Matching Bundles of Axons Using Feature Graphs

Axonenbündel mit Feature Graphs zuordnen
Bachelor-Thesis von Matej Zecevic aus Frankfurt am Main
Tag der Einreichung:

1. Gutachten: Prof. Dr. Jan Peters
2. Gutachten: Prof. Dr. Moritz Helmstaedter
3. Gutachten: Dr. Marcel Beining und M.Sc. Dorothea Koert

TECHNISCHE
UNIVERSITÄT
DARMSTADT

Matching Bundles of Axons Using Feature Graphs
Axonenbündel mit Feature Graphs zuordnen

Vorgelegte Bachelor-Thesis von Matej Zecevic aus Frankfurt am Main

1. Gutachten: Prof. Dr. Jan Peters
2. Gutachten: Prof. Dr. Moritz Helmstaedter
3. Gutachten: Dr. Marcel Beining und M.Sc. Dorothea Koert

Tag der Einreichung:

Za sviju koji su srcem uvijek uz mene, oni koji me vole - ja vas volim isto.

Erklärung zur Bachelor-Thesis

Hiermit versichere ich, die vorliegende Bachelor-Thesis ohne Hilfe Dritter nur mit den angegebenen Quellen und Hilfsmitteln angefertigt zu haben. Alle Stellen, die aus Quellen entnommen wurden, sind als solche kenntlich gemacht. Diese Arbeit hat in gleicher oder ähnlicher Form noch keiner Prüfungsbehörde vorgelegen.

Darmstadt, den 7. November 2017

_____

(Matej Zecevic)

Abstract

Matching thin neuronal processes on nanometer scale between neighboring image volumes which are affected by distortions and noise is hard, especially when the processes appear in bundles. Classical image alignment methods mostly use feature points to minimize pixel-to-pixel dissimilarities. This, however, renders knowledge about the data like physical and biological properties useless by not involving it into the matching, which can lead to erroneous results in problematic matching regions. In this work, a pipeline approach to semantically match precarious bundles of axons using feature neighborhood graphs (FNG), in addition to a rather classical alignment method based on correspondences, is presented. The FNG approach provides a framework for data preprocessing, feature extraction, matching and evaluation applicable to the assignment of fiber bundles, considerable as collections of similar components, across overlapping datasets. Theoretical analysis on the efficiency of the developed methods is provided, showing that, though simple, FNG matching performs reasonably well on drastically distorted data and is highly effective when utilizing learned predictions from a trained model to assess match compatibility.

Zusammenfassung

Das korrekte Zuordnen von dünnen neuronalen Prozessen im Nanometer-Regime zwischen benachbarten Bildvolumen, die Verzerrungen und Störungen ausgesetzt sind, ist schwer, vor allem wenn diese Prozesse in Bündeln auftreten. Klassische Bildausrichtungsmethoden nutzen meist spezifische Bildmerkmale, um die Pixel-zu-Pixel Differenzen der Bilder zu minimieren. Dabei bleibt Wissen über die physischen und biologischen Eigenschaften ungenutzt, da dieses nicht im Zuordnungsprozess berücksichtigt wird, was zu fehlerhaften Resultaten in schwierigen Regionen führen kann. In dieser Arbeit wird ein Pipeline-Ansatz vorgestellt, der problematische Axonbündel auf semantische Art zuordnet mittels Feature Neighborhood Graphs (FNG), zusätzlich zu einem eher klassischen Bildausrichtungsansatz basierend auf Korrespondenzen. Der FNG-Ansatz bietet ein Framework für Datenvorverarbeitung, Feature-Extrahierung, Zuordnung und Evaluation einsetzbar für das Zuordnen von Faserbündeln, prinzipiell verständlich als Ansammlungen von ähnlichen Komponenten, zwischen überlappenden Datensätzen. Eine theoretische Analyse der Effizienz der entwickelten Methoden wird behandelt und zeigt unter anderem, dass auch wenn einfach, die Ergebnisse des FNG-Ansatzes auf stark verzerrten Daten vernünftig sind und der Ansatz sehr effektiv ist, wenn er zusammen mit gelernten Vorhersagen von trainierten Modellen, die die Kompatibilität von Zuordnungen messen, genutzt wird.

Acknowledgments

I wish to take the opportunity in this short section to say thank you.
First of all, I want to begin with the person that brought me here in the first place, the person with whom this all began, thank you, Moritz, for giving me the opportunity to participate in a wonderful and cool lab giving me insights into top research and making me grow on so many different levels. I am very grateful for the past months and everything I was able to learn. And I will never forget how cool it felt when we first met back at the ICNF lecture and you gave me your visiting card telling me to stop by anytime soon for a visit. The foundation to all of this.
Thank you, Marcel, for always being there to talk, to help and so much more - it was so cool with you.
Thank you, Manuel, for always explaining complex concepts as if they were a tale from a children's book.
Thank you, Alessandro, for always bringing in your great knowledge and helping when one was stuck.
Thank you, Benedikt, for always being there smiling and with a game plan even if something seemed hard.
Thank you, Sahil, for always keeping it cool, pushing productivity, but still always there for a laugh.
Thanks to all other lab members that helped me, that showed me and that discussed with me.
Thanks to Robin who is always there where I am, expanding my horizon, showing me that there exist no crazy ideas but only the limitations one puts onto oneself.
Finally, I want to thank the university side of things as well.
Dorothea, thank you for always being there to help, always having ideas, always smiling, always being true to yourself.
Jan, thank you for inspiring me back in my first semester to go the road of 'learning', which I personally think is the most interesting one, sparking my flame, and ultimately enabling me to do this thesis at the TUD.

Contents

Figures and Tables

List of Figures

List of Tables

Abbreviations, Symbols and Operators

List of Abbreviations

Notation	Description
AD	Apical Dendrite
AIS	Axon Initial Segment
CNN	Convolutional Neural Network
EM	Electron Microscopy
FNG	Feature Neighborhood Graph
LM	Light Microscopy
LMNN	Large Margin Nearest Neighbor
PC#	Piezo Column # dataset
PCA	Principal Component Analysis
ROI	Region of Interest
SBEM	Serial Blockface Electron Microscopy
SEM	Scanning Electron Microscopy
SIFT	Scale-Invariant Feature Transform
STEM	Scanning Transmission Electron Microscopy
SURF	Speeded-Up Robust Features
SVM	Support Vector Machine
TEM	Transmission Electron Microscopy

1 Introduction

To introduce the questions being asked to both motivate and guide this work, the subsequent chapter will introduce the reader to the fundamental motivations behind the study of the brain in general and more specifically Connectomics as a field and the challenge of increasing the size of volumes that are analyzable. After that, the problem at hand is being stated and specified. Finally, a short overview that guides the reader structurally through this work chapter by chapter.

1.1 Motivation

In science it is all about the accumulation of indepedently verifiable facts about the world connected into a coherent framework of consistent principles and the interdisciplinary field of neuroscience is no exception. But why study the human brain, what makes it so interesting? The brain is the most complex component of the human body, an organ that is responsible for intelligence, initiation of body movement, interpretation of sensory information and essential in control of all our behaviors. Concepts like memories and feelings are all somehow encoded within this network of neuronal signals that pass through the various types of cells that come in different shapes and sizes all with their specific function. However, due to the vast complexity of the brain, much remains yet to be discovered. The brain has enabled the human race to come up with highly complex solutions to a variety of problems spread across lots of different domains. When this highly capable organ, however, starts to malfunction then life can become very limited. Brain disorders, mental and neurological, represent an enormous burden on both individuals and societies. The broad impact of brain disorders is threatening the life quality of millions of citizens because of its consequences on the sustainability of health systems. As the European brain council reports in a study, the treatment for neuronal diseases applied in hospitals is estimated to be around one trillion USD per year solely in Europe [1]. Because of this high economical strain, brain disorders are considered as a major public health issue. Many individuals with such disorders remain untreated although effective treatments exist. To only overcome these problems, besides seeking to understand the underlying mechanisms of the brain, should be considered as sufficiently motivating. This work is based on the progress in the neuroscientific field of Connectomics, a field that is concerned with the vast connections that neurons in the brain make differing from all other human cells within their respective organ. In Connectomics, huge datasets recorded by highly specialized microscopes are being processed and analyzed. An essential part to gain volume i.e., increasing the scope of neuronal tissue covered by the analysis is to connect subvolumes properly across their boundaries. This local alignment problem makes the search for a global alignment obsolete, which would be highly influenced by artifiacts and distortions that occur during imaging and therefore lead to the propagation of errors. A challenge to this local alignment problem is the robust alignment/matching of very thin processes as they appear in for instance bundles of axons. To efficiently, correctly and robustly handle such collections of problematic neuronal processes is the main motivation behind this work. The goal is to (at least to some extent) contribute to the solution of the general problem which then eventually leads to the progress of Connectomics itself to ultimately improve the understanding of the human brain.
The key contributions of this work are as follows:

- An image alignment method that uses manual correspondence points for robust and efficient matching, especially when confronted with larger regions of neuronal volume data
- A novel graph-based approach for feature-based matching, which focusses on matching problematic matching regions i.e., regions that contain bundles of axons and thereby considers the relationship between these neuronal processes
- A theoretical and comparative analysis on the efficiency of the developed methods with proposals on how to improve performance and utilize the best of both approaches

1.2 Problem Statement

The problem to be solved is generally that of identifying specific processes like axons across dataset boundaries i.e., if the very same axon appears in multiple datasets then its identity should be known for each of these volumes to assure a proper connectome in the end. For the first approach, the problem is defined in the sense of classical image alignment i.e., the images to be aligned should fit each other properly, which means that the processes within the images that are aligned must represent the same physical process. The main focus of this work, however, is the matching using feature graphs,

here the problem is stated as follows. Given some initial agglomeration of segmented EM data, nodes are extracted that describe features of the neuronal processes and these are connected via edges that represent the neighborhood of the respective physical process within the recorded tissue to form a graph structure. This is done for processes within an overlap region between two datasets and therefore two graphs are created which should be matched. A matching is the assignment of nodes from one graph to a corresponding node in the other graph with respect to the fact that nodes that make up a match (pair) should represent the same physical process. As there exists only literature regarding problems that share to some extent similarity to the problem at hand, this is a novel approach utilizing the inherent neighborhood relationship that appears within a bundle, which is considered as a remarkable feature of these structures, as the initial motivation behind the use of graphs for matching.

1.3 Overview

The following Chapter 2 (Background and Related Work) is divided into two main sections. The background section will introduce the reader to all basic concepts within Connectomics necessary to follow the ideas and methods presented in this work. Furthermore, the data preparation is explained ranging from the early phases of acquisition through imaging to the final stages of cubing to create the data volume where the objects of interest namely bundles of axons are contained. The related work section covers all fundamental concepts and methods used throughout this work from image alignment over the combinatorial assignment problem to popular concepts/techniques in the field of machine learning like PCA and neural networks. Chapter 3 (Methods) will introduce the first approach addressed as correspondence alignment and its conceptual difficulties concerning the solution of the problem stated, which then leads to the discussion of the main method of this work being the feature neighborhood graph (FNG) matching. After a presentation of the underlying mechanisms of the methods, Chapter 4 (Experiments) will cover various scenarios that measure the capabilities of the developed approach. Finally, Chapter 5 (Conclusions and discussion) summarizes the important insights of this work and the developed methods. Additionally, some future work regarding possible improvements and extensions are being proposed. In a separate Appendix Chapter, the identification of reoccurring, specific structures of recognizable nature like axonal bundles will be discussed.

2 Background and Related Work

This Chapter is divided into two subsections: background and related work. First, the Background Section will make the reader familiar with the basic concepts of the field of Connectomics and answer significant questions on the datasets used for the methods to be discussed in this thesis. The Related Work Section that follows will introduce basic concepts from image alignment and machine learning that play a key role in the developed methods.

2.1 Background

This Section introduces Connectomics as a neuroscientific field and establishes the necessary tools required to acquire data from neuronal tissue. Towards the end, the specific datasets are analyzed in more detail and the subject of interest is being portraited.

2.1.1 Neuroscience: Connectomics

Being regarded as the most complex living structure observable in the known universe, the human brain - a spongy on average three-pound heavy mass of tissue - is what contains all the important keys to the mysteries of how we humans interact with the world we perceive. The human brain is a single organ that controls the body and shapes the thoughts and beliefs of each single individual being. The human brain is capable of storing more information than any supercomputer known to be existent and also spans a network of connections that far surpasses any social network to date. In a study, it has been established that the human brain on average consists of 86 billion neurons where each of these cells is capable of creating connections to a large number of other neuronal cells [2]. The capability of neurons having lots of connections is one of the aspects that make the brain so interesting when compared to other organs like the liver for instance, while a typical liver cell connects on average to four other cells, a neuron connects to thousand making it a difference of (almost) three orders of magnitude. Therefore one can already assume the sheer computational freedom that comes from having a lot of connections i.e., parameters that can be adapted for calculations of any kind. Neuroscience i.e., the study of the human brain and in general the nervous system has made a lot of progress in the last century, but there are yet many unsolved mysteries. Arguably the field of neuroscience is a very interesting one, further given the anyhow curious nature of humans which thrive to understand everything. Furthermore, neuronal disorders make up for over a third of the burden of all diseases in Europe [3]. By today, the field developed to an interdisciplinary field with various different approaches that study the molecular, developmental, structural or computational aspects of the nervous system.

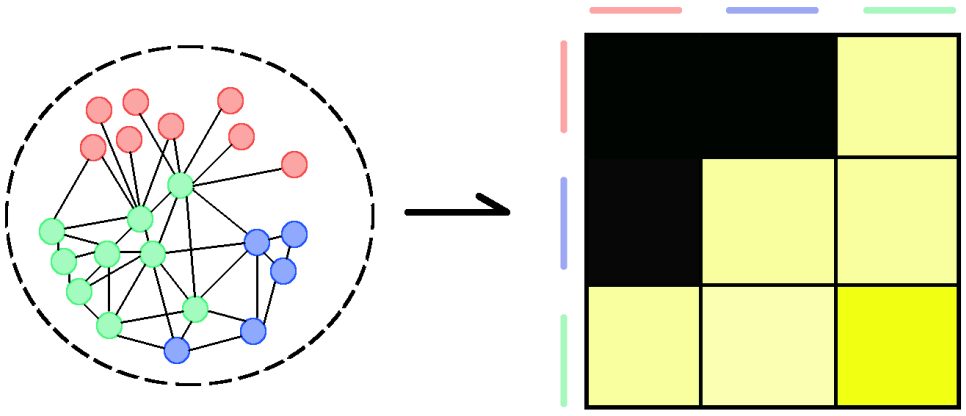

Figure 2.1.: Connectome. A schematic illustration of a connectome. On the left, a pool of entities representing different neuronal cell types (and subtypes) which are connected in a certain way to each other. On the right, the extracted connectome that shows that for instance the red subpopulation only connects to the green subpopulation, while the green subpopulation connects to all but strongest to entities of its own class.

A communication map for neural circuits, a so-called connectome, is the subject of interest being investigated in Connectomics. The main goal of this branch of neuroscience is to decipher the yet unsolved mysteries of the human brain by cutting down the hypothesis space through the analysis of connections and structure of the billions of neuronal processes that space the network of the human brain. As mentioned before, the sheer amount of connections made by cells within the brain make it already a unique attribute of this organ when compared to all others. Therefore comprehensive structural information is often the final arbiter between competing mechanistic models of biological processes and can serve as inspiration for new hypotheses. The structure gives hints on the function because all algorithms of the brain are ultimately encoded by this network. Therefore knowing all those connections should at least provide the data to distinguish between models of neural computations [4]. Measuring similarities between the neural networks in the cortex of different individuals in search for the algorithms of sensory perception and to understand alterations in the structure of those networks in psychiatric diseases are of high interest in Connectomics. To achieve those goals one requires cutting-edge methodology in electron microscopy. What especially matters is speed because the more neuronal tissue volume can be scanned the larger the circuits that one can analyze. Definitive structure at atomic resolution is available while the information on the structure of the whole brain is much less complete. Even more challenging than image acquisition itself is the data analysis that follows, which extracts insights from these neuronal circuits. The sensory expectation i.e., the ability to classify objects into different categories like apples and cars are not decoded in our genomes by default but imprinted into our brains during our lives and with Connectomics the hope is there to systematically figure out the mentioned. In the 1970s, the field started to rise with the study of the small nervous system of a worm and has further gained general interest, especially within the last decade, due to technical and computational advances offering the possibility of mapping even large mammalian brains [5].

