogram is 9. In Model 2, the size of each patch was $41 * 41$ to train a 2D CNN. Note that we used 20 epochs to train the 2D CNN model. In Model 3, the size of each patch was $41 * 41 * 7$ to train a 3D multi-channel CNN framework; we used 30 epochs to train the 3D multi-channel CNN model. Training the entire framework required several hours. It takes approximate 20 seconds to process one slice (1344*1024*3), and less than 10 minutes to obtain the detection and counting results of one stack of zebrafish images containing 35 slices.

Evaluation Criterion

We used labelled information as ground truth to evaluate the performance of the different methods. If the distance between labelled location (x_t, y_t, z_t) and detected location (x_d, y_d, z_d) was closer than d, then this location was considered to be a true positive (N_{TP}). Constant d was set to 10, which is approximately the closest distance

of the two smallest TH-labelled neurones in our training images. Each labelled location can only have at most one corresponding detected location as correctly detected true positive, if there are more detected results around one labelled location, only the closest detected location can be considered as true positive.

Aside from the number of true positives N_{TP}, we also counted the number of false positives N_{FP} and false negatives N_{FN}. After these counts were accumulated, we calculated performance measures, including recall, precision, and F_1 score for each of the test z-stack images. For each stack of images, we calculated the evaluation scores with those performance measures. Finally, scores for each evaluation method were averaged to yield a final evaluation score for $Recall(R)$, $Precision(P)$ and F_1-score (F_1), respectively. The *Recall* and *Precision* metrics are defined as:

$$Recall = \frac{N_{TP}}{N_{TP} + N_{FN}}, \tag{4.4}$$

$$Precision = \frac{N_{TP}}{N_{TP} + N_{FP}}, \tag{4.5}$$

and F_1-score is the harmonic mean of *Precision* and *Recall*:

$$F_1 = 2 \cdot \frac{Recall \cdot Precision}{Recall + Precision}. \tag{4.6}$$

Optimal Structure

To select the optimal structure for our deep learning architecture, we first tested the sensitivity of the size of the input patches, then tested the sensitivity of the filter size at each convolutional layer. We randomly selected a subset of data points from all training examples, and tested with the same 1 million examples for each case.

Sensitivity of patch size:

Through our analysis, we concluded that the patch size should be odd and the patch should be symmetric extracted from the central voxel. In the seed expansion step,

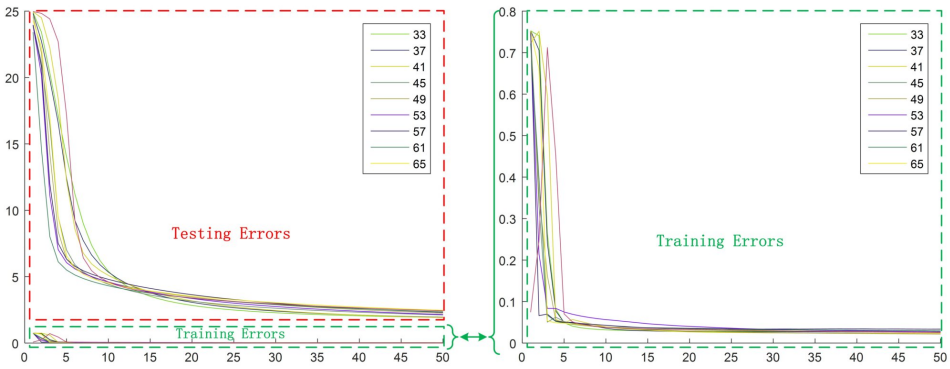


Fig. 4.17 Training and testing errors with different sizes of input patches; (a) training error and testing error are drawn in one figure; (b) enlargement of the training error shown in (a).

Table 4.3 Sensitivity of patch sizes.

Patch size x * y	Testing error %	Training time $M^{-1}Epoch^{-1}s$	Testing time $M^{-1}s$
33*33	2.7630	126.1592	14.3604
37*37	2.6377	174.0946	16.7188
41*41	**2.3622**	213.6202	19.7119
45*45	2.9483	269.2293	23.0471
49*49	2.7661	347.4515	26.7166
53*53	2.6945	414.2774	30.5176
57*57	3.3608	475.8913	34.8540
61*61	2.9506	547.0197	39.8501
65*65	2.5568	628.2490	44.8104
69*69	2.5428	708.9567	50.5970

$M^{-1}Epoch^{-1}s$: Seconds per million patches per epoch.
$M^{-1}s$: Seconds per million patches.

the TH-labelled neurones are modelled using a circle with averaging 33 pixels in diameter. The patch size was tested from the smallest size (*i.e.* 33*33 pixels) to the largest size (*i.e.* 69*69) at an interval of four pixels. In total, there were nine networks trained with patches extracted from the same set of training and testing points. We tested using the same structure in the 2D CNN model with seven layers, including an input layer, two convolutional layers (each with filter size 4*4), two sampling layers (each with filter size 2*2) and two fully connected layers. In Fig. 4.17, we show training and testing errors with different patch sizes. From our results, we note that training and testing errors had very small differences after 15 epochs. Further, after 30 epochs, all testing errors did not change too much at all.

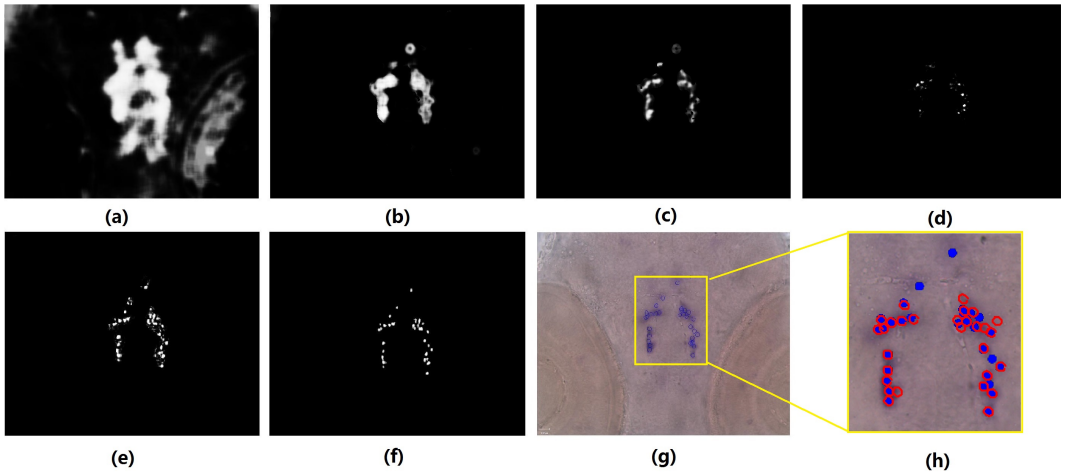


Fig. 4.18 Decision map of one test z-stack of images in each stage; (a-d) the central frame of the decision map processed by the first iteration of Model 1 and the second iteration of Model 1, Model 2 and Model 3; (e) the maximum projection of the final decision map processed by our cascade framework; (f) decision map after post-processing; (g) final detection and counting results, shown as blue circles and (h) comparison with human counting results, shown as red circles.

We measured testing error, training time in seconds per million patches in one epoch ($M^{-1}Epoch^{-1}s$), and testing time in seconds per million patches ($M^{-1}s$) after training a CNN structure for 50 epochs. Our results here were only marginally sensitive to patch size as shown in Table 4.3; the training and testing times gradually increase as patch sizes increased. Although there were small differences in training and testing errors between different sizes of input patches, for the sake of consistency, we had to select one patch size for subsequent training and testing. As a result, we selected patch size 41*41 to consistently train all deep learning architectures. We also used this patch size because a sufficient number of positive patches can be produced after seed expansion; further, the computational time was much lower than if we used a larger patch size.

Sensitivity of filter size

As noted above, we fixed the input patch size to be 41*41; further, we set the filter size in the max pooling layer to be 2*2 and in the fully connected layer to be 61*2 and 2*1. Given this, we tested different structures with different filter sizes to

Table 4.4 Sensitivity of filter sizes

First C layer x*y	Second C layer x*y	Testing error %
2*2	3*3	2.4474
	5*5	2.3711
	7*7	2.3828
4*4	2*2	2.3949
	4*4	2.3622
	6*6	2.3658
6*6	3*3	2.5723
	5*5	2.5988
	7*7	2.5785
C: Convolutional Layer.		

evaluate the sensitivity the filter size had on the convolutional layers. Although the pooling layer can reduce computational time and produce space invariant features, some information and important features may be lost if the pooling size is too large. Nonetheless, Table 4.4 shows that results are not very sensitive to filter sizes.

Other settings

We also tested the number of layers, such as 5, 7, 9, 11 and 13 layers, but there is no obvious differences. Theoretically, the more layers can have more parameters to produce better results, but it also needs more resources, such as more training data, time and powerful equipments. Due to the limitation of the number of training data and our experimental equipments, 7 layers for 2D-cnn and 9 layers for 3D-MC-CNN are selected based on its training time and producing results. Normally, more hidden layers lead to more trainable weights and lower errors. Therefore, compared with Model 2, we added one more convolutional layer to Model 3 to make the classification ability progressive; however, because of the computational speed and the limitations of physical memory, only seven slices were extracted in constructing each input patch, thereby resulting in a 41*41*7 patch size for each channel of Model 3. Note that we did not sample along the z-dimension in each sampling layer, as there are fewer pixels along the z-direction. Due to limitations of the optimisation method in the CUDA computing library, the feature maps for each layer can only be set to an integral multiple of 16. Therefore, in the 2D CNN, we set 16 feature maps in each convolution layer; further, we set the number of feature

maps to 32 in each convolutional layer of the 3D multi-channel CNN structure. As a result, optimal structures for Model 2 and Model 3 were selected, with structure details shown in Table 4.5 and 4.6.

Table 4.5 Structure details of two-dimensional deep architecture.

Layer	Type	# of Maps	Feature size	Filter size
0	Input	3	41*41	-
1	C	16	38*38	4*4
2	MP	16	19*19	2*2
3	C	16	16*16	4*4
4	MP	16	8*8	2*2
5	F	1	64*1	1*1
6	F	1	2*1	1*1

C: Convolutional Layer. MP: Max Pooling Layer.
F: Fully Connected Layer.

Table 4.6 Structure details of three-dimensional deep architecture.

Layer	Type	# of Maps	Feature size	Filter size
0	Input	1	41*41*7*3	-
1	C	32	38*38*3	4*4*3
2	MP	32	19*19*3	2*2*1
3	C	32	16*16*3	4*4*3
4	MP	32	8*8*3	2*2*1
5	C	32	6*6*3	3*3*3
6	MP	32	3*3*3	2*2*1
7	F	1	64*1*3	1*1
8	F	1	2*1*3	1*1
9	FU	1	2*1	1*1

C: Convolutional Layer. MP: Max Pooling Layer.
F: Fully-Connected Layer. FU: Fusion Layer.

Table 4.7 Comparison of different high-throughput open-source software

Software Name	*Preprocess Needed?*	*Automatic?*	*Method*	F_1	*Time/slice*
Proposed	Yes	Yes	Supervised detectors	0.8585	10s
Ilastik v.1.2.2 (detection) [14]	No	Interactive input	Supervised classifier	0.2650	68s
Ilastik v.1.2.2 (counting) [14]	Yes	Interactive input	Regression	-	231s
MINS v.1.3 [118]	No	Parameters tuning	Segmentation	0.1215	195s
3D-MLS v.1.02 [119]	Yes	Parameters tuning	Segmentation	0.1602	321
3D Counter v.2.0.1 [120]	Yes	Parameters tuning	Segmentation & Detection	0	20s

Comparison Methods & Evaluation Results

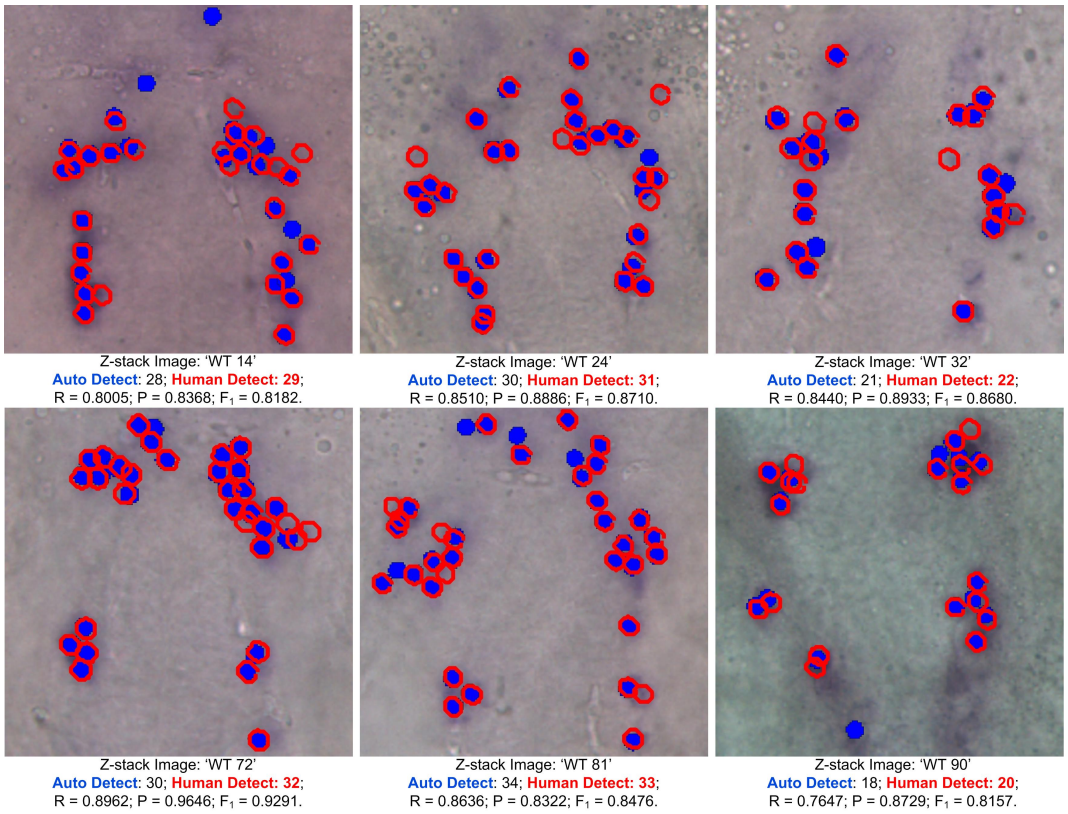


Fig. 4.19 Typical results in which 2D projections of six typical results are generated; here, blue filled circles represent results detected by our proposed method (*i.e.* **Auto Detect**), and red circles represent ground truth identified by human experts (*i.e.* **Human Detect**); in each image, evaluation scores were calculated, including *recall* (R), *precision* (P), and the F_1-score (F_1).

