January 2023

Computational Framework For Neuro-Optics Simulation And Deep Learning Denoising

Cecilia Ling

COMPUTATIONAL FRAMEWORK FOR NEURO-OPTICS SIMULATION AND DEEP LEARNING DENOISING

by

Cecilia Ling

Bachelor of Science, University of North Dakota, 2020

A Thesis

Submitted to the Graduate Faculty

of the

University of North Dakota

in partial fulfillment of the requirements

for the degree of

Master of Science

Grand Forks, North Dakota

August

2023

This thesis, submitted by Cecilia Ling in partial fulfillment of the requirements for the Degree of Master of Science from the University of North Dakota, has been read by the Faculty Advisory Committee under whom the work has been done and is hereby approved.

Bo Liang
Name of Chairperson

Emanuel Grant
Name of Committee Member

Min Seok Kim
Name of Committee Member

Aaron Kennedy
Name of Committee Member

This thesis is being submitted by the appointed advisory committee as having met all of the requirements of the School of Graduate Studies at the University of North Dakota and is hereby approved.

Chris Nelson

Dean of the School of Graduate Studies

Date

PERMISSION

Title Computational Framework for Neuro-Optics Simulation and Deep Learning Denoising

Department Computer Science

Degree Master of Science

In presenting this thesis in partial fulfillment of the requirements for a graduate degree from the University of North Dakota, I agree that the library of this University shall make it freely available for inspection. I further agree that permission for extensive copying for scholarly purposes may be granted by the professor who supervised my thesis work or, in his absence, by the chairperson of the department of the dean of the School of Graduate Studies. It is understood that any copying or publication or other use of this thesis or part thereof for financial gain shall not be allowed without my written permission. It is also understood that due recognition shall be given to me and to the University of North Dakota in any scholarly use which may be made of any materials in my thesis.

Cecilia Ling

July 18, 2023

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

Table Page

ACKNOWLEDGEMENTS

I would like to express my deepest gratitude and appreciation to all those who have contributed to the completion of this thesis. First and foremost, I am immensely thankful to my supervisor, Dr. Bo Liang, for his guidance, expertise, and unwavering support throughout this research journey. His valuable insights, constructive feedback, and dedication have been instrumental in shaping the direction of this study. I would also like to extend my sincere appreciation to the members of my thesis committee, Dr. Emanuel Grant, Dr. Aaron Kennedy, and Dr. Min Seok Kim, for their valuable time and constructive feedback. I am immensely grateful to Mr. Aaron Bergstrom and Mr. David Apostal for their support that allowed me to take full advantage of the computer cluster to accelerate the progress.

I am thankful to the University of North Dakota for providing the necessary resources, research facilities, and a conducive academic environment for conducting this study. The support and encouragement received from the faculty and staff have been invaluable.

ABSTRACT

The application of machine learning techniques in microscopic image restoration has shown superior performance. However, the development of such techniques has been hindered by the demand for large datasets and the lack of ground truth. To address these challenges, this study introduces a computer simulation model that accurately captures the neural anatomic volume, fluorescence light transportation within the tissue volume, and the photon collection process of microscopic imaging sensors. The primary goal of this simulation is to generate realistic image data for training and validating machine learning models. One notable aspect of this study is the incorporation of a machine learning denoiser into the simulation, which accelerates the computational efficiency of the entire process. By reducing noise levels in the generated images, the denoiser significantly enhances the simulation's performance, allowing for faster and more accurate modeling and analysis of microscopy images. This approach addresses the limitations of data availability and ground truth annotation, offering a practical and efficient solution for microscopic image restoration. The integration of a machine learning denoiser within the simulation significantly accelerates the overall simulation process, while improving the quality of the generated images. This advancement opens new possibilities for training and validating machine learning models in microscopic image restoration, overcoming the challenges of large datasets and the lack of ground truth.

CHAPTER 1 INTRODUCTION

In the dynamic landscape of modern science, the confluence of fluorescence microscopy and computational image restoration stands as a potent force, opening up new dimensions in the realm of visualization and analysis. Fluorescence microscopy, with its ability to illuminate specific cellular components, has unlocked a microscopic universe once hidden from human eyes. However, the raw images captured by even advanced microscopy techniques often suffer from inherent noise, degradation, and other imperfections. This is where computational image restoration, empowered by machine learning algorithms, steps in as a transformative technology that breathes new life into these blurry, distorted, and obscured images. This chapter embarks on a voyage that explores the intricate interplay between fluorescence microscopy and computational image restoration alongside related research.

1.1 Background

Calcium fluorescence microscopy has emerged as a powerful tool in neuroscience enabling researchers to investigate the dynamic activities of calcium ions (Ca^{2+}) within living cells. Calcium ions play a fundamental role in various biological processes, including neuronal signaling, muscle contraction, and cellular communication. Genetically encoded calcium indicators (GECIs) have profoundly transformed microscopic imaging techniques. These GECIs, such as GCaMP and chameleon, are fluorescent proteins that change in fluorescence intensity or wavelength in response to changes in calcium ion concentration. The fluorescence emission enables the visualization and quantification of calcium dynamics with high spatiotemporal resolution in a wide variety of cell types and tissues (Fig. 1-1., left panel, [1]). By monitoring changes in intracellular calcium concentrations, fluorescence microscopy offers a non-invasive

and real-time approach to studying these dynamic events at the cellular and subcellular levels. Discover a more elaborate depiction of fluorescence microscopy in Chapter 2, Section 1.

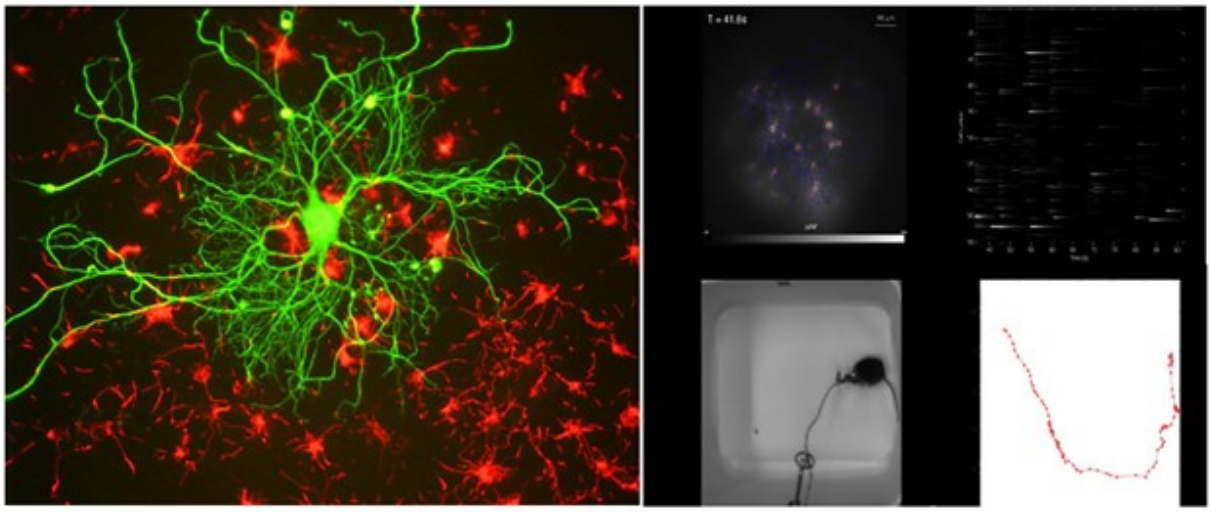

Fig. 1-1. Examples fluorescence images. Left: Structural fluorescence image of two types of brain cells dyed in distinct colors: green for neurons and red for astrocytes [1]. Right: Functional fluorescence imaging of brain activities of mouse striatum with simultaneous recording of behavior by a miniature fluorescence microscope, adapted from [2]. The two sub-images on the top illustrate the neuronal activities (top-left corner) and temporal dynamics (top-right corner) of the brain neurons. The actual and track for the movement of the mouse are demonstrated in the two sub-images at the bottom. Striatum is believed to play a prominent role in motor control, and miniscope provides a means to investigate the neural-behavioral relationship with minimal disruption.

One of the crucial advantages of calcium fluorescence microscopy is its ability to provide insights into the functional activity of individual cells within complex biological systems. Targeting specific cell populations or subcellular compartments with fluorescent calcium indicators allows calcium dynamics to be observed in real time, permitting the investigation of cellular processes. Most importantly, calcium fluorescence microscopy has empowered the study of calcium dynamics in intact, living organisms, furnishing an effective means to understand the functional organization of neural circuits and the mechanisms underlying complex behaviors. Along with advanced imaging techniques like confocal and light-sheet microscopy, calcium

fluorescence imaging captures neuronal activities across large tissue volumes, offering a comprehensive view of neural network dynamics in vivo.

1.1.1 Miniature Fluorescence Microscope

Fluorescence microscopy comprises various types based on illumination process and light collection techniques, such as widefield and confocal. The miniature fluorescence microscope is a compact version of widefield microscopy, known as miniscope, has ushered in a new era in calcium fluorescence microscopy. This groundbreaking device revolutionized neuroscience research by permitting the recording and analysis of neural dynamics in real-time while animals freely engage in natural behaviors (Fig. 1-1., right panel, [2]). Its compact size and lightweight design made it possible to be mounted directly onto the head of an animal, minimizing disruptions to the animal's movements and enabling long-term experiments in a relatively natural environment. Its capability of imaging fluorescently labeled neurons deep in the brain facilitates tracking individual or group of neurons over extended periods, unveiling critical insights into the relationship between neural circuitry and behavior. The miniscope is the primary focus of the present study.

1.1.1a Main sources of image degradation

Despite its remarkable advantages, the miniscope is not exempt from limitations. First, owing to its simple lens and poor optical components alignment due to its compact size miniscope suffers from severe optical aberrations that lead to image degradation. Second, blurring is one of the prominent challenges associated with all widefield imaging as a direct consequence of the indiscriminate collection of both in-focus and out-of-focus light. In widefield imaging, the entire sample is illuminated and the image is formed not only by the light from the

focal plane but also from areas above and below the focal plane (Fig. 1-2., left panel).

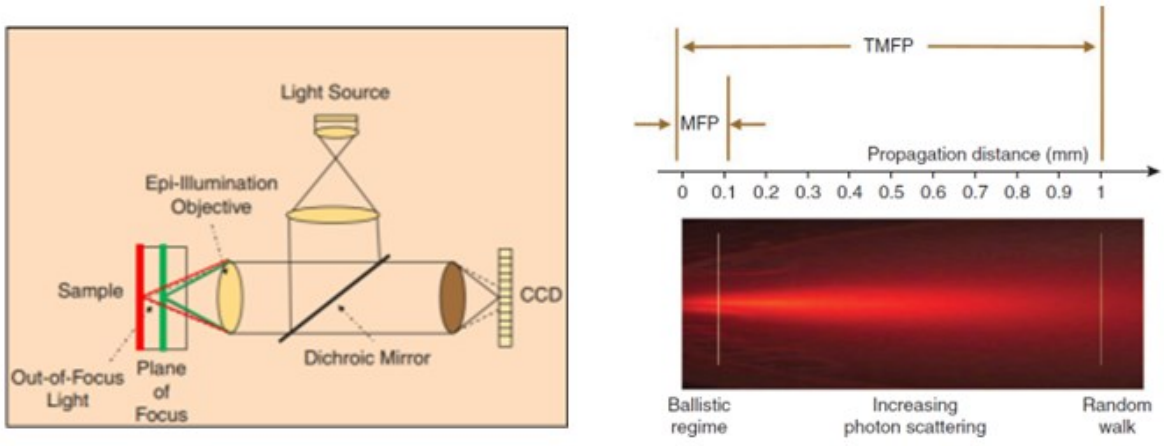

Fig. 1-2. Sources of degradation. Left: Schematic of fluorescence imaging system illustrating the regions of in-focus and out-of-focus [3]. Right: Simulated photon propagation inside brain tissue [4]. The distance of MFP for brain tissue is about 0.1mm, this is the ballistic region. After this distance light experiences increasing number of scattering events, maintaining its initial direction of propagation becomes increasingly difficult.

Another source of blurring is the scattering effect of the tissue. Most microscopy techniques are susceptible to depth limitation resulting from tissue scattering, and widefield is especially vulnerable due to its inability to depth discrimination. The mean free path (MFP) is the average distance a photon moves between two consecutive scattering occurrences inside a scattering medium (e.g., tissue). It defines a ballistic region of light propagation inside the tissue. As the distance approaches the transport mean free path (TMFP), which is the mean distance light maintains relatively in its initial direction, penetration progressively diffuses and results in blurring. The MFP is wavelength dependent and varies across different tissues. Scattering is especially strong in the visible spectral regions where fluorescence lies. For brain tissue, the MFP is typically about the order of 100μm, meaning a photon experiences at least one scattering event while traveling through a tissue of 100μm thickness [4] (Fig. 1-2, right panel). The scattering property of a specific tissue sets the practical depth limit for widefield imaging, and

the thicker the tissue the blurrier the image. This scattering effect will be further uncovered in Chapter 2.

1.1.1b Strategies for image quality improvement

Extensive endeavors have been dedicated to enhancing the quality of fluorescence images via two primary strategies: hardware optimization and computational solution. By leveraging state-of-the-art optical components and cutting-edge techniques, the hardware approach aims to achieve superior image quality by maximizing hardware performance. Adaptive optics is one example that utilizes a wavefront sensor and corrector to minimize optical aberrations caused by tissue scattering [5]. Nevertheless, this approach requires incorporating sophisticated optical elements that exceed the size constraints of a miniscope. Consequently, computational methodology is a more feasible alternative.

1.1.2 Computational Image Enhancement

The computational approach for image improvement involves mathematically modeling the image formation process. An observed image is assumed to result from convolving an observed image results from convolving an object's true image with a blurring function. Computational algorithms are formulated to approximate the true image from observed one by deconvolving the blurring effects. This approach enables the restoration of lost details, noise reduction, and overall image quality enhancement.

1.1.2a Image formation

Physically an image is formed by lenses through the process of refraction, which occurs when light traverses through lenses due to the change of speed and direction caused by the curvature of the lenses. The convergence of light rays resulting from refraction creates a focused

image onto an image sensor. However, an optical system's ability to resolve fine details in an image is limited by diffraction.

Because of the wave nature of light, diffraction happens when the light comes across small openings like aperture, and the interferences among waves generate a characteristic pattern: an Airy disk. Two closely spaced point sources are separable only if the peak of one Airy disk falls on the first minimum of the other. Rayleigh's criterion describes the least distance (r) between two resolvable points as $r = 0.61\lambda/NA$, where λ is the wavelength and NA the numerical aperture. The diffraction limit directly dictates the shape and characteristics of the point spread function (PSF). The PSF is defined as the impulse response of an imaging system to a point source of light and determines how the system blurs the point source.

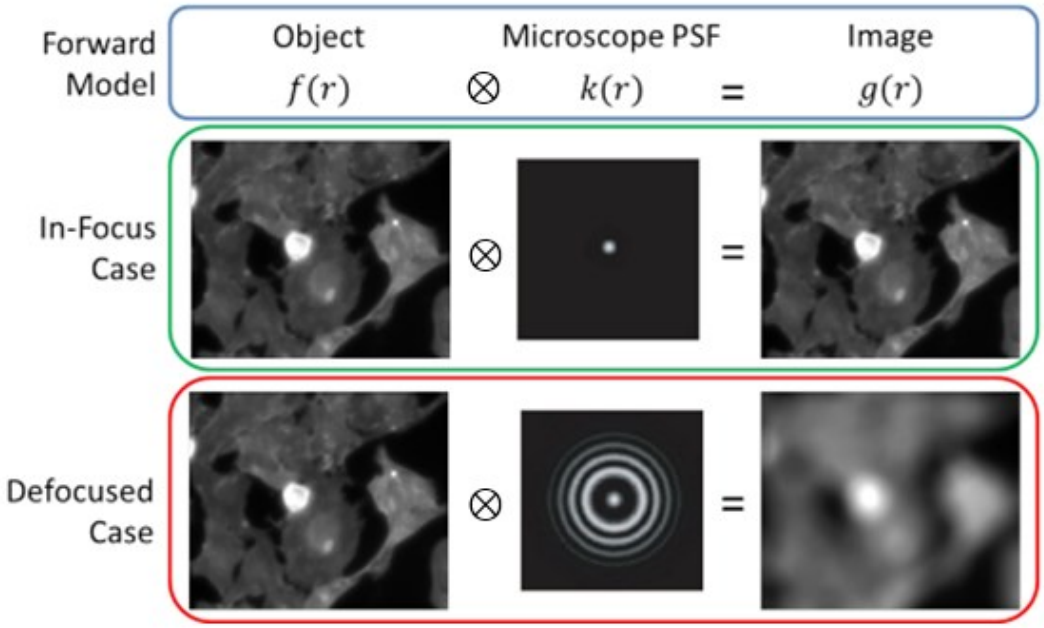

Fig. 1-3., Fluorescence images formed by convolving the same object with different blurry functions (PSFs), adapted from [6].

Mathematically the PSF is the blurring function in the image formation process, and the observed image (I) can be expressed as the convolution result of the true image in object space (O) and the PSF, $I = O \otimes PSF$, illustrated in Fig. 1-3. An undistorted image can thus be obtained by solving the inverse of convolution, called deconvolution.

1.1.2b Deconvolution

Computational deconvolution as a technique for image quality enhancement has existed for decades. Some of the well-known deconvolution algorithms, like the Wiener filter [7] and Richardson-Lucy (RL) iterative methods [8] [9], date back to the 1940s and early 1970s. The conventional approach to deconvolution relies on the assumption of a known PSF obtained either experimentally or theoretically. In practice, however, PSF varies from system to system; noise and unknown aberrations makes it challenging to acquire an accurate PSF either way. Subsequently, deconvolved results are sensitive to the inaccuracies in the PSF and potentially give rise to artifacts in the restored images. Efforts have been made to perform deconvolution without a known PSF, known as blind deconvolution, by simultaneously estimating the PSF and deconvolving [10] [11] [12]. But again, the accuracy of the estimation is often questionable and blind deconvolution processes are usually slow and unstable.

In recent years, data driven machine learning approach has achieved remarkable success and is gradually overperforming traditional methods. Nevertheless, machine learning has its own caveat. It requires a large number of clean images to serve as ground truth during training in supervised learning as well as performance evaluation, regardless of the learning paradigm. This poses a barrier in applying machine learning methods to microscopic images since ground truth images are oftentimes not readily accessible.

1.2 Problem Statement

The miniscope has advanced neuroscience research by enabling concurrent recording of brain activities and behavioral data in freely moving animals. However, the compact size constrains its lens and optical elements' alignments, resulting in optical aberrations. These aberrations distort the PSF from its ideal theoretical shape and introduce image blurring. Moreover, the existence of out-of-focus lights and tissue scattering exacerbates image degradation. These limitations hinder the accuracy of neural signal estimation, undermining the experimental results' reliability and credibility.

Machine learning deconvolution, especially deep learning using artificial neural networks, has demonstrated promising results. But deep learning is data hunger and relies on ground truth images; both can be challenging in microscopic imaging. Furthermore, since cells are not directly observable by naked eye and the determination of “clean” images depends on the subjective perception of clarity by the researchers. What constitutes a clear image could vary based on an individual researcher’s prior knowledge and expectations, resulting in biased assessment. This subjectivity may further complicate the evaluation process. As a result, there is a strong need for an effective computational framework to address these issues and lay the foundation for objective analysis of data and deconvolution performance.

The proposed solution is to create a computer simulation modeling the imaging process of miniscope aiming at generating photo-realistic simulated miniscope fluorescence images, clean and distorted, to be used in training deep learning networks. In addition, simulated clean images can serve as the ground truth for performance assessment. The ultimate goal is to achieve a transferable deconvolution scheme, where the deep learning network trained on simulated data can effectively generalize and be applied to actual miniscope data. The primary focus of the

current study is on developing a reliable simulation that captures the essential characteristics of miniscope images while incorporating a deep neural network to accelerate the simulation.

1.3 Related Works

Over the last decade research efforts have been centered around machine learning methodologies aimed at improving microscopic images. This section explores cutting-edge deep learning image restoration techniques relevant to fluorescence microscopy, followed by a brief introduction to the Monte Carlo (MC) framework of modeling optics-tissue dynamics. The methods employed in these studies are not directly applicable to miniscope images; nonetheless, they serve as the source of inspiration for this study. Example deep learning applications in fluorescence microscopy are shown in Fig. 1-4.