2.1.2 Electron Microscopy: Data Acquisition

To analyze and extract useful information from neuronal data assumes that there exists such data in the first place so the following section will introduce the reader to the process of microscopic imaging used in Connectomics to acquire data from neuronal tissue. In science, objects of interest to be studied can vary significantly in their scale. Biologists, for instance, study living organisms and therefore also the tiny units of which those systems consist, cells invisible to the human eye. For the latter, the invention of the microscope has been essential. Microscopes enhance our sense of sight i.e., they allow the observation of things on atomic scale. Two important concepts to be differentiated when it comes to microscopes are magnification and resolution. Magnification describes the ability to enlarge the appearance of something that is observed through an optical instrument, while not changing its physical size. Resolution, on the other hand, describes the level of detail observable i.e., the ability to distinguish between two objects. Therefore information gain for a given observable is only expectable with high magnification when also coupled with high resolution.
In Connectomics, when three-dimensional tissue structure needs to be reconstructed over hundreds of micrometers with a resolution sufficient to follow even the thinnest cellular processes or identify small organelles such as synaptic vesicles, the necessity of powerful microscopic methodologies becomes self-evident [6]. Therefore the usage of Electron Microscopy (EM) has been adopted widely in the natural sciences. The EM is a type of microscope that utilizes a beam of electrons to create an image. It has been developed to overcome the limitations of light microscopy (LM), which is restricted by the wavelength of visible light ($\sim$400-700 nm compared to $\sim$1 nm for the electron beam), and is therefore capable of significantly higher magnification and resolution. However, EM, in comparison to LM, is more restrictive concerning cost, availability and preparation. The microscopes used in EM are large, expensive pieces of equipment that in general stand alone in small, specially designed rooms and require trained personnel to operate them. Two main types of electron microscopes exist: transmission EM (TEM) and scanning EM (SEM). One of the most important differences is that TEM uses a broad static beam through the sample, while SEM utilizes a fine point strategy scanning the sample line by line, contributing also to differences in practical application. In 1986, Binnig and Rohrer, which where the first to produce images of atoms on the surface of a crystal of gold with their scanning tunneling microscopes (STM) shared the Nobel prize in physics with Ruska considered the original pioneer of EM [7]. The rapid development of electron microscopes over the last decades and the necessity of neuronal data acquisition, essential to the reconstruction of substantial volumes that should provide a further understanding of cellular networks, led to the development of the Serial Blockface EM (SBEM) [6]. As illustrated in Figure 2.2, SBEM basically consists of an ultramicrotome that utilizes a diamond knife, mounted in a vacuum chamber of an SEM. Samples are fixed typically with e.g. formaldehyde and then stained using heavy metals like osmium. The basic procedure consists of cutting thin slices off a sample after scanning that certain slice that has been on top before the cutting step. Upon completion, this process results in a stack of two-dimensional scans which are later on processed to a three-dimensional volume.

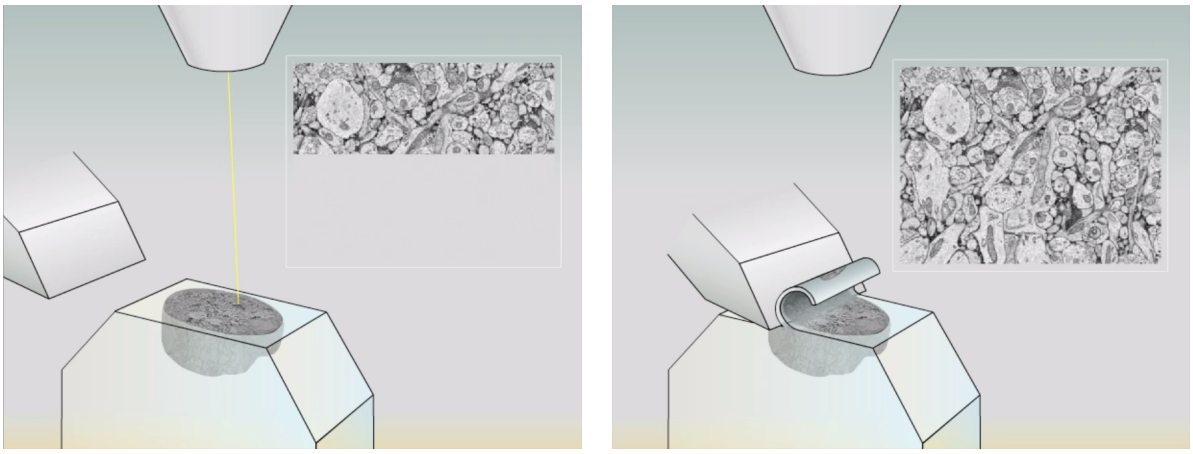

Figure 2.2.: Serial Blockface EM (SBEM). The left image shows the first phase of the SBEM procedure that consists of scanning the tissue using a beam of electrons. The right image shows the second phase in which the diamond knife cut is done to clear the slice. (Source: Max Planck Institute for Medical Research)

2.1.3 Piezo Columns and Bundles of Axons

The datasets, on which the methods to be discussed in later sections are applied to, originate from the whisker barrel cortex of a mouse. This cortex region is in overall part of the somatosensory system. The whisker touch represents a major channel through which a mouse is able to collect information from the nearby environment. Questions like 'where' and 'what' can thereby be encoded in a neuronal representation [8]. To address the question of volume generation, a process called 'cubing' is applied to the stack of raw EM images. Cubing generally describes the alignment of single EM images/slices to form 3D datasets. In principle there exist different approaches to the cubing process, but the datasets considered throughout this work are uniformly generated by having inplane overlap. The overlap approach assumes that the microtome cuts are fine enough i.e., there should not be a significant difference between consecutive slices. Overlap simply means that there exists a region in the respective datasets which contain at least parts of the same neuronal processes. Features for aligning consecutive slices are then extracted and used to find the transformation to generate a volume. In the best case, a simple rigid transformation is sufficient, in the worst case, which is the case for the datasets considered here, an affine, non-rigid transformation is required. The reason for this is due to oscillations in the recording resulting in parts of single images being either squashed or stretched. A last substep to the cubing process prepares the aligned volume in 128 vx^3 cubes, which are then uploaded to a cluster and accessible via *WebKnossos*. *WebKnossos* is a fully cloud- and browser-based 3D annotation tool for distributed large-scale data analysis in LM and EM based Connectomics [9]. This tool is especially useful for fast navigation and annotation of neuronal data. The datasets used here can be accessed internally with *WebKnossos*, the resolution is $11.24 \times 11.24 \times 28\ nm^3$ ($x \times y \times z$). The lower resolution in z is due to the cutting procedure meaning that the voxels (units that represent a value in the image volume space) are anisotropic. Furthermore, these same datasets are named 'Piezo Column 3' and 'Piezo Column 4'. Piezo stands for the type of electric motor used for the recording: a piezo motor. This motor is based on the change in shape of a piezoelectric material when an electric field is being applied. Due to this process, the motors are capable of very fine steps, giving a precision in nanometer scale. Additionally, the motors are also capable of working in strong magnetic fields. The identifiers '3' and '4' simply describe that the piezo columns (PC) used here represent data that is originally neither from a beginning nor end, but from some intermediate, recording phase. A major challenge that will become relevant in later sections are the deformations occurring in the datasets. As illustrated in Figure 2.3, the PCs both share some overlap region that contains at least parts of the same neuronal processes, but these processes, even though being physically the same, look different. Especially over z this problem becomes worse making the matching a hard problem. To conclude on PCs, these deformations are due to the cubing because the transformations applied to the overlap region depends on the part that is unique to the respective PC. Simply meaning that the parameters that are applied to the same region are different, which of course results in two different interpretations of that very same region.
Now that the questions concerning the origin of the datasets have been addressed, the question on the subject of interest remains. Before this question is answered, the neuronal processes encountered in these volumetric datasets acquired from neuronal tissue need to be addressed in more detail first. As touched upon before, the neuron is the cell type which dominates the volume of the brain. These neurons have specialized projections called dendrites and axons, which both grow out of the soma (cell body). While a neuron generally has many dendritic components, it has only a single axon. Dendrites bring in external information and act therefore as a sort of antenna for the neuron. Axons, however, push information out. Information flows from one neuron to another across what is called a synapse. Therefore axons act

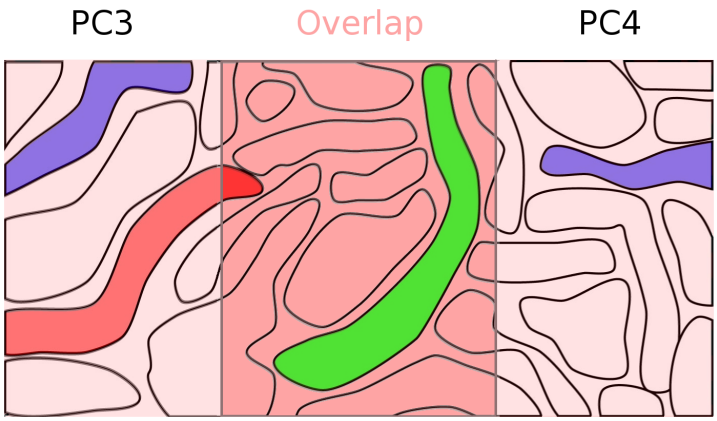

Figure 2.3.: Overlap. Schematic illustration of the overlap concept between two piezo columns. The darker red shade shows the region that both PCs have in common. Three different cases are being distinguished here: the green process is contained fully within the overlap, the red process is partially contained and the blue processes in each set are part of the unique regions portraited by the respective piezo column.
(inspired by Dr. Marcel Beining from the Max-Planck Institute for Brain Research)

as primary transmission lines in the nervous system. To explain the communication step more precisely: the electrical impulse travels down the axon until it reaches the synaptic terminal, also called synaptic boutons (for instance the end parts of the branches of an axon), where then an emission of neurotransmitters is triggered. These neurotransmitters, which are stored in vesicles beforehand, travel the synaptic cleft over to the communication partner i.e., the other neuron. Axons are evidently of very interesting nature being a key player in the whole nervous system. These sometimes very thin processes are the subject of interest for the application of the methods developed later on in this work. To be more precise, collections of such axons that all flow together are of said interest. Such bundles can be encountered at various places in the brain and they usually occur in the vicinity of somata. While the developed methods are especially interested in these bundles because they can be portraited as a group of individual axonal processes, they generally should be applicable to any kind of fiber bundle and therefore to other domains as well. Figure 2.4 below shows an example of a random axonal bundle as encountered in the EM dataset through the search in *WebKnossos*.

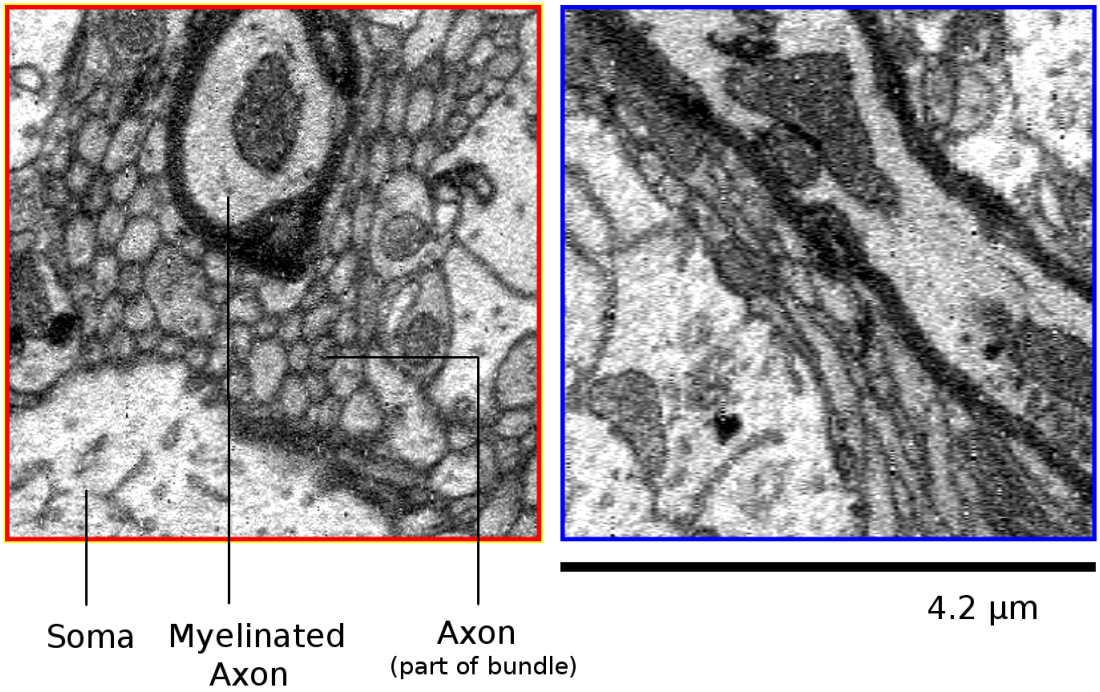

Figure 2.4.: Bundle of Axons. Two viewports from *WebKnossos* that show an axonal bundle in the PC3 dataset of the mouse whisker barrel cortex. The red quadrat shows the XY view, while the blue one shows YZ. The bundle flows in the vicinity of a soma surrounding an axon covered by a myelin sheath. The bundle flows orthogonal to cutting direction (see blue viewport). (dataset 2017-04-21_st143_PC3_CubingKSMB at 4.2 μm viewport width)

2.2 Related Work

This Section introduces all concepts necessary to understand the developed methods discussed throughout Chapter 3 (Methods) of this work. It introduces the reader to the problem of image alignment, which is likely the most intuitive way of solving the matching problem when working with images. Then the assignment problem as the foundation for the semantic matching of the FNG approach is explained and assisted by the concepts behind graph matching. Finally, two machine learning concepts are discussed, which are used to improve the matching procedure.

2.2.1 Image Alignment

This Section will introduce the reader to some real-world applications, define motion models which allow to transform one image onto another and discuss briefly past approaches to the alignment problem. This Section lays the foundation for the correspondence points approach discussed later in Chapter 3 (Methods). First of all the problem at hand needs to be defined. The image alignment problem is often defined as follows: images, typically from one scene, that are related by a motion model which parameters need to be estimated. Therefore, the 'stitching' of images that all belong to the same scene, also called panoramic mosaicing, becomes an intuitive application for the image alignment problem [10]. Furthermore, video stabilization or color channel alignment are applications also solvable by image alignment. In Connectomics and in general the context of this thesis, the alignment of images can be used to connect (parts of) whole datasets in order to e.g., be able to track connections of processes that are of greater size than the actual dataset, which is essential for the retrieval of a proper connectome. However, before the transformation for the alignment of two images can be applied, it is necessary to establish mathematical relationships that map pixel coordinates from one image to another. These mappings are essentially what is called parametric motion models and there exists a variety of those reaching from simple two-dimensional transformations over perspective models to complex mappings onto non-planar surfaces. In the following, a Cartesian coordinate system is assumed and the motion models of interest are two-dimensional. Given a pixel coordinate $\mathbf{x} = (x, y)^\intercal$, its *homogeneous* or *projective 2D coordinate* is then given by $\tilde{\mathbf{x}} = (x, y, 1)^\intercal$.
One of the most basic motion models is *translation* and it can be written as

$$\mathbf{x}' = \begin{bmatrix} \boldsymbol{I} & \boldsymbol{t} \end{bmatrix} \tilde{\mathbf{x}} \tag{2.1}$$

where $\boldsymbol{I}$ is the (2×2) identity matrix and $\boldsymbol{t}$ is the translation vector which describes the shift in x and y direction.
The next motion model is the *Euclidean transformation* (since Euclidean distances are preserved), also known as *rigid body motion*, basically adding a rotational component to the translation. It can be written as

$$\mathbf{x}' = \begin{bmatrix} \boldsymbol{R} & \boldsymbol{t} \end{bmatrix} \tilde{\mathbf{x}} \tag{2.2}$$

where

$$\boldsymbol{R} = \begin{bmatrix} cos\theta & -sin\theta \\ sin\theta & cos\theta \end{bmatrix} \tag{2.3}$$

is an orthonormal rotation matrix which satisfies $\boldsymbol{R}\boldsymbol{R}^\intercal = \boldsymbol{I}$.
Next, the *similarity transformation*, which preserves angles between lines but not their length. An arbitrary scale factor s is added to the Euclidean transform, this can be written as

$$\mathbf{x}' = \begin{bmatrix} s\boldsymbol{R} & \boldsymbol{t} \end{bmatrix} \tilde{\mathbf{x}} = \begin{bmatrix} s \cdot cos\theta & s \cdot (-sin\theta) & t_x \\ s \cdot sin\theta & s \cdot cos\theta & t_y \end{bmatrix} \tilde{\mathbf{x}} \tag{2.4}$$

When only parallel lines are preserved then the motion model utilized is the *affine transform*, it is written as

$$\mathbf{x}' = \boldsymbol{A}\,\tilde{\mathbf{x}} = \begin{bmatrix} a_{00} & a_{01} & a_{02} \\ a_{10} & a_{11} & a_{12} \end{bmatrix} \tilde{\mathbf{x}} \tag{2.5}$$

where $\boldsymbol{A}$ is an arbitrary (2×3) matrix.