In Fig. 4.19, typical results are shown and evaluated. Many of the detections were consistent with the ground truths provided by human experts. In our previous work [10], we trained a region detector and a CNN (*i.e.* Model 1 + Model 2) using the same dataset. In [72], CNN with PS is applied to solve the class-imbalanced problem present in training deep neurone networks for document classification. In Table 4.7,

Table 4.8 Comparison of results on test stacks.

Method	Recall	Precision	F1
Proposed	**0.7996** ±0.0496	**0.8168**±0.0589	**0.8081**±0.0339
SVM-RGBHist + 3DMCCNN	0.7864±0.0542	0.7536±0.0595	0.7697±0.0408
SVM-RGBHist + 3DCNN	0.7745± 0.0653	0.7263±0.0502	0.7496±0.0388
SVM-RGBHist + CNN [10]	0.7692±0.0797	0.6452±0.0399	0.7018±0.0430
3DCNN [112] + PS	0.6578±0.0418	0.7269± 0.0295	0.690±0.0284
CNN + PS [72]	0.6292±0.0434	0.6057±0.0430	0.6172± 0.0307
CNN [31]	0.6071±0.0806	0.3542±0.0550	0.4474±0.0294
CNN (Grey)	0.4429±0.0518	0.3227±0.0614	0.3767±0.0396
SVM-HOG	0.5600±0.0662	0.2593±0.0436	0.3544±0.0423
SVM-RGBHist+RGBSift	0.4839±0.0993	0.1136±0.0508	0.1840±0.0650
SVM-RGBSift	0.3704±0.0974	0.0962±0.0338	0.1527±0.0654

3DMCCNN: Three-dimensional multi-channel convolutional neural network.

3DCNN: Three-dimensional convolutional neural network.

CNN: Convolutional neural network.

PS: Probabilistic sampling.

SVM: Support vector machine.

we tested some existing software and tools using our zebrafish dataset, and also recorded the running time for processing each slice.

In Table. 4.8, results using different combinations are shown, with relevant methods tested, including Refined CNN (Model 1 + Model 2) [10], 2D CNN (Model 2) with PS [72], 3D CNN [112], 2D CNN (Model 2) [31] and SVM (Model 1) with a HOG descriptor. Due to computational time, memory limitations and types of images, when using our dataset, it is not easy to apply other types of detection methods, such as cascade methods with low level features [78] and adaptive boosting (AdaBoost) based methods for detection [121].

4.6 Conclusion

In this chapter, we mainly focused on cell detection in 3D multi-channel light microscopy zebrafish images. By exploring different features with a basic SVM detector, important handcrafted features were found. By applying CNN, the automatic framework with learned high-level features was designed. To solve the severe class-imbalanced problem, a cascade dynamic framework was introduced. In the

framework, centres of cells were detected with three cascade detectors, including a SVM detector, a 2D CNN detector and a 3D multi-channel CNN. Compared with different methods, the proposed method has its own potential for cell detection in 3D multi-channel light microscopy images.

Chapter 5

Cell Counting by Regression in Fluorescent Microscopy Images

5.1 Introduction

Cell counting in images must be fast, robust and accurate. This work is motivated by the observation that, in PD disease research, compared to wild-type zebrafish controls, the mutant zebrafish model shows a reduction of approximately 25% of their dopaminergic neurones as early as three days post fertilisation [1]. To answer fundamental questions about disease pathogenesis, dopaminergic neurones must be counted with computer-aided intervention. However, by experimenting and searching the literature, there is no definite clue to indicate which method is the best; however, some methods may perform well under certain constraints, e,g, certain shape, fixed cell size, large background noise, overlapping or occlusion situations. In this chapter, we aim to design a cell counting method for two-dimensional fluorescence microscopy images that can operate robustly with few constraints.

The current problem with counting by detection is that it is very difficult to detect an individual cell due to several major challenges. One common issue is that light microscopy images contain background noise and out-of-focus light, especially in

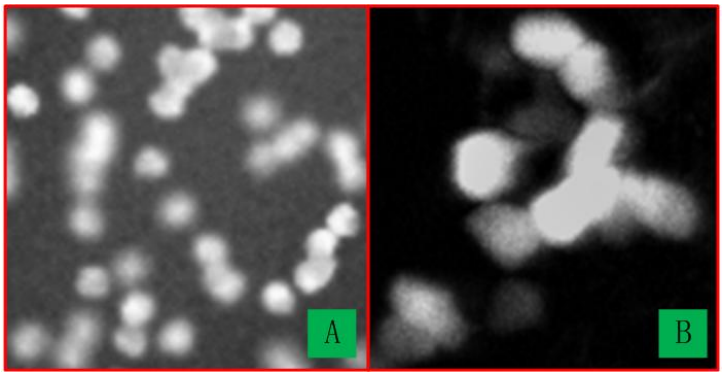


Fig. 5.1 Crop images from two different datasets: A: synthetic image dataset with overlapping cells [122]. B: our zebrafish WF fluorescent image dataset.

WF microscopy images. During the imaging process, in-focus information is mixed with out-of-focus information from regions outside the focus plane of interest. The out-of-focus light makes the cell boundary fuzzier and makes touching cells harder to detect.

Furthermore, cell images may vary widely, and Figure 5.1 shows two cropped fluorescent microscopy images from two different datasets. The sizes, shapes and contrast levels of the cells can differ significantly. Due to different imaging conditions, mitosis, overlapping and touching situations, cells may appear to have different intensity and different shape. Many researchers have recently focused on cell detection in microscopy images [65, 69, 10]. However, for our dataset, those methods cannot produce satisfied results. The occlusion and touching situations significantly limit the performance of detection methods.

Although many regression models with real-valued output variables have been designed for different regression tasks, in our situation, the output value for each cell cluster must be a non-negative integer, that represents an exact count. Poisson and negative binomial regression can be used to solve this problem. Standard Poisson regression was first analysed in a Bayesian interface [123]. They proposed a closed Gaussian approximation to the posterior weight distribution using log-gamma

distribution. Alternatively, [124] and [125] extended this approximation for crowded pedestrian counting with low-level features extracted from segmented regions.

Even the rounding process can produce integer result with normal regression method, but the error will become large after rounding. With the proposed method, the error function is minimized based on integer number, and the error rate will be reduced compared with the rounding process with normal regression method.

The RVM is a Bayesian sparse kernel technique [88] that brings a sparse solution to the kernel-based model [126]. The sparse model avoids the principle limitations of the kernel-based model and makes prediction performance much faster while maintaining comparable generalisation error [88]. The original RVM model defines a conditional distribution that follows Gaussian distribution. In this chapter, we propose a Sparse Bayesian Poisson Regression (SBPR) model based on Bayesian Poisson regression (BPR) and RVM regression. By changing the RVM model with Poisson distribution, we obtain improved versions BPR or RVM that solve their inherent limitations.

The methods in previous chapter is designed for 3D detection. The method in current chapter is designed for cell counting in 2D fluorescence microscopy image. In the 2D fluorescence microscopy, the neurones are gathering closer and it is not easy to detect severe overlapping and touching condition in 2D projection images. We also compared 2D CNN with the proposed method in this chapter, and the proposed cell counting by regression method is more suitable for processing this kind of 2D fluorescence microscopy images with overlapping and touching situations. Figure. 5.2 illustrates the workflow of our cell counting by regression method.

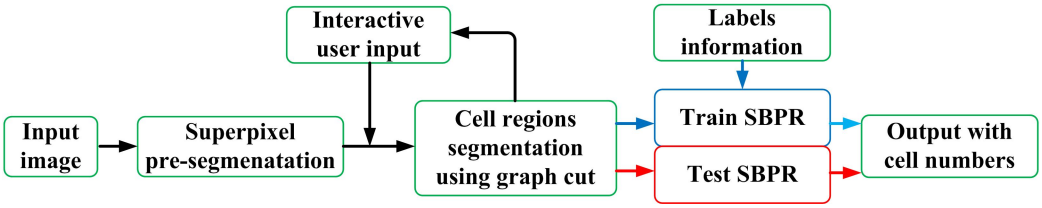


Fig. 5.2 Workflow of the cell counting by regression method.

5.2 Segmentation with Superpixel-based Graph Cut

5.2.1 Region Segmentation with Graph Cut

The cell segmentation problem can be posed as a binary graph cuts problem. Here, we briefly revisit the traditional graph cut segmentation framework. Assume that the given image is a directed graph $\mathscr{G} = (\mathscr{P}, \mathscr{E})$ with pixels $\mathscr{P}$ and edges $\mathscr{E}$. This graph consists of a neighbourhood system $\mathscr{N}$ that contains all unordered pairs $\{p,q\}$ of neighbouring pixels in $\mathscr{P}$. In this framework, the energy function is minimised to seek the optimal labelling $f \in \{0,1\}^{|P|}$ for each pixel $p \in \mathscr{P}$ in the image. The standard form of the energy function is as follows:

$$E(f) = \lambda \cdot \sum_{p \in \mathscr{P}} E_{data}(f_p) + \sum_{p,q \in \mathscr{N}: f_p \neq f_q} E_{smooth}(f_p, f_q), \tag{5.1}$$

where λ is a positive coefficient that specifies the relative importance of the data term E_{data} versus the smooth term E_{smooth}. $E_{data}(f_p)$ is the data term that evaluates the cost of assigning label f_p to pixel p:

$$E_{data}(f_p) = -log\,\Pr(p|f_p), \tag{5.2}$$

where $\Pr(p|f_p)$ is the probability that pixel p matches labels f_p. Here, $E_{smooth}(f_p, f_q)$ is a smooth term that measures the penalty for assigning neighbour pixels p and q to different labels:

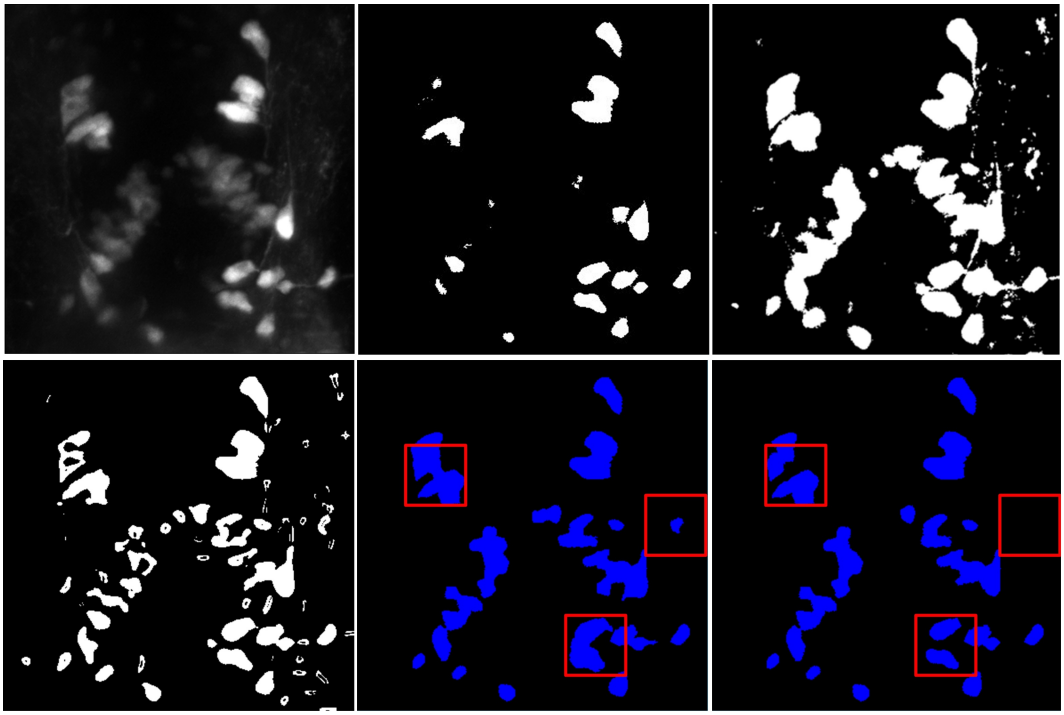


Fig. 5.3 Comparison of different segmentation methods. Left (first row): original image. Middle (first row): auto-thresholding with Otsu. Right (first row): region selection with Ilastik. Left (second row): pixel-based graph cut. Middle (second row): superpixel-based graph cut. Right (Second row) superpixel-based graph cut with learned probability term

$$E_{smooth}(f_p, f_q) = \begin{cases} e^{-\frac{\|x_p - x_q\|}{2\sigma^2}}, & if\ f_p \neq f_q \\ 0, & otherwise \end{cases} \tag{5.3}$$

where x_p and x_q are intensity values from pixels p and q.