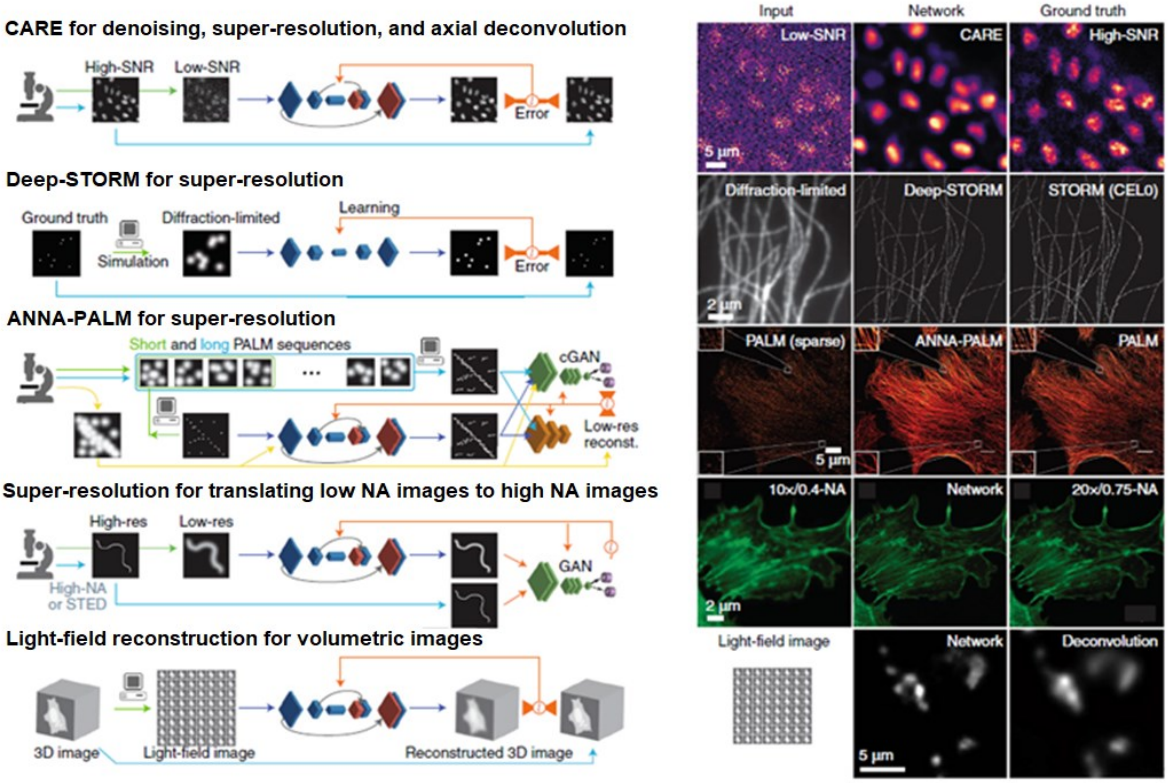

Fig. 1-4. Applications of deep learning techniques for image quality enhancement in fluorescence microscopy, adapted from [13].

1.3.1 Deep Learning Deconvolution without Clean Image

With its superior performance in almost every image processing task Convolution Neural Networks (CNNs) quickly gain recognition in the optical correction community. In early times a CNN is frequently used in deconvolution as a PSF estimator [14] or feature extractor [15]. Its capability of image-to-image deconvolution was not fully realized until a few years ago when the Generative Adversarial Network (GAN) [16] and U-Net [17] were introduced.

One of the earliest image-to-image CNN-based blind deconvolution for confocal fluorescence images was proposed in [18]. A vital aspect of this method is eliminating ground truth in training by applying self-super-resolution, a technique first appearing in [19]. The idea is to increase spatial resolution from information within a dataset, e.g., adjacent pixels of an image or neighboring frames of a volumetric or temporal set. Consequently, the need for clean, undistorted ground truth during training is eliminated.

The proposed method exploited the fact that optical PSF is elongated along the z-direction, so lateral slices of volumetric images exhibit significantly higher resolution and structural contrast than axial slices [18]. Features extracted from a stack of lateral slices along the vertical axis can therefore be used to reconstruct lateral images. Two CNN-based networks were tested: one with basic CNN architecture and the other with a U-Net-like architecture. The networks were trained on synthetic confocal and light-sheet data, whereas the test data consisted of synthetic and real microscope images. The deconvolution results of the two were compared with those obtained through traditional RL algorithms and evaluated using pixelwise peak signal-to-noise ratio (PSNR). Not surprisingly, the deep networks outperformed the RL algorithm, with the U-Net-like model achieving the highest PSNR [18].

A similar approach was advocated in [20] to restore 3D two-photon microscopy images in deeper tissue without prior knowledge of PSF. It adopted a GAN based architecture named spatial constrained cycleGAN (SpCycleGAN). The training data consisted of nuclei-labeled fluorescence images of rat kidneys obtained through two-photon microscopy. The original volumetric data were sliced in three directions: x-y, x-z, and y-z. The generator was trained on a subset of well-defined images selected from the original volume. Restored images were compared to the results obtained from a few traditional algorithms, including RL. As expected, the SpCycleGAN surpassed all other algorithms according to the three evaluating metrics: Blind/Referenceless Image Spatial Quality Evaluator (BRISQUE), the oriented-gradient image quality assessment (OG-IQA), and the microscopy image focus quality assessment (Microscopy IFQ).

These studies pioneered end-to-end machine learning deconvolution of fluorescence microscopy images attempting to minimize the reliance on ground truth data during training. However, their focuses were on the data from more advanced imaging techniques like confocal and light-sheet microscopes; thus, the methods do not apply to miniscope. For instance, the self-super-resolution technique requires information from the axial direction, which is unavailable for widefield images. Although GANs are generally considered unsupervised learning, the training set in [20] contains well-defined, clear images deliberately handpicked by the researchers. This option is also not available for miniscope data.

1.3.2 Self-supervised Learning

Self-supervised learning of signal reconstruction introduced in [21] was built on a simple statistical concept: random errors affect variance but not the mean; true values of a physical quantity can be acquired through a series of unreliable measurements. Logically, learning the

image restoration process by observing corrupted examples without explicit priors or likelihood models is possible. The test result with natural images showed that the self-supervised model can sometimes outperform its supervised counterpart. A more detailed account of this self-supervised learning is given in Chapter 4. One later variation of the same strategy demonstrated that this approach applies to single cell gene expression data [22]. According to [23], a self-supervised denoising neural network can be trained effectively using a single noisy image, and its denoise performance on natural and synthetic images was comparable to the popular denoising algorithm BM3D [24]. However, its performance on widefield fluorescence data was far from satisfactory.

1.3.3 Simple Lens Deconvolution

While more advanced imaging systems like 3D confocal or two-photon microscopy garnered significant attention and research efforts, the less complex devices have been largely neglected. One study addressing the optical correction of images captured with simple lenses adopted a more traditional deconvolution approach [25]. The method involves matching power-wise gradients in different color channels to improve the suppression of color fringing artifacts caused by chromatic aberration in simple lens systems. The proposed algorithm incorporates a convex cross-channel color prior to the deconvolution process, ensuring global convergence. By jointly deconvolving multiple channels and leveraging information from other color channels, high-quality images comparable to those obtained with sophisticated lens systems can be achieved [25]. The idea is similar to the self-super-resolution, except it uses information from other color channels. This method can obtain a high-quality image comparable to sophisticated lens systems. However, same as most traditional deconvolution algorithms, it requires known PSF. Additionally, this method is not directly applicable to miniscope data as it is designed for color images, while miniscope images are typically grayscale.

1.3.4 Modeling Tissue-Optics Dynamics

Computer simulation has been a broadly applied technique in microscopy studies to gain an in-depth understanding of neural dynamics and generate synthetic data for model evaluation. One example is the neural anatomical optical microscopy simulation (NAOMi), which can generate simulated two-photon neuronal images [26]. However, as mentioned earlier, the optics of two-photon microscopy differs sufficiently from that of widefield. The tissue-optics dynamics of widefield are better modeled by the Monte Carlo (MC) model of light propagation [27] [28]. This method has been widely accepted as the gold standard for simulating photon-tissue interaction in a scattering tissue volume. It was adapted in a 3D fluorescence model to evaluate the microvasculature geometry of data acquired from two-photon microscopy [29]. Another variation was devised to characterize the performance of different aperture configurations of confocal microscopy in the human brain [30]. These studies shed light on light-tissue interactions, but rare attempts were made to explore the image formation process. Nevertheless, the NAOMi and MC models constituted two main components in the present study, and a detailed account of the two is given in chapters 2 and 3, respectively.

1.4 Study Overview

The long-term goal of our research is to develop a computational framework (Fig. 1-5) to improve the image quality of miniature microscope data. There will be a total of three technical steps:

1) Creating a neuro-optics model to generate realistic miniscope images.
2) Developing machine learning models that will be trained on the simulated dataset generated in Step 1.
3) Enhancing the miniscope image quality using the pretrained (on synthetic data) model.

This thesis primarily focuses on Step 1: simulation and synthetic data generation. Simultaneously, it also introduces a novel approach by incorporating a deep learning network to expedite the simulation process. Based on the physical principles of light-tissue interaction in the widefield miniscope imaging system, the neuro-optics model consists of three modules: a tissue model, an optics model, and a detector model.

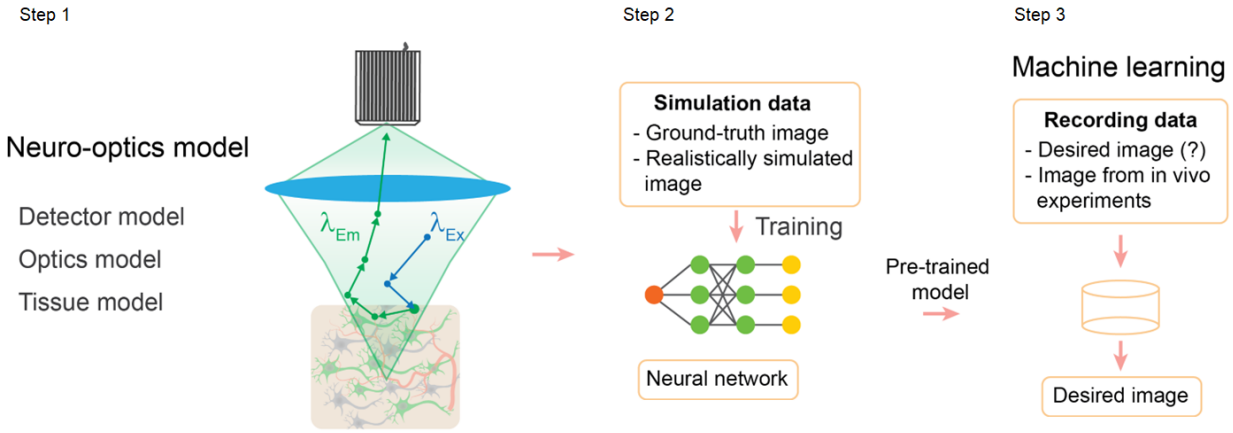

Fig. 1-5. Overview of the current study and long-term goal. Step 1 is to create a neuro-optics simulator modeling the interactions between neural tissue and light propagating inside the brain volume and generating photo-realistic miniscope images. Step 2 involves the training of deep learning networks on simulated images for image quality improvement, including denoising and deconvolution. Steps 3 transfers the deconvolution result to real miniscope data from simulated data.

In the following chapters, a comprehensive groundwork for the components of this computational framework is given. Chapter 2 is dedicated to how the fluorescence microscope works and the NAOMi neural volume generator. A universally recognized MC tissue-optics model is provided in Chapter 3. A few cutting-edge deep networks for image processing are summarized in Chapter 4. The implementation results of this study are presented in Chapter 5. In the end, Chapter 6 summarizes the study with the conclusions drawn from the results.

1.4.1 Motivation

This study is motivated to bridge the gap in research by shifting the efforts towards widefield imaging, specifically the miniscope. The primary focus of the study is to develop a reliable simulation that accurately captures the essential optical characteristics of miniscope images. The simulation consists of elements adapted from the NAOMi and MC models. It aims to generate distorted and clean images for future training of deep neural networks. A self-supervised deep learning network is incorporated into the simulation. This integration serves the dual purpose of accelerating the simulation process and demonstrating the effectiveness of the self-learning strategy in the context of widefield imaging. The objective is to lay the groundwork for applying these techniques in future widefield imaging research.

1.4.2 Significance of This Study

The lack of ground truth has been a significant obstacle in applying data driven techniques to microscopy image reconstruction. The proposed neuro-optics simulation provides a means to generate ground truth images and enables objective evaluation of the restored results. While serving as the benchmark for simulating light propagation in turbid media, it is worth noting that the MC model is highly computationally intensive. The current study offers a promising solution to accelerate the MC process by combining the simulation with a self-supervised denoiser. This innovative approach presents a simple yet effective solution that significantly reduces the time required to obtain a satisfactory synthetic image without resourcing expensive hardware upgrades. The limitations posed by the absence of ground truth can be overcome by utilizing accelerated neuro-optics simulation. This study paves the road for unlocking the potential of state-of-the-art data-driven machine learning techniques in widefield imaging research.

CHAPTER 2 NEURAL TISSUE SIMULATION

Microscopy provides a vital window into the world of cells. However, the accuracy of microscopic imaging is often compromised by optics aberrations, casting doubts on the reliability of acquired data. In silico observation offers a compelling alternative for exploring cellular and subcellular dynamics. Before diving into simulation techniques, it is essential to revisit the principles of fluorescence microscopy to comprehend its complexities and the necessity for studying image formation in a controlled virtual environment. This chapter is devoted to the fundamental principles of fluorescence microscopy, covering essential aspects of excitation and emission processes, and the primary sources of blurring in widefield imaging. Then a thorough introduction to the NAOMi simulation of two-photon microscopy is presented. This simulation holds particular relevance to the present study because a portion of it has been integrated into the proposed neuro-optics model.

2.1 Fluorescence Microscopy

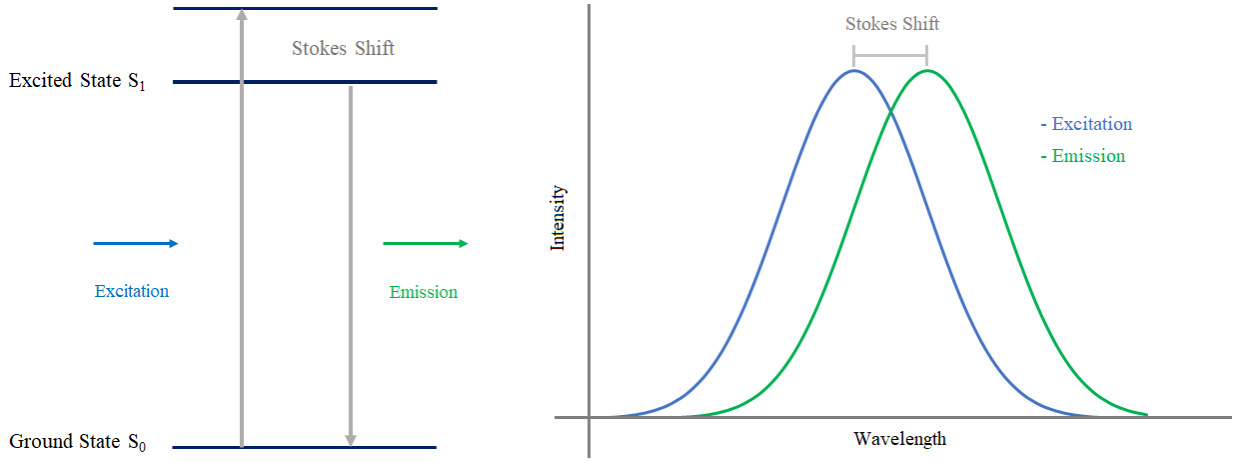

Fig. 2-1. Illustration of fluorophore excitation. Left: Jablonski diagram of fluorescence emission. Fluorophores absorb illumination light, excite electrons to high energy state. When the electrons return to ground state light of slightly longer wavelengths is emitted. Right: Stokes shift equals the difference in wavelengths between excitation and emission.

Regardless of imaging techniques, fluorescence microscopy operates on the same principle of fluorophore excitation. Fluorophores are molecules labeled by fluorescent dyes or genetically encoded fluorescent proteins. When illuminated by a beam of a specific wavelength the fluorophores are excited and emit light of longer wavelengths, seen as fluorescence (Fig. 2-1, left panel). The difference between the excitation and emission wavelength is known as Stokes' shift (Fig. 2-1, right panel).

2.1.1 Illumination in Different Imaging Systems

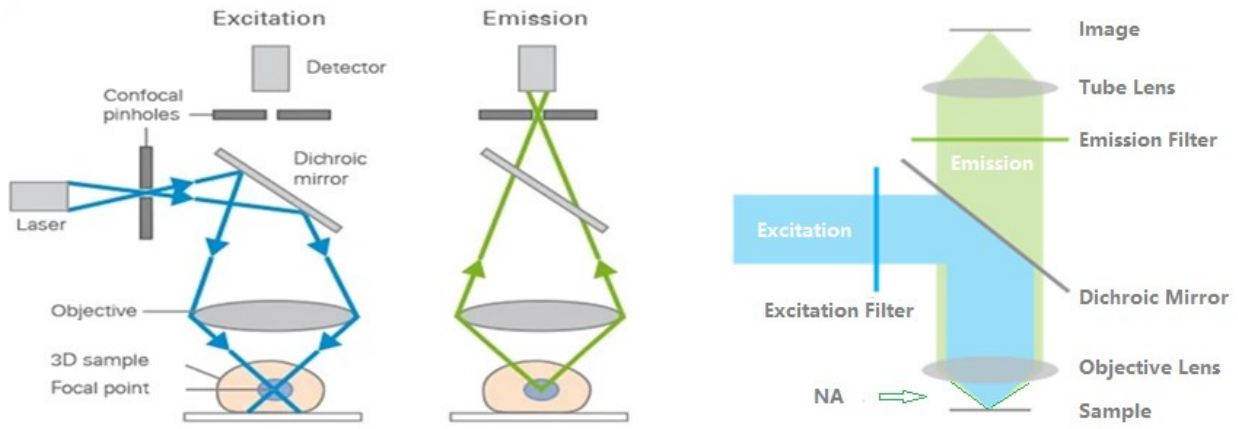

Fig. 2-2. Schematic of fluorescence microscopy. Left: Confocal imaging [31]. A pinhole is placed before the detector to block out-of-focus fluorescence emission. Right: widefield imaging, adapted from [32]. The entire tissue is illuminated and fluorescences within NA, in-focus or not, all being collected by the sensor.

The distinction between imaging systems, such as confocal and widefield microscopy, lies in the imaging process and how the excitation beam is directed onto the sample, illustrated in Fig. 2-2. In confocal microscopy, the excitation beam is focused onto a narrow region, resulting in a correspondingly narrow emission. This is especially true for two-photon imaging, where two beams of longer wavelength target the excitation to a small, localized spot. Excitation and fluorescence emission occurs exclusively at the spot where both beams hit simultaneously, with

the precise location of this spot can be determined by the geometry of two excitation beams. A layered scanning process collects fluorescence signals from focus areas to construct volumetric images. To further enhance image clarity a confocal pinhole is placed in front of the detector to block the out-of-focus signals. In contrast, widefield microscopy illuminates the entire sample, capturing both in-focus and out-of-focus signals within the NA. The divergence in illumination and light collection schemes significantly impacts on the dynamics of tissue-optics interactions among different imaging systems.

2.1.2 Tissue Scattering and Depth Limitations

As mentioned in Chapter 1, tissue scattering determines the depth limit for widefield imaging. Two related metrics: MFP and TMFP, measure the behavior of light transportation in the presence of tissue scattering. The MFP is defined as $MFP = 1/\mu_t$, the inverse of the transport coefficient (μ_t), which equals the total attenuation due to combined effects of scattering and absorption. MFP represents the average length a photon travels without being scattered. The mean distance a photon can relatively maintain its initial direction after multiple scattering collisions is given by $TMFP = 1/\mu_s(1-g)$, where μ_s denotes the scattering coefficient and g the anisotropy factor. The anisotropy factor quantifies the degree of forward scattering and can take on values in the range of [-1, 1]. Positive values indicate a forward scatter tendency, whereas negative values represent backward scattering, and g equals zero signifies isotropic scattering.

While most imaging techniques are subject to depth limit imposed by tissue scattering properties, it is more prominent in widefield imaging. Consider the example of mouse cortex tissue, the MFP is calculated to be 0.047mm (47μm) and the TMFP at 0.265mm (265μm), using parameter values of g = 0.82, $\mu_s = 21\text{mm}^{-1}$ and $\mu_a = 0.33\text{mm}^{-1}$, which are commonly observed in brain

tissue. The focal plane of the miniscope is typically positioned at a depth of 100μm. It means that within this range, multiple scattering could occur, affecting not only the out-of-focus light but also the in-focus signals. Additionally, the fluorescence emission is presumably isotropic, meaning that the initial direction of emitted photons may not necessarily be directed upward. These photons can travel in any direction and still reach the surface due to scattering and contributing to out-of-focus signals. Confocal imaging systems, on the other hand, are less susceptible to these concerns as they are equipped with localized scanning. This capability allows for more precise control over the detection of signals. As a result, the optics processes of the confocal and widefield differ significantly.

2.2 In Silico Two-Photon Imaging

Computer simulation is a versatile tool to investigate complex phenomena by utilizing mathematical algorithms to create virtual environments mimicking real-world scenarios, allowing the analysis of intricate behaviors that are otherwise difficult or impossible to study through traditional methods. As deep learning image improvements continue to gain ground in fluorescence microscopy, simulated data emerged as an appealing approach to address the challenges arising from the lack of ground truth data. One example is the NAOMi simulation, specifically designed for modeling two-photon microscopy. This section delves into the elemental design aspects that contribute to its accurate representation of the imaging process.