The last motion model discussed is the *projective transformation* or *homography*. It only guarantees preservation of straight lines. It is defined for an arbitrary (3×3) matrix $\boldsymbol{H}$ which itself is homogeneous. The transformed coordinate $\boldsymbol{H}\tilde{\mathbf{x}}$ therefore needs to be normalized, this can be written as

$$x' = \frac{h_{00}x + h_{01}y + h_{02}}{h_{20}x + h_{21}y + h_{22}} \quad \text{and} \quad y' = \frac{h_{10}x + h_{11}y + h_{12}}{h_{20}x + h_{21}y + h_{22}} \tag{2.6}$$

where $\mathbf{x}' = (x', y')$.
As can be observed from the mathematical notations, with increasing degrees of freedom in the parameters the complexity of the motion model rises, but with that of course also the possibility of change for the image to be transformed. Figure 2.5 below illustrates intuitively the presented motion models.

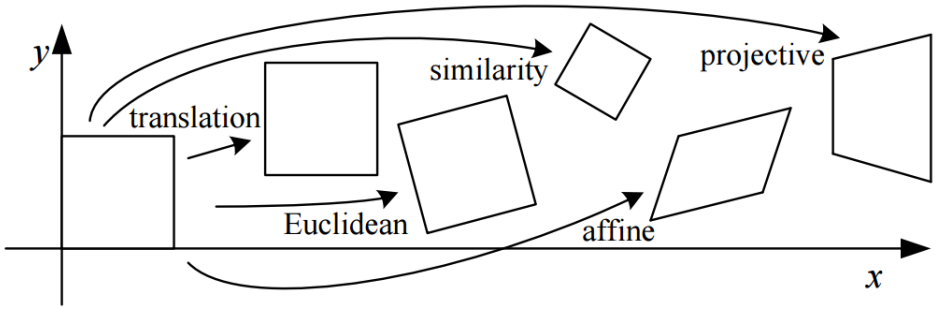

Figure 2.5.: Motion Models. A quadrat in a two-dimensional space is being transformed using all different parametric motion models with increasing degree of freedom. While simple translation only shifts the pixel values a more sophisticated motion model like projective transform is able to create $3D$ (perspective) effects. (Source: Szeliski et al. 2006)

2.2.2 Assignment Problem

This Section covers the assignment problem and further the graph matching problem which lay the foundation for the later developed feature graph method for matching bundles of axons. The assignment problem is considered as an important combinatorial optimization problem, which can be illustrated with the following scenario: as a sales manager who is in charge of salespeople that need to interact with buyers at different locations, it is necessary to have the personnel reach their destinations by e.g., flight. Flights cost money and the price is in a way dependent on the distance traveled, therefore the different salespeople produce a different cost for each location. The goal is then to find a way to organize all salespeople to reach exactly one destination while minimizing the overall airfare. Generally speaking, the problem is defined by agents and tasks, where each agent is assigned to exactly one task and the overall cost is minimized. The problem can be solved by the Hungarian algorithm, which finds the global solution while being computationally feasible (discussed in more detail in the next section).
The problem can be written formally as follows: given a set of agents A and a set of tasks T (in this case of equal size) together with some cost function $C : A \times T \to \mathbb{R}$. The task is to find a bijective function $t : A \to T$, which assigns all agents to exactly one task each, so that the term

$$\sum_{a \in A} C(a, t(a)) \tag{2.7}$$

is minimized. The problem can also be written in terms of matrices, where the cost of a specific assignment pair is part of a $(N \times N)$ cost matrix in following way

$$\mathbf{C} = \begin{bmatrix} C(a_1, t_1) & C(a_1, t_2) & \dots & C(a_1, t_N) \\ \vdots & \vdots & \ddots & \vdots \\ C(a_N, t_1) & C(a_N, t_2) & \dots & C(a_N, t_N) \end{bmatrix} \tag{2.8}$$

where $N = |A| = |T|$ and a_i/t_j represent concrete agents/tasks. Searching for a solution then means to find a path through this matrix that covers exactly one element in each row/column (the optimal solution is then the path that yields minimal cost).
The search space for this problem grows with $N!$ as there are that many ways of assigning N agents to N tasks. In comparison to the mathematical notions from above, the assignment problem can be rephrased in terms of graph theory. A bipartite graph where each agent a_i is connected via an edge to a task t_j which has a weight w_{ij}. With the goal then to find the minimum-weight matching. Figure 2.6 below visualizes this concept.

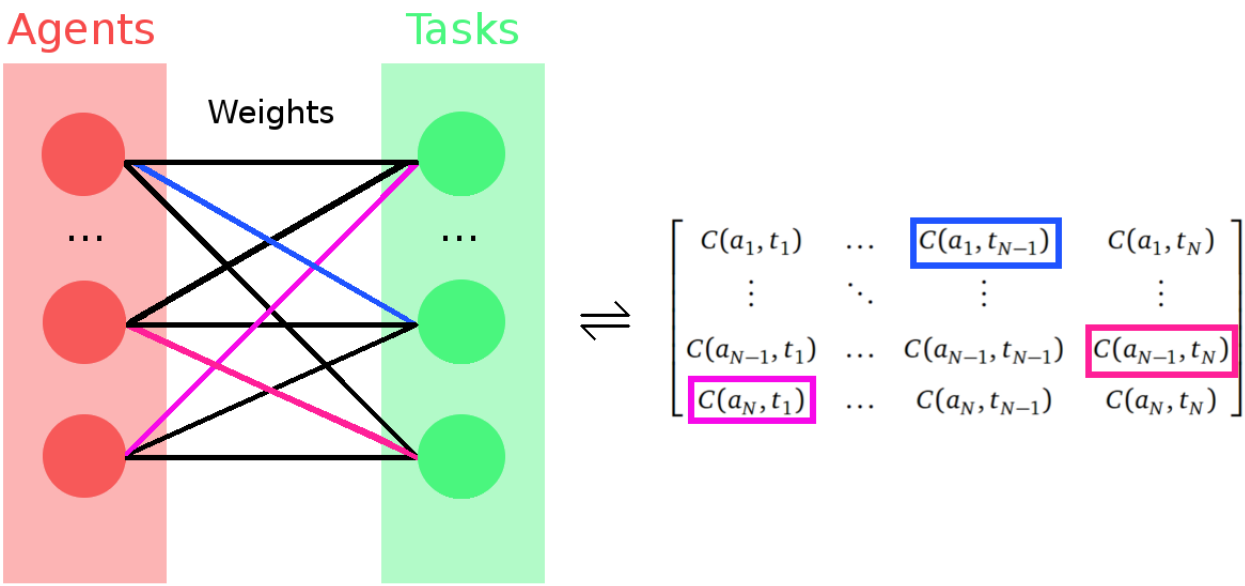

Figure 2.6.: Assignment Problem as Bipartite Graph. The left side shows the graph representation of the assignment problem. The red set contains the agents while the green set contains the tasks. Each agent can only be assigned to exactly one task. The weights are represented by the black edges and the goal is to find the edges that overall minimize the total weight/cost. The right side shows the equivalent matrix representation. An example solution is presented with specific weights/costs chosen (illustrated with different colors).

In Chapter 3 (Methods) when the feature graph matching method is discussed, agents and tasks will both be represented by nodes of each graph. To stay within the context of graph theory: many computer vision problems can be formulated as attributed graph matching problems [11]. In graph matching, patterns are modeled as graphs where nodes correspond to features and edges represent the relational aspect between the features. Graph matching can be considered as a quadratic assignment problem (which is, like the generalized form of the assignment problem, considered as NP-hard), but when talking about the assignment problem generally the linear assignment problem is assumed. Linear in this case simply means that solely the nodes between the graphs are being matched i.e., more technically formulated, a linear term in the objective function encodes node compatibility functions.

2.2.3 Hungarian Algorithm

The last section presented the assignment problem as a fundamental problem in combinatorial optimization, which is essentially the class of the problem to be solved by the later discussed feature graph method. However, in order to find a proper solution for a given cost function, it is necessary to have an algorithm which guarantees to deliver the global optimum (i.e., minimum) each time it is applied. One solution to that is the so-called Hungarian algorithm also known as Munkres algorithm, named after the mathematician which reviewed the original algorithm and discovered it being strongly polynomial [12]. This algorithm is a combinatorial optimization algorithm that solves the assignment problem. It finds an optimal assignment for a given cost matrix in polynomial time. Originally, the algorithm was considered to have an asymptotic complexity of $O(n^4)$, through improvements however at a later point in history the algorithm was improved to $O(n^3)$. Just as the assignment problem can be considered either in terms of graph theory or matrices, the same applies to the Hungarian algorithm. In the following, the algorithm and how it operates is explained in terms of the latter on behalf of simplicity.

Although the global solution is guaranteed to be found by the Hungarian algorithm, the algorithm is maximally as good as the cost function which measures and compares different solutions to the problem at hand. If the desired solution to the problem does not correspond to the optimal solution of the given cost function then the usage of Hungarian is expected to be ineffective.

Input : A $(n \times n)$ cost matrix C
Output: An optimal assignment of n agents to n tasks

Subtract row minima: for each row find lowest element and subtract it from each element
Subtract column minima: same as 1 but for each column
while *Optimality not satisfied* **do**
  Cover zeroes: cover all 0s with a minimum number of lines
  if *(# of lines)* $< n$ **then**
    Create zeroes: take the minimal element k that is not covered by a line, subtract k from all uncovered elements and add it to all covered elements
  else
    Optimality satisfied. **Stop.**
  end
end

Algorithm 1: Hungarian (Munkres) algorithm. An optimization algorithm which solves the assignment problem by finding an assignment that minimizes the given cost.

2.2.4 Principal Component Analysis

This Section briefly covers a relatively old but very popular technique in the field of machine learning: Principal Component Analysis (PCA) [13, 14]. The first step to any data related task lies within the exploration of the data itself. As the data, more precisely the quality and quantity of it, is what enables learning in the first place. The data is the medium from which information (and later knowledge) is being extracted from, meaning that if the data is not 'good' enough then the learning model most likely will fail to generalize and therefore not perform as desired. Data is essential to machine learning and data comes in different variations. A renown problem around data in machine learning is the curse of dimensionality. It basically describes that certain learning algorithms will perform poorly on high-dimensional datasets. Ten different values are assumed (e.g. single digits) with N variables, this means the learner needs to distinguish 10^N configurations of the N-dimensional input vector, which for $N \to \infty$ is very fast in reaching numbers that exceed the number of atoms in the observable universe [15]. Furthermore, visual exploration becomes impossible in practice for these high-dimensional datasets. PCA is an orthogonal linear transformation that transforms the data into a new coordinate system such that the variance is maximized by the first coordinate called the principal component (analogously for the other principal components). In practice this means that PCA is able to solve the problems discussed sufficiently well, reducing dimensionality and making visualization possible. This all while keeping as much information as possible. Figure 2.7 illustrates the application of PCA for a two-dimensional example dataset. In the context of neuronal volume data, PCA is an easy and fast method to obtain the basic directionality of a given process by basically applying the method to a point cloud sample of the respective agglomerate.

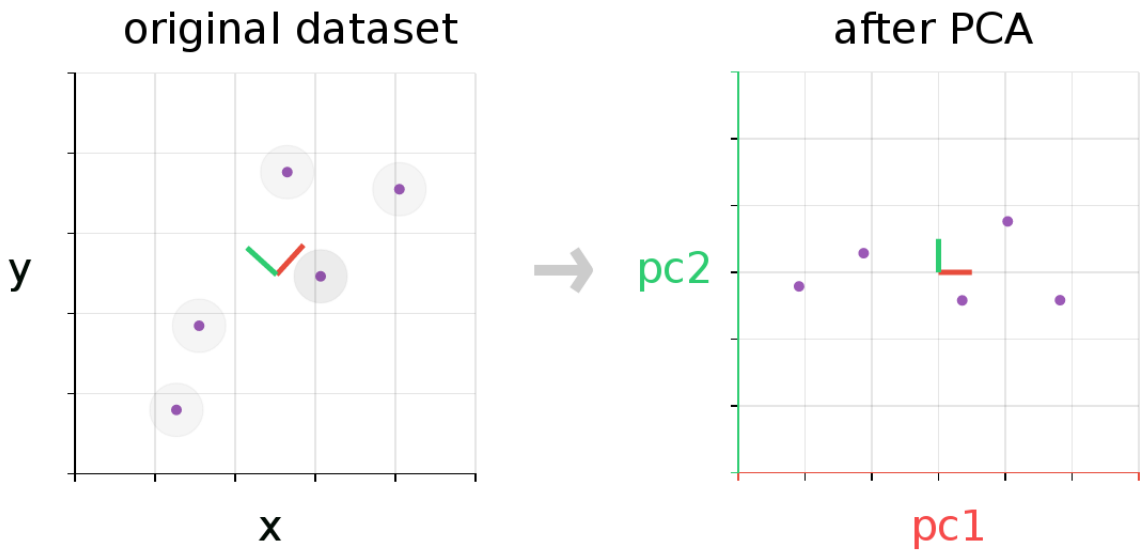

Figure 2.7.: Principal Component Analysis. The left graph shows the original two-dimensional dataset with some example data points marked purple and emphasized with grey circles for illustration purposes (as a note, this example should illustrate the basic concept of maximizing variance, which is of course visually not possible for the usual application on very high dimensional data). The right graph shows the new coordinate system given the principal component (pc). As expected the first pc maximizes the variance (covers the axis on which the original data is spread across the most). (inspired by Victor Powell from setosa.io)

2.2.5 Large Margin Nearest Neighbor

Large Margin Nearest Neighbor (LMNN) is considered as a statistical machine learning algorithm for metric learning. LMNN learns a Mahalanobis distance metric (for k-Nearest Neighbor classification) by semidefinite programming [16]. Semidefinite programming is a generalization of linear programming where the objective is to minimize some function that is subject to some set of constraints. The ultimate goal of LMNN is to group samples from the same class together while seperating these samples from ones from a different class. This separation is induced by a (large) margin where, similar to Support Vector Machines (SVM) [17], a greater margin indicates a greater confidence level i.e., more 'space' for a decision to be made. In general, LMNN shows many parallels to SVMs which, similar to PCA, are very popular in the machine learning community. Just as SVMs, LMNN includes a convex objective function based on hinge loss. However, in comparison to SVMs, there is no further modification necessary for problems with large numbers of classes. In the past, LMNN was able to boost classification among various datasets (e.g., the famous MNIST handwritten digits dataset) [16]. Figure 2.8 illustrates the basic intuition behind LMNN for a given example scenario.
The formal definition of the optimization problem can be written as

$$\begin{gathered}\min_{\mathbf{M}} \sum_{i,j\in N_i} d(\vec{x}_i,\vec{x}_j) + \sum_{i,j,l} \xi_{ijl} \\ \forall_{i,j\in N_i,l,y_l\neq y_i} d(\vec{x}_i,\vec{x}_j) + 1 \leq d(\vec{x}_i,\vec{x}_l) + \xi_{ijl} d(\vec{x}_i,\vec{x}_j) + 1 \leq d(\vec{x}_i,\vec{x}_l) + \xi_{ijl} \\ \xi_{ijl} \geq 0 \\ \mathbf{M} \succeq 0\end{gathered} \tag{2.9}$$

where $\mathbf{M}$ is the matrix being optimized for that completes the pseudometric $d(\vec{x}_i,\vec{x}_j) = (\vec{x}_i - \vec{x}_j)^\top \mathbf{M}(\vec{x}_i - \vec{x}_j)$.
N_i is the set of target neighbors for a data point $\vec{x}_i$ and y_i represents the class label of the i-th data point. Target neighbors share the same class as a given data point $\vec{x}_i$, while imposters not i.e., for an impostor $\vec{x}_l$ it holds that $y_l \neq y_i$. ξ_{ijl} is a slack variable which is there to handle the violations of the impostor constraints. The first term minimizes the average distance between instances and their target neighbors, while the second term constrains impostors $\vec{x}_l$ by pushing them further away than the target neighbors. The last constraint guarantees that the matrix $\mathbf{M}$ is positive semi-definite.
By default, PCA is utilized as a means of linear preprocessing transforming the input into a variance-maximizing subspace. The output of LMNN can be considered either as a distance measure or as a linear transformation to be applied to the data. The matrix $\mathbf{M}$ on which the distance metric is defined is a Mahanabolis matrix, which corresponds to the linear transformation $\mathbf{L}$ in a way that $\mathbf{M} = \mathbf{L}^\intercal\mathbf{L}$. Applying the distance metric using $\mathbf{M}$ or using the Euclidean distance metric (where $\mathbf{M}$ is equal to the identity matrix) on the $\mathbf{L}$ transformed input data, is indifferent as the two ways are equivalent.