The total energy $E(f)$ in equation (5.1) can be minimized using the Max-flow/Min-cut algorithm [129].

5.2.2 Superpixel-based Graph Cut

Pixel-wise prediction using a supervised classifier can be noisy, and predicting an image with mega-pixels also takes significant time.

Superpixel Generation

Several methods have been proposed to generate superpixels in image segmentation, and such methods have been proved useful in image segmentation tasks. Here, we use a simple linear iterative clustering algorithm (SLIC) [130] for superpixel generation. Compared to other superpixel methods, the SLIC algorithm has its own advantages to aggregate nearby pixels into superpixels in fluorescent images, e.g. it is simple to use, fast, memory efficient, has only one tunable parameter, and it has excellent boundary adherence.

In the SLIC algorithm, cluster centres are first initialised from sampling on a regular grid. Then, the initialised centres are moved toward the lowest gradient position in a certain distance. For each new centre, its best matching points are searched iteratively based on intensity, colour and spatial proximity from the neighbourhood to form temporary cluster. If new centres and previous centres are sufficiently close, then the iteration stops. Finally, post-processing is required to enforce connectivity. Additional details about superpixel generation using the SLIC method can be found in the literature [130].

Graph cut, which is primarily based on image intensity, is an unsupervised technique for image cuts and segmentation. To maintain segmentation consistency and use rich features from superpixels, a supervised SVM classifier is trained to predict the data term in equation (5.1). The SVM injects important information into the probability term $Pr(p|f_p)$.

Intensity is the main difference between cell regions and the surrounding background, intensity means, variances and histograms from the superpixel are extracted. It is important to include features that reflect the differences between cell and background regions. Therefore, features in neighbour superpixels are selected to predict the probability term.

5.2.3 Local Feature Extraction from Clusters

Those features are manfully selected, and those are most effective features for cell counting by regression.

Area Size

The number of pixels in the cluster.

Intensity

Mean value, variance and intensity histogram (6-bin) in each segmented cluster.

Perimeter

Number of pixels on each segment perimeter.

Perimeter-area ratio

The ratio between segment perimeter and area.

Internal Edge Feature

The total pixel count of edges within each cluster. Note that Canny edge detection is used to locate edges. Each pixel is weighted by the square root of its value in the density map.

Texture Feature

Energy: total sum-squared energy, $e_\theta = \sum_{i,j} p(i,j|\theta)^2$

Shape Convexity Feature

For most situations, the cell appears to be elliptical, which can be used as a prior for elliptical object segmentation. Since segmentation with a superpixel-based graph cut

can avoid holds in segmented regions, we only analyse the shape convexity along the region boundary. The number of convexity defects and the convexity ratio are recorded as a shape feature. Convexity is the ratio between the convex polygon perimeter and the perimeter of the shape itself.

$$Convexity = \frac{ConvexPerimeter}{Perimeter} \tag{5.4}$$

Finally, a feature vector is formed by concatenating all features into a vector, which is then used as input to the regression module.

5.3 Sparse Bayesian Poisson Regression

The cell number in each cluster must be non-negative integer $y \in \mathbb{Z}_+ = \{0,1,2,\cdots\}$, which should be the output of the counting method. However, typical regression methods are designed for problems with real-valued output variables. Bayesian treatment for a typical Poisson regression model addresses this limitation [123–125]. Here, we modify the original RVM [88] by modeling the output variable as a Poisson distribution, thereby resulting in SBPR.

5.3.1 Sparse Modelling

For a regression problem, given a set of example input-target pairs $\{\mathbf{x}_n, y_n\}_{n=1}^N$, the Poisson regression model between input $\mathbf{x}$ and scalar target y can be described as follows:

$$y \sim Poisson(\lambda(\mathbf{x})), \;\; \lambda(\mathbf{x}) = e^{f(\mathbf{x})} \tag{5.5}$$

where $\lambda(\mathbf{x})$ is the arrival rate (or mean of y) and $f(\mathbf{x})$ is the log of the arrival rate. We define the log of the arrival rate $f(\mathbf{x})$ following the assumption in the RVM model

[88], which typically assumes the following linearly weighted model for $f(\mathbf{x})$:

$$\begin{aligned} f(\mathbf{x}) &= \sum_{n=1}^{N} K(\mathbf{x}, \mathbf{x}_n) w_n, \\ &= \boldsymbol{\phi}(\mathbf{x})^T \mathbf{w} \end{aligned} \tag{5.6}$$

where $\mathbf{w} = (w_1, ..., w_N)^T$ is the weight vector of the linear model, $K(\mathbf{x}, \mathbf{x}_n)$ is a predetermined basis function centred on the training sample $\mathbf{x}_n$, and $\boldsymbol{\phi}(\mathbf{x}_n) = [K(\mathbf{x}_n, \mathbf{x}_1), K(\mathbf{x}_n, \mathbf{x}_2), ..., K(\mathbf{x}_n, \mathbf{x}_N)]^T$. Our goal is to determine a sparse solution for the weight vector $\mathbf{w}$ in equation (5.6).

The output variable y is Poisson distributed; thus, the likelihood of y given observation $\mathbf{x}$ is:

$$p(y|\mathbf{x}, \mathbf{w}) = \frac{1}{y!} e^{-\lambda(\mathbf{x})} \lambda(\mathbf{x})^y. \tag{5.7}$$

The zero-mean Gaussian prior distribution over the weight vector $\mathbf{w}$ is defined as:

$$p(\mathbf{w}|\boldsymbol{\alpha}) = \prod_{i=0}^{N} \mathcal{N}(w_i|0, \alpha_i^{-1}), \tag{5.8}$$

with $\boldsymbol{\alpha}$ a vector of N hyperparameters. We define $\mathbf{A} = \text{diag}(\alpha_1, ..., \alpha_N)$.

5.3.2 Bayesian Poisson Regression Inference

Having defined the prior, the posterior distribution over the weight $\mathbf{w}$ is given by:

$$p(\mathbf{w}|X, \mathbf{y}) = \frac{p(\mathbf{y}|X, \mathbf{w}) p(\mathbf{w})}{\int p(\mathbf{y}|X, \mathbf{w}) p(\mathbf{w}) d\mathbf{w}} \propto p(\mathbf{y}|X, \mathbf{w}) p(\mathbf{w}) \tag{5.9}$$

where $X = [\mathbf{x}_1, ..., \mathbf{x}_N]$ is the matrix of observed input vectors $\mathbf{x}_i$, and $\mathbf{y} = [y_1, ..., y_N]^T$ is the vector of the corresponding output y_i. In equation (5.9), we are forced to adopt some approximation due to the lack of conjugacy between the Poisson likelihood and the Gaussian prior. We follow a procedure that approximates a log-gamma distribution with a Gaussian distribution [123–125].

Given a Gamma random variable $\mu \sim Gamma(a,b)$,

$$p(\mu|a,b) = \frac{1}{\gamma(a)b^a}\mu^{a-1}e^{-\frac{\mu}{b}}. \tag{5.10}$$

The random variable $\nu = \log\mu$ has a log-Gamma distribution $\nu \sim LogGamma(a,b)$. For a large a, the log-gamma distribution is approximately Gaussian:

$$\begin{aligned} p(\nu|a,b) &= \frac{1}{\gamma(a)b^a}e^{\nu(a-1)}e^{-\frac{e^\nu}{b}} \\ &\approx \mathcal{N}(\nu|\log a + \log b, a^{-1}). \end{aligned} \tag{5.11}$$

From equation (5.7) (5.11), we can obtain:

$$\begin{aligned} & p(\mathbf{y}|X,\mathbf{w}) \\ &= \prod_{i=1}^{N}\left[\frac{e^{f(\mathbf{x}_i)(y_i+c)}e^{-e^{f(\mathbf{x}_i)}}}{\Gamma(y_i+c)}\right]e^{-cf(\mathbf{x}_i)}\frac{\Gamma(y_i+c)}{y_i!} \\ &\approx \prod_{i=1}^{N}\left[\mathcal{N}(f(\mathbf{x}_i)|\log(y_i+c),(y_i+c)^{-1})\right]e^{-cf(\mathbf{x}_i)}\frac{\Gamma(y_i+c)}{y_i!}, \end{aligned} \tag{5.12}$$

where f is log-Gamma distributed with $f \sim \text{LogGamma}(y+c,1)$, which can be approximated by a Gaussian $\mathcal{N}(f|\log(y+c),(y+c)^{-1})$ [123]. As $\lambda(\mathbf{x}) = e^{f(\mathbf{x})}$, λ is approximately Gamma distributed.

Substituting into equation (5.9),

$$\begin{aligned} & \log p(\mathbf{w}|X,\mathbf{y}) \propto \log p(\mathbf{y}|X,\mathbf{w}) + \log p(\mathbf{w}) \\ &\approx -\frac{1}{2}\|\mathbf{\Phi}^T\mathbf{w} - \log(\mathbf{y}+c)\|^2_{\mathbf{\Sigma}_y} - c\mathbf{1}^T\mathbf{\Phi}^T\mathbf{w} - \frac{1}{2}\|\mathbf{w}\|^2_{\mathbf{A}^{-1}} \\ &\propto -\frac{1}{2}\|(\mathbf{w}-\hat{\boldsymbol{\mu}}_\mathbf{w})\|^2_{\hat{\mathbf{\Sigma}}_\mathbf{w}} + \frac{1}{2}\mathbf{m}^T\hat{\mathbf{\Sigma}}_\mathbf{w}^T\mathbf{m} \end{aligned} \tag{5.13}$$

Then, the posterior distribution is approximately Gaussian

$$p(\mathbf{w}|X,\mathbf{y}) \approx \mathcal{N}(\hat{\boldsymbol{\mu}}_\mathbf{w}, \hat{\mathbf{\Sigma}}_\mathbf{w}) \tag{5.14}$$

with mean and covariance

$$\begin{aligned}\hat{\boldsymbol{\mu}}_{\mathbf{w}} &= (\boldsymbol{\Phi}\boldsymbol{\Sigma}_y^{-1}\boldsymbol{\Phi}^T + \mathbf{A})^{-1}\mathbf{m},\\ \hat{\boldsymbol{\Sigma}}_{\mathbf{w}} &= (\boldsymbol{\Phi}\boldsymbol{\Sigma}_y^{-1}\boldsymbol{\Phi}^T + \mathbf{A})^{-1},\end{aligned} \tag{5.15}$$

where $\boldsymbol{\Sigma}_y = \text{diag}[\frac{1}{y_1+c}...\frac{1}{y_N+c}]$, $\mathbf{m} = \boldsymbol{\Phi}\boldsymbol{\Sigma}_y^{-1}(\log(\mathbf{y}+c) - c\boldsymbol{\Sigma}_y\mathbf{1})$, c is a constant ($c \geq 0$), $\boldsymbol{\Phi}$ is the $N \times N$ symmetrical matrix with $\boldsymbol{\Phi} = [\boldsymbol{\phi}(\mathbf{x}_1), ..., \boldsymbol{\phi}(\mathbf{x}_N)]$, wherein $\boldsymbol{\phi}(\mathbf{x}_n) = [\phi_1(\mathbf{x}_n), \phi_2(\mathbf{x}_n), ..., \phi_N(\mathbf{x}_n)]^T$ with $\phi_i(\mathbf{x}_n) = k(\mathbf{x}_n, \mathbf{x}_i)$.

Given a new observation $\mathbf{x}_\star$, the predicted log-arrival rate $f_\star = \boldsymbol{\phi}(\mathbf{x}_\star)^T\mathbf{w}$ will approximately follow a Gaussian distribution with $p(f_\star|\mathbf{x}_\star, X, \mathbf{y}) \approx \mathcal{N}(\mathbf{f}_\star|\hat{\boldsymbol{\mu}}_f, \hat{\sigma_f^2})$, where

$$\begin{aligned}\hat{\boldsymbol{\mu}}_f &= \phi(\mathbf{x}_\star)^T\hat{\boldsymbol{\mu}}_{\mathbf{w}}\\ \hat{\sigma_f^2} &= \boldsymbol{\phi}(\mathbf{x}_\star)^T\hat{\boldsymbol{\Sigma}}_{\mathbf{w}}\boldsymbol{\phi}(\mathbf{x}_\star)\end{aligned} \tag{5.16}$$

The new prediction distribution for $y_\star$ is

$$p(y_\star|\mathbf{x}_\star, X, \mathbf{y}) = \int_0^\infty \underbrace{p(y_\star|f_\star)}_{\text{Poisson}}\underbrace{p(\lambda|\mathbf{x}_\star, X, \mathbf{y})}_{\text{Gamma}}d\lambda_\star. \tag{5.17}$$

The Gamma distribution is conjugate prior for Poisson likelihood, thus, the predictive distribution of $y_\star$ can be approximated as a negative binomial distribution [124, 125]:

$$y_\star|\mathbf{x}_\star, X, \mathbf{y} \sim NegBin(e^{\hat{\boldsymbol{\mu}}_f}, \hat{\sigma}_f^2) \tag{5.18}$$