2.2.1 The Design of NAOMi Simulation

NAOMi simulation incorporates five individual modules simulating the anatomical volumes of the mouse cortex, temporal spiking activities, and the optics and imaging process of three dimensional (3D) two-photon confocal microscopy [26]. Fundamental to the NAOMi simulator are flexibility and efficiency, driven by the need to expand capacity to tackle various

aspects of the calcium imaging process on a simple standalone machine. The five modules run in sequence, each serving a distinct simulation role. The neuron module is responsible for creating a single neuron to be assembled by the volume module into tissue volume. Then the activity module generates calcium traces for neuropils and neurons. The optics module approximates optical properties and PSF for the scanning module to form image frames of two-photon microscopy. An overview of the NAOMi design is shown in Fig. 2-3.

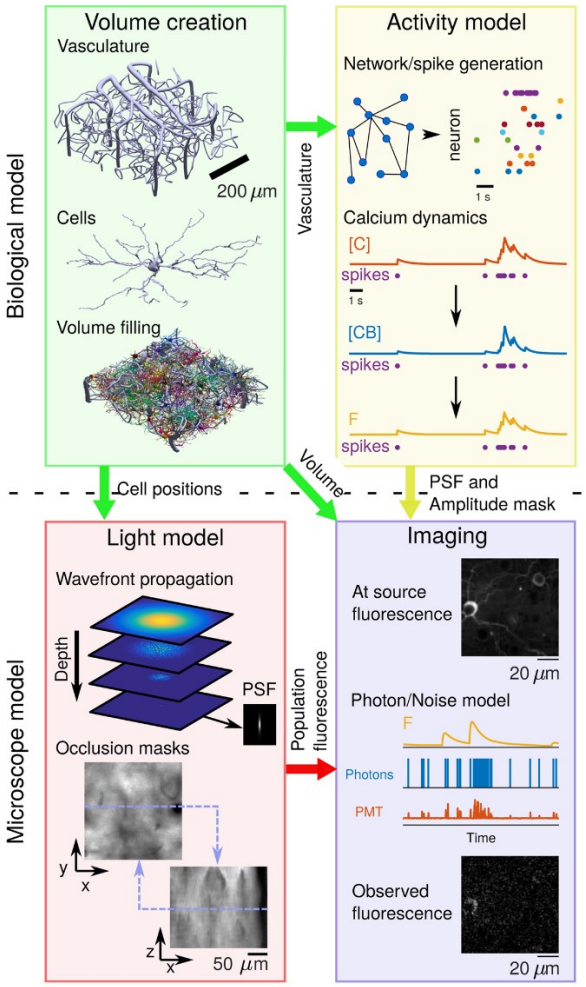

Fig. 2-3. Block diagram of two-photon NAOMi simulator, adapted from [26]. Top-Left: neural volume generation process Top-Right: spiking activity generation Bottom-Left: light propagation model Bottom-Right: scanning and image formation. Colored arrows indicate how modules are related to others.

2.2.1a Neuron module

The creation of neurons begins with shaping cell bodies, i.e., somas, via an isotropic Gaussian process defined over a sphere, where points are sampled uniformly first. It is then followed by sampling independently and identically distributed (i.i.d) Gaussian random numbers for each point. The points are smoothed according to their spatial relationships. The height, i.e., distance from the center of the sphere, is sampled from a Gaussian distribution with the covariance matrix dependent on the distances between the points. $r_i \sim N\ (0, K), Ki, j = e^{-d(P_i, P_j)/l}$, where l denotes the length magnitude applied in controlling the smoothness of the cell body. The distances are calculated using the arc length along the great circle connecting the two random points, P_i, $P_j \in R^3$, on a unit sphere. Between two points the shortest path is defined $d(P_i, P_j) = 2 \sin^{-1}\left(\frac{1}{2}\left\|P_i - P_j\right\|_2\right)$ [26]. Rescaling is applied to radii values to constrain the radial height within the region of realistic deformations. The base radius at each point is treated as a function of its location on the sphere to account for the pyramidical neurons. After the shapes of cell bodies are adequately adjusted, nuclei are generated through shrinking and smoothing the cell walls defined as $r_i^* = (r_{50\%})\,(|r_i| - min_i\, r_i)/(r_{5\%} + r_i) + r_{min} - (\Delta r)_{min}$ [26].

Dendrites are grown to cell bodies by the neuron module but only after the somas have been placed into the neural volume. Once the somas are in position, the dendrites are generated by identifying two endpoints for each dendrite, then iteratively filling the volume in between via a stochastic process while avoiding other neuronal structures already in place. Thicker and more axially oriented apical dendrites are grown similarly, except those in deeper layers are built from the bottom of volume upwards to the surface.

The statistical process involved in neuron creation was carefully tuned according to real morphological data from electron and optical microscopy, ensuring simulated neuron shapes accurately represent the observed characteristics.

2.2.1b Volume module

Anatomical volume is constructed by loading neural components into an empty volume, modeled as 3D grids. The volume is initialized with blood vessels. Surface vasculature is formed through smoothly varied and dilated connections of random nodes on the surface. Diving arterioles join the endpoints of surface vasculature to the volume bottom, with capillaries growing from diving arterioles semi-randomly [26].

Neuron somas generated by the neuron modules are then loaded to fill the space randomly with a predetermined minimum distance. The density of neurons can be controlled through modifications of the random process. Potential overlapping of neurons is solved by a scheme similar to Cuckoo hashing, where newcomers, e.g., the neurons placed later, replace neurons already in place. Dendrites are simulated by the neuron module, attached to the somas sequentially, and steered clear of existing structures. The remaining space in the volume is filled with locally grouped axon segments, which are assigned to cells either by minimizing centroid distance or randomly [26]. The parameters of concentration, diameter, orientation, and branching levels for the vessels are verified to conform with the mouse vasculature data obtained through two-photon microscopy.

2.2.1c Activity module

Two different mechanisms were employed to model the temporal spike trains for neurons. The elementary independent spike activities of fluorescent proteins are obtained through a statistical model simulating the typical behaviors of rise and decay. More dynamic activities of

molecular kinetics over time are modeled by a Hawkes process to account for self-excitation and interneuron correlations [26].

The statistical method simulates the bursting activity of each neuron as a series of rapid and repetitive firing at independent, exponential intervals defined by $P(\Delta t_{burst}) = \lambda_{burst} e^{-\Delta t_{burst}/\lambda_{burst}} \; for \; \Delta t_{burst} > 0$ [26]. The bursting rate λ_{burst} is either predefined or drawn from a Gamma distribution with a given mean rate and fixed shape parameter $\alpha = 1$. The number of spikes for a burst, N_{burst}, is governed by a Poisson distribution $N_{burst} = 1 + Poisson(\lambda_N)$, where λ_N is the parameter determining the bursting length. Inter-spike time is uniformly distributed in the range of [5, 7] milliseconds. Alternative time intervals can be attained via modifications to the parameters λ_{burst}, α, and λ_N [26].

An adjacency matrix is used in the dynamic scheme to correlate interneuron firing via Watts-Strogatz small-world network model. This network captures the dynamics of small dynamic networks by combining regular and random structures. It has a lattice structure with each node (representing a neuron) connected to its nearest neighbors. Randomness is introduced through a rewiring process where nodes undergo random reconnection to distant nodes beyond their immediate neighbors [33]. Rewiring is a one way process in which neurons are allowed to affect background, but the reverse is largely suppressed. This process facilitates local clusters and long-range linkage commonly found in natural neural networks.

The resulting matrix is normalized and undergoes the Hawkes process using Lewis' method. This method is convenient for simulating nonhomogeneous point processes with time-varying intensities [34]. Hawkes model simulates spike trains exhibiting self-exciting behavior, burst, and temporal correlation by combining baseline intensity and excitation function [35]. The

baseline intensity represents the expected rate of spontaneous spikes in the absence of any external influence. In contrast, the excitation function describes how past spikes influence the probability of future spikes. Baseline fluorescence is generated by $\beta_i = |1 + z|$, where z is the Gaussian random variable. Auto regression, AR-p dynamics, with p degrees of freedom, is used to model the calcium and fluorescence impulse response, which is solved by the inverse Laplace transformation to create temporal activity per neuron. The fluorescence level at a specific time is determined by the number of calcium ions bound to the indicators through the binding/unbinding dynamics of free calcium ions [Ca^{2+}] in the cell [26]. The conversion of calcium concentration to fluorescence time traces is done using the Hill equation $\Delta F/F = 1/1 + (K_D/[Ca^{2+}])^{nH}$ with the dissociation constant (K_D) and Hill coefficient (nH) taken from the measured value in [36].

2.2.1d Optics module

At the core of the optics module is the simulation of PSF of confocal microscopy. The shape of PSF is assumed to remain constant throughout the scanned volume but with varied intensity. It means the blurring is independent of location, and optics aberrations are modeled through amplitude modulation. It is approximated by propagating a scalar field across the tissue volume. The scalar field at the front aperture of the objective lens is described as a Gaussian with a circular aperture and spherical phase. Optics aberrations can be added at this stage. Fresnel diffraction integral and the split-step beam propagation method are employed to estimate the field, considering the effect of inhomogeneity within the sample volume.

The optical phase masks are calculated based on the phase differences owing to refractive index inhomogeneity within the sample. These masks are multiplied after every optical propagation step to compute the scalar field at each position. The resultant field is used to obtain the two-photon PSF at each location over the field of view (FOV), which is averaged to derive the PSF

for scanning. For computational consideration, the PSF near the focal plane is sampled at volume resolution, whereas out-of-focus PSF and scaling mask are sampled at a reduced resolution. An absorbance mask was computed based on estimated signal reduction due to vasculature absorption. The spatial signal scaling mask is the product of the absorbance mask and optical excitation mask, which is the sum of PSF intensity over the entire field [26].

2.2.1e Imaging module

The last module creates volumetric image frames for the two-photon microscopy based on the time traces and PSF. The process begins with assigning fluorescence levels to each neuron using fluorescence distribution and time traces. The background level is set similarly by repeating the same procedure on the neuropil. The fluorescence values are convolved with PSF to form raw images and masked with an optical mask to produce initial images. Motion is simulated through either a per-line motion or shearing. Poisson distributed photon shot noise and Gaussian electrical noise are added before signal conversion. Finally, an analog-to-digital accumulator is used to model the bleed effect [26]. Photons incoming in one pixel's accumulation period can result in an analog form that bleeds over to the buildup for the next pixel.

2.2.2 NAOMi data validation

The NAOMi simulator was carefully designed to capture the essential properties of two-photon microscopic data. All parameters used in the simulation were verified to conform with either experimental data or values well established in the literature [26]. Simulated neural activities and fluorescence images were checked against real two-photon microscope recordings. Fig. 2-4 demonstrates the comparison between NAOMi simulated data and two-photon recordings.

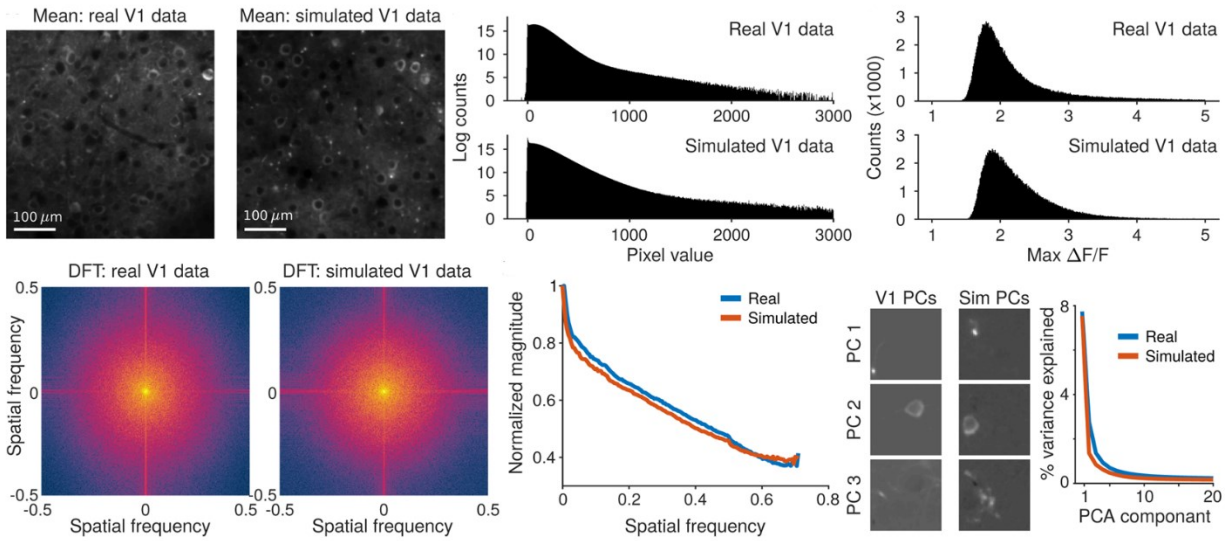

Fig. 2-4. Comparison between NAOMI generated data and GCaMP6f labeled mouse V1 L2/3 recorded by two-photon imaging, adapted from [26]. The results demonstrated that simulated data accurately resembles the characteristics of real data in terms of pixel value distribution, spatial frequency content, maximum neuronal response (ΔF/F), and principal component decomposition.

Undoubtedly, the NAOMi simulation presents a robust framework for evaluating two-photon microscopy, offering a computationally efficient means to generate anatomical volumetric data. Once again, two-photon confocal microscopy optics and image processes differ substantially from those of widefield imaging for miniscope. A distinct model is needed to accurately simulate the intricate tissue-optics interactions and imaging procedures specific to widefield microscopy.

2.3 Virtual Neural Volume

One of the most desirable features of the NAOMi simulator is the modular flexibility, allowing component modules to function independently. This permits the leverage of its accurate tissue modeling capabilities by selectively integrating its tissue related modules into the widefield simulation. Fig. 2-5 shows examples outputs of neural and volume modules. The

volumetric data captures the spatial distribution and characteristics of fluorescent signals within the tissue.

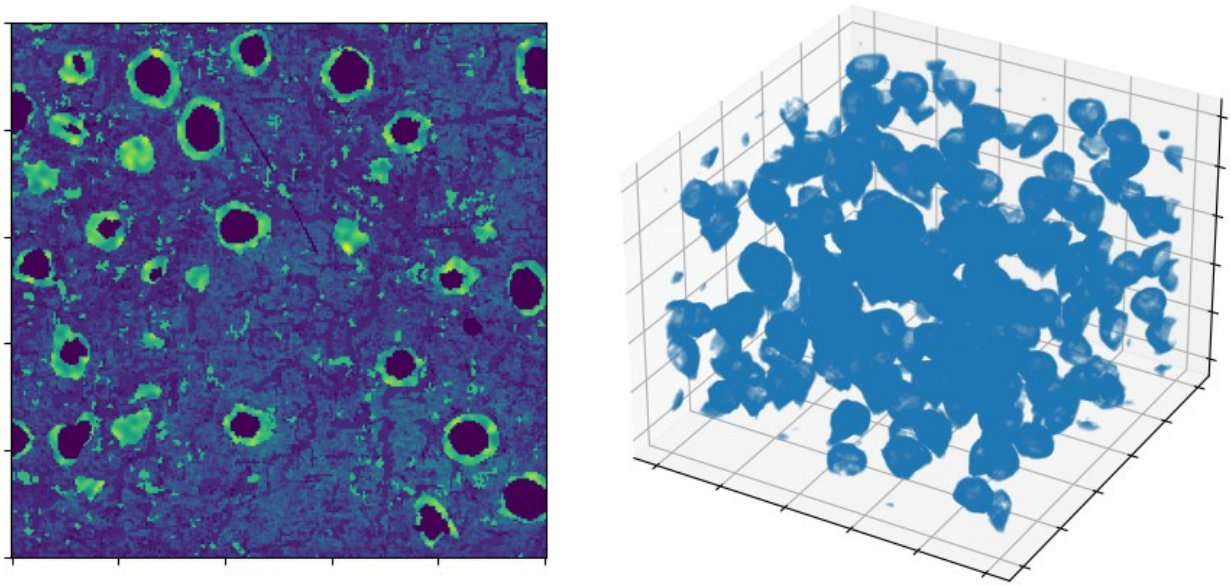

Fig. 2-5. NAOMi generated neural volume. Left: a segment of tissue with fluorescent labeled somas sliced from the volume. Right: isolated somas without fluorescence.

The NAOMi simulation is an open source MATLAB software. It is directly employed in this project to generate brain volume. Next step is to construct an optics process that accounts for the tissue-light interactions in widefield imaging. This is handled by the MC model, introduced in Chapter 3.

CHAPTER 3 MONTE CARLO NEURO-OPTICS SIMULATION

The Monte Carlo (MC) model has achieved extensive acclaim as the leading approach for simulating interactions between light and tissues. This chapter offers a historical overview of the milestones of the MC models and reveals key contributions from pioneers who laid the groundwork for its application in tissue optics. Moreover, this chapter dives into the specific MC method employed in the current study, elaborating on the intricacies of its implementation together with the fundamental principles underpinning the MC simulation.

3.1 Monte Carlo Method

The MC method was initially introduced in 1949 [37] to address physical phenomena involving a cascading chain of semi-random processes. An example of such a phenomenon is the procreative nuclear events resulting from interactions between cosmic rays and Earth's atmosphere [37]. While the likelihood of producing a specific particle with a particular energy in a collision depends on the energy the incoming particle carries, the motion direction follows a stochastic process governed solely by a particular probability distribution. Neither traditional mathematical frameworks via analytical mechanics using differential equations nor statistical schemes based on probability theory are fully equipped to comprehend the behaviors of such a process.

The MC algorithm was designed to bridge the gap between these two classical approaches by combining a deterministic process with a stochastic one into a repetitive procedure. It can be likened to calculating the probability of outcomes in a game of solitaire, assuming the probability of each possible outcome is given, changes reflecting the presumed probability distribution can be materialized by simulating numerous games of chance (thus the name Monte Carlo) [37]. A physical process is simulated as a sequential flow of operations using two sets of parameters that represent the characteristics of the physical event: one set is deterministic and manipulated

algebraically, while the other set is randomly generated based on prescribed frequency distributions. The essential aspect of the MC algorithm is the complete elimination of complex integration with a straightforward sequence of random sampling processes.

At the core of the stochastic process lies the random sampling of uniformly distributed uncorrelated numbers within the range of [0,1]. This set of random numbers can be utilized to derive any desired predefined probability distribution f(x), which, in turn, underlies the stochastically determined value y = g(x) resembling the distribution f(y) [37]. Repeating the sampling process can generate a collection of all possible outcomes, enabling statistical analysis of both genealogical properties and distributions at a specific time. The MC algorithm offers a notable advantage in tackling intricate problems in high-dimensional spaces, where analytical solutions are often unfeasible [38]. Its versatility has led to its application in various scientific fields, ranging from astronomy [39] [40] to microscopy [29] [41].

3.2 Monte Carlo Neuro-Optics Model

The MC model developed in this project is built on the framework established in a series of studies conducted by the research group at the University of Texas in the 1990s [27] [28] [42]. The MC method has proven its capability to accurately simulate tissue-light interactions, yielding meaningful results for various physical quantities, including volumetric heating, fluence rate, and more. With necessary modifications, the proposed MC model aims to provide an effective tool for modeling the neuro-optics dynamics in widefield miniscope imaging.

Overall the MC process repeatedly releasing photons from specific locations in neural tissue. The step size and direction of propagation are determined by the tissue's optical property and i.i.d sampling from a probability distribution. By repeating the procedure a sufficient number of times the statistical pattern of the behavior of light in tissues can be obtained. The simple variance

approach in [28] was adopted to boost computation efficiency. It allows the photons to be treated as a packet constituting any arbitrary number of photons. This feature makes the model more efficient and flexible because it enables the scaling of recorded physical quantities with respect to changes in photon energy without undergoing the simulation process. From this section onwards, "photon(s)" is used interchanbealbe with "photon packet(s)."

3.2.1 Basic Assumptions of MC Neuro-Optics Process

The MC process employed in this study models the propagation of excitation light and fluorescence emission. The excitation light is a collimated beam of specific diameter and wavelength perpendicularly incident onto the tissue surface. The tissue is assumed to be a homogeneous single layer of infinite width and depth. This assumption is reasonable for mouse anatomical tissue in a widefield microscopy environment where the tissue size is sufficiently larger than the FOV. For simplicity consideration, the wave nature of light is omitted; in other words, the diffraction effect is left out. This simplification limited the validity of simulated images because diffraction plays a significant role in widefield image formation; nonetheless, it helps to reduce the computational complexity of the simulation. This strategy still produces satisfactory results for demonstrating the feasibility of applying computer simulations to generate synthetic widefield images and perform subsequent deconvolution.