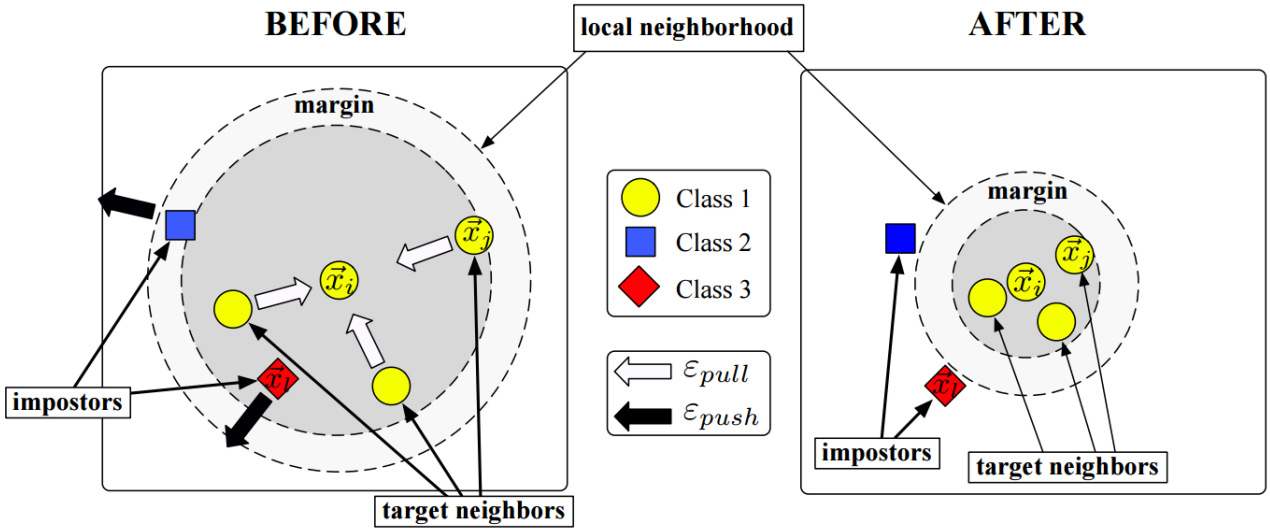

Figure 2.8.: Large Margin Nearest Neighbor. A schematic illustration of the LMNN learning algorithm. After learning the distance metric, target neighbors are closer to the data sample than the impostor which are held away by a (large) margin. (Source: Weinberger et al. 2009)

2.2.6 Neural Networks

Back in 1958, psychologist Rosenblatt developed a simplified mathematical model called the perceptron, which was inspired by the neurons in the human brain [18]. These perceptrons make the foundation of the neural networks, as they are the basic computational units which are connected and formed layer-wise. To briefly emphasize on key events in the history of neural networks, perceptrons had the problem that they couldn't learn effectively which was first adequately discussed and portraited by the literature published by Minsky in 1969 [19], lastly leading to a recurrent theme in neural network history - the 'AI winter'. Backpropagation was developed and allowed the efficient training of networks. The convolutional neural network (CNN) was then developed [20] and the universal approximator theorem established [21]. Unsupervised learning with Boltzmann machines and reinforcement learning with some results in the domain of the Backgammon game were one of the last things before the limitations of backpropagation, being the vanishing/exploding gradient problem, led again to the fall of neural networks. In 2006, deep learning as a field started to gain immense influence rapidly after the few left working on neural networks (Hinton, LeCun and Bengio to name a few) were able to effectively train deep layers [22]. Since then the field around machine learning especially deep learning with neural networks has boomed and become very popular.
Nowadays, neural networks come in all kinds of forms and shapes, but the basic concept behind all is that they are a collection of these 'neural' components. The most basic version of such a network is a feed-forward, non-recurrent network. A network has an input layer, hidden layer(s) and an output layer. Neural networks are in general classifiers and there most prominent use is in supervised learning (where the goal is to classify new, unseen data based on training with labeled data i.e., where we know the right answers to the question asked/input given). An example of supervised learning: the popular MNIST handwritten digits dataset, an image processing task, where the classifier is given an image of a handwritten digit in form of features, in this case pixel values ranging from $0-255$, and then tries to output a class it believes the input belongs to, in this case $0-9$. Given this example, Figure 2.9 illustrates the structure of this basic neural network and how the network predicts for a given example digit image from the dataset. A very important aspect regarding neural networks or in general machine learning is the assumption that the underlying real relationship between the features (here the pixel values for a given position in the image) is actually able to describe the class, as the classifier essentially tries to learn this correlation i.e., coming up with a function that resembles this real relationship. Further to the technical aspect of the networks, non-linear activations, which resemble the biological behavior of neurons regarding their firing rate, are used to allow for complex computations i.e., giving the networks the power they are advertised for. Training, which is essentially the optimization of parameters to be able to fit the data and at the same time generalize to new data, is done by a process called backpropagation as mentioned in the paragraph on the history. Backpropagation describes the computation of the parameters or weights, given some error function that estimates how well the network performs, in order to improve the performance to lastly reach closer to the underlying distribution.

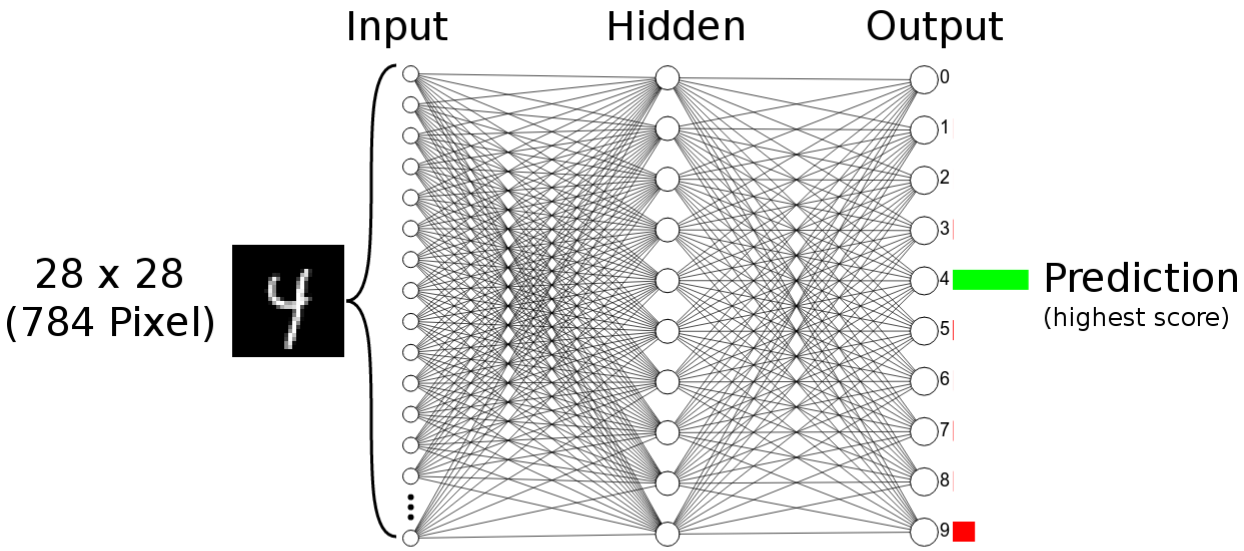

Figure 2.9.: Prediction of a Neural Network. The portraited simple feed-forward network consists of an input layer with 784 units each corresponding to a single pixel in the image of the 28x28 handwritten digits from the MNIST dataset, a hidden layer of a chosen fixed size and an output layer consisting of the 10 different possible digits. The prediction for the example image of '4' shows that the networks confidence is highest for the right class (shown with the green bar), but the networks second highest score would be for the wrong class which is '9'. For instance, bridging the gap of the upper strokes in the image would complete a 9, which is interesting as it seems like these two classes are actually more similar as when compared to some other classes.
(inspired by ml4a (Machine Learning for Artists))

3 Methods

This Chapter is divided into two subsections that cover two approaches developed, the correspondence points approach and the feature neighborhood graph approach. At first, the correspondence alignment will be introduced being the more intuitive/natural approach to solving the alignment of image data. After evaluation of an experiment of this procedure, the conceptual problem i.e., restriction given this approach will be discussed which leads to the development of the main approach: feature graphs. The feature neighborhood graph pipeline will be established within this chapter, reaching from data collection over feature extraction up to the evaluation of the matching of graphs.

3.1 Correspondence Alignment

This Section introduces the correspondence alignment approach. 'Correspondence' simply means that the leading principle behind this procedure is that of image features, which will be extracted from the images to be aligned and basically should represent the shared property between the images. These points that represent the same feature in each image will be called *correspondence points*. The strategy used in this approach is similar to classical image alignment strategies like e.g., SIFT/SURF that are feature-based [23, 24]. Feature-based approaches are considered to be more robust against scene movement, when compared to techniques that minimize pixel-to-pixel dissimilarities, and potentially faster. Additionally, the ability to "recognize panoramas" i.e., detecting adjacency relationships among images is regarded as a significant advantage [25].
The goal of local alignment, as the fundamental problem introduced in the beginning of this work, is to make the search for a global alignment obsolete. Warping artifacts and gradual shifts that occur during electron microscopy imaging make it hard to get the whole dataset aligned. The error that is observed due to these practical problems propagates with increasing dataset size. At some point e.g., whole brain (area) reconstruction this strategy might also become computationally infeasible. To know how to align two neighboring datasets would therefore suffice to break down the problem into local domains in which the error is easier to cope with or even irrelevant.

3.1.1 Marking Correspondences

As first introduced in Subsection 2.1.3, the concept of overlap is crucial to the alignment of neighboring datasets. All discussed methods are based on the assumption of having regions of the datasets that intersect. For the functionality of the correspondence alignment approach, this assumption is even crucial in the sense that correspondences can only be found if the points that represent the same physical process actually exist in each respective dataset space. Identification of correspondence points is therefore the first step in this procedure. A prerequisite to this is that, after the two neighboring datasets have been chosen, an overlap region is being defined as this is mandatory for the application of this approach. In comparison to the latter feature graph approach, the alignment is not mainly evolved around bundles of axons but rather the alignment of the images in general and therefore all processes contained within. But the motivating idea behind this approach is to evaluate how many correspondence points are sufficient to reach a good alignment. Therefore the bundles of axons, which are considered difficult from matching persepctive because of their thin structure, are of most interest for the development of the methods presented in this chapter and will be considered for later experiments.
The datasets used are the piezo column datasets PC3 and PC4 of the mouse whisker barrel cortex (introduced in Section 2.1.3). To begin with, the datasets have been preprocessed with *SegEM* [26]. *SegEM* is a semi-automated volume segmentation toolset for circuit reconstruction in fully stained 3D-EM image data and acts as the foundation for almost any analysis of neuronal volume data. However, the segmentation is not of technical relevance to this approach, but rather of perceptional making it easier to visualize separate processes. Segmentation will play a key role in the main approach discussed later in Section 3.2. Because bundles of axons are considered in this scenario, at least parts of the axonal bundle must lie within the considered overlap region so that correspondences can be extracted to match the bundle across the image volumes. Figure 3.1 shows the region of interest defined within a bounding box, this region will be considered for evaluation later on.
The reader may observe that the bundle of axons flows orthogonal to the z (cutting) direction. Originally the idea behind the identification of such a bundle was to restrict the volume to be observed because the probability that a process would be distorted in a way that it could not be assigned should be lower for this case. The bundle has been spotted manually by navigating the dataset in *WebKnossos*. The ROI is of course defined for each dataset (image volume) i.e., the search in the overlap means that we find the processes we search for in both datasets, so the above is done twice basically. Now that

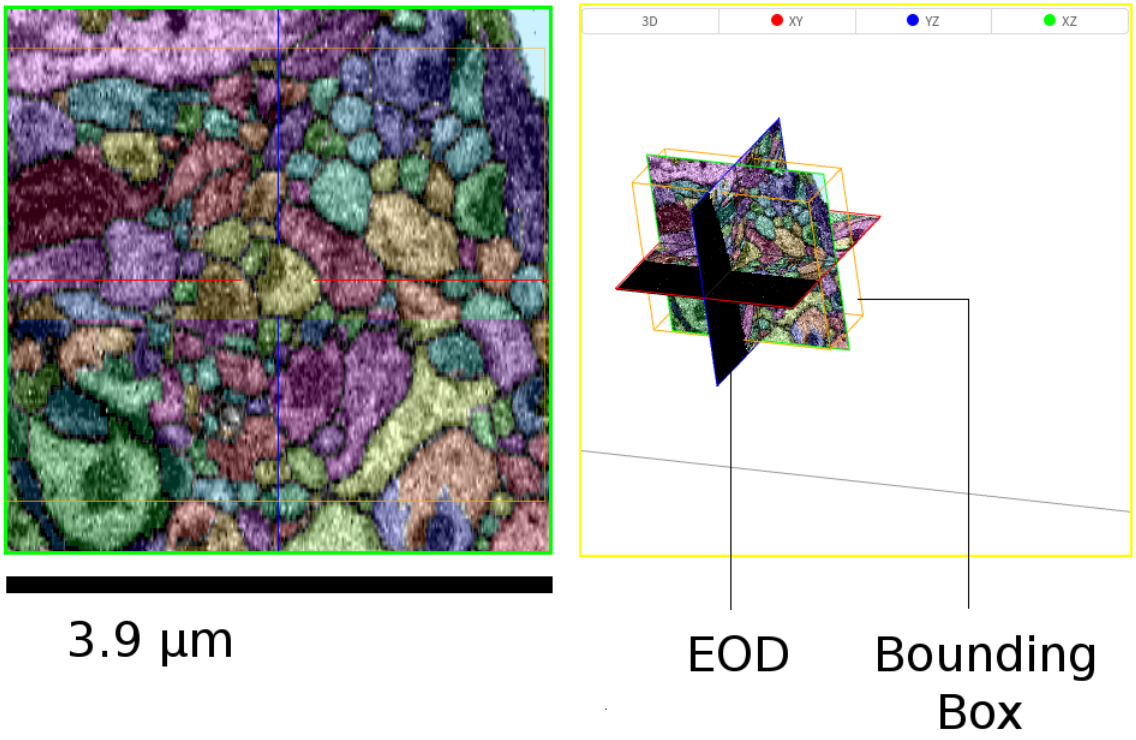

Figure 3.1.: Region of Interest. The left (green) viewport from *WebKnossos* shows the cross-section of an axonal bundle in the xz plane. The dataset after application of *SegEM* is portraited, as can be easily observed by the various colors that the processes are visualized with (each corresponding to a unique ID). The right viewport shows the 3D view, where EOD stands for 'End of Dataset', which is described by the black region that appears in y direction of the bounding box. The bounding box (in yellow) represents the region of interest (ROI) that lies within the overlap and covers most of the bundle. (dataset 2017-04-21_st143_PC3_CubingKSMB at 3.9 μm viewport width)

the ROI is identified, the correspondences need to be found and marked. All code developed throughout this project was done in a *MATLAB* environment. To mark correspondences it is necessary to find them, this is done by visualizing the data in the developmental environment. For this purpose, a volumetric visualization was developed to easily browse the (segmented) data and mark correspondences which are defined by points in the respective volumetric space. The browsing is done slice-wise meaning in each step a distance of the resolution in z (being $\sim$30 nm) is covered. The marking is done manually either by searching the visualization directly or through navigating *WebKnossos* for the points to be marked. Figure 3.2 shows the same slice i.e., depth level of the respective bounding box for both datasets marked with a correspondence point. As the reader may observe, this point describes the same point in the neuronal tissue but through different perspectives given the two datasets. Furthermore, the difference in color is given due to the segmentation which is applied separately to each dataset. More interesting is that the segmentation differs for the same physical processes, however, the representation of the same processes between the datasets is not the same to begin with. As discussed in Section 2.1.3, the reason for this is that the overlap region, which contains (to some extent) the shared processes, is cubed differently as the cubing process is influenced by the unique region that is within the respective piezo column. Looking closely at the bottom dataset, PC4, the reader may observe a clean separation in the center area where the colors assigned to the segments change for processes that lie within this separation. This same aspect can be observed in the upper row of both datasets. This 'border' illustrates the division of the dataset into smaller cubes with which *SegEM* operates (these *SegEM cubes* are of $512 \times 512 \times 256$ vx^3 size). However, there is no technical relevance to this besides that this division is needed to handle these huge datasets, but when visualizing this aspect can be observed so therefore this short note. Now that correspondence points have been marked across the datasets the application of the motion models discussed in Section 2.2.1 will be utilized to align the datasets. The leading question is: how many correspondences are needed to achieve a good matching? To answer this question, an experiment will be evaluated in the last section discussing this method.