5.3.3 Optimising Hyperparameters

Updating Hyperparameters

The optimal values $\hat{\boldsymbol{\alpha}}$ of the hyperparameters, given a training set $\{X, \mathbf{y}\}$, can be found by *type-II maximum likelihood* [127], which maximises the marginal likelihood

as follows:

$$\hat{\boldsymbol{\alpha}} = \arg\max_{\boldsymbol{\alpha}} \log p(\mathbf{y}, X|\boldsymbol{\alpha}), \tag{5.19}$$

where

$$\begin{aligned}
p(\mathbf{y}, X|\boldsymbol{\alpha}) &= p(\mathbf{y}|X, \boldsymbol{\alpha}) p(X) \propto p(\mathbf{y}|X, \boldsymbol{\alpha}) \\
&= \int p(\mathbf{y}, \mathbf{w}|X, \boldsymbol{\alpha}) d\mathbf{w} \\
&= \int p(\mathbf{y}, X|\mathbf{w}) p(\mathbf{w}|\boldsymbol{\alpha}) d\mathbf{w} \\
&\approx \int \prod_{i=1}^{N} \left[\mathcal{N}(f(\mathbf{x}_i)|\log(y_i + c), (y_i + c)^{-1}) \right] \\
&\quad \cdot e^{-cf(\mathbf{x}_i)} \frac{\Gamma(y_i + c)}{y_i!} \prod_{i=0}^{N} \left[\mathcal{N}(w_i|0, \alpha_i^{-1}) \right] d\mathbf{w} \\
&= \int \frac{e^{-\frac{1}{2}\|\boldsymbol{\Phi}\mathbf{w} - \log(\mathbf{y}+c)\|^2_{\boldsymbol{\Sigma}_y} - c\mathbf{1}^T \boldsymbol{\Phi}\mathbf{w}}}{(2\pi)^{\frac{N}{2}} |\boldsymbol{\Sigma}_y|^{\frac{1}{2}}} \\
&\quad \cdot \frac{e^{-\frac{1}{2}\|\mathbf{w}\|^2_{\mathbf{A}^{-1}}}}{(2\pi)^{\frac{N+1}{2}} |\mathbf{A}^{-1}|^{\frac{1}{2}}} d\mathbf{w} \prod_{i=1}^{N} \frac{\Gamma(y_i + c)}{y_i!}
\end{aligned} \tag{5.20}$$

Then, the marginal log-likelihood can be approximated as

$$\begin{aligned}
&\log p(\mathbf{y}, X|\boldsymbol{\alpha}) \\
&\propto \int -\frac{1}{2}\|\mathbf{w} - \hat{\boldsymbol{\mu}}_{\mathbf{w}}\|^2_{\hat{\boldsymbol{\Sigma}}_{\mathbf{w}}} d\mathbf{w} + \frac{1}{2}\mathbf{m}^T \hat{\boldsymbol{\Sigma}}^T_{\mathbf{w}} \mathbf{m} - \frac{1}{2}\log|\mathbf{A}^{-1}|
\end{aligned} \tag{5.21}$$

The marginal likelihood can be approximate as:

$$p(\mathbf{y}|X, \boldsymbol{\alpha}) \propto \frac{|\hat{\boldsymbol{\Sigma}}_{\mathbf{w}}|^{\frac{1}{2}}}{|\mathbf{A}^{-1}|^{\frac{1}{2}}} e^{\frac{1}{2}\mathbf{m}^T \hat{\boldsymbol{\Sigma}}^T_{\mathbf{w}} \mathbf{m}} \tag{5.22}$$

Then, we can get the objective function

$$\mathscr{L}(\boldsymbol{\alpha}) = \log p(\mathbf{y}, X|\boldsymbol{\alpha}) \propto \frac{1}{2}(\log|\hat{\boldsymbol{\Sigma}}_{\mathbf{w}}| - \log|\mathbf{A}^{-1}| + \mathbf{m}^T \hat{\boldsymbol{\Sigma}}^T_{\mathbf{w}} \mathbf{m}). \tag{5.23}$$

By differentiation for $\boldsymbol{\alpha}$ and equating to zero, it gives

$$\begin{aligned}\frac{\partial \mathscr{L}(\boldsymbol{\alpha})}{\partial \alpha_i} &\propto \frac{\partial}{\partial \alpha_i}(\log|\hat{\boldsymbol{\Sigma}}_{\mathbf{w}}| - \log|\mathbf{A}^{-1}| + \mathbf{m}^T\hat{\boldsymbol{\Sigma}}_{\mathbf{w}}\mathbf{m}) \\ &= -\mathrm{Tr}(\hat{\boldsymbol{\Sigma}}_{\mathbf{w}}\frac{\partial(\mathbf{A})}{\partial \alpha_i}) + \mathrm{Tr}(\mathbf{A}^{-1}\frac{\partial \mathbf{A}}{\partial \alpha_i}) \\ &\quad - \mathbf{m}^T\hat{\boldsymbol{\Sigma}}_{\mathbf{w}}\frac{\partial(\mathbf{A})}{\partial \alpha_i}\hat{\boldsymbol{\Sigma}}_{\mathbf{w}}\mathbf{m} \\ &= -\hat{\boldsymbol{\Sigma}}_{\mathbf{w}_{ii}} + \frac{1}{\alpha_i} - \hat{\boldsymbol{\mu}}_{\mathbf{w}_i} = 0\end{aligned} \tag{5.24}$$

where $\mathbf{B} = \boldsymbol{\Phi}\boldsymbol{\Sigma}_{\mathbf{y}}^{-1}\boldsymbol{\Phi}^T$, $\hat{\boldsymbol{\Sigma}}_{\mathbf{w}_{ii}}$ is the i-th diagonal element of the posterior weight covariance, and $\hat{\boldsymbol{\mu}}^2_{\mathbf{w}_i}$ is the i-th posterior mean weight from equation (5.15). Then, we can obtain the new α_i:

$$\alpha_i^{new} = \frac{1}{\hat{\boldsymbol{\Sigma}}_{\mathbf{w}_{ii}} + \hat{\boldsymbol{\mu}}^2_{\mathbf{w}_i}} \tag{5.25}$$

By giving an initial value for $\boldsymbol{\alpha}$, the mean and covariance of the posterior can be evaluated using equation (5.15). Then $\boldsymbol{\alpha}$ can be re-estimated using equation (5.25), and the posterior mean and covariance can be re-estimated using equation (5.15), until a suitable convergence criterion is satisfied. When hyperparameter α_i is driven to large values, the weight parameters w_i has posterior distribution with zero mean and variance. Then, the corresponding basis function $\phi_i(\mathbf{x})$ is removed to make the entire model sparse. Although this update method is guaranteed to locally maximize the marginal likelihood, the process is inefficient. Based on [128], faster marginal likelihood maximization is used to estimate hyperparameters in SBPR.

Faster marginal likelihood maximization

From equation (5.23), the marginal log-likelihood can be approximated as follows:

$$\begin{aligned}
\mathscr{L}(\alpha) &\propto \frac{1}{2}(\log|\hat{\boldsymbol{\Sigma}}_{\mathbf{w}}| - \log|\mathbf{A}^{-1}| + \mathbf{m}^T\hat{\boldsymbol{\Sigma}}_{\mathbf{w}}^T\mathbf{m}) \\
&\propto -\frac{1}{2}\log|\,\boldsymbol{\Phi}\boldsymbol{\Sigma}_y^{-1}\boldsymbol{\Phi}^T + \mathbf{A}| + \frac{1}{2}\log|\mathbf{A}| + \frac{1}{2}(\mathbf{t}^T\boldsymbol{\Sigma}_y^{-1}\boldsymbol{\Phi}^T\hat{\boldsymbol{\Sigma}}_{\mathbf{w}}^T\boldsymbol{\Phi}\boldsymbol{\Sigma}_y^{-1}\mathbf{t} - \mathbf{t}^T\boldsymbol{\Sigma}_y^{-1}\mathbf{t}) \\
&= -\frac{1}{2}\log(|A||\boldsymbol{\Sigma}_y^{-1}||\boldsymbol{\Sigma}_y + \boldsymbol{\Phi}^T A^{-1}\boldsymbol{\Phi}|) + \frac{1}{2}\log|\mathbf{A}| - \frac{1}{2}\mathbf{t}^T(\boldsymbol{\Sigma}_y + \boldsymbol{\Phi}^T A^{-1}\boldsymbol{\Phi})^{-1}\mathbf{t} \\
&= \frac{1}{2}\log|\boldsymbol{\Sigma}_y| - \frac{1}{2}\log|\boldsymbol{\Sigma}_y + \boldsymbol{\Phi}^T A^{-1}\boldsymbol{\Phi}| - \frac{1}{2}\mathbf{t}^T(\boldsymbol{\Sigma}_y + \boldsymbol{\Phi}^T A^{-1}\boldsymbol{\Phi})^{-1}\mathbf{t} \\
&\propto -\frac{1}{2}\log|\mathbf{C}| - \frac{1}{2}\mathbf{t}^T\mathbf{C}^{-1}\mathbf{t},
\end{aligned} \tag{5.26}$$

where

$$\begin{aligned}
\mathbf{C} &= \boldsymbol{\Sigma}_y + \boldsymbol{\Phi}^T A^{-1}\boldsymbol{\Phi}, \\
\mathbf{t} &= \log(\mathbf{y} + c) - c\boldsymbol{\Sigma}_y\mathbf{1}.
\end{aligned} \tag{5.27}$$

By pulling out the contribution from α_i [128], $\mathbf{C}$ in equation (5.27) can be decomposed as:

$$\begin{aligned}
\mathbf{C} &= \boldsymbol{\Sigma}_y + \sum_{j\neq i}\alpha_j^{-1}\boldsymbol{\Phi}_j\boldsymbol{\Phi}_j^T + \alpha_i^{-1}\boldsymbol{\Phi}_i\boldsymbol{\Phi}_i^T \\
&= \mathbf{C}_{-i} + \alpha_i^{-1}\boldsymbol{\Phi}_i\boldsymbol{\Phi}_i^T
\end{aligned} \tag{5.28}$$

where $\boldsymbol{\Phi}_i$ denotes the i^{th} column of $\boldsymbol{\Phi}$ and $\mathbf{C}_{-i}$ denotes the matrix $\mathbf{C}$ with the contribution from basis function i removed. We can then express the marginal likelihood with basis function $\boldsymbol{\Phi}_i$ omitted as follows:

$$\begin{aligned}
\mathscr{L}(\boldsymbol{\alpha}) &\propto -\frac{1}{2}\log|\mathbf{C}| + \boldsymbol{t}^T\mathbf{C}^{-1}\boldsymbol{t} \\
&= -\frac{1}{2}\log(|\mathbf{C}_{-i}||1 + \alpha_i^{-1}\boldsymbol{\Phi}_i^T\mathbf{C}_{-i}^{-1}\boldsymbol{\Phi}_i|) - \frac{1}{2}\mathbf{t}^T\mathbf{C}_{-i}^{-1}\mathbf{t} \\
&\quad + \frac{1}{2}\mathbf{t}^T\frac{\mathbf{C}_{-i}^{-1}\boldsymbol{\Phi}_i\boldsymbol{\Phi}_i^T\mathbf{C}_{-i}^{-1}}{\alpha_i + \boldsymbol{\Phi}_i^T\mathbf{C}_{-i}^{-1}\boldsymbol{\Phi}_i}\mathbf{t} \\
&= \mathscr{L}(\alpha_{-i}) + \xi(\alpha_i),
\end{aligned} \tag{5.29}$$

where $\mathscr{L}(\alpha_{-i})$ is the marginal log-likelihood with basis function $\mathbf{\Phi}_i$ omitted and $\xi(\alpha_i)$ is defined by:

$$\xi(\alpha_i) = \frac{1}{2}\left(\log\frac{\alpha_i}{\alpha_i + s_i} + \frac{q_i^2}{\alpha_i + s_i}\right) \tag{5.30}$$

where

$$s_i = \mathbf{\Phi}_i^T \mathbf{C}_i^{-1} \mathbf{\Phi}_i, \quad q_i = \mathbf{\Phi}_i^T \mathbf{C}_i^{-1} \mathbf{t}. \tag{5.31}$$

To find the maximum of function $\boldsymbol{L}(\boldsymbol{\alpha})$ with respect to α_i, the first derivative of the function is set to zero.

$$\frac{\partial \mathscr{L}(\boldsymbol{\alpha})}{\partial \alpha_i} = \frac{1}{\alpha_i} - \frac{1}{\alpha_i + s_i} - \frac{q^2}{(\alpha_i + s_i)^2} = 0. \tag{5.32}$$

This results in several stationary points by solving equation (5.32). Note that there are two possible solutions. If $q_i^2 < s_i$, then $\alpha_i = \infty$. If $q_i^2 > s_i$, then α_i can be obtained from equation

$$\alpha_i = \frac{s_i^2}{q_i^2 - s_i}. \tag{5.33}$$

In practice, it is convenient to use matrix identity equations that

$$s_i = \frac{\alpha_i Q_i}{\alpha_i - S_i}, \quad q_i = \frac{\alpha_i S_i}{\alpha_i - S_i}, \tag{5.34}$$

where

$$\begin{aligned} S_i &= \mathbf{\Phi}_i^T \mathbf{\Sigma}_y^{-1} \mathbf{\Phi}_i - \mathbf{\Phi}_i^T \mathbf{\Sigma}_y^{-1} \mathbf{\Phi} \hat{\mathbf{\Sigma}}_w \mathbf{\Phi}^T \mathbf{\Sigma}_{\mathbf{y}}^{-1} \mathbf{\Phi}_i, \\ Q_i &= \mathbf{\Phi}_i^T \mathbf{\Sigma}_y^{-1} \mathbf{t} - \mathbf{\Phi}_i^T \mathbf{\Sigma}_y^{-1} \mathbf{\Phi} \hat{\mathbf{\Sigma}}_w \mathbf{\Phi}^T \mathbf{\Sigma}_y^{-1} \mathbf{t}. \end{aligned} \tag{5.35}$$

This practical algorithm provides significant speed advantage when updating hyperparameters [128] and determines whether each candidate basis vector should

be included in the model. The entire process of the SBPR algorithm is described in Algorithm 4.