3.2.2 Mathematical Framework

The deterministic parameters of the model describe the properties of a light beam and tissue layer. Tissue parameters comprise the refractive index, the absorption coefficient μ_a, the scattering coefficient μ_s, and the anisotropy factor g. Light parameters include beam types (e.g., collimated or isotropic) and wavelengths. Since the tissue is assumed to be infinitely deep, only the refractive index of the top ambient medium is given, which is the refractive index of the GRIN lens. Two of

the most critical stochastic parameters are the step size and direction of photon propagation at each interaction site. These are constructed such that a physical quantity to be determined corresponds to the expected value of a given random. It is equivalently an alternative expression for bridging a predetermined probability distribution f(x) to a uniformly distributed variable. It is formulated in [28] as follows. Consider a random variable X whose probability distribution is defined over the interval [a, b], characterized by the probability density function:

$$\int_a^b p(X)dX = 1 \tag{1}$$

Suppose there is a uniformly distributed random variable ξ over the interval [0, 1] described by the cumulative distribution function (CDF) as:

$$F_\xi(\xi) = \begin{cases} 0 & if\ \xi\ \leq 0 \\ \xi & if\ 0 < \xi\ \leq 1 \\ 1 & else \end{cases} \tag{2}$$

Linking the two distributions requires a one-to-one mapping between them, equating the probabilities

$$P\{a < X\ \leq X_1\} = P\{0 < \xi < \xi_1\} \tag{3}$$

Based on the definition of CDF, this leads to

$$F_X(X_1) = F_\xi(\xi_1) \tag{4}$$

$$\int_a^{X_1} p(X)dX\ = \begin{cases} 0 & if\ \xi_1\ \leq 0 \\ \xi_1 & if\ 0 < \xi\ \leq 1\ =\ \xi_1\ for\ \xi_1 \in [0,1] \\ 1 & else \end{cases} \tag{5}$$

$$\int_a^{X_1} p(X)dX = \xi_1 \qquad for\ \xi_1 \in [0,1] \tag{6a}$$

Assuming X = f(ξ) is non-decreasing, this is equivalent to

$$\int_a^{X_1} p(X)dX = 1 - \xi_1 \qquad for\ \xi_1 \in [0,1] \tag{6b}$$

Eq. (6) lays the groundwork for the stochastic sampling procedure using a uniformly distributed random variable ξ.

3.2.3 Geometry and Coordinate Systems

A three dimensional homogeneous gird system represents a tissue volume with infinite width and depth. The tissue surface is the horizontal plane intersecting the origin of the vertical axis, that is, z = 0. To simplify computation, the volume is inverted along the vertical axis, where positive z coordinates represent vertical positions inside the tissue and negative z values vertical positions above the tissue surface. Since the volume is assumed to be infinitely wide and deep, the only boundary between the two media is at the tissue surface.

A three dimensional Cartesian coordinate system is set corresponding to the cubic anatomical volume generated by the NAOMi simulator. It tracks photon propagation, locates fluorescence values of specific neurons and stores simulated physical quantity. Coordinates are denoted by the standard notation of the x, y, z tuple as a unit vector [28]. Two arrays based on the Cartesian system are employed to record photon energy. A two dimensional array, D(x, y), stores the fluorescence photon weights collected by the detector. It reflects the intensity of the fluorescence image. The three dimensional array, A(x, y, z), records the excitation photon energy tissue absorbed. The excitation energy, the NAOMi generated fluorescence values, and a predefined quantum efficiency value jointly determine the fluorescence emission level at each grid point (e.g., voxel).

A moving spherical coordinate system is utilized to sample the directional change of a photon packet. Sampling is performed with respect to the deflection angle (θ) and azimuth angle (ψ) [28]. This coordinate system tracks the most current direction of photon propagation as directional cosines. The x, y, and z components are updated before the next interaction with tissue. For the sake of optimizing time and space efficiency, only the current direction is registered at each step. Photon trajectory can be tracked when necessary by storing the directional cosines at every step.

A cylindrical coordinate system is employed primarily for model validation purpose, aiming to replicate the results from the original Monte Carlo Modeling of Light Transport (MCML) simulation proposed by [28]. It shares the same origin and z-axis with the Cartesian system. This cylindrical coordinate system is implemented as a separate module and is not typically used in the normal execution of the Monte Carlo simulator.

3.2.4 Stochastic Parameters

The stochastic parameters described in this section are derived from the framework summarized previously in section 3.2.2. In actual implementation the random variable ξ takes on values in [0, 1) instead of [0 1], which is a common practice with computer generated random numbers. The mathematical derivation of the stochastic parameters listed in the upcoming subsections follows the framework in [28].

3.2.4a Step size

The step size of the photon packet is governed by the transport coefficient and probability distribution of a photon's free path s ∈ [0, ∞). Given a PDF:

$$p(s) = \mu_t e^{-\mu_t s} \tag{7}$$

According to eq. (6) this can be expressed as

$$\xi = \int_0^{s_1} \mu_t e^{-\mu_t s} ds = 1 - e^{-\mu_t s_1} \tag{8}$$

Rearrange the equation

$$e^{-\mu_t s_1} = 1 - \xi \tag{9}$$

Solve for s_1

$$s_1 = -\frac{ln(1-\xi)}{\mu_t} \tag{10a}$$

Based on eq. (6) it is equivalent to

$$s_1 = \frac{-\ln(\xi)}{\mu_t} \tag{10b}$$

S_1 is the straight distance a photon travels at each move.

3.2.4b ***Directional angles***

The directional change of a photon packet at an interaction depends on the deflection angle (θ) and azimuth angle (ψ). These angles are governed by the phase function P(θ), which describes the intensity of an incident ray normalized to the integral of scattered intensity at all angles [43] and is defined as:

$$P(\theta) = \frac{F(\theta)}{\int_0^{\pi} F(\theta) \sin\theta d\theta} \tag{11}$$

In tissue optics, the phase function is often approximated using the Henyey-Greenstein function in terms of anisotropy factor g, where the probability distribution of deflection angle (θ) is expressed in terms of cosine:

$$p(\cos\theta) = \frac{(1-g^2)}{2(1+g^2-2g\cos\theta)^{3/2}} \tag{12}$$

The cosine of deflection is sampled via random variable ξ:

$$\cos\theta = \begin{cases} \frac{1}{2g}\left[1+g^2-\left(\frac{1-g^2}{1-g+2g\xi}\right)^2\right] & if\ g>0 \\ 2\xi-1 & if\ g=0 \end{cases} \quad for\ \theta \in [0,\pi) \tag{13}$$

The azimuth angle ψ is directly drawn through random variable ξ:

$$\psi = 2\pi\xi \quad for\ \psi \in [0, 2\pi) \tag{14}$$

3.2.5 Deterministic Parameters

The deterministic parameters are related to the optical properties of the tissue that controls the photon propagation and physical quantities of interest, such as the intensity of the image. These parameters include the specular reflectance (R_{sp}), scattering coefficient ($\mu_{s)}$, absorption coefficient ($\mu_{a)}$, albedo, and Anisotropy factor (g).

3.2.5a Specular reflectance

Specular reflectance refers to the reflection at the boundary between two media possessing different refractive indices [44] [45]. In the current model, the tissue and bordering glass (GRIN lens) are both assumed to be homogenous, each having uniform optical properties. As a result, the specular reflectance takes place only at the surface when fluorescence signals transmit from tissue to glass. The specular reflectance is defined as:

$$R_{sp} = \left(\frac{n_1 - n_2}{n_1 + n_2}\right)^2 \quad (15)$$

where n_1, n_2 denotes the refractive indices of the two-bordering media. The specular reflectance decrements the photon's weight during its propagation, detailed in 3.2.6c.

3.2.5b Scattering coefficients

The scattering coefficient (μ_s) gives the likelihood of photon scattering per unit length traveled. It dictates a photon's interaction with tissue and the subsequent change in direction [28]. The unit of scattering coefficient in this study is specified in mm^{-1}.

3.2.5c Absorption coefficients

The absorption coefficient (μ_a) represents the likelihood of photon absorption per unit length traveled in the tissues. It quantifies the amount of light energy absorbed by the tissue and subsequently converted into heat. The unit of absorption coefficient is the same as the scattering coefficient.

3.2.5d Albedo

Albedo (α) is the ratio of scattering to the total attenuation of light in the tissue. Its value ranges between 0 and 1, with higher values signifying a higher scattering-to-absorption ratio. It is calculated as follows:

$$\alpha = \frac{\mu_s}{\mu_a + \mu_s} \in [0, 1] \quad (16)$$

The albedo directly controls the excitation energy deposited in the tissue, determining the fluorescence emission level.

3.2.5e Anisotropy factor g

The anisotropy factor (g) provides an aggregate description of a tissue's scattering profile. It specifies the probability of a photon being scattered in a specific direction. The anisotropy factor is wavelength dependent. In tissue optics g typically ranged from 0.8 to 1. The values experimented in the current MC process are based on the measurements documented in [46].

3.2.6 Photon Propagation

The overall propagation directions for excitation light and fluorescence emission are nearly the opposite. The excitation beam starts at the tissue surface, with its initial direction set straight downward. In contrast, fluorescence emission travels from tissue to surface. The fluorescence values of NAOMi generated neural volume command the initial emission location. The emission exhibits isotropic characteristics, with the tissue's anisotropy factor regulating directional changes after each scattering event. The distance and direction are updated at every interaction site by sampling random variable ξ as described in 3.2.4.

3.2.6a Updating photon position

The current position of a photon packet is represented by two vectors, the Cartesian coordinates (x, y, z) and directional cosines (μ_x, μ_y, μ_z). The position vector is updated by:

$$x' = x + \mu_x * s$$
$$y' = y + \mu_y * s \quad (17)$$
$$z' = z + \mu_z * s$$

where x', y', and z' denote the vector components of a new position. In actual implementation, the position vector (x, y, z) is updated inline, and only the current position is tracked. Similarly, the directional cosines are updated as follows:

$$u'_x = \begin{cases} \sin\theta\cos\psi & if\ |\mu_z| > c \\ \frac{\sin\theta}{\sqrt{1-\mu_z^2}}\left(\mu_x\mu_z\cos\psi - \mu_y\sin\psi\right) + \mu_x\cos\theta & else \end{cases}$$

$$u'_y = \begin{cases} sin\,\theta\,sin\,\psi & if\ |\mu_z| > c \\ \frac{sin\,\theta}{\sqrt{1-\mu_z^2}}\left(\mu_y\mu_z\,cos\,\psi + \mu_x\,sin\,\psi\right) + \mu_y\,cos\,\theta & else \end{cases} \quad (18)$$

$$u'_z = \begin{cases} \pm\,\mu_z\cos\theta & if\ |\mu_z| > c \\ -\sin\theta\cos\varphi\sqrt{1-\mu_z^2} + \mu_z\cos\theta & else \end{cases}$$

where c = 1-10^{-12} represents a nearly vertical incident. The border condition is checked every time the photon packet moves. The z components of the position vector and directional vector provide information about whether the photon packet reaches the surface and if it falls within the NA. Every position vector represents a photon-tissue interaction site where absorption and scattering occur, resulting in attenuation of the photon weight.

3.2.6b Internal reflection

Reflection or transmission occurs at the interface of two media. In this case it is the top tissue surface where z = 0. Since there is only one boundary condition it can be checked by monitoring the value of the z coordinate. The internal reflectance R_i is given by

$$R_i = \frac{1}{2}\left[\frac{sin^2(\alpha_i - \alpha_t)}{sin^2(\alpha_i + \alpha_t)} + \frac{tan^2(\alpha_i - \alpha_t)}{tan^2(\alpha_i + \alpha_t)}\right] \quad (19)$$

Whether a photon packet is transmitted or being internally reflected is determined stochastically by comparison of R_i to a number randomly drawn from random variable ξ. A photon packet is internally reflected if $\xi \leq Ri$; else, it is transmitted.

3.2.6c Photon weight

The weight of a photon packet at the origin is always initialized to 1 and decreases as the packet propagates in the tissue. This reduction represents the loss of photon energy. The drop in photon weight resulting from tissue absorption is:

$$\Delta W = W\left(1 - \frac{\mu_s}{\mu_t}\right) \tag{20}$$

Another factor reducing the weight is the specular reflectance:

$$W = 1 - R_{sp} \tag{21}$$

The weight of a photon packet immediately falls to zero if the packet escapes, which occurs exclusively at the tissue surface. Owing to the infinite tissue volume, a photon packet can travel almost forever as long as it stays away from the surface, even though its weight has diminished to a negligible level. Allowing such photons to continue the transportation no longer helps improve the simulation's accuracy but sufficiently increases computational intensity. Therefore, it is necessary to terminate the photon if its weight drops below a predefined threshold value. In [28] a technique called roulette is used to prevent skewing the distribution of photon deposition by improper termination. This technique terminates photons in an unbiased manner and conserves energy by granting the photon packet a second chance:

$$W = \begin{cases} cW \; if \; \xi \leq \frac{1}{c} \\ 0 \quad else \end{cases} \tag{22}$$

The weight of surviving photon packet is reset based on a predefined chance factor. The roulette method is retained in this MC neuro-optics model.

3.2.6d Procedure

Photon propagation involves launching a photon packet from a specific location. When a photon is moved to a new position, the boundary condition is checked to update the weight. Meanwhile, the direction of the escaped photon is compared to the NA to determine whether it is

detected by the sensor. Once a photon packet is moved to a new position it experiences absorption and scattering. As a result, its energy level drops and propagation direction changes. When a photon's energy level falls below a certain threshold the packet undertakes a roulette process to determine if it will be terminated. Algorithm 1 summarizes the propagation procedure of a single photon packet.

Algorithm 1

Algorithm Propagation(P,O):		
Input:	Dictionary P contains optics parameters 1D position vector O represents the launching point	
Output:	3D array Ai storing excitation energy deposited at each voxel 2D array Di storing fluorescence intensity collected by the sensor	
1	Ai ← zeros	Initialize 3D array Ai with the shape of neural volume
2	Di ← zeros	Initialize 2D array Di with the shape of image
3	w ← 1.0	Initialize weight factor for photon packet
4	(x,y,z), (μ_x,μ_y,μ_z), R_{SP} ← initialization()	Initialize coordinate, direction, specular reflectance
5	**while** w > 0.0	
6	(x,y,z), w, Di ← move()	Move photon packet with one step and update Di
7	w, Ai ← deposit()	Deposit energy at the interaction site (equals to absorption)
8	(μ_x,μ_y,μ_z) ← rotate()	Obtain new propagation direction after being scattered
9	w ← roulette()	Check weight level
10	**return** Ai, Di	Ai, Di will be accumulated for all launching locations

3.3 Optics and Detector model

This section discusses a seamless integration of the detector model into the MC neuro-optics simulation. It begins with highlighting the fundamental optics principles that underpin the MC process. Next, the role of CMOS sensors is introduced, elucidating how they facilitate the conversion of photons into electrons for data acquisition. Additionally, significant sources of noise associated with CMOS sensors are examined.

3.3.1 The Optics

The excitation light is assumed to be a column of infinitely narrow beams injected into the tissue at the surface orthogonally. Each constituent beam is implemented as a photon packet

launched at a specific grid point (x, y) on the tissue surface. During its journey in the tissue a photon packet constantly experiences attenuation at every interaction site, specified by (x, y, z). Excitation energy deposited at the site is assumed to be re-emitted as fluorescence based on a predefined quantum efficiency. The default value is 100%, though not very realistic; however, considering the photon packet eventually consists of any arbitrary number of photons a linear relationship exists between the photon energy and quantum conversion efficiency. Consequently, a more realistic value can be obtained by scaling the output; in other words, quantum efficiency plays more of a theoretical role in the process.

The excitation and emission processes are set to run independently to promote the simulation's flexibility and efficiency. The outputs of each process are stored apart in separate files that can be accessed by the other. The segregation of the two processes does not compromise the validity of the outputs because the excitation beam is in the blue spectrum, while the emitted fluorescence is in the green spectrum. This arrangement remains consistent with the miniscope configuration, where the dichroic mirror filters the two. Whether a fluorescence photon packet is made to the NA depends on the z component of directional cosines μ_Z. The photon packet is collected if $\sin^{-1}(\mu_z) \leq \sin^{-1}(NA/n_2)$.

3.3.2 Detector Model

Miniscope employs a CMOS sensor for photon detection. Its on-chip analog-to-digit converter performs element-wise charge-to-voltage conversion at a pixel site. The number of electrons converted is the product of the number of photons reaching the detector and quantum efficiency. This quantum efficiency is generally provided by the manufacturer and is not the same as excitation-emission conversion. An amplifier is usually exploited to boost the electrical signal. Since both quantum efficiency and amplifier are constants and photon packets can be any

arbitrary number, the conversion eventually does not impact the final image and becomes unnecessary.

There are several noise sources in CMOS sensors, with the single most prominent one being the fixed pattern noise that arises from the variations in photoelectrons produced by the sensor responding to the same number of striking photons. The effect is modeled in [47] by

$$K_p = \gamma_p Pois\left[S_p(\tau)\right] + N(0, \sigma_R) + \beta_p \quad (23)$$

K_p is the noise value of the pixel. The Poisson distributed S_p is a function of exposure time (τ). $Pois[S_p(\tau)]$ is the Poisson-distributed photon shot noise. The multiplicative factor γ_p and reference voltage β_p both can be calibrated. In computer simulation it is modeled by:

$$K_p = \gamma_p \varsigma + \eta + \beta_p \quad (24)$$

where ς and η are random numbers draw from the Poisson and Gaussian distributions, respectively. By setting $\gamma_p = 1$ and $\beta_p = 0$, the model is simplified to

$$K_p = \varsigma + \eta \quad (25)$$

Eq. $K_p = \varsigma + \eta$ (25) shows that the noise can be added by sampling two values from Poisson and Gaussian distributed random variables.

3.4 Implementation

The MC neuro-optics simulator is implemented in Python code with the virtual detector wrapped into the MC process. A contrast between the miniscope setup and the MC simulation is illustrated in Fig. 3-1.

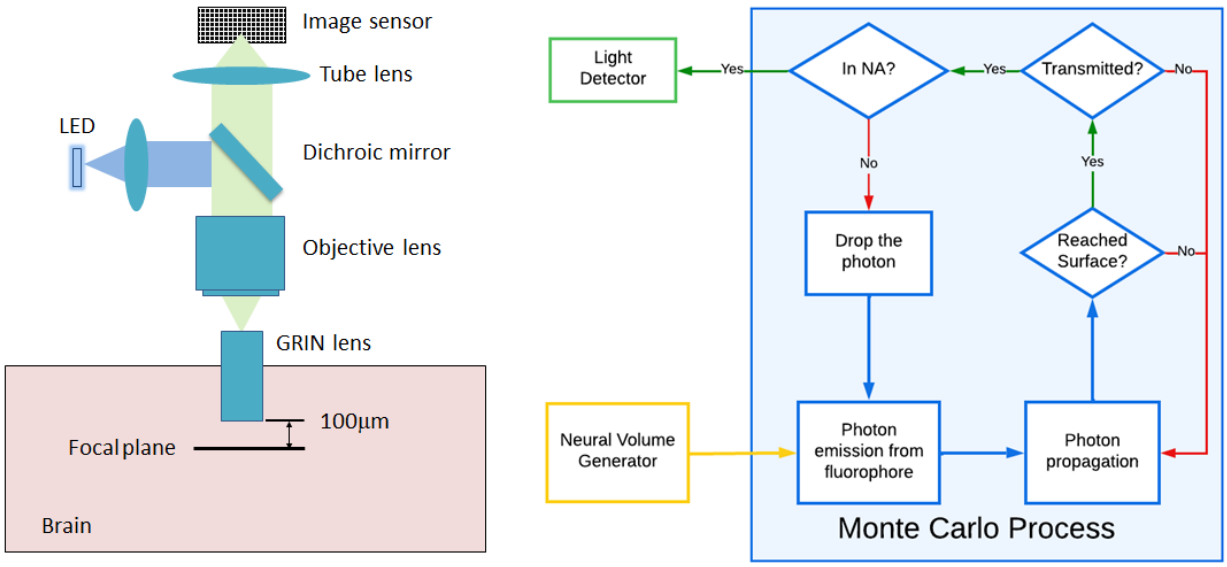

Fig. 3-1. Optical system setup and MC simulation flowchart. Left: schematic of miniscope setup for brain imaging. The dichroic mirror separates emission light from excitation light. This simplifies the simulation as the excitation and emission can be run separately. Right: flowchart of MC simulation of fluorescence emission.

The reason to choose Python is because its NumPy array data structure is convenient for storing image data. Furthermore, most popular machine learning library packages like TensorFlow and Scikit Learn are Python based. These packages provide comprehensive tools for model construction and data visualization; therefore, Python offers better portability and more flexibility.

The primary concern with Python is its runtime efficiency because it is generally slower than compiled languages like C. This is especially vital for the MC process because the simulation itself is computationally extensive. One solution to accelerate the process is a parallel scheme similar to the one in similar to [48]. Another factor contributing to the slowdown is the stochastic nature of the process, which requires a sufficient number of repetitions to achieve the desired statistical pattern. As depicted in Fig. 3-2, the noise level in the output decreases with increasing number of iterations. To minimize the number of iterations and further expedite the

simulation, a deep learning denoiser is exploited to denoise the outputs of the MC simulator. This denoiser is discussed in Chapter 4. The Python codes of MC simulator and deep learning denoiser are provided in the appendix.