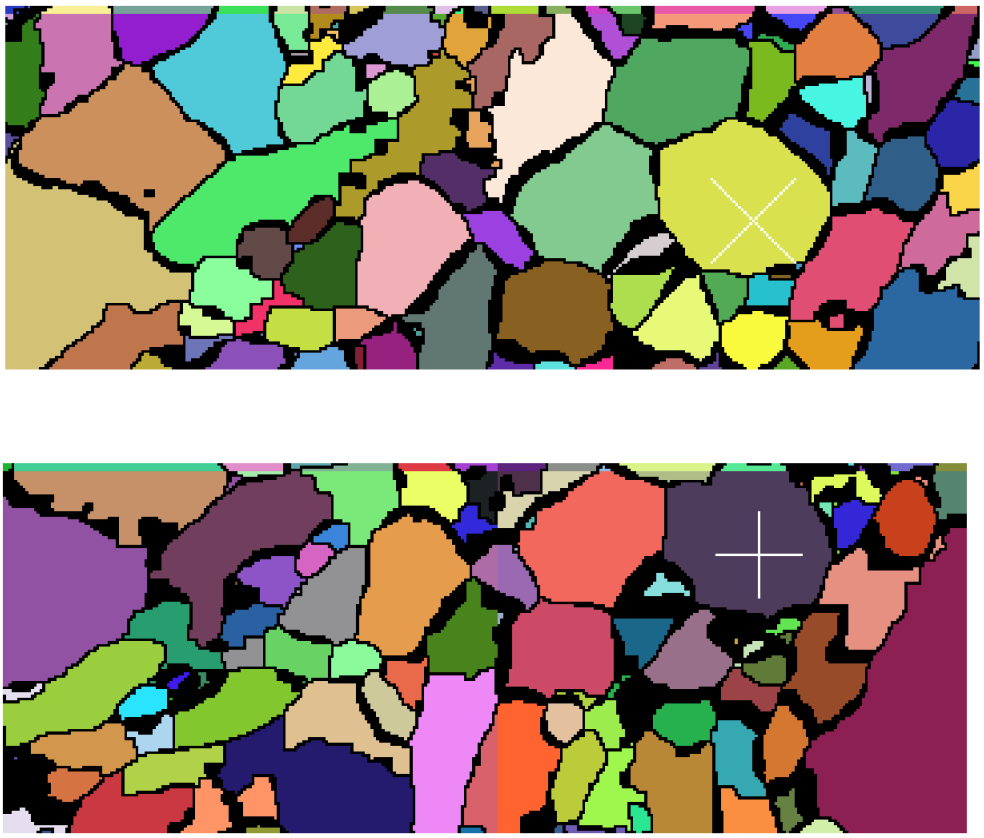

Figure 3.2.: Correspondences. The two regions of interest from each piezo column dataset are shown (with their respective size in voxel at a resolution of $11.24 \times 11.24 \times 28\ nm^3$). The segmented data visualized in this figure is exported directly from the developmental environment. It shows the same physical process across the two sets marked with a white cross at the respective point in each dataset - this is a correspondence point (pair). (dataset 2017-04-21_st143_PC3_CubingKSMB and dataset 2017-04-21_st143_PC4_CubingKSMB)

3.1.2 Motion Models

As introduced in Section 2.2.1, motion models are parametric models that describe transformations which can be applied to align images. The method is implemented in a way that a different amount of correspondence point pairs per slice leads to the usage of a different motion model i.e., the more correspondences per slice exist the more sophisticated the model that is applied. Using a more complicated model which transforms the data more freely may however not always be the best option because the warping artifacts are not evenly distributed across the datasets. I.e., sometimes a simple translation can wield better results than an affine transformation.
The method operates in the following way: when only 1 correspondence for a given slice exists across the two PCs then a simple translation is applied. Given 2 correspondences for a slice then a similarity transformation is utilized. For >3 correspondences an affine transformation is chosen. The data is processed in z (cutting) direction, meaning that the xy planes are aligned (default setting as being the most intuitive, can be changed however), and the transformations are applied stack-wise. I.e., the first transformation defines the transformation for all slices below up until new correspondences are encountered for a later slice. The first transformation is considered an exception as it also defines the transformation applied to the region above, which is the case if the correspondences found are below the first slice considered (i.e., below the beginning of the ROI defined by the respective bounding box with below meaning in z direction). This concept is illustrated schematically for a three-dimensional volume in Figure 3.3.
Now that transformations can be applied to the datasets, one PC needs to be transformed onto the other to evaluate how well the motion models perform.

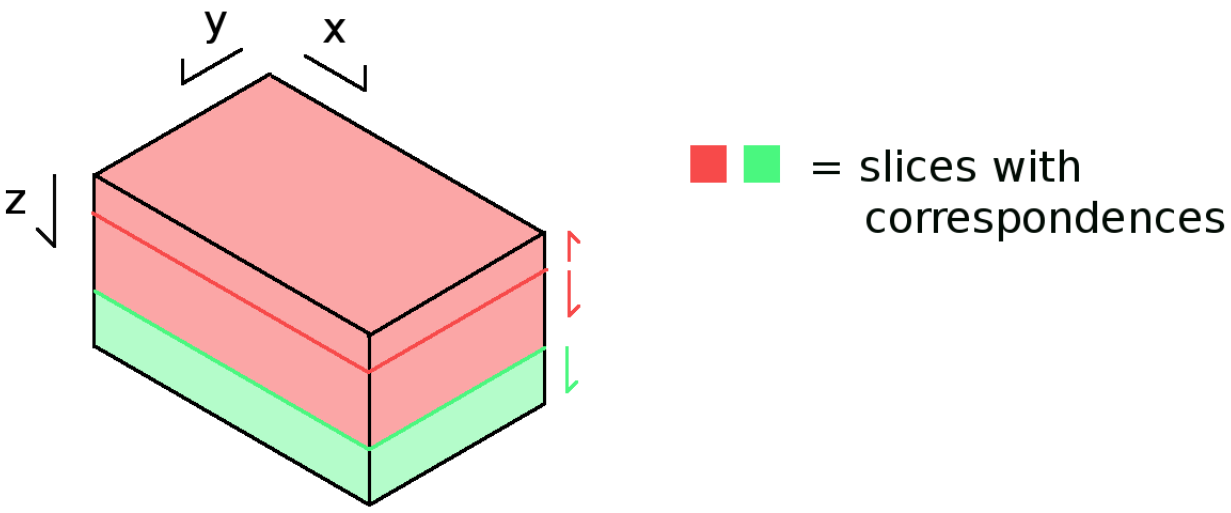

Figure 3.3.: Stack-wise Transformations. A schematic illustration of how the transformations, given the slices that contain the correspondences, are applied to the dataset to be transformed. The red slice contains correspondences for one specific motion model and the resulting transformation is applied up to the green slice, where a new transformation is going to be determined. Note that the red one is the first and therefore also determines the transformation applied to all slices above i.e., correspondences do not have to be found in the first slice of the ROI.

3.1.3 Evaluation and Conceptual Problem

This Section covers an experiment to measure the performance of the transformations given the parametric motion models that are utilized by the method as discussed in the last section. In this experiment PC3 is transformed onto PC4, the results after application of the method are evaluated on a ground truth set and then proof-checked visually. The ground truth contains 35 correspondence point pairs that have been matched manually between the two piezo columns. These points serve only evaluation purpose, the correspondences used to determine the transformations during alignment are separate and independent. The experiment is divided into three categories on which the method is evaluated: *translation only*, *all motion models* and *translation with interpolation*. The first category simply assumes a simple translation motion model i.e., only one correspondence per slice, where different counts of slices that contain a correspondence are compared. In the second category, all discussed motion models are utilized, the correspondences are chosen randomly over the volume i.e., some slices may contain 1 while others contain 3 or more correspondences. The last category is basically the first with the addition of interpolation i.e., this should find out whether given fewer correspondences a good alignment can still be achieved by interpolating between the transformations per stack. Interpolation is done linearly and is only implemented for translation.

Evaluation is done by measuring the *hits* against the total matches within the ground truth. Furthermore, *borderhits* are measured which are points that have been transformed onto a pixel that has a segmentation value of 0 which is usually 'border' (e.g., the membrane of a neuronal process) or sometimes a pixel that lies out-of-bounds. When a borderhit occurs, the implemented method tries to correct this wrong transformation by considering the immediate 26-neighborhood and simply assigning the ID of the winner by majority vote. Table 3.1 shows the results of the experiment. After evaluation of the experiment another category was added, which evolved around the centroids of the segmented data, but the results of that sub-experiment did not yield any insight on a possibility of improvement for this method which is the reason for it not being displayed here. Anyhow, for completeness reasons it should be mentioned.

Looking at Table 3.1, it is noticeable that for the translation-only motion model the increase in the number of corresponding points (which is simply an increase in the number of slices that contain a correspondence) the overall score increases as well. The same holds true for the category that utilizes all motion models. More interesting however is that the best score reached in the first category slightly surpasses the strongest contender in the second category. This observation may suggest that for the transformations needed (at least for the ROIs measured) a simple translational model is sufficient. Examining the last category, translation with interpolation yields the best overall score when only two correspondence slices and multiple interpolation steps are considered. At this point, no points are transformed out of the image bound-

Category	# of cpoints	# of interp. steps	# of borderhits (out of bounds)	Score (Hits/Total)
Translation only	1		23 (15)	0.229 (8/35)
	2		20 (12)	0.286 (10/35)
	3		14 (8)	0.343 (12/35)
	4		10 (5)	0.371 (13/35)
	5		11 (4)	**0.543** (19/35)
All motion models	2		25 (17)	0.229 (8/35)
	4		21 (14)	0.286 (10/35)
	6		16 (9)	0.343 (12/35)
	8		13 (4)	**0.514** (18/35)
With interpolation	2	1	7 (4)	0.486 (17/35)
	2	7	5 (0)	**0.771** (27/35)
	2	21	4 (0)	0.743 (26/35)
	3	1	6 (0)	0.657 (23/35)
	3	7	7 (0)	0.571 (20/35)
	3	21	6 (0)	0.514 (18/35)

Table 3.1.: Correspondence Alignment Experiment. This table summarizes, for the discussed categories, how the developed method performs for varying counts of correspondences considered. The best results in each category are marked with bold font.

aries and only a few borders are hit for which no decision can be made successfully. This observation may suggest that the slices within a stack are affected by (at least something close to a) linearly changing distortion as the linear interpolation performs well. Furthermore, this is rather good as the need for extraction of correspondences, which is done manually here, is lower. The computational aspect is, however, more expensive (even though for ROIs of this small size irrelevant).
To get a feeling of the functional behavior of the alignment when applied to the data, the visualization can be taken into account. Figure 3.4 illustrates for a given correspondence slice how the method performs for a concrete example from the ground truth set. The reader may observe that the alignment that happens in the slice that contains the correspondences is very good as the only major differences seem to be in the different segmentations which are expected. Moving in z (cutting) direction, down the stack, the alignment gets 'outdated' as the distortions in the data differ at different depth locations, just as they may differ within regions of the very same plane. As can be observed, the prediction does not fit the target perfectly at some later point in the stack. Therefore, having more correspondences across different depth levels to account for this problem is reasonable.
To summarize the method developed in the last sections, the correspondence alignment approach is based on correspondences i.e., (feature) points that represent the same physical process within the different datasets to be aligned. Different counts of correspondences per slice result in the consideration of different motion models, which are then applied stack-wise to the dataset. As Table 3.1 suggests, translation with interpolation performs best on the considered piezo columns and chosen ROIs (for the ground truth set). This approach scales to greater datasets as the required computations are kept minimal and if needed could be improved through parallelization. Additionally, this approach is very robust in the sense that it is, similar to the feature-based image alignment methods as mentioned in Section 2.2.1, based around finding features (here called correspondences) to find the optimal alignment. At the same time, this concept is the methods doom i.e., all important and useful information on the physical/biological properties of the processes/data is being ignored completely and therefore thrown away. A semantic approach to the alignment turns the problem into an assignment problem where it is about 'real' matching of processes that represent the same physical property. This idea is the main motivation for the developed method surrounding feature graphs, which will be established in the following sections.

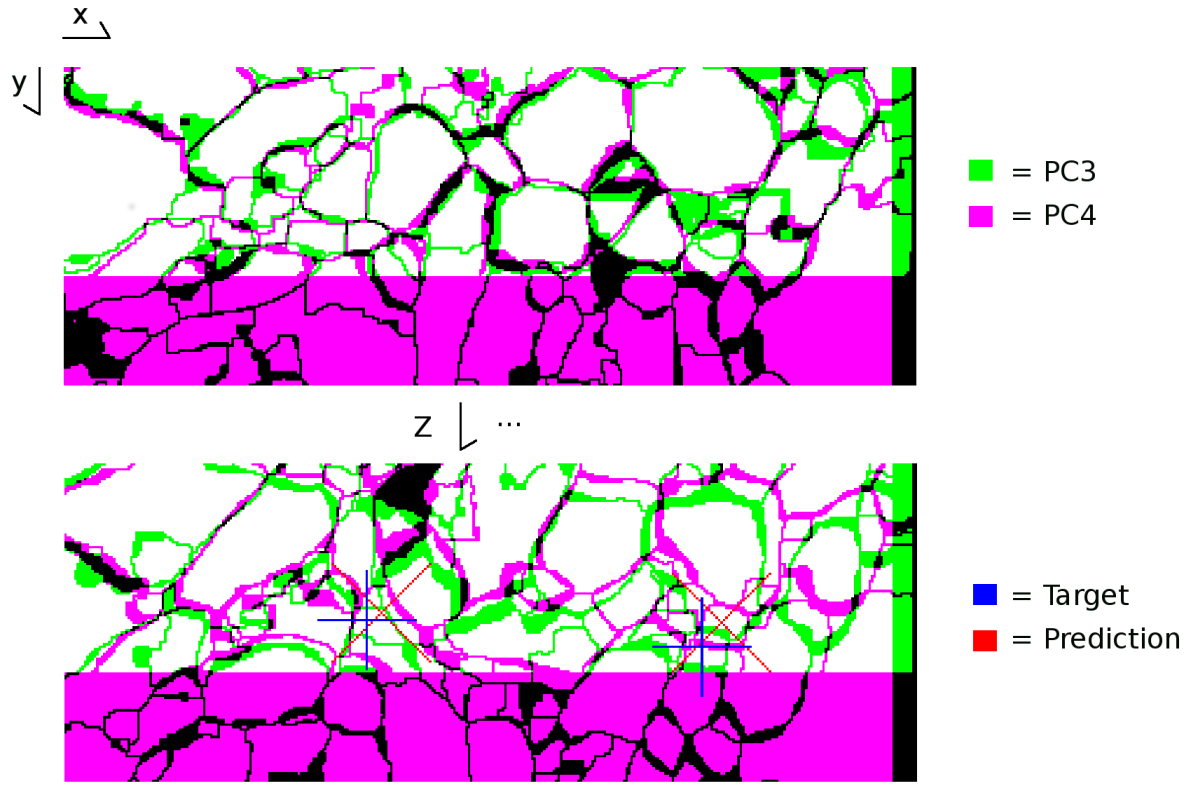

Figure 3.4.: Alignment Visualized. Green represents PC3 which is transformed onto PC4 represented by magenta. The black region is due to the difference in size between the ROIs selected. The upper version of the aligned images is from a correspondence slice i.e., at this depth level the transformation applied to the following stack should be the best because following this slice there exist no correspondence points (until some are encountered and a new motion model is applied). The arrow marked with z and dots (...) represents movement down the stack. The blue cross shows the target point from the ground truth set for PC4, while the red cross is the predicted point onto which the corresponding ground truth point for PC3 is being transformed to.

3.2 Feature Neighborhood Graph Matching

This Section introduces the feature neighborhood graph (FNG) approach. As briefly touched upon in the last section, a classical image alignment approach ignores all information of concepts portraited in the images and their properties i.e., physical/biological knowledge about neuronal tissue. For this purpose, the FNG approach which is going to be discussed in the following sections was implemented. The graph approach which is based around features (but in comparison to significant points or correspondences used for aligning, rather in the sense of attributes as typical for machine learning methods) gives the alignment problem a semantic character. The alignment problem becomes technically a matching problem, which is basically the assignment problem discussed in Section 2.2.2. This holds true as the graphs constitute of nodes that represent the different neuronal processes which need to be matched accordingly. As the neighborhood of a given axon that is part of the bundle is modeled by a graph, the problem can be considered as an instance of graph matching. At first, the different applicational scenarios will be considered. After that, the workflow i.e., pipeline behind the FNG approach will be explored in detail.

3.2.1 Scenarios

After extracting the graphs from the two datasets considered, the matching can be an instance of one of the following four cases. The different cases can be distinguished by two aspects that each can increase the difficulty of the matching process. The first aspect is whether the matching is equal or unequal regarding the number of objects to be matched i.e., the resulting pair matrix that contains all possible matching pairs is either a square matrix or not. The second aspect to increase the overall complexity of the problem is the degree of overlap between the sets considered or the question if a right $1:1$ matching is possible or not i.e., if it is not the case then unique properties of the graphs exist that should not

be matched as for a given process that should be matched the respective partner simply is not part of the other graph (simply put, the overlap does not cover the whole set considered).
Assuming an equal matching (square matrix): in the *first case*, this is the best case scenario, all member of one set have exactly one corresponding member in the other set, the matching is therefore just a matching problem. In the case of an equal matching but no exact correspondence, all member of one set have at most one corresponding member in the other set i.e., some members should actually be unmatched as they belong to the unique fraction of the given set. This means the *second case* actually adds a decision problem to the original matching problem as for some processes the method applied needs to decide whether the process should actually be matched as a partner may not exist. Unequal matching (non-square matrix) makes the original two cases more difficult. Considering the the *third case*, which is essentially the first with an unequal matching, one set is simply the subset of the other as there exists a $1:1$ matching but the sizes of the sets differ making the matching problem more difficult in the sense that more options for a member of the smaller set exist to be matched with. In the *fourth case*, it is unequal matching with non-corresponding sets which is essentially the second case just more difficult as the additional options can be considered as noise that acts as a distractor to the matching. The last case is also the worst case as it is both a matching and decision problem but with overall more pairs to be considered than in the third case. Figure 3.5 below illustrates the discussed scenarios.