Algorithm 4 Sparse Bayesian Poisson regression

Initializations $\boldsymbol{\alpha} = \infty$
Iteration
1) Compute $\hat{\boldsymbol{\mu}}_{\mathbf{w}}$ and $\hat{\boldsymbol{\Sigma}}_{\mathbf{w}}$ using equation (5.15).
2) Compute s_i and q_i using equation (5.34) for all basis functions.
3) Find a candidate basis function Φ_i that can yield the maximum increase in $\mathscr{L}(\boldsymbol{\alpha})$ (equation (5.29)).
4) If $q_i^2 > s_i$ and $\alpha_i < \infty$, Φ_i will already be in the model. Then, update α_i using equation (5.33).
5) If $q_i^2 > s_i$ and $\alpha_i = \infty$, then add Φ_i to the model and evaluate α_i using equation (5.33).
6) If $q_i^2 \leqslant s_i$ and $\alpha_i < \infty$, then remove Φ_i from the model and set $\alpha_i = \infty$.
7) If converged, terminate; otherwise return to 1).
Output $\hat{\boldsymbol{\mu}}_{\mathbf{w}}$, $\hat{\boldsymbol{\Sigma}}_{\mathbf{w}}$ and $\boldsymbol{\alpha}$

5.4 Experiments and Evaluation

The proposed cell counting approach was tested using two different datasets.

5.4.1 Cell Counting Datasets

Synthetic Bacterial Cells Dataset

The first experiment was performed using a synthetic bacterial cell fluorescence microscopy image dataset [87] generated using [122]. This dataset contains 200 images with an average of 174 ± 64 cells per image.

Real Dataset

The real dataset is a zebrafish image dataset that contains 40 images with an average of 38 ± 7 cells per image obtained from 40 different zebrafish embryos scanned using a spinning disc confocal microscope. The objective has 20 times magnification and

a numerical aperture of 0.8. The resolution is 2.9004 pixels per micron. The size of the images is 1000*1000.

Table 5.1 Comparison result of relevant methods for zebrafish dataset with fixed size training vectors. The 'Vector' column shows how many feature vectors are used for training and the 'Used' column is the number of relevant feature vectors used for testing. The root-mean-square-error for vectors (RMSE/Vector), and mean-absolute-error for images (MAE/Image) were evaluated using 100 test images.

Methods	**#Vector**	**#Used**	**RMSE/Vector**	**MAE/Image**
SVR	2458	2458	0.51	4.2
GPR	2458	2458	0.39	4.1
BPR	2458	2458	0.38	3.9
RVM	2458	1038	0.36	3.6
SBPR	2458	256	0.33	3.4

Table 5.2 MAE for cell counting with the test set from synthetic cell dataset. Several state-of-the-art cell counting methods were evaluated with different numbers of training images. In the last six columns, N is the number of images used in the training set. Standard deviations correspond to five different draws of training image sets.

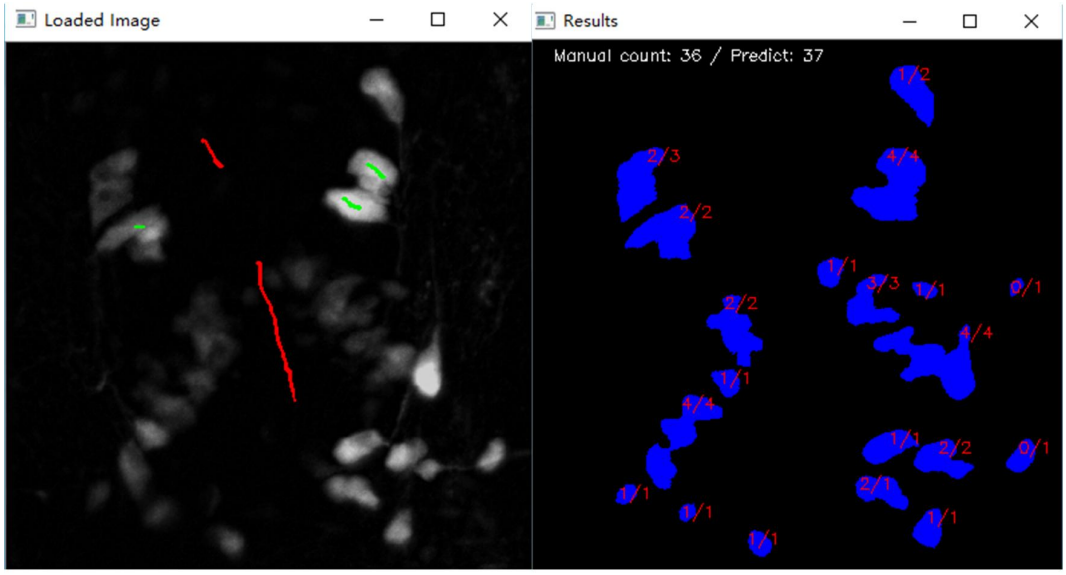
Methods	**N = 1**	**N = 2**	**N = 4**	**N = 8**	**N = 16**	**N = 32**
Image Level Ridge Regression [87]	60.4±16.5	38.7±17.0	18.6±5.0	10.4±2.5	6.0±0.8	5.2±0.3
Ilastik [86]	20.9±2.6	12.4±1.5	11.0±0.9	8.2±0.7	7.5±0.5	6.3±0.3
SVM + SIFT Detection [87]	20.8±3.8	20.1±5.5	15.7±2.0	15.0±4.1	11.8±3.1	12.0±0.8
Density MESA [87]	9.5 ± 6.1	6.3±1.2	4.9±0.6	4.9±0.7	3.8±0.2	3.5±0.2
Fully Convolutional Regression Networks [85]	N/A	N/A	N/A	4.1±0.5	3.7±0.3	3.3±0.2
Ridge-Regression (No smooth)[84]	13.8 ± 3.6	11.3±3.1	10.6±2.3	9.9±0.7	10.6±1.1	10.2±0.4
Ridge-Regression [84]	9.6±5.9	6.4±0.7	5.5±0.8	4.5±0.6	3.8±0.3	3.5±0.1
SBPR (Proposed)	9.0±3.7	5.6±1.0	4.3±0.7	4.1±0.5	3.6±0.3	3.3±0.1

5.4.2 Experimental Setup

Each dataset was split into training sets and testing sets. For the synthetic bacterial cell dataset, we followed the experimental setting in literature [87]. The entire dataset was divided into two groups with 100 images in each group for training and testing. Different training sizes can affect the final result significantly; thus, different numbers of training images were tested. In the training procedure, between 1-32 images were selected randomly from the training set, and the procedure was repeated for five different sets.

Five-fold cross-validation was used to evaluate the proposed method with the real zebrafish cell dataset. The dataset contains two different types of zebrafish embryos, *i.e.* wild-type and PINK1 type zebrafish, and each type has 20 images. Typically, the number of dopaminergic neurones in the PINK1-type should be less than that of the wild-type zebrafish. Even though there are only a few number of images in this dataset, an additional experiment was performed to determine whether this type of reduction can be detected between two different types of images. We manual labelled centre pixels of the cells in each image.

Accuracy was evaluated by $RMSE = \frac{1}{M}\sum_{i=1}^{M}(\hat{c}_i - c_i)^2$ and $MAE = \frac{1}{N}\sum_{i=1}^{N}|\hat{c}_i - c_i|$, where c_i and $\hat{c}_i$ are the true and estimated counts, M is the number of vectors and N is the number of images. Note that performance is strongly affected by the selected features, different subsets of features were tested, including area features $\mathscr{F}_a$, intensity-based features $\mathscr{F}_i$, parameter-based features $\mathscr{F}_p$, edge-based features $\mathscr{F}_e$, texture features $\mathscr{F}_t$, shape-based features $\mathscr{F}_s$, area and shape feature $\mathscr{F}_{as}$, and a full set of features $\mathscr{F}_{full}$.



(a) Input image with user input (b) Results

Fig. 5.4 The interactive interface.

The Figure. 5.4 illustrates the interface of the interactive cell counting software. Users need to draw labels in two different colors on the loaded image, for examples, drawing red lines on the background and green line on the cell regions. Then, the cell region will be automatic segmented based on user inputs with superpixel based graph cut method. For each segmentation region, the proposed method will estimate the number of cells in each isolated blue region.

5.4.3 Results with Synthetic Bacterial Cell Dataset

Comparison with Relevant Methods

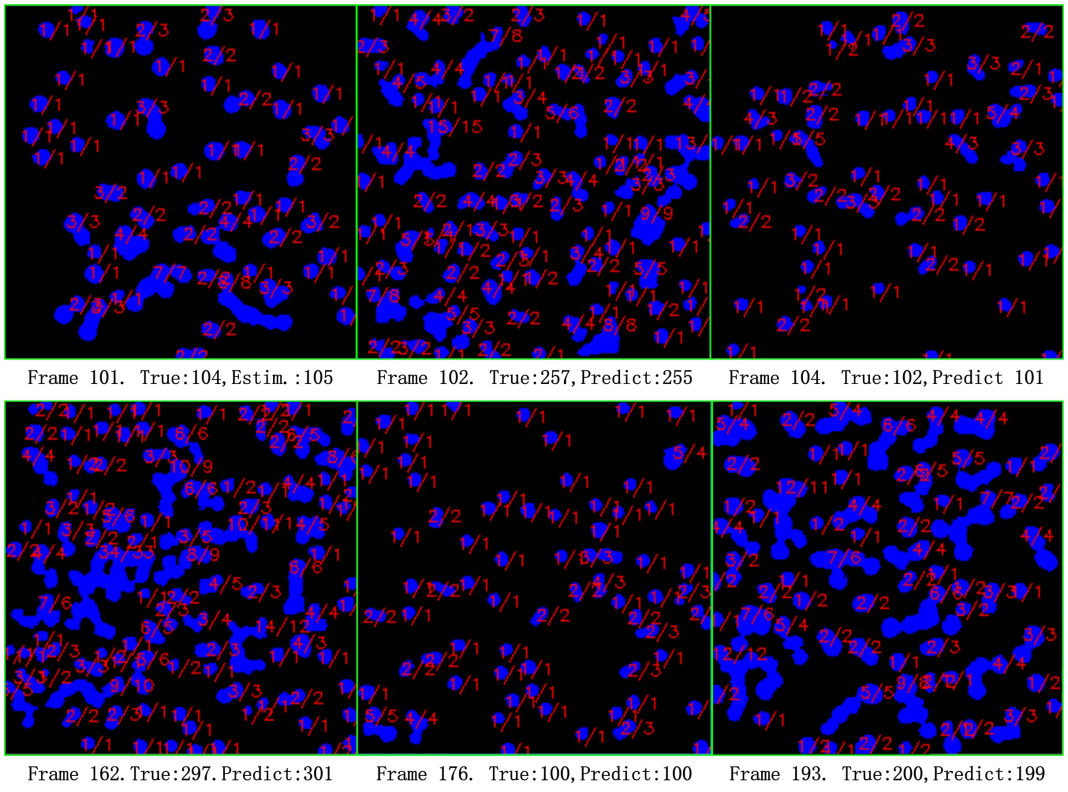


Fig. 5.5 Typical results for synthetic dataset.

As shown in Table 5.1, the proposed and relevance methods were evaluated using the synthetic dataset. The feature vectors and corresponding count numbers were collected for training and testing. In total, 2458 feature vectors from the first 32 images in the training set were used for training, and 7179 feature vectors were

collected from all 100 test images. RMSE for vectors and MAE for images were calculated.

Comparison with Other Methods

Table 5.2 shows a comparison with previous state-of-the-art methods with the synthetic cell dataset. Lempitsky and Zisserman [87] introduced a counting-by-regression method by optimising loss based on the MESA distance. Arteta et al. [84] first enabled interactive counting with ridge-regression, and Fiaschi and Nair [86] used an ensemble of randomised regression trees to estimate the object density map. An open source implementation is provided in the Ilastik framework [14]. Three different features (intensity, edge, and texture) with fixed sizes around dot annotations can be selected to train a random forest. However, this framework only enables users to count dot-like objects with very similar appearance. To ensure that the real number estimates are close to the true count, users must manually label the centres of most objects in the images to collect sufficient training data. Xie et al. [85] applied CNNs to regress cell spatial density across an image. Although a deep learning technique has been applied to cell counting, a final detection process is required to find the local maxima in the density map as the locations of cells. There are some drawbacks to find local maxima in a density map with noise, such as smooth kernel parameters. In addition, thresholding should be selected manually. As a result, several maxima may be found in a single cell region, and the overlapping situation can reduce accuracy significantly. The proposed method considerably outperforms previously proposed state-of-the-art approaches for all training set sizes.

Figure 5.5 shows five typical results selected from 100 test images. The blue regions are cell regions segmented by the superpixel-based interactive segmentation method. In each isolated region, the ground truth count number and the predicted

integer number are shown in red and separated by '/'. As can be seen, the predicted results are very close to the ground truth numbers.

5.4.4 Results with Zebrafish Dataset

We tested the proposed method with different segmentation methods, including the 'Otsu' auto-segmentation method [116], pixel-based graph cut [131], superpixel-based graph cut and our modified superpixel-based graph cut. The results shown in table 5.3 demonstrate that the proposed method can generate more precise results than other methods, such as pixel-based graph cut and auto thresholding segmentation method. It indicates that, for this real dataset, area, perimeter and shape features are more useful compared to other features.