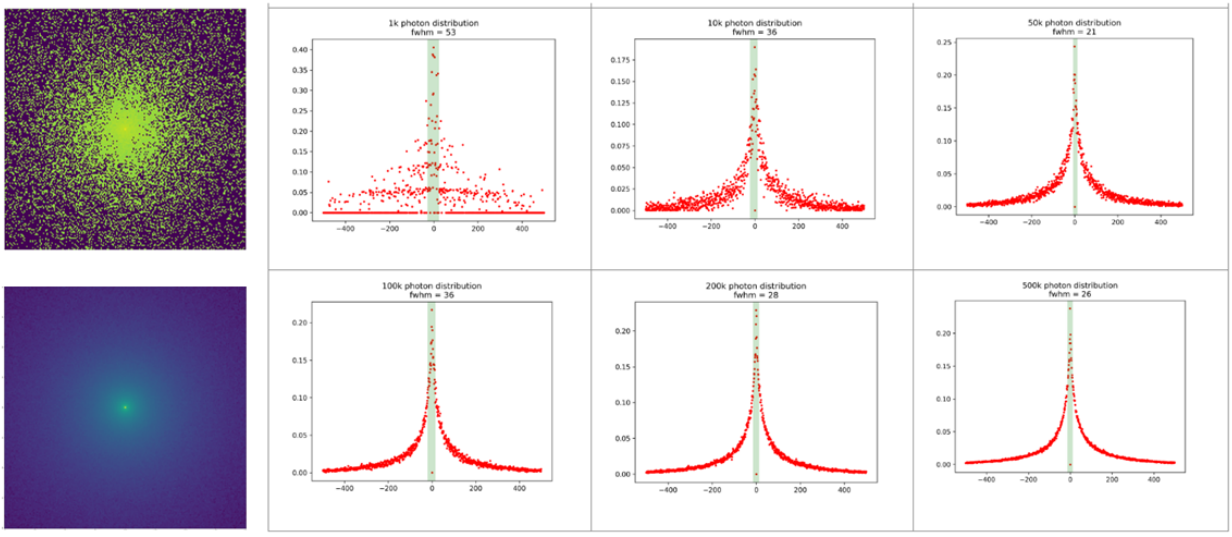

Fig. 3-2. The MC simulated image and distribution of various numbers of photons. Left: a contrast between the images of a point generated by launching 10^4 photons (top) and 10^6 photons (bottom). The point appears more focused as the number of photons increases. Right: the photon distributions of various numbers: 10^3, 10^4, $5*10^4$, 10^5, $2*10^5$, and $5*10^5$. As the number of photons increases, the distribution becomes progressively more concentrated, indicating reduced noise levels.

CHAPTER 4 DEEP LEARNING DENOISING

Although deep learning networks have gained significant attention in image enhancement, supervised learning approaches have been less appealing in microscopic image quality improvement due to the challenges associated with the need for ground truth data. Chapter 1 presented multiple self-learning strategies to address this limitation and reduce the reliance on ground truth data. This chapter focuses on three key concepts: CNN, U-Net, and self-supervised learning, which collectively contribute to the denoiser employed in this project. Then the MC outputs will be reviewed before outlining the denoiser's architecture design.

4.1 Convolutional Neural Networks

Almost immediately after its first appearance [49] CNN architecture wins the dominant position in image processing with its ability to take raw images as input instead of feature vectors. The prototype CNN classifier is a backpropagate network consisting of three hidden layers; the first two are convolution layers and the third is a fully connected layer. The input layer accepts normalized images of size 16 x 16. The output layer contains ten output units yielding the probability of ten possible outcomes. Both convolution layers employ twelve 5 x 5 kernels facilitating a down sampling process to extract localized feature maps with position information mostly discarded. This reduction in information may appear counterintuitive, but it eventually proves to be an elegant strategy. By ignoring the position feature the network can generalize local features globally to broader contexts.

This CNN classifier was trained and tested on a dataset of over 9,000 segmented handwritten zip codes, splitting into a training and test set at roughly an 8:2 ratio. It achieved an error rate as low as 0.14% with only 23 training cycles. However, the test error of 50% suggests the network was heavily overfitted [49]. Despite the poor performance the network opens the door for end-to-end

learning machines in image processing. The dataset was later extended to comprise 60,000 images of handwritten digits. It became one of the standard databases widely known as MNIST data and is frequently used to train machine learning models. This project utilizes the MNIST dataset to help design and finetune the denoiser architecture.

4.2 U-Net

The U-Net architecture [17] further advances the CNNs to tackle image-to-image tasks. It was initially devised to address cell segmentation challenges with insufficient training data. Encouraged by its stunning performance, the U-Net architecture found widespread adoption in various image processing applications including deconvolution and denoising. It gains the name U-Net by its U-shaped architecture that typically consists of an encoder-decoder pair that performs feature contraction and expansion. The encoder is usually a conventional convolutional network comprising multiple convolutional layers, each followed by a pooling layer. It obtains hierarchical attributes from the input images and acquires contextual information through downsampling. The decoder is like an inverted CNN that mirrors the encoder with upsampling convolution layers. It reconstructs the image by merging the feature maps derived from the contracting process. The most distinctive trait of U-Net is the skip connection, which helps to improve accuracy by preserving high-level context. The U-Net configuration is demonstrated in Fig. 4-1.

The U-Net architecture serves as the prototype for the denoiser implementation in the current study, aiming to effectively remove the noise of MC outputs and speed up the overall simulation process. This denoiser's architectural design is detailed in section 4.4.2.

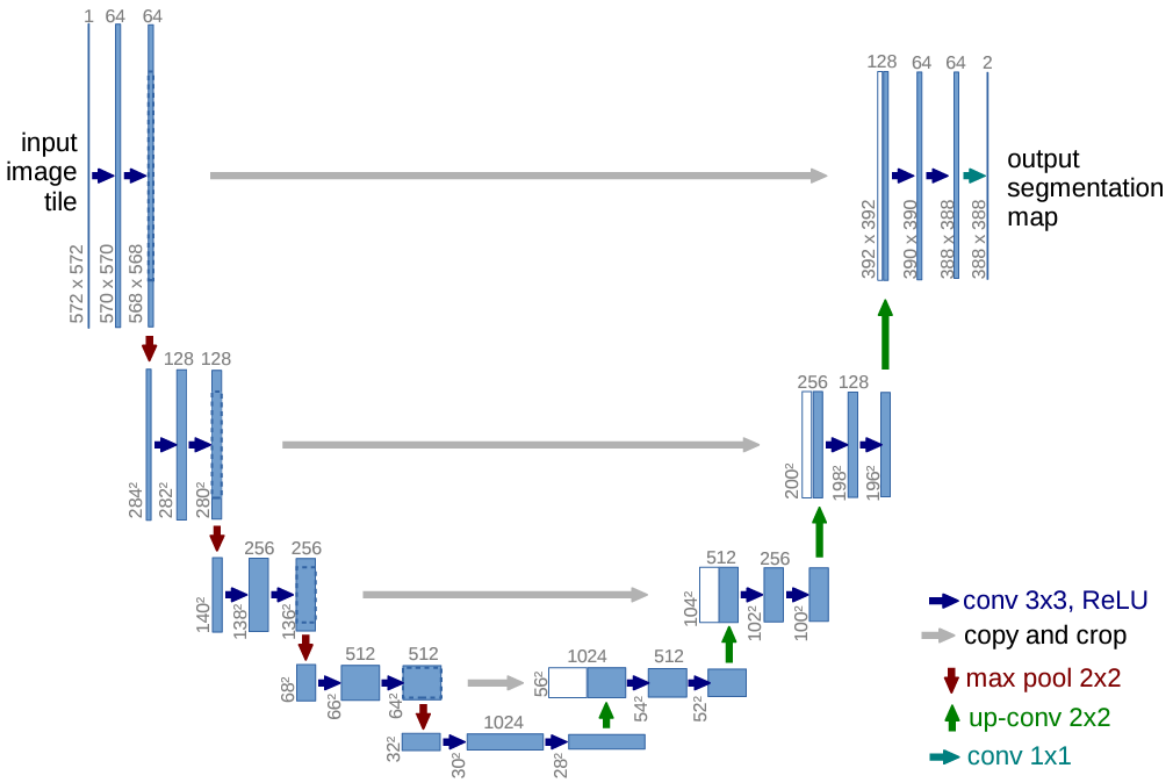

Fig. 4-1. U-Net architecture, adapted from [17]. Blue boxes correspond to multi-channel feature map with the number of channels specified on top of box. Feature maps copied from preceding layer. Skip connections are represented by gray arrows.

4.3 Self-Supervised Learning

A profoundly innovative self-supervised learning paradigm is introduced in [21] to eliminate the reliance on ground truth by applying the fundamental statistical assumption to signal reconstruction. It proves that a deep learning network can learn image restoration solely by observing corrupted examples without explicit priors or likelihood models (i.e., self-supervised).

The underlying idea of self-supervising is straightforward: to approximate a physical quantity from a set of unreliable measurements, say the diameter of a disc, a common strategy for estimating the true value is to find a value (x) that minimizes average deviations from the

measurements through regression. Given a loss function $L(x, y) = (x - y)^2$, where y is the measured value, the minimum error is found at the arithmetic mean of the observed measurements. Since random errors tend to have a mean residual error of zero, the true value can be recovered by minimizing the loss function L(x, y). This concept is illustrated in Fig. 4-2.

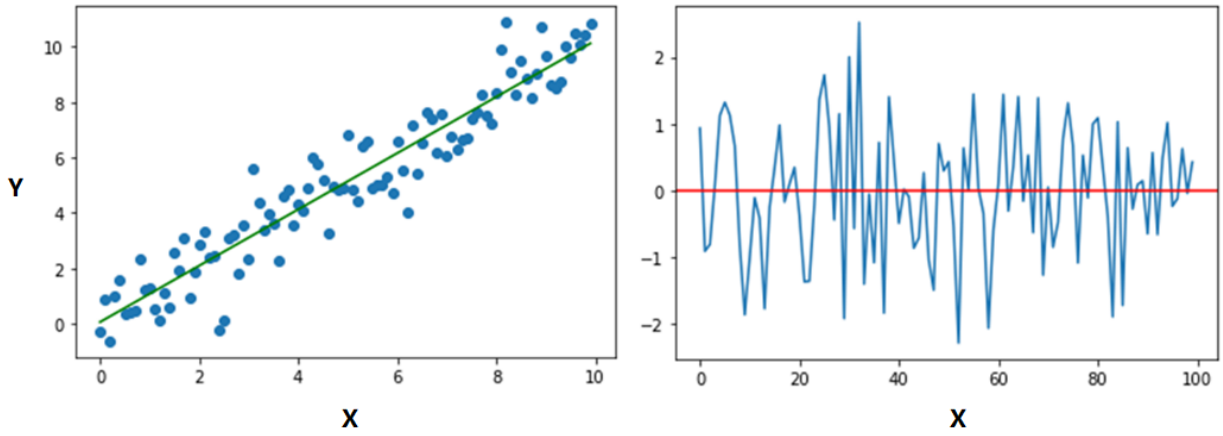

Fig. 4-2. A set of unreliable measurements associated with random errors. Left: linear regression. Right: residual errors

The self-supervised model in [21] expands the original U-Net architecture with additional convolutional layers. Generally, the deeper the networks the more powerful it is. Comparison between supervised and self-supervised learning strategies was evaluated based on three tasks: removing photographic noise, denoising MC rendered synthetic natural scene images, and reconstructing under-sampled Magnetic Resonance Imaging (MRI) scans. The denoised results show that self-supervised can sometimes outperform its supervised counterpart trained on clean data, as measured by PSNR [21].

However, when tested MC generated fluorescence images the model failed to produce any meaningful output. The most likely reason is that the network was designed to handle three-channel images; it is overly sophisticated for grayscale fluorescence images.

4.4 Deep Learning Denoiser

The noise associated with MC generated images arose from the randomness of the MC process. It makes the resulting images a perfect candidate for the self-supervised denoising. Consider other acceleration techniques, such as distributed multi-processing; deep learning denoising offers a faster solution. In order to better comprehend the rationale behind the denoiser it is necessary to briefly review how the MC algorithm works and its resulting outputs.

4.4.1 Outputs of MC Simulator

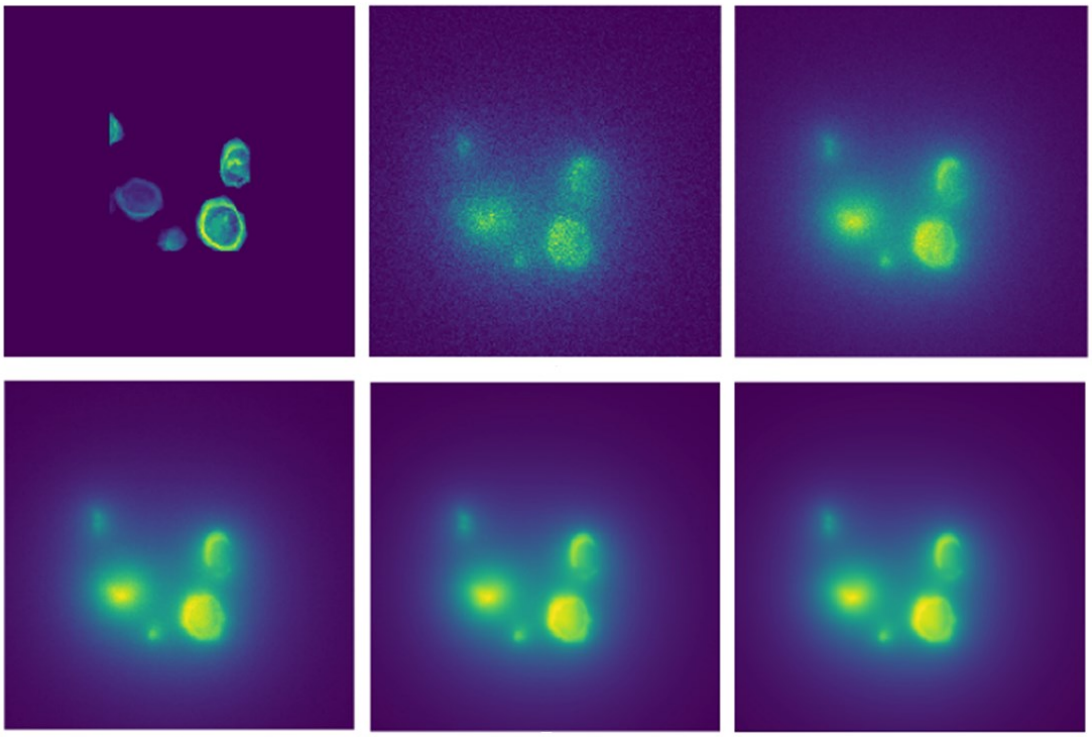

Fig. 4-3. Images of somas simulated by launching varied numbers of photons per voxel. The top-left panel is a top-view of somas created by NAOMi, serving as a based reference. Starting from top-middle the number of photons launched per voxel are 10^2, 10^3, 10^4, 10^5, and 10^6, in the order of left to right and top-down. Since cells at deeper positions appear to be blurrier in an image due to the depth limit, the depth information is indirectly reflected in the MC generated fluorescence image. Therefore, the plain view of NAOMi somas is not directly comparable to MC simulated images. It serves as a visual reference for the overall shape of somas but cannot be used to evaluate the accuracy of fluorescence images.

As mentioned in Chapter 3, the MC algorithm involves a stochastic process that demands many trials for the model to converge. In the case of photon propagation numerous photon packets must be launched to minimize the noise in the simulated result. Example outputs of MC simulator are shown in Fig. 4-3.

Table 4-1 Time required to generate a soma image with various number of photons

Number of Photons	10^2	10^3	10^4	10^5	10^6
Time in Minutes	29	54	185	1,245	14,880

As mentioned in Chapter 3, the MC algorithm involves a stochastic process that demands many trials for the model to converge. In the case of photon propagation numerous photon packets must be launched to minimize the noise in the simulated result. Example outputs of MC simulator are shown in Fig. 4-3.

Table 4-1 shows that the time required to generate an image increases exponentially as the number of photons increases. Because of the background fluorescence associated with the NAOMi neural volume data, it typically takes a sufficiently extended time to simulate an image of neural volume than isolated cell bodies (somas), which do not have fluorescence values attached. In order to produce a reasonable amount of training data for the denoiser, the training set contains images of somas only.

It is worth noting that the plain view of NAOMi somas does not conserve the depth information, illustrated in Fig. 4-4. In addition, the synthetic images of somas differ from those simulated from whole volume data, where the fluorescence labels determine the number of photons to launch, but this is not the case for somas. Further elaboration is discussed in Chapter 5.

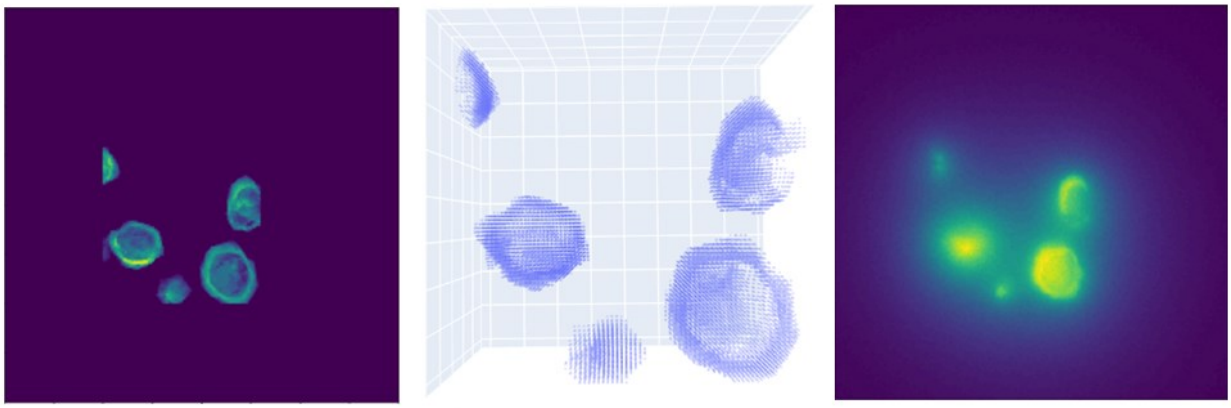

Fig. 4-4. Isolated somas viewed from different perspectives. Left: plain view of NAOMi generated somas. Middle: 3D view from top. Right: MC simulated image of the same somas. Comparing the plain view to the 3D view it can be seen that density of somas is preserved in the plain view, but depth information is lost.

4.4.2 Denoiser Design

The deep learning denoiser adopted the U-Net architecture of the encode-decoder pair, except it does not have a skip connection. The contraction path consists of three 2D convolutional layers each accompanied by a maximum pooling layer and Rectified Linear unit (ReLU) activation. The expansion path mirrors the encoder with three 2D transpose layers, each employing a 2 x 2 strides and ReLU activation. The input layer takes images of size 120 x 120, the same as the output layer. The denoiser design is shown in Fig. 4-5. The denoiser will adopt the self-supervised strategy introduced in 4.3. The learning process is evaluated by a categorical cross-entropy loss function. The actual implementation is given in Chapter 5

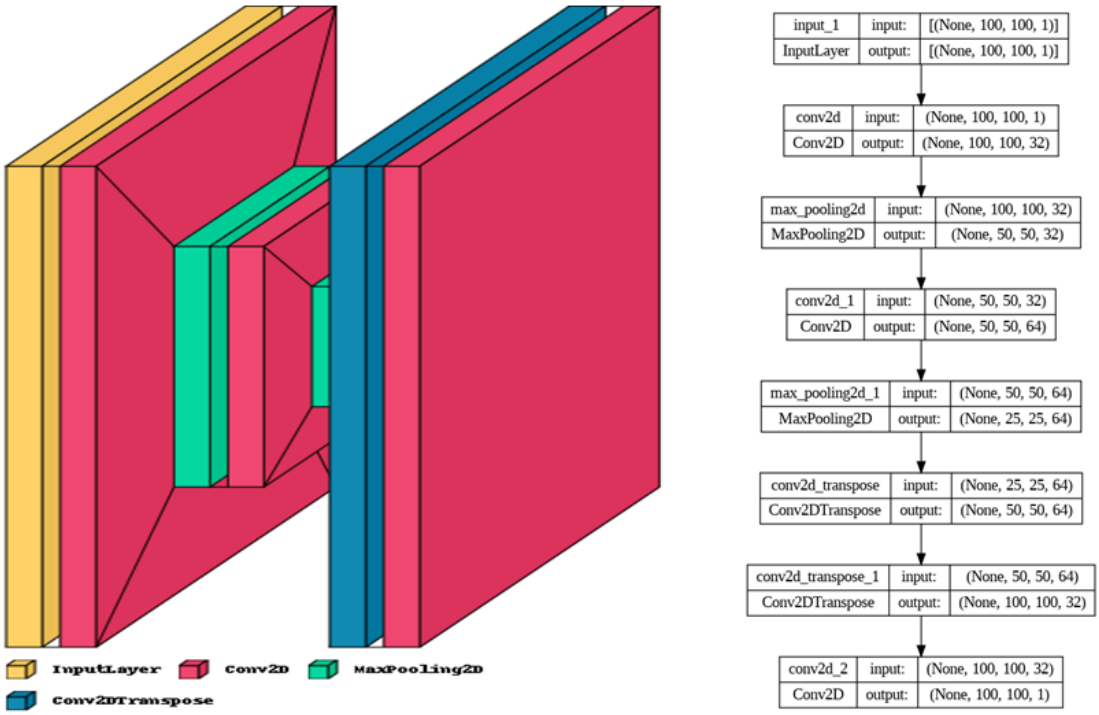

Fig. 4-5. Denoiser design. Left: the encode-decoder configuration. Right: the model graph

CHAPTER 5 METHODS, RESULTS, AND DISCUSSION

This chapter begins by outlining the integration process of the three modules comprising the widefield miniscope simulator. Flowed by the validation of the MC model and the data generation process is provided. An analysis of synthetically generated miniscope fluorescence images and subsequent denoising results are presented next. While a brief discussion is included in this chapter, a more comprehensive examination is reserved for Chapter 6.