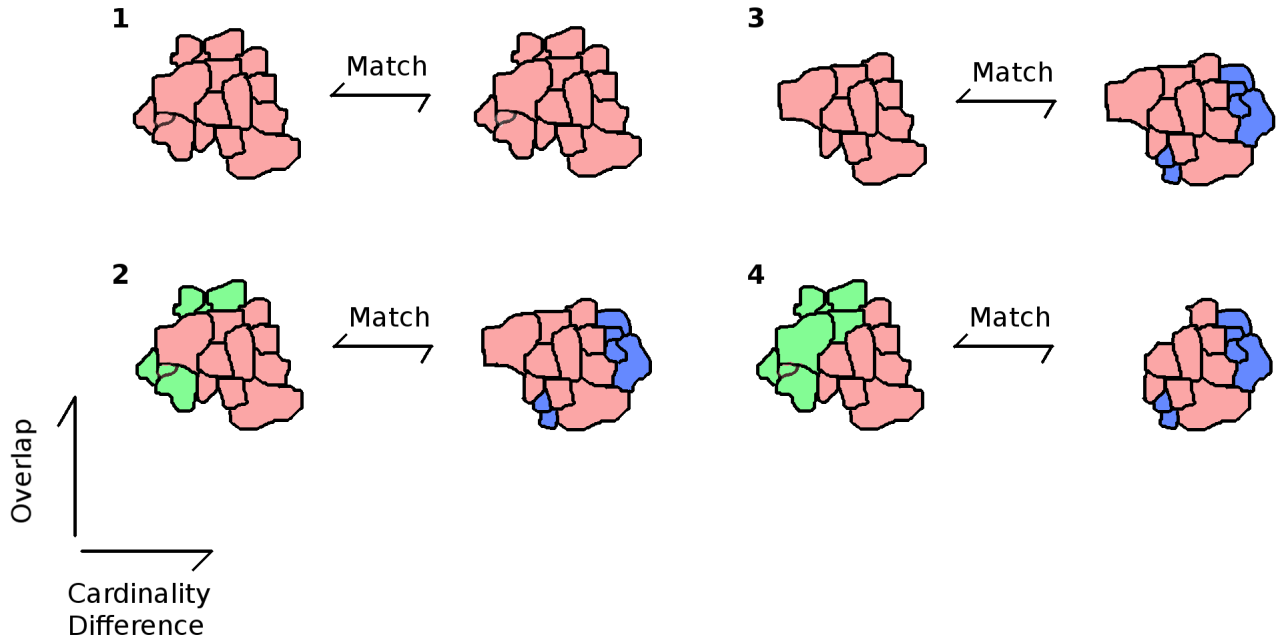

Figure 3.5.: Matching Scenarios. The four discussed cases for matching illustrated. The shapes resemble neuronal processes (mostly axons) that should be matched across the two (data)sets. Following the overlap axis the intersection of the two sets increases and therefore the number of unique members, that make the matching harder, decreases. The cardinality difference axis measures the difference in set sizes which of course makes the matching more complex and computationally expensive (just as when increasing the overall problem size i.e., to be matched members). The first case being the best while the fourth and last case is the worst (from applicational perspective).

3.2.2 Pipeline

To handle the different matching scenarios that may occur when trying to match a bundle of axons, the FNG approach has been developed in a modular and parametric way. In this Section, the reader will be introduced to the pipeline of the developed method, which separates all important steps cleanly and allows for the structured application of the method with ease-of-use. The pipeline divides the developed approach into different categories for which the input and output are clearly defined and which make the approach modular in the sense that steps can be improved by use of custom methods. These categories or steps will be introduced each at a time in detail to complete the presentation of the whole process, which performance will then be tested in experiments in the following chapter. As input, the pipeline

takes the region of interest (ROI) i.e., bounding box for the bundle to be matched, a tracing which will be explained in more detail, a ground truth set to evaluate the overall performance and 'strategies' concerning different aspects of the matching procedure. The first two steps are about preprocessing the data, after which the which output is fed into the feature extraction step where the features for graph creation are prepared. After acquiring the graphs the actual matching process is applied given some selection of the features that should be taken into account and an optimization and cost strategy which should be followed. This procedure then returns the predictions for the matching. In the last step of evaluation, the overall performance on the ground truth set is returned and the matching and its correlation with the cost are being visualized. Figure 3.6 below shows the different steps, their respective Input/Output, the overall parameters the method is fed with and its results that are being returned.

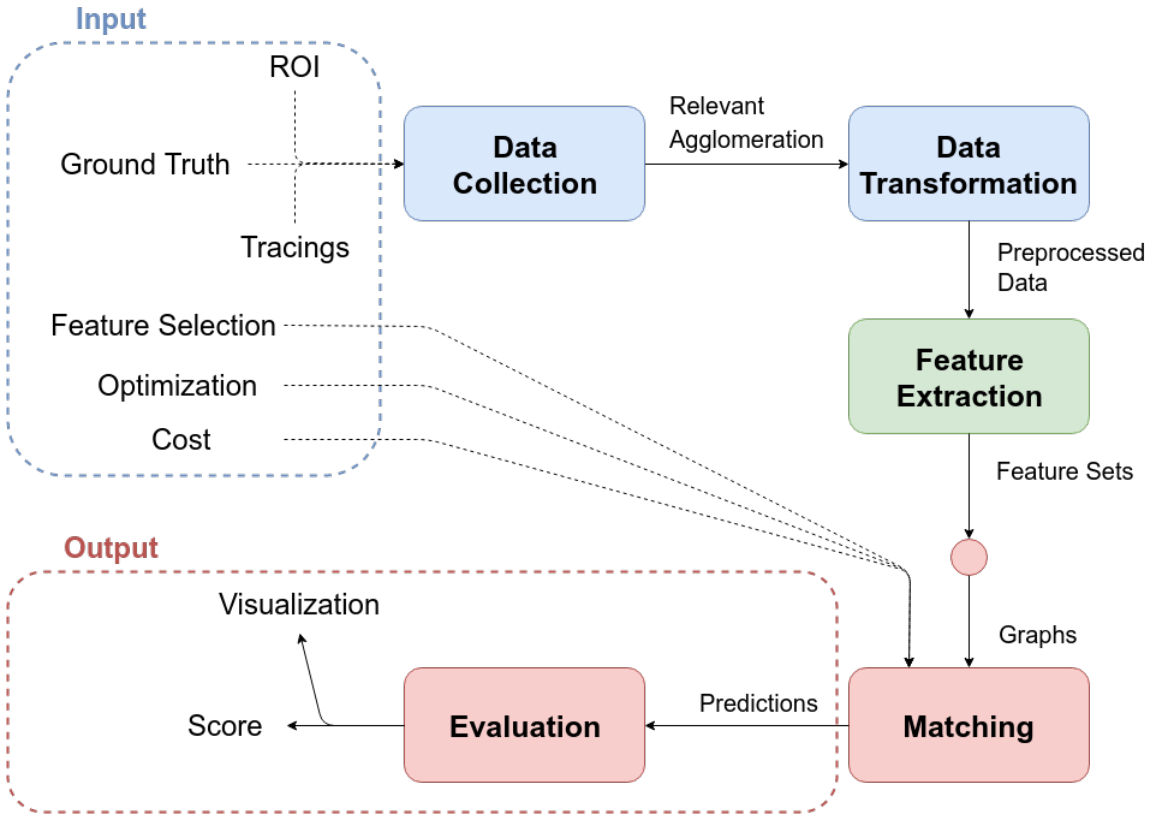

Figure 3.6.: FNG Pipeline. The pipeline of the feature neighborhood graph approach. The input can be considered as split into data-related and strategy-related. The pipeline itself consists of five modular steps that can be categorized into data preprocessing, graph creation and application. The output is then defined by the predictions, the score measured on the ground truth set and visualizations that can be used for analysis purpose.

Data Collection

For the matching, it is required to have datasets that constitute the components that need to be assigned accordingly. Therefore the first step in the pipeline of this approach is *Data Collection*. At first, given some bounding box that marks the ROI that should contain the neuronal bundle, the segmentation data in this area is loaded. In Connectomics, huge datasets are being processed and these are stored on a cluster (briefly mentioned in Section 2.1.3). The segmentation, as compared to the alignment approach (see Section 3.1.1), is here of high importance because the segmentation is the basis for the agglomeration applied later, which is there to separate processes and give them a sort of identification which then again is used to create the nodes of the graphs. So the segmentation of the data is mandatory to the whole procedure. To shortly summarize the segmentation using *SegEM* [26], essentially using a CNN to predict membranes followed by a watershed algorithm resulting in a segmentation for which probabilities of connections between segments are kept in a graph structure to basically assign each process an identity. Based on this segmentation data the tracings can be used to create sets of segments that belong to the same physical process. These sets are then called *agglomerates* and the tracings are essentially manually merged segments through following the traces of neuronal processes in *WebKnossos*. These human-annotated tracings contain therefore the agglomeration which is assumed to be sufficiently correct for

the applied method because initial errors in the agglomerates will most likely result in erroneous matching. Both the loading of the segmentation data and the tracings are done twice as two datasets are being considered for the matching. Just once, however, a ground truth set is created that contains true matches between agglomerates of the datasets. To summarize briefly: the datasets, the tracings and a ground truth are being read-in at first. After the data collection, the segmentation is masked to the relevant area and transformed to an agglomeration using the tracings as described. The transformation is done by assigning new IDs to the resulting sets of segments that make up the agglomerates. The same IDs are assigned across the two datasets for the matches contained within the ground truth set. For visualization purposes, this 'recoloring' step is being backed up and can be accessed to get a view/feel for the datasets to be matched.

Data Transformation

The main use case of the FNG approach is to match bundles of axons (even though everything is kept abstract not just for application on fiber bundles but in general data that can be portraited as collections of components, as the method solves the assignment problem). This matching is done either in $2D$ or $3D$, at first however in $2D$ space. For this purpose, the original idea was to reslice the bundle in an, to the main flow direction of the bundle, orthogonal way (which is invariant to $3D$ matching). To reslice, it is necessary to extract the main flow direction at first. This is done by applying PCA (discussed in Section 2.2.4) on each of the agglomerate's surface point cloud (of each respective dataset) and then taking the mean over that. An example of the sampled surface of an agglomerate is shown in Figure 3.7 below.

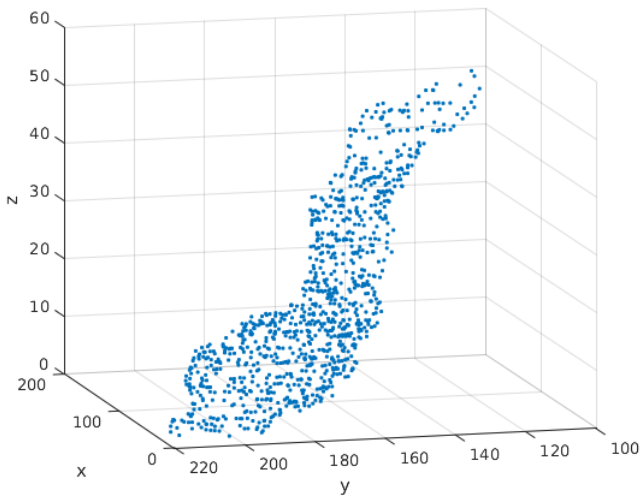

Figure 3.7.: Agglomerate Surface as Point Cloud. An example of a randomly surface sampled point cloud of an agglomerate which is used to calculate the main flow direction of the bundle.

Because PCA extracts the most variance explained with the first principal component, this component is the one that describes the flow direction of a process and taking the mean over all extracted single directions finally results in the collective flow direction of the bundle. This process i.e., the point clouds of the surfaces (or whole processes) can then be visualized. Outliers that detract the direction can be removed manually or by considering the respective variance explained (of that process). After axis extraction, the agglomeration data needs to be resliced i.e., a new 'fly path' along the calculated axis needs to be acquired by transforming the respective dataset accordingly. Figure 3.8 shows the collective point clouds of an example bundle from which the flow axis has been extracted using PCA and the orthogonal slice which is then the new perspective plane i.e., when 'flying-through' the data in two dimensions. This orthogonal reslice plane is defined by the vector that is orthogonal to the main flow vector and the vector that is orthogonal to the xy plane (unit vector in z direction). At the end of the *Data transformation* step, the data is collected in a *MATLAB* structure that contains various versions of the data (raw, transformed, etc.).

Feature Extraction

The next step in the pipeline is about extracting attributes, features, that carry enough information for the matching to discriminate the right pair from the wrong assignments. The critical feature to this method is the *neighborhood*. The neighborhood for a given agglomerate in a bundle is what is unique to this collective structure of neuronal processes because an axon flows with (lots of) other axons. Properties like shape, direction and others are therefore similar for

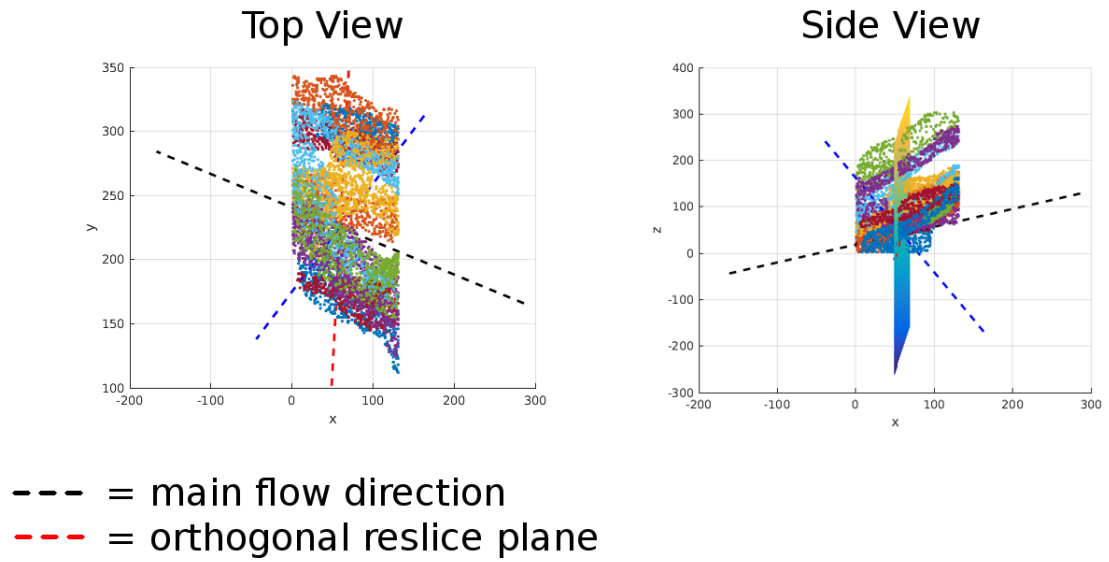

Figure 3.8.: Bundle Main Flow Direction. Two views for a collection of surface point clouds of agglomerates that are part of a bundle are shown. The dashed black line is the main flow direction calculated by PCA, the red one is the orthogonal vector along which the reslice plane (shown in the right view) is set. This information is used to calculate a new flight path along the data. The blue line is simply another orthogonal vector for illustration.

these processes. For instance, by the count of neighbors the center becomes distinguishable from the outer parts of the bundle. Neighborhood is an essential feature that is encoded later in the graph structure being the connections (edges) between the processes (nodes). A lookup table that maps segment IDs to agglomerate IDs is used to collect all edges from the *SegEM* graph mentioned earlier, which is enough to extract the neighbors for each agglomerate within the preprocessed data. Furthermore, other features like *areas* for specific slices ($2D$) or the *volume* ($3D$), *centroids*, *k-nearest centroids* and *bounding rectangles* are collected. All features are collected even if later certain ones are not selected by the feature strategy which is also part of the input to the pipeline. After collection, the features are being normalized to assure equal scales and therefore influence on the matching later on. Finally, all collected and normalized features are saved within feature sets from which, as explained in the next section, the graphs to be matched are created.