Table 5.3 RMSE and MAE comparison with different feature vectors

SBPR	RMSE			MAE		
Features	Auto	Pixel-GC	SP-GC	Auto	Pixel-GC	SP-GC
F_a	1.56	1.12	0.98	10.5	8.5	6.8
F_i	2.13	1.45	1.23	23.6	13.6	10.1
F_p	1.92	1.78	1.04	15.3	8.3	7.3
F_t	1.81	1.49	1.36	19.4	13.4	12.3
F_s	1.76	1.34	1.12	16.5	11.5	9.6
F_as	1.45	0.97	0.72	8.3	8.3	5.2
F_full	1.32	0.78	**0.54**	9.7	5.2	**2.3**

Table 5.4 RMSE and MAE comparison of different methods with real dataset.

Methods	RMSE	MAE
Ilastik [14]	-	50.6±10.7
MINS [118]	-	5.1±1.3
CellDetect [132]	-	12.5±5.6
GPR-rr	0.65	3.1±0.2
BPR	0.58	2.7±0.1
SBPR	0.54	**2.3**±**0.1**

RMSE and MAE comparison of different methods with real dataset is shown in Table 5.4. The results demonstrate the proposed method can generate smallest error by calculating RMSE and MAE with different software and relevance methods.

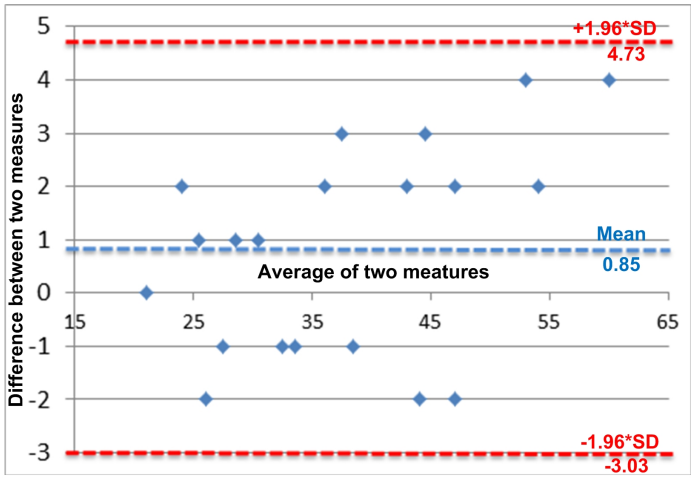


Fig. 5.6 Bland-Altman plot of counts by two experts.

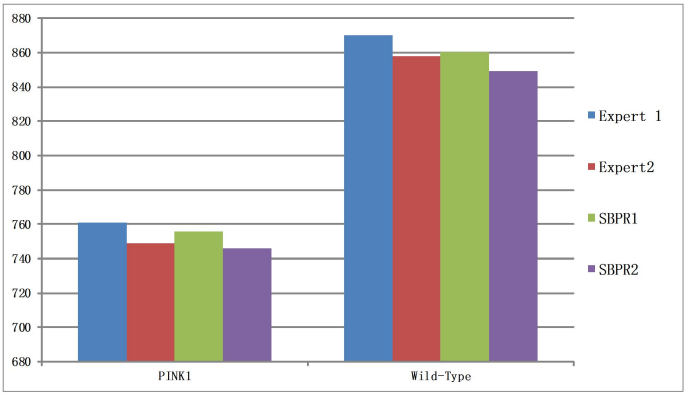


Fig. 5.7 Total count number of two types (PINK1 and wild-type). Blue and red bars are counts by two different experts, and green and purple bars are counts by proposed method trained with a ground truth counted by a corresponding expert.

Figure 5.6 shows a Bland-Altman plot, which is a method to evaluate agreement among two different measurements. The mean difference is the estimated bias and the SD of the differences measures random fluctuations around this mean. It is common to compute 95% agreement limits for each comparison (average difference $\pm 1.96 \times$ standard deviations), which indicate how far apart measurements by two methods are likely to be for most individuals. This figure shows that all points lay in the two boundaries, which indicates that these two measurements have strong agreement. As can be seen in Figures 5.6 and 5.7, the reduction of dopaminergic neurones in the PINK1 type images could be detected the proposed method.

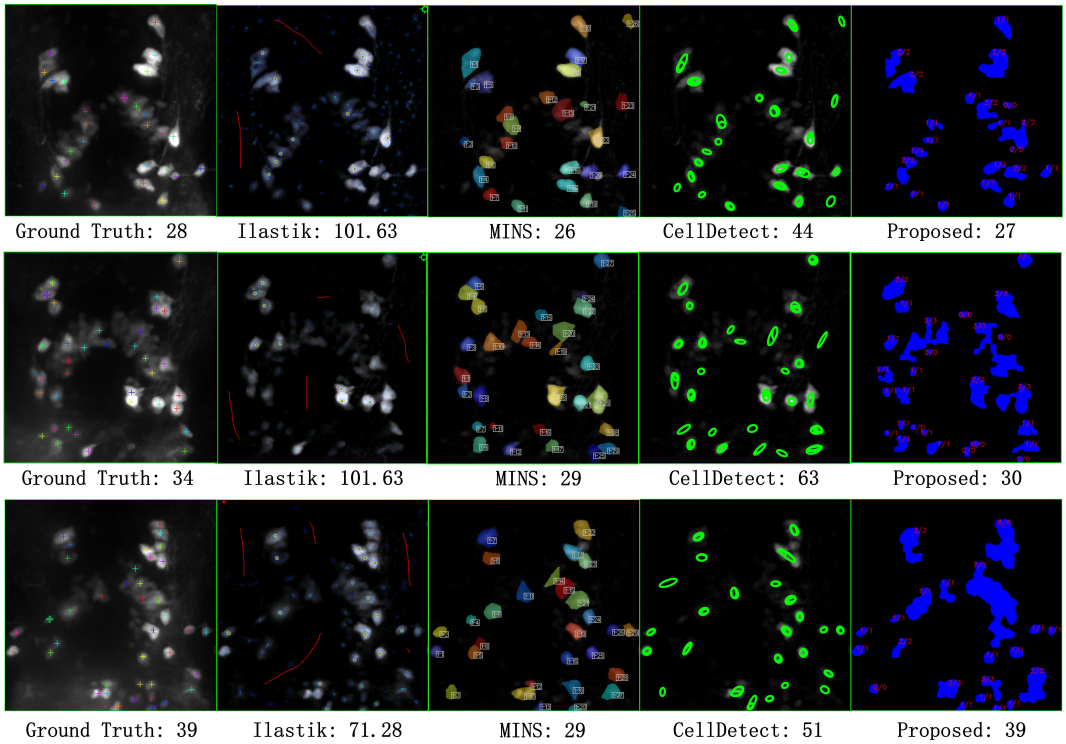


Fig. 5.8 Comparison of different methods for zebrafish data.

5.5 Conclusions

We presented an interactive counting method to solve the cell counting problem by regression in fluorescent microscopy images. In addition, a novel SBPR method based on the RVM framework was proposed. The proposed method integrates a superpixel-based graph cut method for cell region segmentation. SBPR can generate an accurate and discrete number of cells in each cluster and can use significantly fewer relevance vectors for testing, which reduces overall computational cost. Note that the proposed framework is easy to train and use. The proposed method was compared to different state-of-the-art methods, and the results show that it has great potential for counting objects in different types of microscopy images.

Chapter 6

Conclusion and Future Work

As conventional and convenient instruments, light microscopes are used in many different fields. With current technology, it will possible to produce high-resolution large-scale images using high-throughput imaging techniques without human interaction. However, it becomes impossible for a human operator to process all images to obtain relevant and useful information; thus, computational image analysis methods and computer-aided interventions are required. In this thesis, we have focused on methods and software for PD research. Methods and software have been explored for specific problems in light microscopy zebrafish image analysis, including deconvolution/deblurring, object/feature detection and object counting.

6.1 Conclusions and Discussions

In Chapter 3, a deconvolution method with maximum a posteriori was introduced for three-dimensional widefield fluorescence microscopy zebrafish images. In the deconvolution framework, a global sparse prior with Hyper-Laplacian distribution and a local mask were applied for removing out-of-focus light and reducing the ringing artefacts, respectively. With the proposed method, non-blind and blind deconvolution processes were tested with several datasets, including synthetic and

real fluorescence microscopy dataset. By comparing with different existing state-of-the art methods and softwares, it illustrated the proposed method can produce comparable results. However, there are also some drawbacks of using the proposed method. One drawback is the parameter should be manually tuned. During the blind deconvolution, the final result can be sensitive to the estimated initial point spread function.

In Chapter 4, we have presented three different parts for neurone detection in zebrafish z-stack images, including neurone detection using support vector machine with hand-crafted features, neurone detection with 2D convolutional neural network, and neurone detection with a high-throughput automatic framework.

In the first section, we manually extracted many different features, including LBP, HOG, LBP+HOG, HOG3D, colour histogram, SIFT, and colour SIFT features, from the neural images to detect one specific type of neurone. We have demonstrated cell detection results obtained using these different features with a basic SVM detector. Although these handcrafted features do not work particularly effectively, we know that colour is the most important feature and it performs the best with the colour SIFT feature.

In the second section, an automatic deep-learning-based cell detection approach for light microscopy zebrafish z-stack images was introduced. The results demonstrate that the proposed framework has potential for 3D cell detection and outperforms techniques based on hand-crafted features.

In the final section, a high-throughput automatic method based on detection to solve the problem of dopaminergic cell counting by detection in a zebrafish model of PD using TH-labelled neurones in WF microscopy 3D z-stack RGB images. A dynamic cascade framework was used to solve the class-imbalanced problem when training deep learning architectures by combining SVM-RGBHist, a 2D CNN and a 3D-MC-CNN to capture the unique 3D blurry structure of TH-labelled neurones.

The cascade framework can reduce the searching region rapidly in three stages and predict the 'hardest' candidates using a final 3D-MC-CNN deep structure to locate the centre of the neurone. Our zebrafish dataset has been published so that further research can have a benchmark for comparison. Our experimental results demonstrate that, using our dataset, the proposed method has potential for 3D cell detection and counting and outperforms other methods using our dataset. The proposed cascade framework can find 'hard examples' for progressive classification using three different models. Moreover, the 3D-MC-CNN model has more power to classify 3D light microscopy images than 3D CNN or 2D CNN models. The deep neural network algorithm performed better than handcrafted features with a linear SVM. Furthermore, the results demonstrate that even deep learning architectures can learn important high-level features automatically; however, performance can be limited by the method used to select training data from a dataset with large variance.

However, there are still some false positives and false negatives based on human counting criteria. There are two situations that may influence the final results. First, human counting criteria are subjective and some TH-labelled neurones are very ambiguous. For example, if a neurone is stained too lightly, it may not be counted by experts. Second, the quality of a test image can also influence the accuracy of the final result. If the quality is too low, multiple neurones may be gathered and recognised as a single neurone due to out-of-focus light.

In Chapter 5, to count cells in desired regions of two-dimensional fluorescence microscopy images, a interactive counting by density regression framework was proposed. Superpixel-based graph cut technique was used to select cell regions with user inputs. In the framework, a novel SBPR regression method was designed to estimate the number of cells in each isolated selected region. The proposed method was tested with one synthetic fluorescence microscopy dataset and one real zebrafish fluorescence microscopy dataset. Compared to other state-of-the art methods and softwares, the result shows that the proposed method has its own potential to count

cells in low-quality fluorescence microscopy images with severe cell overlapping situations. One advantage is that users can select the region by themselves, which makes our software more user-friendly. Furthermore, the proposed sparse Bayesian Poisson regression not only can produce discrete integer counting number, but it also can produce sparse model to make the prediction performance much faster.

6.2 Suggestions for Future Work

To improve and extend the discovery of this thesis, some potential future work will be summarised based on methods and applications in each chapter.

For the 3D deconvolution method in Chapter 3, potential future work can be developed based on several aspects. For the first aspect, to avoid manually tuning parameters, methods with self-adaptive parameters can be explored. Further, as the proposed method is designed for fluorescence microscopy image, the future work is to extend the deconvolution method to process different modalities of light microscopy images, such as brightfield images and RGB optical sections. In this area, blind deconvolution techniques are not fully explored, which makes more possibility for further improvement and development. For the deconvolution software, we have implemented software with MATLAB. To improve the usability and computational speed, the software can be rewritten using c++ and CUDA computing with GPU acceleration.

For cell detection method in Chapter 4, future work can focus on evaluating the proposed approach using two observers' label information and comparing it with other state-of-the art approaches and programmes. Further, to explore the usability of proposed method, other similar three-dimensional dataset can be tested. For the zebrafish dataset, it is quite possible to obtain z-stack images with higher quality using modern high-throughput microscopy screening techniques. For the cell detection implementation, the optimization can be improved.

For cell counting by regression method in Chapter 5, the future work will focus on developing more powerful shape features for cell counting by regression methods in fluorescence microscopy images. The method can be directly extended to 3D and tested with large-scale 3D datasets. Furthermore, deep learning regression can be a good direction for cell counting in whole images. Currently, most deep learning detection and regression method is based on patches. Image based training procedure is more convenient and efficient for regression task. For the cell counting software, a Windows version is finished and delivered to neuroscientists, and a Mac version is considered for development.

Even though the initial goal of our proposed methods is to process zebrafish microscopy images for PD research, our proposed methods have potential to be used for other purposes, *e.g.* our deconvolution method can be used in the telescope and natural images; the cell detection method can be used for a significant step of pedestrian/cell tracking in videos; the counting software can be used for more complex object counting in natural images.