5.1 Methods

The computational framework developed in this project consists of three main components: a neural volume generator (NAOMi simulation), a neuro-optics simulator, and a deep learning denoiser. The miniscope detector simulator is wrapped into the MC neuro-optics simulator and is given the name MC Neuro-Optics Imager is used to reflect this integration. The neural volume generator is an open source MATLAB software. The imager and denoiser are implemented in Python. A lightweight bridging program (Bridger) links the three core elements, as depicted in Fig. 5-1.

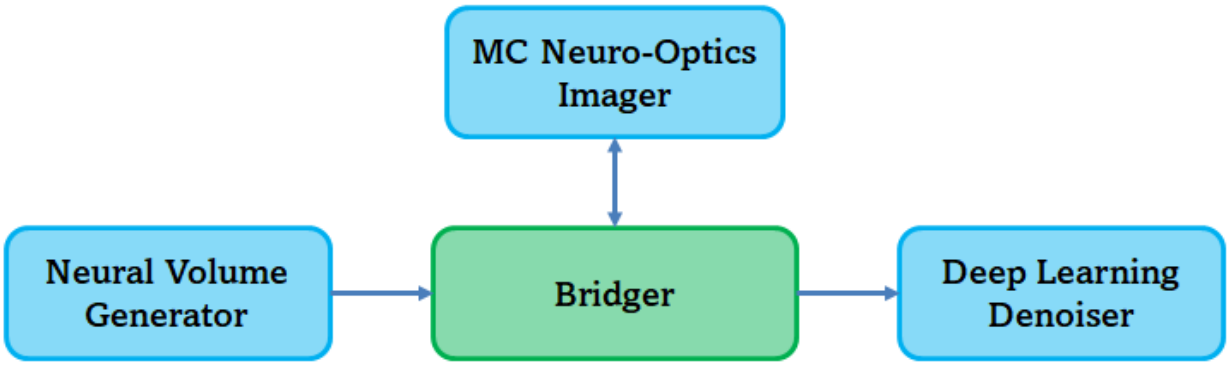

Fig. 5-1. Schematic of widefield tissue-optics simulation process. The outputs of each module are stored in separate files. The Bridger has access to all and is responsible for managing these files.

This modular arrangement provides excellent flexibility and portability as each module can be executed separately on a machine that best fits its computational demand. Among the core components NAOMi neural volume generator is the least computationally demanding one and runs effortlessly on any laptop with moderate computational power. In contrast, the MC imager is highly computationally extensive while requiring zero interactive I/O operations. The high performance computer cluster is thus the most suitable environment for the MC module. The deep learning denoiser requires graphical processing unit (GPU) power during training. The denoiser training is conducted in Google Colab to take advantage of the advanced GPU offered by Google. The runtime presented in the results is the time required to run on Tesla T4 GPU.

5.1.1 MC Neuro-Optics Model Validation

The MC model is built based on the framework of [28] with modifications to account for the imaging process. Since it is implemented in Python with parallel photon launching to boost time efficiency the process significantly differs from the original ANSI-standard C implementation in [28]. In order to validate the MC model, the same physical quantities with the same parameter setting as of the C code are compared. The simulator models an infinitesimally narrow excitation beam injected into a homogeneous tissue of infinite width and depth. Same as the C code, 10^5 photon packets are launched at the origin (0,0,0) and travel downward (Fig. 5-2, panels a and b). Optical parameters are also set to match the parameters used in the original C code. As shown in Fig. 5-2, the resulting energy distribution resembles the symmetry of intensity profile [28] and diffuse pattern of light rays in deeper tissue [4], suggesting the MC optics model accurately captures the light-tissue dynamics.

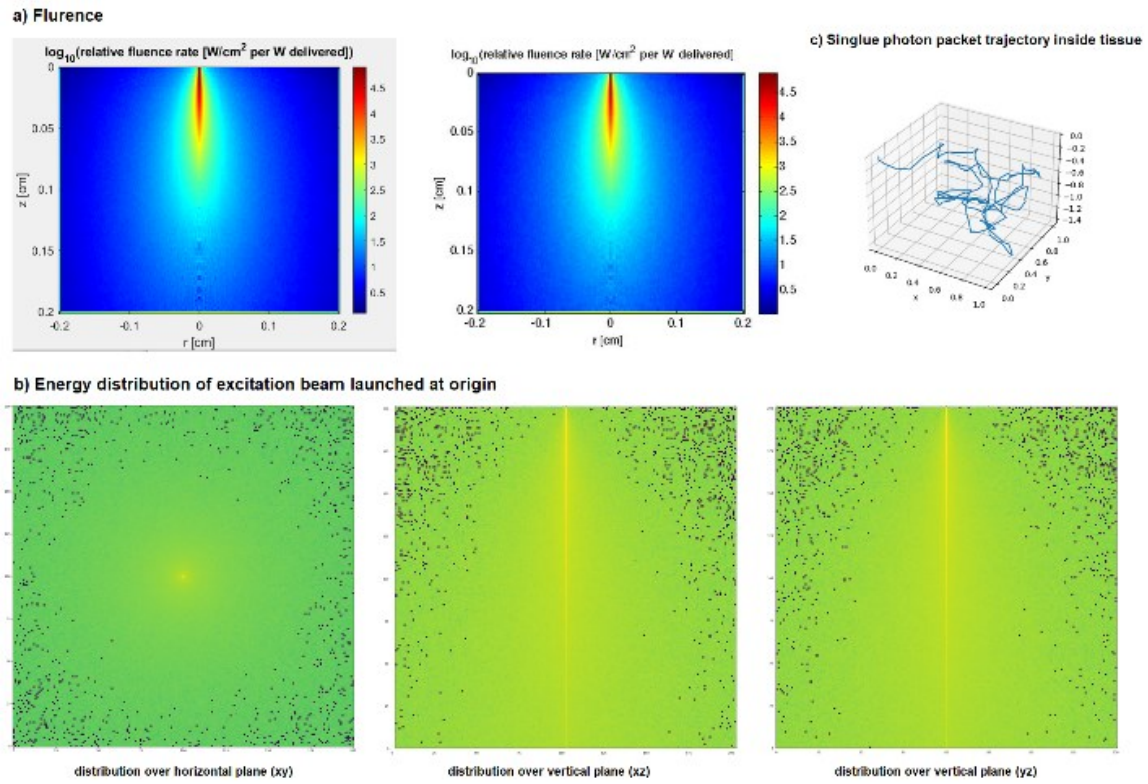

Fig. 5-2. Visualization of MC optics simulator. a) Comparison of fluence rate simulated by current simulator (left) and original C code (right); b) Energy distribution of excitation light launched at origin; c) Example single photon packet path inside tissue.

5.1.2 Detector Validation

The USAF 1951 resolution test chart is a commonly used device in optical engineering to validate imaging systems. This method is adopted to analyze the output of the MC Imager. From Fig. 5-3, it can be seen that a clear image is obtained when photons are launched at a distance equal to the work distance (100μm) the objective lens is located. As the launching plan moves farther away from this work distance the lights are diffused, and the images become more and more blurry.

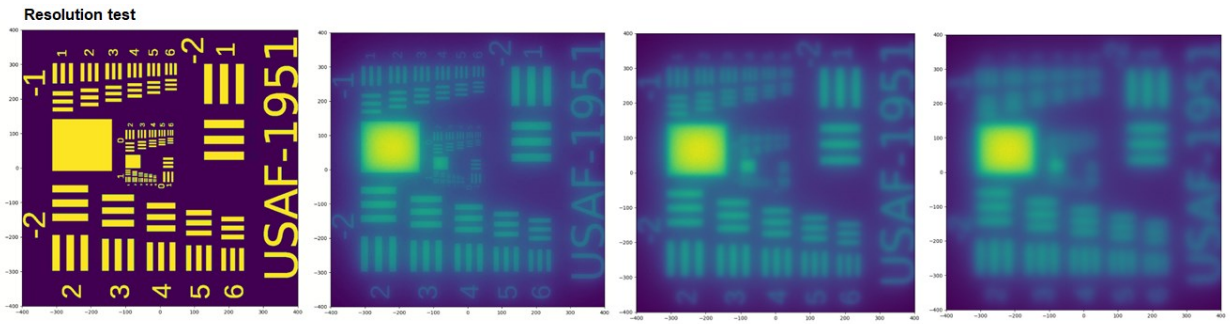

Fig. 5-3. Image formed by launching photons at different depth in tissue. Left most is the USAF 1951 resolution test chart. All photons are launched from a single horizontal plan intersecting the depth of 100μm, 110μm, and 120μm, corresponding to 2nd to 4th images (left to right), respectively.

5.1.3 Model Benchmarking

In order to gain insights into the runtime complexity of the MC Imager so that the efficiency of the denoiser can be better evaluated the runtime required for launching photon packets from a single launching point is compared between MC Imager in Python and the ANSI-standard C code [28]. The result is shown in Table 5-1.

Table 5-1 Comparison between Python and C (both run on a Linux machine) on user CPU time for launching different number of excitation photons from a single location.

Number of Photons	User CPU Time (second)	
	Python	C
10^2	2.146	0.072
10^3	2.323	0.114
10^4	3.135	0.508
10^5	8.965	4.363
10^6	77.692	42.927
10^7	858.037	420.843

It is worth noting that the two implementations differ in various ways, including the coordinate systems. As a result, a direct comparison between the two is meaningless. Nevertheless, it can shed some light on the general runtime required for the Python version of the MC model. The excitation process is chosen because the two simulators share more similarities with the excitation process than the emission. Table 5-1 demonstrates that for less than 10^4 photons C is significantly faster than Python, nearly 30 times faster for 10^2 photons. Then the gap narrows to about 2x faster for 10^5 photons and stabilizes afterward. This is hardly any surprise because Python incurs sufficient overhead than C. As the number of photons launched for generating a single fluorescence image can easily exceed tens of millions, it is reasonable to assume that for a given number of photons launched at each grid point, the C code is typically 2x faster than Python. Since the increase in time is substantially higher for each order of magnitude increase in the number of photons launched, reducing the number of photons per location will effectively shorten the total run time.

5.2 Results

This section presents the outputs and results from each module alongside the parameters employed to generate the data. The basic setup and a thorough analysis of the results are also provided.

5.2.1 Virtual Cells

The virtual neural volume generated by the NAOMi simulator has been discussed in Chapter 2 and Chapter 4. This section focuses on how the virtual tissue is configured for the MC Imager to produce synthetic fluorescence images.

The NAOMi simulator can simulate comprehensive neural volume containing whole cells, including nuclei, somas, dendrites, blood vessels, capillaries, and arterioles. In addition,

fluorescence labels are associated with targeted cells. The fluorescence concentration of any layer can be visualized by slicing the volume vertically and horizontally (Fig. 5-4, left panel). On the other hand, it is also possible to isolate parts of cells, e.g., somas, from the volume. The isolated components do not have the fluorescence labels associated with them. They are visualized by meshing the grid points of position vectors (Fig. 1-1, right panel).

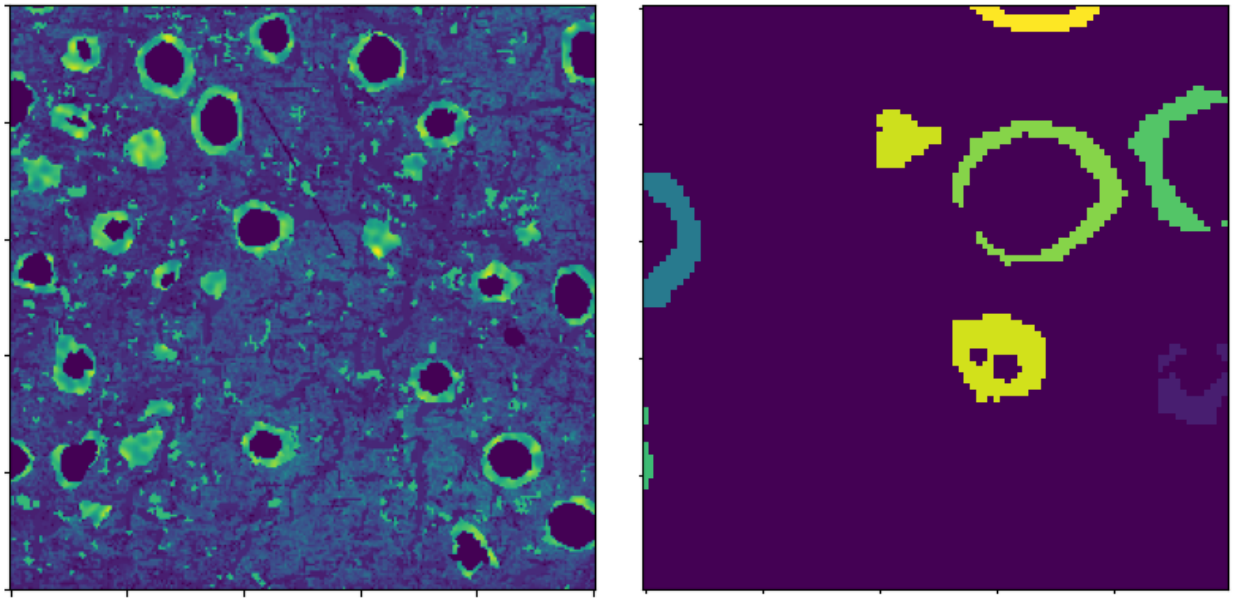

Fig. 5-4. Simulated neurons. Left: a slice taken from a neural volume of 200 x 200 x 200μm. Right: a slice of isolated somas taken from a neural volume of size 100 x 100 x 20μm. It has no fluorescence label associated with it and cells appear in solid color.

The distinction between the two types of virtual cell data makes a difference in the number of photon packets launched at each position in the MC neuro-optics simulation. For a whole neural volume the number of photons launched at each location depends on voxelwise fluorescence concentration values generated by the NAOMi simulator, a predetermined quantum efficiency value, and a base number. The base number represents the desired image clarity, the larger the base number the clearer the resulting image. A threshold value can be added to regulate the background signal level. This configuration ensures that the simulated images consistently

capture the characteristics of the virtual neural volume. On the other hand, isolated somas are formed by grid coordinates and serve the sole purpose of simplifying the simulation process. The number of photons used for generating somas images does not vary across voxels.

Again, the top or sectional view of NAOMi neural volume data is a reference for how the cells look. It is not directly comparable to the synthetic images generated by the MC Imager because the fluorescence intensity in the simulated images reflects the vertical positional information of the cells. However, this information is lost in the sectional view of volume data.

5.2.2 Synthetic Neural Images

To produce simulated fluorescence images an anatomical neural volume of desired size is placed at the targeted depth level. The same volume is allowed to be shifted up and down to observe the difference resulting from various depths and produce in-focus and out-of-focus images. An illustration of the tissue position with respect to focal plane is depicted in Fig. 5-5.

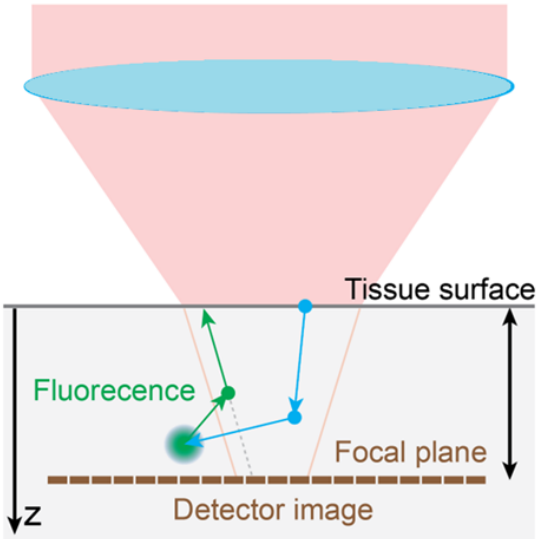

Fig. 5-5. Tissue position in relation to focal plane. The focal plane in the tissue mirrors the working distance of objective lens.

For whole volume simulation with fluorescence concentration the number of photons launched at each voxel is computed as the product of a base number, normalized fluorescence concentration values, and quantum efficiency (default 1). The number of photons mentioned in the results represent the based number instead of actual number. The optical parameters experimented are listed in Table 5-2.

Table 5-2 Optical parameters for excitation beam and fluorescence emission light used in MC optics simulation, referenced the work in [46].

Parameters	Notation	Unit	Excitation Beam	Fluorescence
Wavelength	λ	nm	480	530
Scattering Coefficient	μ_s	mm^{-1}	21	21
Absorption Coefficient	μ_a	mm^{-1}	0.33	0.55
Anisotropy Factor	g	--	0.80	0.82
Work Distance	d	μm	--	100

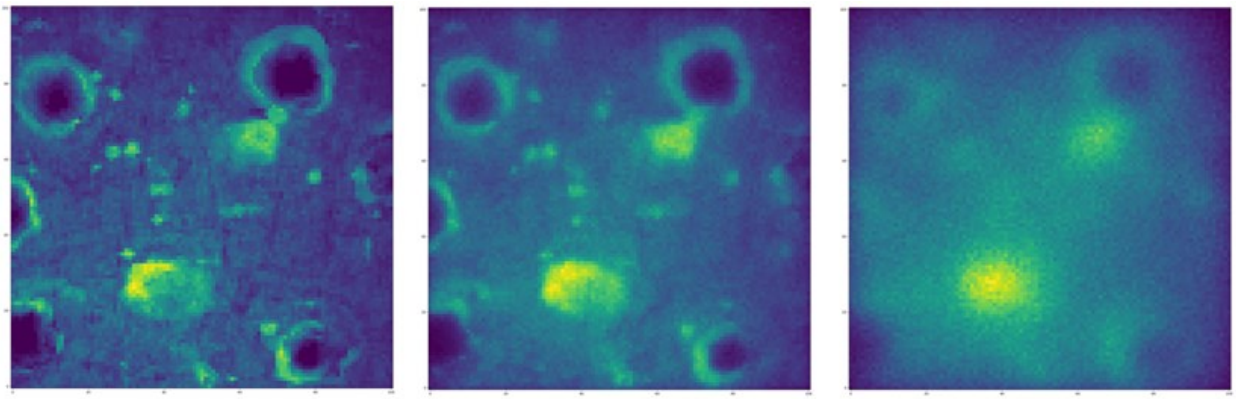

Fig. 5-6. Example synthetic images. Left: NAOMi generated neural volume. Middle: simulated miniscope in-focus image. Right: simulated out-of-focus image.

Fig. 5-6 shows one example image generated from a neural volume of size 100 x 100 x 10μm. The focal plane is at a distance of 100μm. The in-focus image (Fig. 5-6, left panel) is obtained by positioning the volume at 96μm from the surface with photons launched at depths ranging from

96 to 105μm. The out-of-focus image is produced by shifting the volume down 15μm (Fig. 5-6, right panel).

The base number of photon packets per voxel is set to 10^4. It can be seen that the cell structure is hardly recognizable in the out-of-focus image. This can be explained by computing the MFP, which equals 46μm ($1/\mu_t = 1/21.55mm^{-1} = 0.0455mm$). It means that when the photons travel from depth a of 96μm up to the surface more than one scattering could have occurred and blurs the image. Consequently, even the in-focus image is slightly blurred. Besides being blurry, the images also appear to be noisy as well, due to the stochastic process constituents of the MC model. A reasonably clean image requires over 10^5 photon packets per voxel for a tissue volume containing complete neurons and blood vessels. It may take tens of days to generate one image, which is why a deep learning denoiser is necessary.