Graph Creation and Matching

From the feature sets created in the last section, the graphs are finally created. The graphs are neighborhood graphs i.e., the edges represent connections between agglomerates, which again are represented as nodes. These nodes contain the features/attributes, therefore the name to this approach is given by feature neighborhood graph matching.
For the *Matching*, a $(N \times M)$ pair matrix $\mathbf{P}$ is created where N and M represent the cardinality of the graphs. This matrix contains all possible combinatorial assignments. As introduced in Section 2.2.2 the search space grows factorial. For all possible combinations, a cost, given some cost function and features that should be considered, is calculated resulting in a cost matrix $\mathbf{C}$. This cost matrix $\mathbf{C}$ is then optimized given some initial strategy which can either be to walk *greedily* through the cost space i.e., taking always the best option considerable in the specific moment or to take the results of the Hungarian/Munkres algorithm (discussed in Section 2.2.3). The result of the latter is the guaranteed global minimum for the cost matrix. For the optimization to return desired results the cost function must be well chosen/designed. The FNG pipeline allows for different cost functions to be applied, following three are ready-to-use. To formally define the mathematical notion of the FNG approach, G_i for $i \in 1,2$ represent the graphs to be matched and $x \in G_i$ is a N-dimensional vector that is encoded within a graph as a node. The matching problem is then defined as

$$\begin{gathered} M := \{m := (x,y) \,|x \in G_1 \wedge y \in G_2\} \\ \forall x \in G_1 : \forall y \in G_2 : (x,y) \in M \implies (\neg\exists z \in G_1 : (z,y) \in M \wedge \neg\exists z \in G_2 : (x,z) \in M) \end{gathered} \tag{3.1}$$

where the M is a set of matches m between pairs of nodes from the graphs called matching and the matching is exact i.e., a node appears only once in a matching. Further an error/cost function E is defined as $E : G_1 \times G_2 \rightarrow \mathbb{R}$, $(x, y) \mapsto c$ where the goal and final result of the matching can be written as

$$\arg\min_{M} \sum_{i=1}^{d} E(m_i) \tag{3.2}$$

where d is the number of processes to be matched (in the case of a square matrix $d = N$).
The first option for the cost strategy is the usage of a *custom error function* which can be either a Manhatten norm given $\|\boldsymbol{x}\|_1 := \sum_{i=1}^{n} |x_i|$, the Euclidean norm given $\|\boldsymbol{x}\|_2 := \sqrt{x_1^2 + \cdots + x_n^2}$ or a weighted norm over different combinations of features given some free parameter(s) α_i with $i \in 1, ..., F$ (where F is the number of considered features).
The second option is the use of LMNN as first introduced in Section 2.2.5 by learning a distance metric. The method is trained on the set of correct matches given the ground truth to produce the transformation matrix $\mathbf{L}$ which is then reapplied to the data from the graphs. In the resulting transformed space the correct matches are closer to each other while wrong combinations are separated by a margin (all under Euclidean distance).
The last option is the usage of learned scores from a trained neural network as presented in Section 2.2.6. A simple, feed-forward neural network is trained on a set of automatically generated labeled data that consists of (duplicates of) correct and wrong matches. The neural network based scores therefore measure the compatibility of two given nodes from each set, ideally discovering a correlation underlying the features that is sufficient to discriminate right from wrong matches. All presented options for the cost strategy have been implemented and can be utilized in the pipeline. Finally, to conclude the matching step, it is important to again consider the role of optimization. With Hungarian/Munkres the global optimum is guaranteed but the matching must not necessarily be the desired result. Only if the optimum for the given cost represents the desired result, i.e., the error function truly measures the compatibility of a match, then the matching is solved perfectly.

Evaluation and Visualization

The last step of the pipeline is the *Evaluation* and therefore measuring the performance of the method on the data. A score is calculated by simply comparing the result against the ground truth set i.e., hits divided by total amount of confirmed paired matches. In order to understand further how the different configurations of the method perform qualitatively, visualization methods were implemented. The cost matrix $\mathbf{C}$, as discussed in the last section, in addition to the ground truth solution and the predicted matching can all be visualized. Additionally, for a two-dimensional feature space, the data points of the graphs with the matchings applied can also be visualized. The visualizations give insight into the nature of predictions. The following chapter will present different experiments run with the presented method and use the just mentioned visualization techniques to illustrate the results.

4 Experiments and Results

This Chapter puts the developed methods under test and covers thereby seven experiments over three different scenarios, each measuring different aspects of the FNG approach. All experiments have been run using an implemented testbench and are therefore reproducable with the given code and data. The visualizations used are part of the last step of the pipeline discussed in Section 3.2.2. The datasets used for testing are the same as in the experiment for the correspondence alignment approach, the piezo columns PC3 and PC4. At the end of the chapter, a table is shown to summarize the results of the different experiments.

4.1 Matching within a Ground Truth Set

The first part of this experiment series is the matching within only the ground truth set. As discussed in the last chapter, the pipeline takes a ground truth set, consisting of agglomerate pairs that have been matched over the PCs as input, to measure the performance in the final evaluation step. In this scenario only this ground truth set is considered as the data given and is used for running the FNG pipeline. It is over different piezo columns, but on a small scale and therefore only suitable to prove basic functionalities of the developed method. The very first experiment assumes a custom cost function (being just the basic Euclidean distance between pairs of nodes/processes) and $2D$ features being *area* and *k-nearest centroids* where $k = 4$ (set arbitrarily). The optimization strategy was set to greedy, the results are shown in the following figure (score 0.15). This configuration is only able to match two pairs correctly on the ground truth.

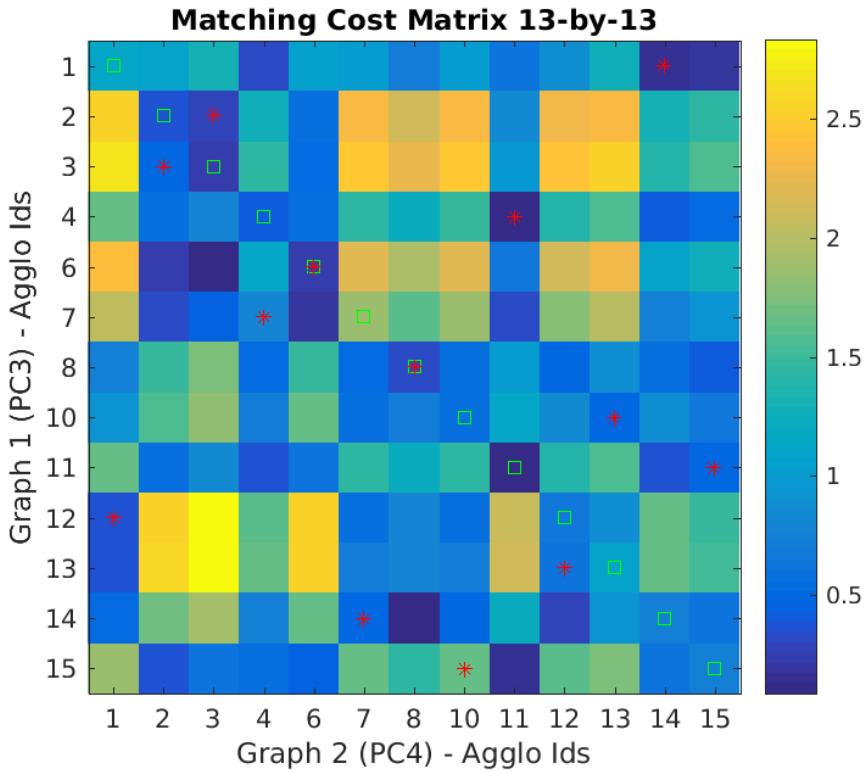

Figure 4.1.: Experiment 1. Features: area, k-nearest centroids. Optimization: greedy. Cost: Euclidean distance. Score: 0.15 (2/13).

The green rectangles represent the ground truth matches, they lie on the diagonal of the matrix because the pairs from the ground truth set or the agglomerates of the different PCs that make up these matches are 'recolored' the same way as discussed in Section 3.2.2. The red stars are matches predicted by the method. The remaining different colors in the cost matrix visualize the cost of each combination, a darker color being a low and therefore preferable cost.

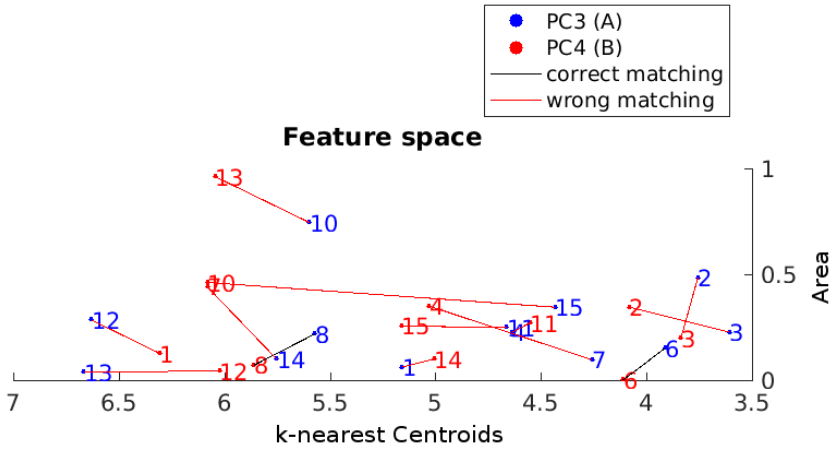

Figure 4.2.: Feature Space for Experiment 1. Score: 0.15 (2/13).

The above Figure visualizes the two-dimensional feature space and the matching predictions. The red lines represent wrong matches, while the black ones represent correct matches. As the reader can see the Euclidean distance between match pairs is not enough as the features considered position the corresponding nodes/processes at different places in that space.
This first experiment shows that there are a variety of different parameters that can be optimized to improve the matching. The following second experiment utilizes the Hungarian/Munkres algorithm introduced in Section 2.2.3 instead of a greedy optimization to return the global minimum of the computed Euclidean cost matrix. Figure 4.3 below shows the cost matrix. In comparison to the greedy approach, the global solution (score 0.23) comes closer to the desired solution

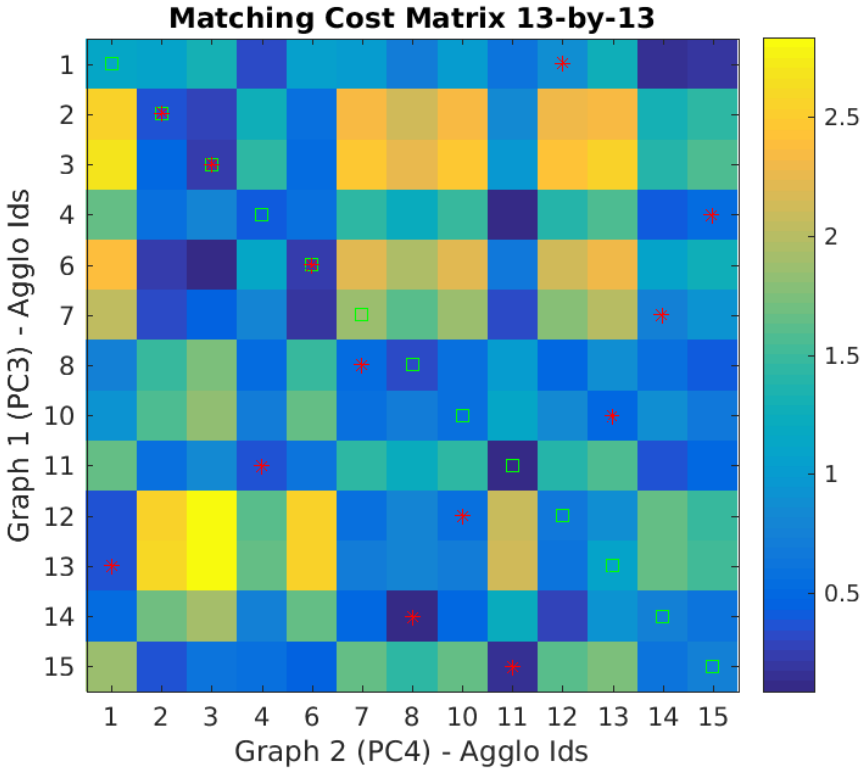

Figure 4.3.: Experiment 2. Features: area, k-nearest centroids. Optimization: global. Cost: Euclidean distance. Score: 0.23 (3/13).

(even though still far from optimal). Looking closely the mixed up matching between pairs (2,2) and (3,3) are corrected as the global approach sees that this combination is better than (2,3) and (3,2), as short-sightedly chosen previously by the greedy strategy. The third and last experiment as part of the matching within the given ground truth set again

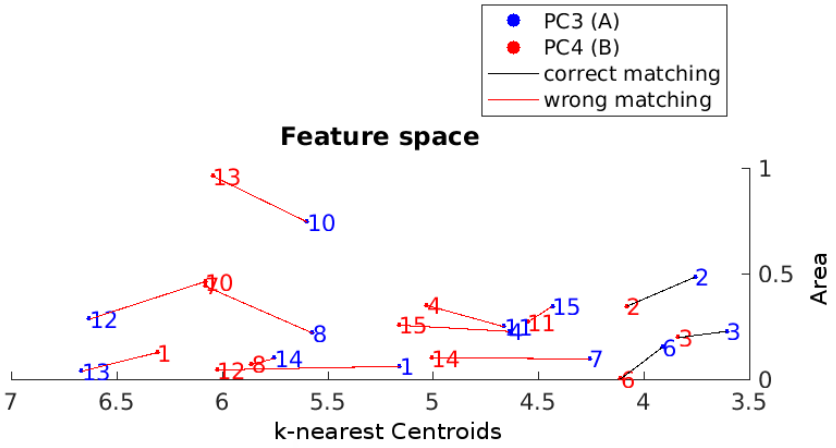

Figure 4.4.: Feature Space for Experiment 2. Score: 0.23 (3/13).

optimizes for a global solution but this time the error function is different. While Euclidean distance is kept, the data is transformed beforehand using a transformation matrix learned with LMNN on the ground truth set, where the matches are labeled as part of the same class. The resulting cost matrix clearly shows that the correct matches actually have the lowest cost. Overfitting does not seem to be the case as similar nodes (in the sense that the same process across the datasets should be matched) result in a similar cost. However, this argument is not proven i.e., there has been no further test on a different bundle that is unknown to the configuration with LMNN learned cost.
A perfect matching is achieved even though the real datasets are used, in which the processes differ drastically in shape over the depth cutting direction as first mentioned in Section 2.1.3. These deformations are due to the cubing process which is influenced by the unique parts of the datasets and they affect both the $2D$ and $3D$ features.

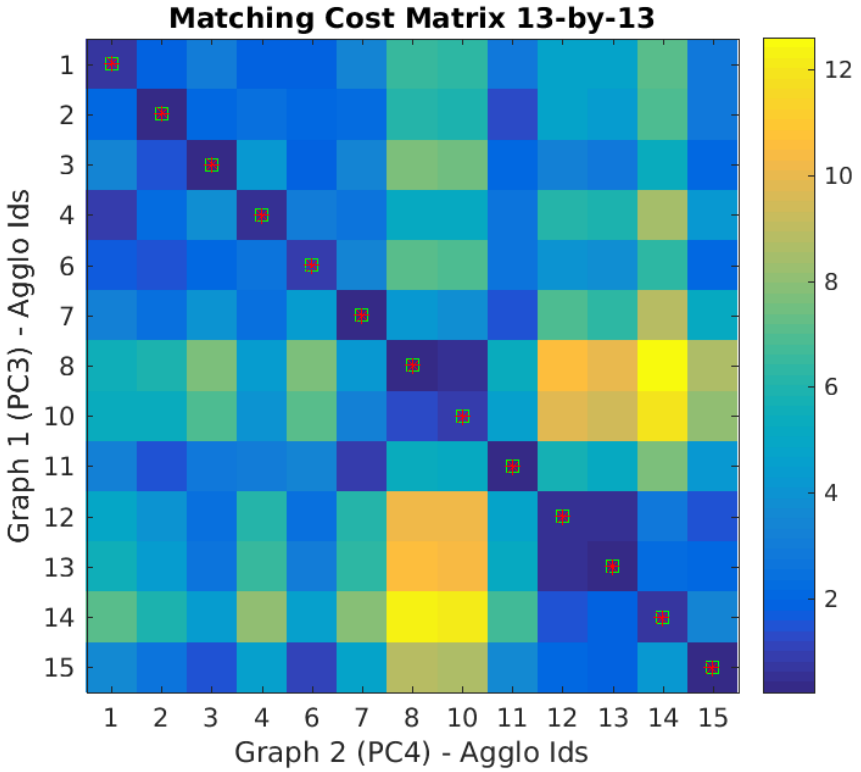

Figure 4.5.: Experiment 3. Features: area, k-nearest centroids. Optimization: global. Cost: LMNN learned. Score: 1.00 (13/13).

The feature space visualization also clearly illustrates the conceptual behavior of the LMNN distance metric learning algorithm as inter-class distance is minimized while the intra-class distance is maximized.