References

[1] S. Bretaud, C. Allen, P. W. Ingham, and O. Bandmann, "p53-dependent neuronal cell death in a dj-1-deficient zebrafish model of parkinson's disease," *Journal of neurochemistry*, vol. 100, no. 6, pp. 1626–1635, 2007.

[2] P. J. Shaw, "Comparison of widefield/deconvolution and confocal microscopy for three-dimensional imaging," in *Handbook of biological confocal microscopy*, pp. 453–467, Springer, 2006.

[3] I. T. Young, "Image fidelity: characterizing the imaging transfer function," *Methods in cell biology*, vol. 30, pp. 1–45, 1989.

[4] M. Bertero, P. Boccacci, G. Desidera, and G. Vicidomini, "Image deblurring with poisson data: from cells to galaxies," *Inverse Problems*, vol. 25, no. 12, p. 123006, 2009.

[5] H. He and E. A. Garcia, "Learning from imbalanced data," *Knowledge and Data Engineering, IEEE Transactions on*, vol. 21, no. 9, pp. 1263–1284, 2009.

[6] R. Wollman and N. Stuurman, "High throughput microscopy: from raw images to discoveries," *Journal of cell science*, vol. 120, no. 21, pp. 3715–3722, 2007.

[7] M. Oberlaender, V. J. Dercksen, R. Egger, M. Gensel, B. Sakmann, and H.-C. Hege, "Automated three-dimensional detection and counting of neuron somata," *Journal of neuroscience methods*, vol. 180, no. 1, pp. 147–160, 2009.

[8] B. Dong, L. Shao, A. F. Frangi, O. Bandmann, and M. D. Costa, "Three-dimensional deconvolution of wide field microscopy with sparse priors: Application to zebrafish imagery," in *22nd International Conference on Pattern Recognition*, pp. 865–870, 2014.

[9] M. Karakaya, R. A. Kerekes, S. S. Gleason, R. A. Martins, and M. A. Dyer, "Automatic detection, segmentation and characterization of retinal horizontal neurons in large-scale 3D confocal imagery," in *SPIE Medical Imaging*, pp. 79622J–79622J, International Society for Optics and Photonics, 2011.

[10] B. Dong, L. Shao, O. Bandmann, M. D. Costa, and A. F. Frangi, "Deep learning for automatic cell detection in wide-field microscopy zebrafish images," in *Biomedical Imaging: From Nano to Macro, 2015 IEEE International Symposium on*, pp. 772 – 776, IEEE, 2015.

[11] X. Xu, X. Xu, X. Huang, W. Xia, and S. Xia, "A high-throughput analysis method to detect regions of interest and quantify zebrafish embryo images," *Journal of biomolecular screening*, vol. 15, no. 9, pp. 1152–1159, 2010.

[12] M. D. Abràmoff, P. J. Magalhães, and S. J. Ram, "Image processing with ImageJ," *Biophotonics international*, vol. 11, no. 7, pp. 36–43, 2004.

[13] J. Schindelin, I. Arganda-Carreras, E. Frise, V. Kaynig, M. Longair, T. Pietzsch, S. Preibisch, C. Rueden, S. Saalfeld, B. Schmid, *et al.*, "Fiji: an open-source platform for biological-image analysis," *Nature methods*, vol. 9, no. 7, pp. 676–682, 2012.

[14] C. Sommer, C. Straehle, U. Köthe, and F. A. Hamprecht, "Ilastik: Interactive learning and segmentation toolkit," in *2011 IEEE international symposium on biomedical imaging: From nano to macro*, pp. 230–233, IEEE, 2011.

[15] F. De Chaumont, S. Dallongeville, N. Chenouard, N. Hervé, S. Pop, T. Provoost, V. Meas-Yedid, P. Pankajakshan, T. Lecomte, Y. Le Montagner, *et al.*, "Icy: an open bioimage informatics platform for extended reproducible research," *Nature methods*, vol. 9, no. 7, pp. 690–696, 2012.

[16] H. Peng, Z. Ruan, F. Long, J. H. Simpson, and E. W. Myers, "V3d enables real-time 3d visualization and quantitative analysis of large-scale biological image data sets," *Nature biotechnology*, vol. 28, no. 4, pp. 348–353, 2010.

[17] Y. Bengio, "Learning deep architectures for ai," *Foundations and trends® in Machine Learning*, vol. 2, no. 1, pp. 1–127, 2009.

[18] J. Ngiam, Z. Chen, D. Chia, P. W. Koh, Q. V. Le, and A. Y. Ng, "Tiled convolutional neural networks," in *Advances in Neural Information Processing Systems*, pp. 1279–1287, 2010.

[19] L. Shao, D. Wu, and X. Li, "Learning deep and wide: A spectral method for learning deep networks," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 25, no. 12, pp. 2303–2308, 2014.

[20] D. Wu and L. Shao, "Multimodal dynamic networks for gesture recognition," in *Proceedings of the ACM International Conference on Multimedia*, pp. 945–948, 2014.

[21] D. Wu and L. Shao, "Leveraging hierarchical parametric networks for skeletal joints based action segmentation and recognition," in *Proceedings of IEEE Conference on Computer Vision and Patter Recognition*, pp. 724–731, 2014.

[22] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, pp. 1097–1105, 2012.

[23] G. E. Hinton, S. Osindero, and Y.-W. Teh, "A fast learning algorithm for deep belief nets," *Neural computation*, vol. 18, no. 7, pp. 1527–1554, 2006.

[24] P. Vincent, H. Larochelle, I. Lajoie, Y. Bengio, and P.-A. Manzagol, "Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion," *The Journal of Machine Learning Research*, vol. 11, pp. 3371–3408, 2010.

[25] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning internal representations by error propagation," tech. rep., DTIC Document, 1985.

[26] R. Hecht-Nielsen *et al.*, "Theory of the backpropagation neural network.," *Neural Networks*, vol. 1, no. Supplement-1, pp. 445–448, 1988.

[27] K. Fukushima, "Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position," *Biological cybernetics*, vol. 36, no. 4, pp. 193–202, 1980.

[28] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[29] S. Lawrence, C. L. Giles, A. C. Tsoi, and A. D. Back, "Face recognition: A convolutional neural-network approach," *Neural Networks, IEEE Transactions on*, vol. 8, no. 1, pp. 98–113, 1997.

[30] R. Collobert and J. Weston, "A unified architecture for natural language processing: Deep neural networks with multitask learning," in *Proceedings of the 25th international conference on Machine learning*, pp. 160–167, ACM, 2008.

[31] D. C. Cireşan, A. Giusti, L. M. Gambardella, and J. Schmidhuber, "Mitosis detection in breast cancer histology images with deep neural networks," in *Medical Image Computing and Computer-Assisted Intervention*, pp. 411–418, Springer, 2013.

[32] S. I. Kabanikhin, "Definitions and examples of inverse and ill-posed problems," *Journal of Inverse and Ill-Posed Problems*, vol. 16, no. 4, pp. 317–357, 2008.

[33] H. W. Engl, M. Hanke, and A. Neubauer, *Regularization of inverse problems*, vol. 375. Springer Science & Business Media, 1996.

[34] A. Levin, Y. Weiss, F. Durand, and W. T. Freeman, "Understanding and evaluating blind deconvolution algorithms," in *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1964–1971, 2009.

[35] J. G. McNally, T. Karpova, J. Cooper, and J. A. Conchello, "Three-dimensional imaging by deconvolution microscopy," *Methods*, vol. 19, no. 3, pp. 373–385, 1999.

[36] A. ĭ. N. Tikhonov, *Numerical Methods for the Solution of Ill-posed Problems*, vol. 328. Springer, 1995.

[37] T. Tommasi, A. Diaspro, and B. Bianco, "3D reconstruction in optical microscopy by a frequency-domain approach," *IEEE Transactions on Signal Processing*, vol. 32, no. 3, pp. 357–366, 1993.

[38] W. H. Richardson, "Bayesian-based iterative method of image restoration," *Journal of the Optical Society of America*, vol. 62, no. 1, pp. 55–59, 1972.

[39] L. Lucy, "An iterative technique for the rectification of observed distributions," *The Astronomical Journal*, vol. 79, p. 745, 1974.

[40] A. Levin, Y. Weiss, F. Durand, and T. W. Freeman, "Efficient marginal likelihood optimization in blind deconvolution," in *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2657–2664, 2011.

[41] S. Cho and S. Lee, "Fast motion deblurring," *ACM Transactions on Graphics*, vol. 28, no. 5, p. 145, 2009.

[42] Q. Shan, J. Jia, and A. Agarwala, "High-quality motion deblurring from a single image," *ACM Transactions on Graphics*, vol. 27, no. 3, p. 73, 2008.

[43] D. Krishnan and R. Fergus, "Fast image deconvolution using hyper-laplacian priors," in *Advances in Neural Information Processing Systems*, pp. 1033–1041, 2009.

[44] J. Portilla, V. Strela, M. J. Wainwright, and E. P. Simoncelli, "Image denoising using scale mixtures of gaussians in the wavelet domain," *IEEE Transactions on Image Processing*, vol. 12, no. 11, pp. 1338–1351, 2003.

[45] A. Levin and Y. Weiss, "User assisted separation of reflections from a single image using a sparsity prior," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 29, no. 9, pp. 1647–1654, 2007.

[46] M. F. Tappen, B. C. Russell, and W. T. Freeman, "Exploiting the sparse derivative prior for super-resolution and image demosaicing," in *IEEE Workshop on Statistical and Computational Theories of Vision*, 2003.

[47] C. V. Stewart, "Robust parameter estimation in computer vision," *Society for Industrial and Applied Mathematics Review*, vol. 41, no. 3, pp. 513–537, 1999.

[48] L. A. Shepp and Y. Vardi, "Maximum likelihood reconstruction for emission tomography," *IEEE Transactions on Medical Imaging*, vol. 1, no. 2, pp. 113–122, 1982.

[49] N. Dey, L. Blanc, Feraud, C. Zimmer, P. Roux, Z. Kam, M. Olivo, C. Jean, and J. Zerubia, "Richardson–Lucy algorithm with total variation regularization for 3D confocal microscope deconvolution," *Microscopy Research and Technique*, vol. 69, no. 4, pp. 260–266, 2006.

[50] M. Keuper, T. Schmidt, M. Temerinac Ott, J. Padeken, P. Heun, O. Ronneberger, and T. Brox, "Blind deconvolution of widefield fluorescence microscopic data by regularization of the optical transfer function (OTF)," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2013.

[51] T. Kenig, Z. Kam, and A. Feuer, "Blind image deconvolution using machine learning for three-dimensional microscopy," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 32, no. 12, pp. 2191–2204, 2010.

[52] N. Dey, L. Blanc-Feraud, C. Zimmer, Z. Kam, J.-C. Olivo-Marin, and J. Zerubia, "A deconvolution method for confocal microscopy with total variation regularization," in *IEEE International Symposium on Biomedical Imaging*, pp. 1223–1226, 2004.

[53] V. Kempen and V. Vliet, "The influence of the regularization parameter and the first estimate on the performance of tikhonov regularized non-linear image restoration algorithms," *Journal of Microscopy*, vol. 198, no. 1, pp. 63–75, 2000.

[54] M. Carlavan and L. Blanc-Féraud, "Sparse poisson noisy image deblurring," *IEEE Transactions on Image Processing*, vol. 21, no. 4, pp. 1834–1846, 2012.

[55] E. F. Hom, F. Marchis, T. K. Lee, S. Haase, D. A. Agard, and J. W. Sedat, "AIDA: an adaptive image deconvolution algorithm with application to multi-frame and three-dimensional data," *The Journal of the Optical Society of America A*, vol. 24, no. 6, pp. 1580–1600, 2007.

[56] L. M. Mugnier, T. Fusco, and J.-M. Conan, "Mistral: a myopic edge-preserving image restoration method, with application to astronomical adaptive-optics-corrected long-exposure images," *The Journal of the Optical Society of America A*, vol. 21, no. 10, pp. 1841–1854, 2004.

[57] B. Dougherty, "Iterative deconvolve 3D." http://www.optinav.info/Iterative-Deconvolve-3D.htm [Online], April 2014. Accessed:2014-01-04.

[58] N. Linnenbrügger, "FFTJ and DeconvolutionJ." http://rsb.info.nih.gov/ij/plugins/fftj.html [Online], April 2014. Accessed:2014-01-04.

[59] P. Wendykier, "Parallel spectral deconvolution 3D." https://sites.google.com/site/piotrwendykier/software/deconvolution/parallelspectraldeconvolution [Online], April 2014. Accessed:2014-01-04.

[60] C. Vonesch and M. Unser, "A fast thresholded landweber algorithm for wavelet-regularized multidimensional deconvolution," *IEEE Transactions on Image Processing*, vol. 17, no. 4, pp. 539–549, 2008.

[61] M. J. Booth, M. Neil, and T. Wilson, "Aberration correction for confocal imaging in refractive-index-mismatched media," *Journal of Microscopy*, vol. 192, no. 2, pp. 90–98, 1998.

[62] G. H. Dunteman, *Principal components analysis*. No. 69, Sage, 1989.

[63] B. Schölkopf, A. Smola, and K.-R. Müller, "Kernel principal component analysis," in *International Conference on Artificial Neural Networks*, pp. 583–588, Springer, 1997.

[64] C. Schmitz, B. S. Eastwood, S. J. Tappan, J. R. Glaser, D. A. Peterson, and P. R. Hof, "Current automated 3D cell detection methods are not a suitable replacement for manual stereologic cell counting," *Frontiers in neuroanatomy*, vol. 8, 2014.

[65] T. Liu, G. Li, J. Nie, A. Tarokh, X. Zhou, L. Guo, J. Malicki, W. Xia, and S. T. Wong, "An automated method for cell detection in zebrafish," *Neuroinformatics*, vol. 6, no. 1, pp. 5–21, 2008.