5.2.3 Denoising Results

The deep learning denoiser advocates the self-supervised strategy introduced in [20]; in other words, the model is trained solely on noisy images without seeing a clean image. The training dataset comprises 2,000 noisy images in size 120 x 120 pixels generated by the MC Imager with isolated somas to accelerate the process. Less sophisticated images also provide better visualization for qualitative analysis. The training set is subsequently split into training and validation set at a ratio of 0.768:0.232. . The test set consists of 20 noisy images and 20 clean images. The training set is split into training and validation sets at a ratio of 0.768:0.232. The test set consists of 20 noisy images and 20 clean images. The clean images are generated by launching 10^4 photons per voxel, in contrast to 10^2 photons per voxel for the training and noisy test set. Categorical crossentropy is employed as the loss function with a learning rate of 10^{-4}. The training involves 2,000 training passes with a minibatch size of 128. On Tesla T4 GPU, the

training takes 4,000 seconds (66 minutes), whereas the denoising of the test set takes 0.07 seconds. Example denoised results are shown in Fig. 5-7

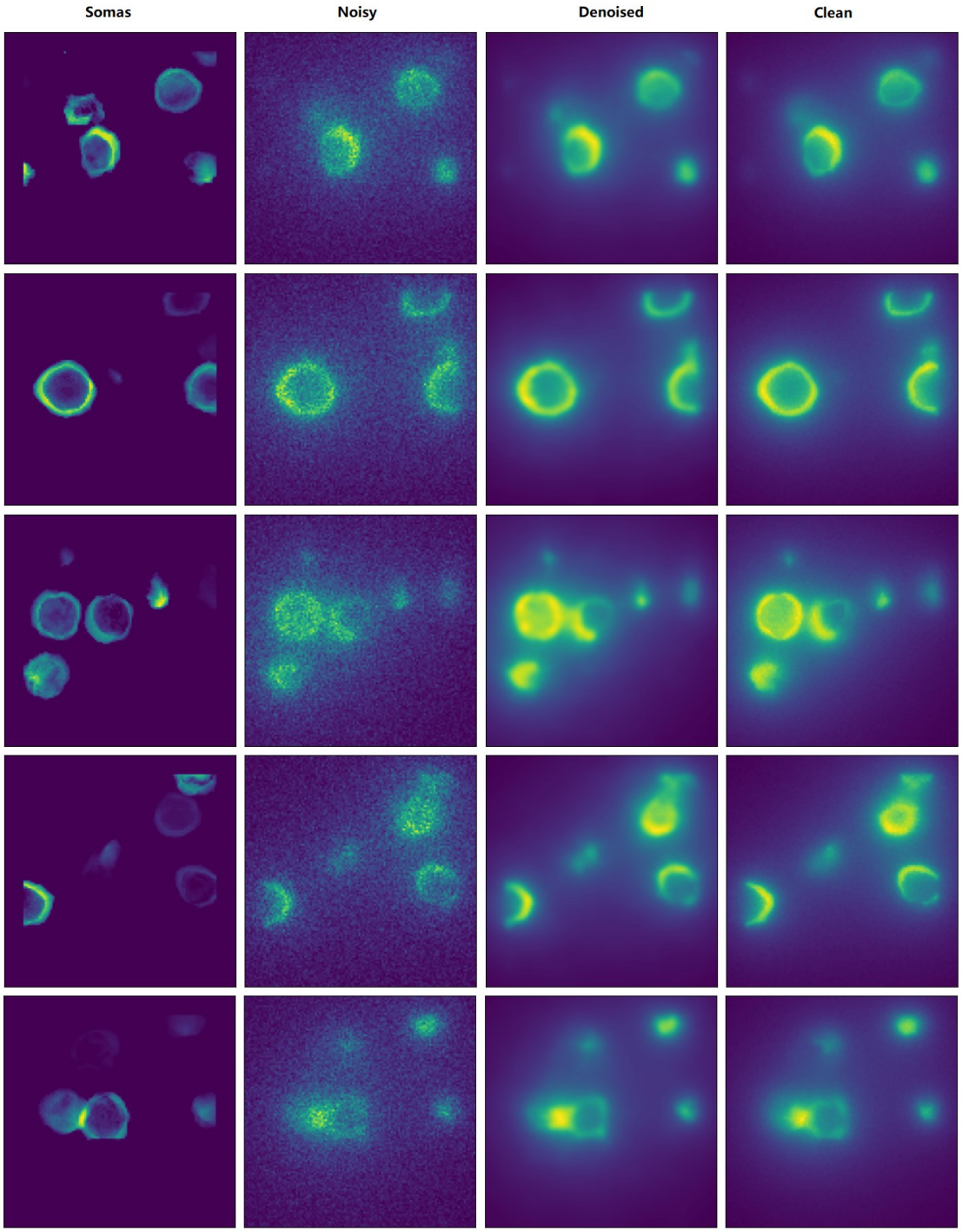

Fig. 5-7. Examples simulated images. From left to right: somas, noisy images, denoised images, and clean images.

Two metrics were used to evaluate the denoise result, structural similarity index measure (SSIM) and signal-noise ratio (SNR). SSIM is a common metric measuring the similarities between two images and evaluating how closely a compressed image represents the original. TensorFlow has a built-in function to compute SSIM between two images. The SNR measures a single image's signal-to-background-noise ratio. It is defined as $SNR = 10\log\frac{P_{signal}}{P_{noise}} = 10\log\frac{\mu}{\sigma}$. The SSIM and SNR results are listed in Table 5-3 and Table 5-4.

Table 5-3. SSIM between images pairs for the example denoised results demonstrated in Fig. 5-7

SSIM	Noisy-Clean	Noisy-Denoised	Denoised-Clean
Row 1	0.471	0.491	0.925
Row 2	0.480	0.501	0.926
Row 3	0.476	0.494	0.922
Row 4	0.503	0.524	0.930
Row 5	0.479	0.502	0.932

Table 5-4. SNR for each image in the example denoised results demonstrated in Fig. 5-7

SNR	Noisy	Denoised	Clean
Row 1	0.692	0.993	0.989
Row 2	0.949	1.254	1.255
Row 3	0.694	0.970	0.999
Row 4	0.817	1.233	1.213
Row 5	0.810	1.199	1.188

The results demonstrate that the denoised-clean pair achieves the highest SSIM, nearly double the value of the two pairs with noisy images. It indicates that the denoised image resembles high structural similarity with the clean image. The SNR value confirms the same pattern, where the

SNR values of the denoised image and clean image are close to each other, while the SNR of the noisy image is significantly lower.

5.3 Discussion

The result proves that the neuro-optics simulator can generate synthetic widefield fluorescence images. These images can then be used for the training of machine learning networks. The contribution of self-learning denoiser is at least three-fold. It helped to shorten the time required to generate reasonably clean images. Secondly, it demonstrated that a self-supervised paradigm can be applied to denoising synthetic images. Additionally, denoising is also part of the deconvolution process. Therefore, it sheds light on the possible directions of future microscopy deconvolution. A more thorough review of the implications and limitations of the study is discussed in Chapter 6.

CHAPTER 6 CONCLUSION

In conclusion, this thesis has shed light on the challenges posed by optics aberrations and the limitations of compact microscopes in the field of microscopy. The study has also highlighted the difficulties in obtaining annotated neural images and ground truth data for training machine learning algorithms. To address these challenges, the proposed solution involved the development of a computer simulation to generate synthetic microscopy images as ground truth, along with the incorporation of a denoiser to enhance image quality.

While the results of this study are promising, it is crucial to acknowledge the limitations that may impact the generalizability of the findings. Firstly, the MC model used in the simulation was oversimplified and did not account for certain important factors. For instance, anatomical tissue is generally anisotropic rather than isotropic, and the assumption of a uniform scattering profile may not always hold true. Therefore, future research should focus on refining the simulation model by incorporating these complexities, allowing for more accurate and realistic image generation.

Secondly, the exclusion of the wave nature of light in the simulation is another limitation of this study. In widefield imaging, diffraction plays a significant role, and by neglecting this aspect, the simulated images may appear cleaner than real images. To overcome this limitation, future work should consider incorporating the wave nature of light propagation into the MC simulation, enabling a more comprehensive and accurate representation of microscopy imaging.

Furthermore, while the denoiser successfully reduced noise levels and enhanced image clarity, it is important to recognize that denoising is just one aspect of the broader computational deconvolution process. The study's focus on denoising presents an incomplete picture of

computational deconvolution in calcium microscopy. Future research should consider integrating other deconvolution techniques to provide a more holistic understanding of the computational deconvolution workflow in microscopy.

The implications of this study are significant for the field of microscopy and biomedical imaging. By addressing the challenges of lacking ground truth data, the proposed simulation-based approach offers researchers a valuable tool for training and evaluating machine learning algorithms without the need for labor-intensive manual annotation. This innovation opens doors for more efficient and accurate experimentation and comparison among different methods.

Moreover, the successful application of the denoiser highlights the potential of data-driven computational deconvolution techniques in mitigating the effects of optics aberrations and noise in microscopy. The ability to reduce noise levels and enhance image quality has implications for various scientific and medical applications that rely on microscopy imaging. It offers researchers the opportunity to extract more precise and reliable information from microscopy data, ultimately leading to improved diagnostics, research outcomes, and advancements in the understanding of biological processes.

For future research, it is recommended to expand the MC simulation to include the wave nature of light propagation, as well as the consideration of anisotropic properties of anatomical tissue. By incorporating these factors, the simulation would better represent the complexities of real-world microscopy imaging, improving the accuracy and reliability of the generated images. Additionally, designing machine learning models specifically tailored to correct optical aberrations in compact microscopes would be a valuable avenue to explore, enabling further advancements in the field of microscopy.

In conclusion, while this study has made significant contributions in addressing the challenges of microscopy, such as optics aberrations and the lack of ground truth data, it is important to acknowledge the limitations and areas for future improvement. By refining the simulation model to account for the complexities of tissue properties and incorporating the wave nature of light, the accuracy and realism of the generated images can be further enhanced. Additionally, the integration of other deconvolution techniques will provide a more comprehensive understanding of computational deconvolution in microscopy. These advancements have the potential to revolutionize the field of microscopy and contribute to advancements in various scientific and medical disciplines, leading to improved imaging techniques and greater insights into the microscopic world.

APPENDIX

1. Python script: Monte Carlo simulation

```
'''
Finalized version with a copy saved in finalized copy directory on 3/1/23
functions tested produced same outputs as C code with same inputs
'''

import numpy as np
import math
import random
import time
import os
import csv
import matplotlib.pyplot as plt
import scipy.io as sio
from scipy.io import loadmat
import plotting as pl
import filing as fl
import coordinates as cr
from scipy import signal

class MonteCarlo():

  def __init__(self,seed=0,n=1,threshold=0.0001,chance=0.1,d=1.,dr=1e3,nv=501):
    self.seed = seed
    self.n = n
    self.threshold = threshold
    self.chance = chance
    self.expower = 200 * 1e-6
    self.na = 0.5
    self.d = d
    self.dr = dr
    self.zfocal = 0.1
    self.A = []
    self.D = []
    self.T = []
    self.nv = nv
    self.mp = math.floor(self.nv/2)
```

```
def reflectance(self,n1,n2,cosi):
  rf = np.zeros(len(cosi))
  if n1 == n2:
    return rf
  b = 1.0 - 1e-12
  vertical = np.where(cosi > b)[0]
  horizontal = np.where(cosi < 1.e-6)[0]
  slant = np.where(((cosi >= 1.e-6) & (cosi <= b)))[0]
  sini = np.sqrt(1-cosi[slant]**2)
  sint = (n1 / n2) * sini
  regular = (sint < 1.0)
  if np.any(regular):
    sint = sint * regular
    cost = np.sqrt(1.0 - sint**2)
    cosp = cosi[slant] * cost - sini * sint
    cosm = cosi[slant] * cost + sini * sint
    sinp = sini * cost + cosi[slant] * sint
    sinm = sini * cost - cosi[slant] * sint
    rf[slant] = regular * (0.5 * sinm**2 * (cosm**2 + cosp**2) / (sinp**2 * cosm**2))
  rf[vertical] = ((n2-n1) / (n2 + n1))**2
  rf[horizontal] = 1.0
  rf[slant] += 1.0 * np.invert(regular)
  return rf

def initialization(self,p,o):

  cors = np.tile(np.vstack(o),self.n)
  dirs = np.zeros((3,self.n))
  rsp = 0.0
  if p['beam'] == 1:
    rdn = np.random.uniform(size=self.n)
    cors[0] += p['r'] * np.sqrt(rdn)
    dirs[2] = 1.0
    nr = p['n1'] / p['n2']
    rsp = pow((1.0 - nr) / (1.0 + nr),2)
  else:
    cosa = np.random.uniform(-1,1,self.n)
    sina = np.sqrt(1.0 - cosa**2)
    psi = 2.0 * math.pi * np.random.uniform(size = self.n)
    cospsi = np.cos(psi)
```

```
    sign = np.where(psi < math.pi, 1, -1)
    sinpsi = np.sqrt(1.0 - cospsi**2) * sign
    dirs[0] = sina * cospsi
    dirs[1] = sina * sinpsi
    dirs[2] = cosa
  return cors, dirs, rsp

def indexing(self,cor,offset=0):

  i = np.rint(cor * self.dr + offset)
  i = np.where(((i < 0) | (i >= self.nv)),self.nv,i).astype(int)
  return i

def detector(self,cor,w,D):
  i = self.indexing(cor[0],self.mp)
  j = self.indexing(cor[1],self.mp)
  inRange = np.where(((i < self.nv) & (j < self.nv)))[0]
  for k in inRange:
    D[i[k],j[k]] += w[k]

def hops(self,cors,dirs,ws,p,D):

  photonAlive = np.where(ws > 0.)[0]
  rdn = np.random.uniform(np.nextafter(0.0, 1.0), 1.0, len(photonAlive))
  s = -np.log(rdn) / (p['mus'] + p['mua'])
  z = cors[2,photonAlive] + s * dirs[2,photonAlive]
  internal = (z > 0.)
  surface = np.invert(internal)
  if np.any(surface):
    uz = dirs[2,photonAlive]
    rdn = np.random.uniform(size=len(uz)) * surface
    ref = self.reflectance(p['n1'],p['n2'],-uz) * surface
    escaped = (rdn > ref) * surface
    reflected = np.invert(escaped) * surface

    ss = np.abs(cors[2,photonAlive] / uz) * escaped
    cors[0,photonAlive] += ss * dirs[0,photonAlive]
    cors[1,photonAlive] += ss * dirs[1,photonAlive]
    cors[2,photonAlive] += ss * dirs[2,photonAlive]
    detected = (np.arcsin(uz) <= math.asin(self.na/p['n2'])) * escaped
    if p['beam'] == 0 and np.any(detected):
```

```
        ss = self.zfocal / uz * detected
        cors[0,photonAlive] += ss * dirs[0,photonAlive]
        cors[1,photonAlive] += ss * dirs[1,photonAlive]
        cors[2,photonAlive] += ss * dirs[2,photonAlive]
        detected = np.where(detected)[0]
        self.detector(cors[0:2,photonAlive[detected]],ws[photonAlive[detected]],D)
    sign = (-1 * reflected + internal + escaped) if np.any(surface) else 1
    internal = (internal + reflected) if np.any(surface) else internal
    ws[photonAlive] *= internal      # this line is corrected on 3/3/23
    cors[0,photonAlive] += s * dirs[0,photonAlive] * internal
    cors[1,photonAlive] += s * dirs[1,photonAlive] * internal
    cors[2,photonAlive] += s * dirs[2,photonAlive] * internal
    cors[2,photonAlive] *= sign
    dirs[2,photonAlive] *= sign

def drops(self,albedo,ws,cors,A,p):

    photonAlive = np.where(ws > 0.0)[0]
    absorb = ws[photonAlive] * (1-albedo)
    ws[photonAlive] -= absorb
    if p['beam'] == 1:
      i = self.indexing(cors[0,photonAlive],self.mp)
      j = self.indexing(cors[1,photonAlive],self.mp)
      k = self.indexing(cors[2,photonAlive])
      inRange = np.where(((i < self.nv) & (j < self.nv)) & (k < self.nv))[0]
      for h in inRange:
        A[i[h],j[h],k[h]] += absorb[h]
    return np.sum(absorb)

def spins(self,dirs,ws,g):
    photonAlive = np.where(ws > 0.0)[0]
    length  = len(photonAlive)
    ux = dirs[0,photonAlive]
    uy = dirs[1,photonAlive]
    uz = dirs[2,photonAlive]
    if g == 0.0:
      cosa = np.random.uniform(-1,1,length)
    elif g == 1.0:
      cosa = np.ones(length)
    else:
      gr = (1.0 - g**2) / (1.0 - g + 2 * g * np.random.uniform(size=length))
```

```
    cosa = (1.0 + g**2 - gr**2) / (2.0 * g)
  sina = np.sqrt(1.0 - cosa**2)
  psi = 2.0 * math.pi * np.random.uniform(size=length)
  cospsi = np.cos(psi)
  sign = np.where( psi < math.pi, 1, -1)
  sinpsi = np.sqrt(1.0 - cospsi**2) * sign
  slant = (1 - np.abs(uz) > 1.e-12)
  perpendicular = np.invert(slant)
  sign = np.where(uz < 0.0, -1, 1)
  zr = np.where(np.abs(uz) != 1.0, np.sqrt(1-uz**2), 1.0)
  uxx = sina * (ux * uz * cospsi - uy * sinpsi) / zr + ux * cosa
  uyy = sina * (uy * uz * cospsi + ux * sinpsi) / zr + uy * cosa
  uzz = -sina * cospsi * zr + uz * cosa
  dirs[0,photonAlive] = sina * cospsi * perpendicular + uxx * slant
  dirs[1,photonAlive] = sina * sinpsi * perpendicular + uyy * slant
  dirs[2,photonAlive] = cosa * sign * perpendicular + uzz * slant

def roulette(self,ws):

  check = np.where(np.logical_and(ws > 0.0, ws < self.threshold))[0]
  if len(check):  # originally not verfied if check is empty (02/27)
    rdn = np.random.uniform(size=len(check))
    ws[check] = np.where(rdn > self.chance, 0.0, ws[check]/self.chance)

def launch(self,p,origin):

  at = 0.0
  albedo = p['mus'] / (p['mus'] + p['mua'])
  D = np.zeros((self.nv,self.nv))
  A = np.zeros((self.nv,self.nv,self.nv))
  cors,dirs,rt = self.initialization(p,origin)
  ws = np.full((self.n),(1.0 - rt))
  while np.any(ws > 0.0):
    self.hops(cors,dirs,ws,p,D)
    at += self.drops(albedo,ws,cors,A,p)
    self.spins(dirs,ws,p['g'])
    self.roulette(ws)
  return A if p['beam'] > 0 else D

def excitation(self,r=1):
```

```
  p = {'mua': 0.33,               # absorption coefficient in ,m^-1
       'mus': 21,              # scattering coefficient in mm^-1
       'g': 0.8,                 # anisotropy
       'n1': 1.5,               # refractive index of medium (tissue)   internal
       'n2': 1.33,                # refractive index outside medium (air) external
       'beam': 1,          # beam type
       'r': 0.0,
       'wv': 480 * 1e-6    # wavelength of excitation light
       }

  self.A = np.zeros((self.nv,self.nv,self.nv))
  origin = np.array((0.0,0.0,0.0))
  self.A = self.launch(p,origin)

def emission(self,origin,r=1):
  p = {'mua': 0.55,    #0.55         # absorption coefficient in mm^-1
       'mus': 21,      #21          # scattering coefficient in mm^-1
       'g': 0.82,       #0.82          # anisotropy
       'n1': 1.5,               # refractive index of internal medium (tissue)
       'n2': 1.33,        #1.5       # refractive index outside external medium (oil)
       'beam': 0,         # beam type
       'r': 0.0,
       'wv': 530 * 1e-6    # wavelength of excitation light
       }
  self.D = np.zeros((self.nv,self.nv))

  print(origin.shape)
  for i in range(len(origin)):     #self.n = int(nv[i] * nbase)
    print(i,self.n,origin[i])
    self.D += self.launch(p,origin[i])
```

2. Python script: deep learning denoising

```
"""dpdenoiser.ipynb

Automatically generated by Colaboratory.

Original file is located at
    https://colab.research.google.com/drive/1GN7D2zq6Z0vLsP4hCES7WS6WyZ6MudXv
"""

import numpy as np
import tensorflow as tf
from tensorflow.keras import layers
from tensorflow.keras.models import Model
from tensorflow import keras
#from tensorflow.keras.datasets import mnist
import matplotlib.pyplot as plt
import os
from scipy.io import loadmat
import scipy.io as sio
from IPython.display import clear_output
import copy
#import mat73

images = np.load('/content/drive/MyDrive/denoise/images_2000_1e02.npy')
xtrain, ytrain = np.split(images,2,axis=0)
print(images.shape, xtrain.shape, ytrain.shape)

testset = np.load('/content/images_testset20.npy')
xtest, ytest = np.split(testset,2,axis=0)
print(testset.shape, xtest.shape, ytest.shape)

"""import shutil
shutil.rmtree('/content/data')
"""

def preprocess(array):
  """normalizes supplied array and reshapes it into appropriate format"""
  if np.max(array) > 1:
    #array = array.astype("float32") / 255.0
    array = array.astype("float32")  / (np.max(array))
```

```
    #array = np.log(1e3 * array + 1) / np.max(array)
  else:
   # array = array.astype("float32") / np.max(array)
    array = array.astype("float32") / np.max(array)
  array = np.reshape(array, (len(array), array.shape[1], array.shape[2], 1)) #size, channel
  return array
def display(array1, array2, array3):
  """Displays ten random iamges from each array"""
  n,c = 1,3
  indices = np.random.randint(min(len(array1),len(array2),len(array3)), size=n)
  images1 = array1[indices, :]
  images2 = array2[indices, :]
  images3 = array3[indices, :]
  plt.figure(figsize = (20,4))
  h,w = array1.shape[1],array1.shape[2]
  for i, (image1, image2,image3) in enumerate(zip(images1, images2,images3)):  # enumerate() returns count
and value, zip() pairs one item from each array as a pair (or tuple for multiple arrays)
    ax = plt.subplot(n, c, i+1)
    plt.imshow(image1.reshape(h,w))
    #plt.gray()
    ax.get_xaxis().set_visible(False)
    ax.get_yaxis().set_visible(False)
    ax = plt.subplot(n, c, i+2) #i+1+n)
    plt.imshow(image2.reshape(h,w))
    #plt.gray()
    ax.get_xaxis().set_visible(False)
    ax.get_yaxis().set_visible(False)
    ax = plt.subplot(n, c, i+3) #i+1+n*2)
    plt.imshow(image3.reshape(h,w))
    #plt.gray()
    ax.get_xaxis().set_visible(False)
    ax.get_yaxis().set_visible(False)
  plt.show()

print(xtrain.dtype, np.max(xtrain), np.min(xtrain))
print(ytrain.dtype, np.max(ytrain), np.min(ytrain))
xtrain = preprocess(xtrain)
ytrain = preprocess(ytrain)
xtest = preprocess(xtest)
ytest = preprocess(ytest)
print(xtrain.dtype, np.max(xtrain), np.min(xtrain))
```

```
print(ytrain.dtype, np.max(ytrain), np.min(ytrain))

input = layers.Input(shape=(xtrain.shape[1],xtrain.shape[2],1))
fc = 32
k = 3
# Encoder
x = layers.Conv2D(fc,(k,k), activation="relu", padding="same")(input)
x = layers.MaxPooling2D((2,2), padding = "same")(x)
x = layers.Conv2D(fc*2,(k,k), activation="relu", padding="same")(x)
x = layers.MaxPooling2D((2,2), padding = "same")(x)
x = layers.Conv2D(fc*4,(k,k), activation="relu", padding="same")(x)
x = layers.MaxPooling2D((2,2), padding = "same")(x)
# Decoder
x = layers.Conv2DTranspose(fc*4, (k,k), strides=2, activation="relu", padding="same")(x)
x = layers.Conv2DTranspose(fc*2, (k,k), strides=2, activation="relu", padding="same")(x)
x = layers.Conv2DTranspose(fc, (k,k), strides=2, activation="relu", padding="same")(x)
x = layers.Conv2D(1,(k,k), activation="sigmoid", padding="same")(x)
# autoencoder
denoiser = Model(input,x)  # the model
#denoiser.compile(optimizer="adam", loss="binary_crossentropy")#, metrics=['accuracy'])
optimizer = tf.keras.optimizers.Adam(learning_rate=1e-4)
denoiser.compile(optimizer=optimizer, loss = "binary_crossentropy")
denoiser.summary()

class showCallbacks(tf.keras.callbacks.Callback):
  def on_epoch_end(self, epoch, logs = None):
    clear_output(wait=True)
    predict = denoiser.predict(xTrain)
    print(f'epoch {epoch+1}')
    display(xTrain,predict,yTest)

callback = [keras.callbacks.EarlyStopping(monitor='val_loss',
                                          min_delta=.5,
                                          patience=5,
                                          verbose=1),
            showCallbacks()]

# test model configuration on traing data(
denoiser.fit(
    x = xtrain,
    y = ytrain,
```

```
    epochs=1,
    batch_size = 128,
    validation_split=0.232,
    shuffle=True,
    #callbacks = callback,
    #validation_data = (xValidation, xValidation)
)

predictions = denoiser.predict(xtest)
display(xtest, predictions, ytest)

history = denoiser.fit(
    x = xtrain,
    y = ytrain,
    epochs = 1000,
    batch_size = 128,
    validation_split = 0.232,
    shuffle=True,
    #callbacks = callback,
    #validation_data = (xValidation, xValidation)
)

predictions = denoiser.predict(xtest)
display(xtest,predictions,ytest)
print(predictions.shape)

import matplotlib.pyplot as plt
print(history.history.keys())
#acc = history.history['accuracy']
#val_acc = history.history['val_accuracy']
loss = history.history['loss']
val_loss = history.history['val_loss']
epochs = range(1, len(loss) + 1)
#epochs = range(501,1001)
#plt.plot(epochs, acc, 'b', label='Training acc')
#plt.plot(epochs, val_acc, 'r', label='Validation acc')
#plt.title('Training and validation accuracy')
#plt.legend()

plt.figure()
plt.plot(epochs, loss, 'b', label='Training loss')
```

```
plt.plot(epochs, val_loss, 'r', label='Validation loss')
plt.title('Training and validation loss')
plt.legend()

plt.show()

i = 19
print(predictions.shape,ytest.shape)
print(tf.image.ssim(predictions[i].astype('float32'),ytest[i].astype('float32'),1.0))
print(tf.image.ssim(predictions[i].astype('float32'),xtest[i].astype('float32'),1.0))
print(tf.image.ssim(ytest[i].astype('float32'),xtest[i].astype('float32'),1.0))

import math
def signaltonoise(a):
    a = a.flatten()
    return (10 * math.log10(a.mean() / a.std()))

snrNoisy = signaltonoise(xtest[i])
snrTarget = signaltonoise(ytest[i])
snrPredict = signaltonoise(predictions[i])
print(snrNoisy, snrTarget, snrPredict)

clean = np.load('/content/somas_testset20.npy')
print(clean.shape)
img = []
for i in range(clean.shape[0]):
  img.append(np.sum(clean[i],axis=2))
img = preprocess(np.array(img))
clean = np.zeros((img.shape[0],120,120,img.shape[3]))
clean[:,10:110,10:110,:] = img
print(clean.shape)

i = 0
print(predictions.shape,ytest.shape)

print(tf.image.ssim(predictions[i].astype('float32'),clean[i].astype('float32'),1.0))
print(tf.image.ssim(ytest[i].astype('float32'),clean[i].astype('float32'),1.0))
print(tf.image.ssim(xtest[i].astype('float32'),clean[i].astype('float32'),1.0))

print(tf.image.psnr(predictions[i].astype('float32'),clean[i].astype('float32'),1.0))
print(tf.image.psnr(ytest[i].astype('float32'),clean[i].astype('float32'),1.0))
```

```
print(tf.image.psnr(xtest[i].astype('float32'),clean[i].astype('float32'),1.0))

img2 = loadmat('/content/somas_1_1e+02_292200')['0']
img3 = loadmat('/content/somas_1_1e+03_284777')['0']
img4 = loadmat('/content/somas_1_1e+04_271509')['0']
img5 = loadmat('/content/somas_1_1e+05_269638')['0']
img6 = loadmat('/content/somas_1_1e+06_2818')['0']
x1,x2 = 30,150
y1,y2 = 22,142
images =
np.stack((img2[x1:x2,y1:y2],img3[x1:x2,y1:y2],img4[x1:x2,y1:y2],img5[x1:x2,y1:y2],img6[x1:x2,y1:y2]),axis=0
)
print(images.shape)

plt.imshow(images[0])
plt.show()

print(images.shape)
print(images[0].shape)

images = preprocess(images)

print(tf.image.ssim(images[0],images[1],1.0))
print(tf.image.ssim(images[1],images[2],1.0))
print(tf.image.ssim(images[2],images[3],1.0))
print(tf.image.ssim(images[3],images[4],1.0))
print(tf.image.ssim(images[0],images[4],1.0))
print(tf.image.ssim(images[1],images[4],1.0))
print(tf.image.ssim(images[2],images[4],1.0))
print(signaltonoise(images[0]),signaltonoise(images[1]),signaltonoise(images[2]),signaltonoise(images[3]),s
ignaltonoise(images[4]))

#fn = 'denoiser2000.h5'
#denoiser.save(fn)
#denoiser = keras.models.load_model(fn)
print(xtest.shape)
print(images.shape)
i = 2
denoised = denoiser.predict(xtest)
plt.imshow(denoised[i])
plt.show()
```

```
plt.imshow(ytest[i])
plt.show()

from keras import models
layerOutputs = [layer.output for layer in denoiser.layers[:8]]
activationModel = models.Model(inputs=denoiser.input,outputs=layerOutputs)

activations = activationModel.predict(xTest)

singleLayer = activations[5]
print(singleLayer.shape)
plt.matshow(singleLayer[0,:,:,0], cmap='gray')

layerNames = []
for layer in denoiser.layers[:8]:
  layerNames.append(layer.name)
imagePerRow = 32
for layerName, layerActivation in zip(layerNames, activations):
  nfeatures = layerActivation.shape[-1]
  print(nfeatures)
  size = layerActivation.shape[1]
  ncols = nfeatures // imagePerRow
  displayGrid = np.zeros((size * ncols, imagePerRow * size))
  for col in range(ncols):
    for row in range(imagePerRow):
      channelImage = layerActivation[0,:,:,col*imagePerRow + row]
      channelImage -= channelImage.mean()
      channelImage /= channelImage.std()
      channelImage *= 64
      channelImage += 128
      channelImage = np.clip(channelImage, 0, 255).astype('uint8')
      displayGrid[col*size: (col+1)*size,
                  row*size: (row+1)*size]=channelImage
  scale = 1./size
  plt.figure(figsize=(scale*displayGrid.shape[1],
                      1))
  plt.title(layerName)
  plt.grid(False)
  plt.imshow(displayGrid,aspect='auto',cmap='viridis')
```

REFERENCES

[1] "What do you know about fluorescence?," *Teledyne Photometrics*. https://www.photometrics.com/learn/microscopy-basics/fluorescence-imaging (accessed Jun. 28, 2023).

[2] G. Barbera *et al.*, "Spatially Compact Neural Clusters in the Dorsal Striatum Encode Locomotion Relevant Information," *Neuron*, vol. 92, no. 1, pp. 202–213, Oct. 2016, doi: 10.1016/j.neuron.2016.08.037.

[3] S. Werth and A. Matlock, "NEURAL NETWORK-BASED MICROSCOPE DEFOCUS CORRECTION WITH POINT SPREAD FUNCTION RECOVERY", [Online]. Available: https://www.bu.edu/vip/files/pubs/reports/SWAM18-02buece.pdf

[4] V. Ntziachristos, "Going deeper than microscopy: the optical imaging frontier in biology," *Nat. Methods*, vol. 7, no. 8, Art. no. 8, Aug. 2010, doi: 10.1038/nmeth.1483.

[5] N. Ji, "Adaptive optical fluorescence microscopy," *Nat. Methods*, vol. 14, no. 4, Art. no. 4, Apr. 2017, doi: 10.1038/nmeth.4218.

[6] S. Werth and A. Matlock, "NEURAL NETWORK-BASED MICROSCOPE DEFOCUS CORRECTION WITH POINT SPREAD FUNCTION RECOVERY".

[7] N. Wiener, *Extrapolation, Interpolation, and Smoothing of Stationary Time Series: With Engineering Applications*. Cambridge, MA, USA: MIT Press, 1949.

[8] L. B. Lucy, "An iterative technique for the rectification of observed distributions," *Astron. J.*, vol. 79, p. 745, Jun. 1974, doi: 10.1086/111605.

[9] W. H. Richardson, "Bayesian-Based Iterative Method of Image Restoration*," *JOSA*, vol. 62, no. 1, pp. 55–59, Jan. 1972, doi: 10.1364/JOSA.62.000055.

[10] G. R. Ayers and J. C. Dainty, "Interative blind deconvolution method and its applications," *Opt. Lett.*, vol. 13, no. 7, p. 547, Jul. 1988, doi: 10.1364/ol.13.000547.

[11] T. J. Holmes, "Blind deconvolution of quantum-limited incoherent imagery: maximum-likelihood approach," *J. Opt. Soc. Am. A*, vol. 9, no. 7, pp. 1052–1061, Jul. 1992, doi: 10.1364/josaa.9.001052.

[12] J. G. McNally, T. Karpova, J. Cooper, and J. A. Conchello, "Three-dimensional imaging by deconvolution microscopy," *Methods San Diego Calif*, vol. 19, no. 3, pp. 373–385, Nov. 1999, doi: 10.1006/meth.1999.0873.

[13] C. Belthangady and L. A. Royer, "Applications, promises, and pitfalls of deep learning for fluorescence image reconstruction," *Nat. Methods*, vol. 16, no. 12, Art. no. 12, Dec. 2019, doi: 10.1038/s41592-019-0458-z.

[14] J. Herbel, T. Kacprzak, A. Amara, A. Refregier, and A. Lucchi, "Fast Point Spread Function Modeling with Deep Learning," *J. Cosmol. Astropart. Phys.*, vol. 2018, Jan. 2018, doi: 10.1088/1475-7516/2018/07/054.

[15] A. A. Ramos and N. Olspert, "Learning to do multiframe wavefront sensing unsupervisedly: applications to blind deconvolution," *Astron. Astrophys.*, vol. 646, p. A100, Feb. 2021, doi: 10.1051/0004-6361/202038552.

[16] I. J. Goodfellow *et al.*, "Generative Adversarial Networks." arXiv, Jun. 10, 2014. doi: 10.48550/arXiv.1406.2661.

[17] O. Ronneberger, P. Fischer, and T. Brox, "U-Net: Convolutional Networks for Biomedical Image Segmentation." arXiv, May 18, 2015. doi: 10.48550/arXiv.1505.04597.

[18] M. Weigert, L. Royer, F. Jug, and G. Myers, “Isotropic Reconstruction of 3D Fluorescence Microscopy Images Using Convolutional Neural Networks,” Sep. 2017, pp. 126–134. doi: 10.1007/978-3-319-66185-8_15.
[19] C. Dong, C. C. Loy, K. He, and X. Tang, “Learning a Deep Convolutional Network for Image Super-Resolution,” in *Computer Vision – ECCV 2014*, D. Fleet, T. Pajdla, B. Schiele, and T. Tuytelaars, Eds., in Lecture Notes in Computer Science. Cham: Springer International Publishing, 2014, pp. 184–199. doi: 10.1007/978-3-319-10593-2_13.
[20] S. Lee, S. Han, P. Salama, K. W. Dunn, and E. J. Delp, “Three Dimensional Blind Image Deconvolution for Fluorescence Microscopy using Generative Adversarial Networks,” in *2019 IEEE 16th International Symposium on Biomedical Imaging (ISBI 2019)*, Venice, Italy: IEEE, Apr. 2019, pp. 538–542. doi: 10.1109/ISBI.2019.8759250.
[21] J. Lehtinen *et al.*, “Noise2Noise: Learning Image Restoration without Clean Data.” arXiv, Oct. 29, 2018. doi: 10.48550/arXiv.1803.04189.
[22] J. Batson and L. Royer, “Noise2Self: Blind Denoising by Self-Supervision.” arXiv, Jun. 08, 2019. doi: 10.48550/arXiv.1901.11365.
[23] A. Krull, T.-O. Buchholz, and F. Jug, “Noise2Void - Learning Denoising from Single Noisy Images.” arXiv, Apr. 05, 2019. doi: 10.48550/arXiv.1811.10980.
[24] K. Dabov, A. Foi, V. Katkovnik, and K. Egiazarian, “Image denoising by sparse 3D transform-domain collaborative ltering,” vol. 16, no. 8.
[25] F. Heide, M. Rouf, M. B. Hullin, B. Labitzke, W. Heidrich, and A. Kolb, “High-quality computational imaging through simple lenses,” *ACM Trans. Graph.*, vol. 32, no. 5, p. 149:1-149:14, Oct. 2013, doi: 10.1145/2516971.2516974.
[26] A. Song, J. L. Gauthier, J. W. Pillow, D. W. Tank, and A. S. Charles, “Neural anatomy and optical microscopy (NAOMi) simulation for evaluating calcium imaging methods,” *J. Neurosci. Methods*, vol. 358, p. 109173, Jul. 2021, doi: 10.1016/j.jneumeth.2021.109173.
[27] S. L. Jacques and L. Wang, “Monte Carlo Modeling of Light Transport in Tissues,” in *Optical-Thermal Response of Laser-Irradiated Tissue*, A. J. Welch and M. J. C. Van Gemert, Eds., in Lasers, Photonics, and Electro-Optics. Boston, MA: Springer US, 1995, pp. 73–100. doi: 10.1007/978-1-4757-6092-7_4.
[28] L. Wang, S. L. Jacques, and L. Zheng, “MCML—Monte Carlo modeling of light transport in multi-layered tissues,” *Comput. Methods Programs Biomed.*, vol. 47, no. 2, pp. 131–146, Jul. 1995, doi: 10.1016/0169-2607(95)01640-F.
[29] M. A. Davis, “Monte Carlo simulation of fluorescence imaging of microvasculature,” thesis, 2011. Accessed: Jun. 22, 2023. [Online]. Available: https://repositories.lib.utexas.edu/handle/2152/ETD-UT-2011-08-4191
[30] A. A. Tanbakuchi, A. R. Rouse, and A. F. Gmitro, “Monte Carlo characterization of parallelized fluorescence confocal systems imaging in turbid media,” *J. Biomed. Opt.*, vol. 14, no. 4, p. 044024, 2009, doi: 10.1117/1.3194131.
[31] “Confocal Microscopy | Principle & Applications | ibidi.” https://ibidi.com/content/216-confocal-microscopy (accessed Jun. 28, 2023).
[32] “Widefield Fluorescence Microscopy: What you need to know,” *Widefield Fluorescence Microscopy: What you need to know*. https://www.scientifica.uk.com/learning-zone/widefield-fluorescence-microscopy (accessed Jun. 28, 2023).
[33] D. J. Watts and S. H. Strogatz, “Collective dynamics of ‘small-world’ networks,” *Nature*, vol. 393, no. 6684, Art. no. 6684, Jun. 1998, doi: 10.1038/30918.

[34] Y. Ogata, “On Lewis’ simulation method for point processes,” *IEEE Trans. Inf. Theory*, vol. 27, no. 1, pp. 23–31, Jan. 1981, doi: 10.1109/TIT.1981.1056305.
[35] A. G. Hawkes, “Spectra of some self-exciting and mutually exciting point processes,” *Biometrika*, vol. 58, no. 1, pp. 83–90, Apr. 1971, doi: 10.1093/biomet/58.1.83.
[36] A. Badura, X. R. Sun, A. Giovannucci, L. A. Lynch, and S. S. H. Wang, “Fast calcium sensor proteins for monitoring neural activity,” *Neurophotonics*, vol. 1, no. 2, p. 025008, Oct. 2014, doi: 10.1117/1.NPh.1.2.025008.
[37] N. Metropolis and S. Ulam, “The Monte Carlo Method,” *J. Am. Stat. Assoc.*, vol. 44, no. 247, pp. 335–341, 1949, doi: 10.2307/2280232.
[38] N. Metropolis, “THE BEGINNING of the MONTE CARLO METHOD,” Accessed: Jun. 23, 2023. [Online]. Available: https://www.semanticscholar.org/paper/THE-BEGINNING-of-the-MONTE-CARLO-METHOD-Metropolis/9630fe755a64231a69519b771fb744d540d626e9
[39] I. Tews, “Quantum Monte Carlo methods for astrophysical applications,” *Front. Phys.*, vol. 8, p. 153, May 2020, doi: 10.3389/fphy.2020.00153.
[40] J. Guo, “The Progress of Three Astrophysics Simulation Methods: Monte-Carlo, PIC and MHD,” *J. Phys. Conf. Ser.*, vol. 2012, p. 012136, Sep. 2021, doi: 10.1088/1742-6596/2012/1/012136.
[41] H. Demers *et al.*, “Three-Dimensional Electron Microscopy Simulation with the CASINO Monte Carlo Software,” *Scanning*, vol. 33, no. 3, pp. 135–146, May 2011, doi: 10.1002/sca.20262.
[42] S. Jacques, “Monte Carlo Modeling of Light Transport in Tissue (Steady State and Time of Flight),” in *Optical-Thermal Response of Laser-Irradiated Tissue*, 2011, pp. 109–144. doi: 10.1007/978-90-481-8831-4_5.
[43] L. G. Henyey and J. L. Greenstein, “Diffuse radiation in the Galaxy.,” *Astrophys. J.*, vol. 93, pp. 70–83, Jan. 1941, doi: 10.1086/144246.
[44] M. Born *et al.*, *Principles of Optics: Electromagnetic Theory of Propagation, Interference and Diffraction of Light*, 7th edition. Cambridge ; New York: Cambridge University Press, 1999.
[45] E. Hecht, *Optics*, 2nd edition. Reading, Mass: Addison-Wesley Pub. Co, 1974.
[46] Y. Ma *et al.*, “Wide-field optical mapping of neural activity and brain haemodynamics: considerations and novel approaches,” *Philos. Trans. R. Soc. Lond. B. Biol. Sci.*, vol. 371, no. 1705, p. 20150360, Oct. 2016, doi: 10.1098/rstb.2015.0360.
[47] B. Mandracchia, X. Hua, C. Guo, J. Son, T. Urner, and S. Jia, “Fast and accurate sCMOS noise correction for fluorescence microscopy,” *Nat. Commun.*, vol. 11, no. 1, Art. no. 1, Jan. 2020, doi: 10.1038/s41467-019-13841-8.
[48] J. Waters, “Sources of widefield fluorescence from the brain,” *eLife*, vol. 9, p. e59841, Nov. 2020, doi: 10.7554/eLife.59841.
[49] Y. LeCun *et al.*, “Backpropagation Applied to Handwritten Zip Code Recognition,” *Neural Comput.*, vol. 1, no. 4, pp. 541–551, Dec. 1989, doi: 10.1162/neco.1989.1.4.541.