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

[67] M. K. K. Niazi, A. A. Satoskar, and M. N. Gurcan, "An automated method for counting cytotoxic T-cells from CD8 stained images of renal biopsies," in *SPIE Medical Imaging*, pp. 867606–867606, International Society for Optics and Photonics, 2013.

[68] S. Wienert, D. Heim, K. Saeger, A. Stenzinger, M. Beil, P. Hufnagl, M. Dietel, C. Denkert, and F. Klauschen, "Detection and segmentation of cell nuclei in

virtual microscopy images: a minimum-model approach," *Scientific reports*, vol. 2, 2012.

[69] T. Chen and C. Chefd'hotel, "Deep learning based automatic immune cell detection for immunohistochemistry images," in *Machine Learning in Medical Imaging*, pp. 17–24, Springer, 2014.

[70] H. Bogunović, J. M. Pozo, M. C. Villa-Uriol, C. B. Majoie, R. van den Berg, H. A. G. van Andel, J. M. Macho, J. Blasco, L. San Román, and A. F. Frangi, "Automated segmentation of cerebral vasculature with aneurysms in 3DRA and TOF-MRA using geodesic active regions: An evaluation study," *Medical Physics*, vol. 38, no. 1, pp. 210–222, 2011.

[71] M. Galar, A. Fernandez, E. Barrenechea, H. Bustince, and F. Herrera, "A review on ensembles for the class imbalance problem: bagging-, boosting-, and hybrid-based approaches," *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on*, vol. 42, no. 4, pp. 463–484, 2012.

[72] T. Grósz and I. Nagy, "Document classification with deep rectifier neural networks and probabilistic sampling," in *Text, Speech and Dialogue*, pp. 108–115, Springer, 2014.

[73] D. Ciresan, A. Giusti, L. M. Gambardella, and J. Schmidhuber, "Deep neural networks segment neuronal membranes in electron microscopy images," in *Advances in neural information processing systems*, pp. 2843–2851, 2012.

[74] M. Kukar, I. Kononenko, *et al.*, "Cost-sensitive learning with neural networks.," in *ECAI*, pp. 445–449, Citeseer, 1998.

[75] C. Elkan, "The foundations of cost-sensitive learning," in *International joint conference on artificial intelligence*, vol. 17, pp. 973–978, Citeseer, 2001.

[76] H. Guo and H. L. Viktor, "Learning from imbalanced data sets with boosting and data generation: the databoost-im approach," *ACM SIGKDD Explorations Newsletter*, vol. 6, no. 1, pp. 30–39, 2004.

[77] C.-W. Wang and A. Hunter, "Robust pose recognition of the obscured human body," *International journal of computer vision*, vol. 90, no. 3, pp. 313–330, 2010.

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

[79] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, "Smote: synthetic minority over-sampling technique," *Journal of artificial intelligence research*, vol. 16, no. 1, pp. 321–357, 2002.

[80] B. E. Boser, I. M. Guyon, and V. N. Vapnik, "A training algorithm for optimal margin classifiers," in *Proceedings of the fifth annual workshop on Computational learning theory*, pp. 144–152, ACM, 1992.

[81] C. Cortes and V. Vapnik, "Support-vector networks," *Machine learning*, vol. 20, no. 3, pp. 273–297, 1995.

[82] A. B. Chan, Z.-S. J. Liang, and N. Vasconcelos, "Privacy preserving crowd monitoring: Counting people without people models or tracking," in *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, pp. 1–7, IEEE, 2008.

[83] K. Chen, C. C. Loy, S. Gong, and T. Xiang, "Feature mining for localised crowd counting.," in *BMVC*, vol. 1, p. 3, 2012.

[84] C. Arteta, V. Lempitsky, J. A. Noble, and A. Zisserman, "Interactive object counting," in *European Conference on Computer Vision*, pp. 504–518, Springer, 2014.

[85] W. Xie, J. A. Noble, and A. Zisserman, "Microscopy cell counting with fully convolutional regression networks," in *MICCAI 1st Workshop on Deep Learning in Medical Image Analysis*, 2015.

[86] L. Fiaschi, U. Köthe, R. Nair, and F. A. Hamprecht, "Learning to count with regression forest and structured labels," in *Pattern Recognition (ICPR), 2012 21st International Conference on*, pp. 2685–2688, IEEE, 2012.

[87] V. Lempitsky and A. Zisserman, "Learning to count objects in images," in *Advances in Neural Information Processing Systems*, pp. 1324–1332, 2010.

[88] M. E. Tipping, "Sparse bayesian learning and the relevance vector machine," *Journal of machine learning research*, vol. 1, no. Jun, pp. 211–244, 2001.

[89] A. C. Cameron and P. K. Trivedi, *Regression analysis of count data*, vol. 53. Cambridge university press, 2013.

[90] P. J. Shaw, "Comparison of widefield/deconvolution and confocal microscopy for three-dimensional imaging," in *Handbook of Biological Confocal Microscopy*, pp. 453–467, Springer, 2006.

[91] I. T. Young, "Image fidelity: characterizing the imaging transfer function," *Methods Cell Biol*, vol. 30, pp. 1–45, 1989.

[92] G. Zack, W. Rogers, and S. Latt, "Automatic measurement of sister chromatid exchange frequency.," *Journal of Histochemistry and Cytochemistry*, vol. 25, no. 7, pp. 741–753, 1977.

[93] R. W. Boyd, "Radiometry and the detection of optical radiation," *John Wiley and Sons*, vol. 1, p. 261, 1983.

[94] S. Hiware, P. Porwal, R. Velmurugan, and S. Chaudhuri, "Modeling of PSF for refractive index variation in fluorescence microscopy," in *IEEE Conference on Image Processing*, pp. 2037–2040, 2011.

[95] S. Frisken Gibson and F. Lanni, "Experimental test of an analytical model of aberration in an oil-immersion objective lens used in three-dimensional light microscopy," *The Journal of the Optical Society of America A*, vol. 8, no. 10, pp. 1601–1613, 1991.

[96] P. Sarder and A. Nehorai, "Deconvolution methods for 3D fluorescence microscopy images," *IEEE Signal Processing Magazine*, vol. 23, no. 3, pp. 32–45, 2006.

[97] P. Pankajakshan, B. Zhang, L. Blanc-Féraud, Z. Kam, J.-C. Olivo-Marin, and J. Zerubia, "Blind deconvolution for diffraction-limited fluorescence microscopy," in *IEEE International Symposium on Biomedical Imaging*, pp. 740–743, 2008.

[98] F. Aguet, D. Van De Ville, and M. Unser, "Model-based 2.5D deconvolution for extended depth of field in brightfield microscopy," *IEEE Transactions on Image Processing*, vol. 17, no. 7, pp. 1144–1153, 2008.

[99] F. Aguet, *Super-resolution fluorescence microscopy based on physical models*. PhD thesis, ÉCOLE POLYTECHNIQUE FÉDÉRALE DE LAUSANNE, 2009.

[100] H. Kirshner, F. Aguet, D. Sage, and M. Unser, "3D psf fitting for fluorescence microscopy: implementation and localization application," *Journal of microscopy*, vol. 249, no. 1, pp. 13–25, 2013.

[101] D. Svoboda, M. Kozubek, and S. Stejskal, "Generation of digital phantoms of cell nuclei and simulation of image formation in 3D image cytometry," *Cytometry*, vol. 75A, no. 6, pp. 494–509, 2009.

[102] Scientific Volume Imaging, *Hugens Essential: deconvolution a few mouse cliks away*. The Netherlands, 2014.

[103] E. Rink and M. F. Wullimann, "Development of the catecholaminergic system in the early zebrafish brain: an immunohistochemical study," *Developmental brain research*, vol. 137, no. 1, pp. 89–100, 2002.

[104] A. E. Carpenter, T. R. Jones, M. R. Lamprecht, C. Clarke, I. H. Kang, O. Friman, D. A. Guertin, J. H. Chang, R. A. Lindquist, J. Moffat, *et al.*, "Cellprofiler: image analysis software for identifying and quantifying cell phenotypes," *Genome biology*, vol. 7, no. 10, p. R100, 2006.

[105] N. Dey, L. Blanc-Feraud, C. Zimmer, P. Roux, Z. Kam, J.-C. Olivo-Marin, and J. Zerubia, "Richardson–lucy algorithm with total variation regularization for 3d confocal microscope deconvolution," *Microscopy research and technique*, vol. 69, no. 4, pp. 260–266, 2006.

[106] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *The Journal of Machine Learning Research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[107] P. Thévenaz, "Point Picker. (A plugin of the ImageJ)." http://bigwww.epfl.ch/thevenaz/pointpicker/ [Online], 2013. Accessed: 2015-06-05.

[108] L. G. Nyu and J. K. Udupa, "On standardizing the mr image intensity scale," *image*, vol. 1081, 1999.

[109] M. Kolesnik and A. Fexa, "Multi-dimensional color histograms for segmentation of wounds in images," in *Image Analysis and Recognition*, pp. 1014–1022, Springer, 2005.

[110] S. Demyanov, "Convnet library for matlab." http://www.demyanov.net/ [Online], 2014. Accessed:2014-09-30.

[111] J. Masci, U. Meier, D. Cireşan, and J. Schmidhuber, "Stacked convolutional auto-encoders for hierarchical feature extraction," in *Artificial Neural Networks and Machine Learning–ICANN 2011*, pp. 52–59, Springer, 2011.

[112] S. Ji, W. Xu, M. Yang, and K. Yu, "3D convolutional neural networks for human action recognition," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 35, no. 1, pp. 221–231, 2013.

[113] G. Chiaruttini, "Identification of vesicular snare proteins involved in the secretoryintracellular trafficking of il-12 in dendritic cells." https://www.openstarts.units.it/dspace/handle/10077/8636 [Online], 2013. Accessed: 2013-04-22.

[114] D. Ciresan, U. Meier, and J. Schmidhuber, "Multi-column deep neural networks for image classification," in *Computer Vision and Pattern Recognition, 2012 IEEE Conference on*, pp. 3642–3649, IEEE, 2012.

[115] S. Demyanov, J. Bailey, R. Kotagiri, and C. Leckie, "Invariant backpropagation: how to train a transformation-invariant neural network," *arXiv preprint arXiv:1502.04434*, 2015.

[116] N. Otsu, "A threshold selection method from gray-level histograms," *Automatica*, vol. 11, no. 285-296, pp. 23–27, 1975.

[117] S. Demyanov, "ConvNet." https://github.com/sdemyanov/ConvNet [Online], 2014. Accessed:2015-04-29.

[118] X. Lou, M. Kang, P. Xenopoulos, S. Munoz-Descalzo, and A.-K. Hadjantonakis, "A rapid and efficient 2d/3d nuclear segmentation method for analysis of early mouse embryo and stem cell image data," *Stem cell reports*, vol. 2, no. 3, pp. 382–397, 2014.

[119] R. Chinta and M. Wasser, "Three-dimensional segmentation of nuclei and mitotic chromosomes for the study of cell divisions in live drosophila embryos," *Cytometry Part A*, vol. 81, no. 1, pp. 52–64, 2012.

[120] S. Bolte and F. Cordelieres, "A guided tour into subcellular colocalization analysis in light microscopy," *Journal of microscopy*, vol. 224, no. 3, pp. 213–232, 2006.

[121] K. Okuma, A. Taleghani, N. De Freitas, J. J. Little, and D. G. Lowe, "A boosted particle filter: Multitarget detection and tracking," in *Computer Vision-ECCV 2004*, pp. 28–39, Springer, 2004.

[122] A. Lehmussola, P. Ruusuvuori, J. Selinummi, H. Huttunen, and O. Yli-Harja, "Computational framework for simulating fluorescence microscope images with cell populations," *IEEE Transactions on Medical Imaging*, vol. 26, no. 7, pp. 1010–1016, 2007.

[123] G. El-Sayyad, "Bayesian and classical analysis of poisson regression," *Journal of the Royal Statistical Society. Series B (Methodological)*, pp. 445–451, 1973.

[124] A. B. Chan and N. Vasconcelos, "Bayesian poisson regression for crowd counting," in *2009 IEEE 12th International Conference on Computer Vision*, pp. 545–551, IEEE, 2009.

[125] A. B. Chan and N. Vasconcelos, "Counting people with low-level features and bayesian regression," *IEEE Transactions on Image Processing*, vol. 21, no. 4, pp. 2160–2177, 2012.

[126] V. Vapnik, S. E. Golowich, A. Smola, *et al.*, "Support vector method for function approximation, regression estimation, and signal processing," *Advances in neural information processing systems*, pp. 281–287, 1997.

[127] J. O. Berger, *Statistical decision theory and Bayesian analysis*. Springer Science & Business Media, 2013.

[128] M. E. Tipping, A. C. Faul, *et al.*, "Fast marginal likelihood maximisation for sparse bayesian models.," in *AISTATS*, 2003.

[129] Y. Boykov and V. Kolmogorov, "An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision," *IEEE transactions on pattern analysis and machine intelligence*, vol. 26, no. 9, pp. 1124–1137, 2004.

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

[131] Y. Y. Boykov and M.-P. Jolly, "Interactive graph cuts for optimal boundary & region segmentation of objects in nd images," in *Computer Vision, 2001. ICCV 2001. Proceedings. Eighth IEEE International Conference on*, vol. 1, pp. 105–112, IEEE, 2001.

[132] C. Arteta, V. Lempitsky, J. A. Noble, and A. Zisserman, "Learning to detect cells using non-overlapping extremal regions," in *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pp. 348–356, Springer, 2012.