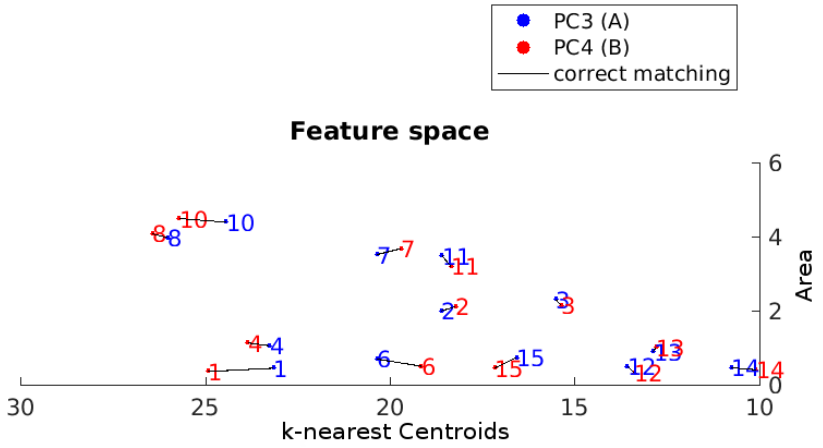

Figure 4.6.: Feature Space for Experiment 3. Score: 1.00 (13/13).

4.2 Matching within the Same Piezo Column

In this second application case the same piezo column is assumed i.e., the same data is utilized for the matching which is trivial to 3*D* features (because it is trivial to match between volumes that are identical and 3*D* features under the assumption of same data create identical sets) but of similar difficulty to the 2*D* features. For the 2*D* features an approximation of corresponding slices is given to the method, however, the challenge is this time increased by considering greater graphs i.e., more nodes/processes. In the fourth experiment the parameters are set like in the second experiment (see Section 4.1). Compared to the last section, the size of the matching is increased. To further increase the complexity, an unequal matching, just as discussed in Section 3.2.1, is performed. Figure 4.7 below illustrates the results (score 0.18). As expected the results are rather poor but the pipeline is also able to handle unequal matching of datasets with

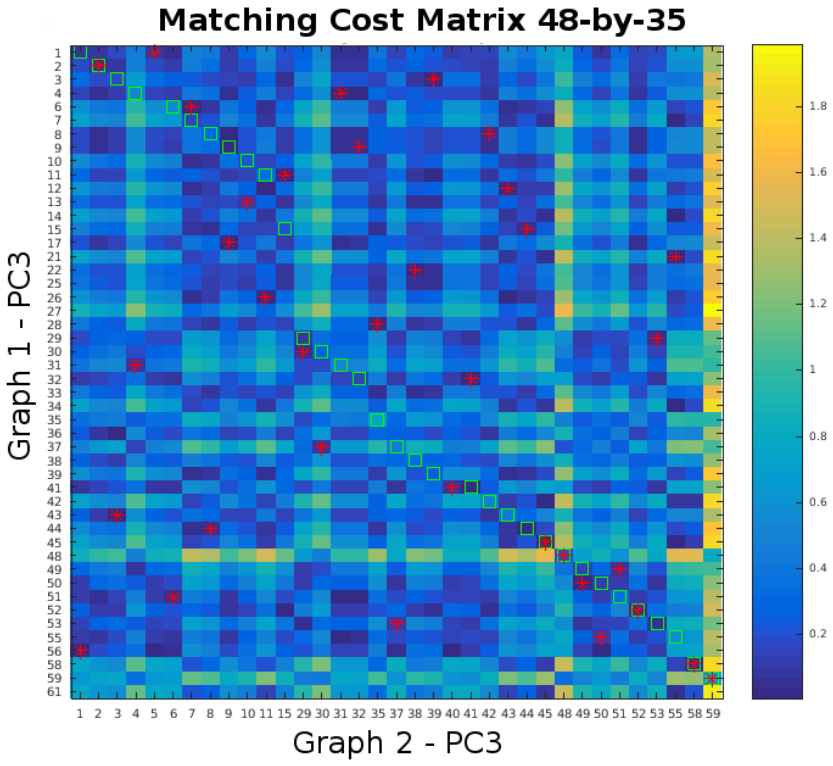

Figure 4.7.: Experiment 4. Features: area, k-nearest centroids. Optimization: global. Cost: Euclidean distance. Score: 0.18 (6/33).

higher cardinality (at least to the extent which is on the scale of the encounterable number of units within a bundle). The discriminative power of the selected features are simply not good enough for this matching scenario, therefore following

fifth experiment switches to 3*D* features. In this case, a perfect matching is expected as the same data is assumed and therefore the matching with the right 3*D* features should be sufficient to identify all corresponding pairs across the graphs. Indeed, the usage of *volume* and *neighborhood* across the whole region (in this case just the count of neighbors)

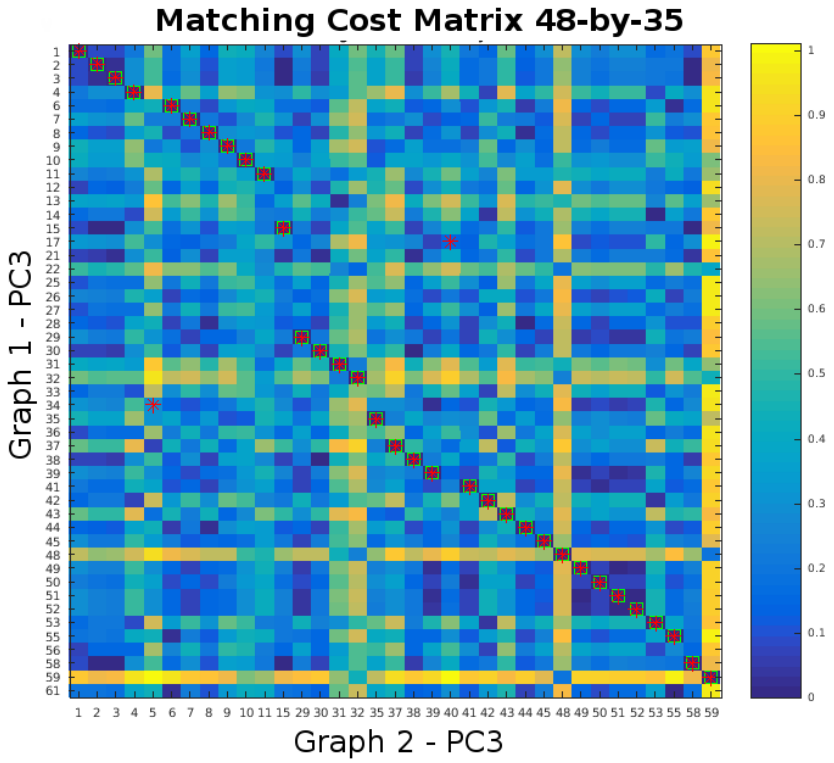

Figure 4.8.: Experiment 5. Features: volume, neighborhood. Optimization: global. Cost: Euclidean distance. Score: 1.00 (33/33).

is sufficient to describe a perfect matching as shown in Figure 4.8. This switch in feature selection illustrates finely the importance of the features as basis for the assignment procedure. As this is an unequal matching (restricted by the cardinality of the smaller set), some of the members of the bigger set (here graph 1) are left unmatched. Most of these unmatched are correct, however, as can be seen for node with ID 40 from graph 2 it is being matched to node with ID 17 from graph 1, which is incorrect as the right (corresponding) node actually does not exist in this matching scenario. Anyhow, this specific match resembles the closest result within this matching scenario to the actual right match.

4.3 Matching over Different Piezo Columns

Matching over different piezo columns (like in the first scenario) with the complete tracings make this third and last scenario the hardest being basically the real world application scenario. As insight from the last two sections, the used optimization and features are crucial. But as the diligent reader may noticed, in Experiment 3 (see Section 4.1), the cost/error function chosen is also of fundamental importance as the influence on the result of this experiment is immense, being able to improve the performance of the procedure by far easily.
Like before, the sixth experiment considers the 3*D* features (volume, neighborhood) under Euclidean distance as the cost with the global solution optimized for by the Hungarian/Munkres algorithm. All tracings of the two PCs are considered making this the hardest scenario, as discussed in Section 3.2.1, where an unequal matching across different datasets (that therefore don't share a complete overlap and have unique parts that disturb the matching) is given. Evaluation is again done on the ground truth set. Figure 4.9 shows the results. The method performs underwhelming for the test set (score 0.23). The Euclidean norm is simply not good enough in 'interpreting' the features at hand in a sense that the desired matching, which should assign the same process across the datasets to itself, can be achieved. Therefore the last and seventh experiment tries a similar strategy to the one in Experiment 3. But instead of learning a transformation matrix to transform the space in such a way that the Euclidean distance is enough to correctly interpret the desired matching, a neural network is being trained. Figure 4.10 illustrates the results, which have improved significantly compared to the last configuration (score 0.46). The matching is far from perfect but as can be seen the predictions of the trained network are actually able to improve the matching score on the ground truth set by a factor of 2.

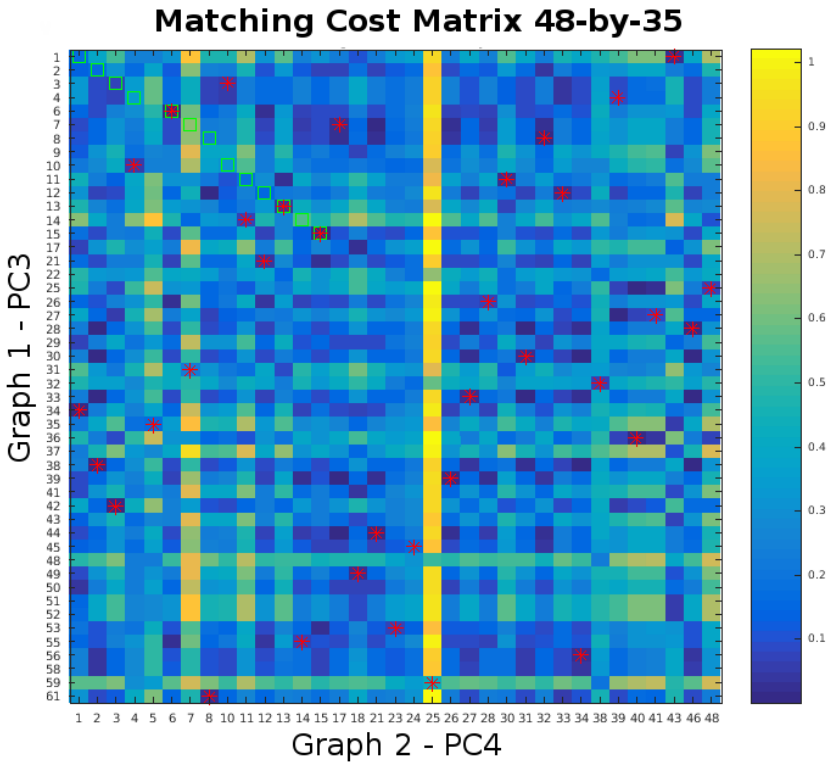

Figure 4.9.: Experiment 6. Features: volume, neighborhood. Optimization: global. Cost: Euclidean distance. Score: 0.23 (3/13).

As first introduced in Section 2.2.6, neural networks are capable of learning functions that are hard (if at all possible) to design manually. By learning in a supervised manner directly from the data at hand, the hope lies in the discovery of the

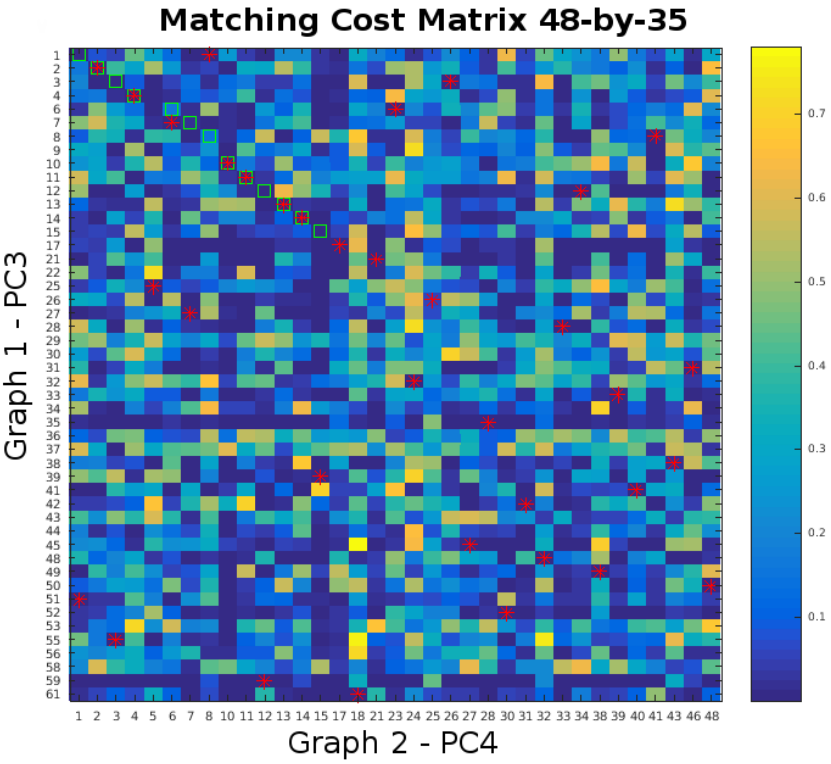

Figure 4.10.: Experiment 7. Features: volume, neighborhood, area, connections. Optimization: global. Cost: NN learned. Score: 0.46 (6/13).

real relationship that is described by the underlying data distribution. A simple feed-forward network was trained with labels that indicate whether a match is correct or not to produce probability scores that should describe whether the two processes (a given combination of nodes) are compatible or not. All features used in the different experiments so far have been used for the training, which is done on automatically generated sets based on the ground truth set. Interestingly, when observing the cost matrix the reader may notice that the neural network seems to predict a pattern that represents a kind of binary 'thinking' of the network as some combinations are completely penalized and not considered but then again some are and therefore very likely to be chosen by the prediction.
Following Table 4.1 summarizes the results discussed in the last three sections of this chapter.

Scenario	Features	Optimization	Cost	Score (Hits/Total)
Ground truth only	area, k-nearest centroids	greedy	Euclidean	0.154 (2/13)
	area, k-nearest centroids	global	Euclidean	0.231 (3/13)
	area, k-nearest centroids	global	LMNN learned	**1.00** (13/13)
Within same PC	area, k-nearest centroids	global	Euclidean	0.182 (6/33)
	volume, neighborhood	global	Euclidean	**1.00** (33/33)
Over different PCs	volume, neighborhood	global	Euclidean	0.231 (3/13)
	v., nb., a., k-nc	global	NN learned	**0.462** (6/13)

Table 4.1.: FNG Matching Experiment. This table summarizes, for the presented experiment scenarios, how the developed method performs across different parameter settings concerning feature selection, optimization and cost/error strategy. The best results within a scenario are achieved by the last experiment within the experiment series. Global optimization is achieved by applying the Hungarian/Munkres algorithm to the cost matrix. All learning based approaches use the full and same ground truth set as in the first experiment scenario.

The first scenario clearly illustrates that the choice of optimization is crucial for the method to robustly find the best matching. Without a proper cost function, however, that evaluates for given matches how suitable they are for the final predictions, even a proper optimization is rendered useless. So the matching is essentially dependent on both of these components. Another dependency has been shown by the second scenario, which illustrated that the choice of features, which essentially describes all information about the data that is available, is also key to a proper matching. The last scenario shows that applying all these principles can lead to good improvement in the overall matching process, however, it also shows that the real-world application is difficult and that there is still lots of room for improvement left. This can be achieved either through enhancement of the three discussed principles or the addition of some new insights that enable progress independent of these principles (or a combination of both). A comparison of this approach to the correspondence alignment can only be considered limited as this approach is (fully) automatic and does not depend on the manual collection of correspondences. Furthermore, the correspondence alignment procedure has been applied to larger datastacks, while this approach focused on a specific structure namely bundles of axons. However, the FNG method utilizes knowledge on the properties of the concepts displayed within the data volumes and is therefore better suited for the alignment/matching of such structures as it actually matches instances of e.g., the bundles compared to simply aligning pixel intensities. Being more specialized means also, however, that the computations are more sophisticated and therefore comparably more expensive.

5 Conclusion and Future Work

First, this work introduced an alignment based on classical image alignment techniques. The correspondence alignment approach is about utlizing correspondences, i.e., points that represent the same process (physical point in space) across the datasets to be aligned, to achieve a robust and satisfying alignment. Motion models, which are key to the proper alignment and account for the transformations applied to the image stacks, have been presented and discussed. Furthermore, visualization possibilities have been shown that can also be used to improve the discussed method. The performance was evaluated within an experiment that covered three different categories to help discovering the strengths and weaknesses. A major disadvantage concluded from the overall concept of the correspondence alignment approach was the fact that accessible biological information about the given neuronal data is rendered useless as it is not considered in any way during application. This observation led to the development of what is considered the main approach of this work, the feature neighborhood graph matching. The second method introduced focused on a special structure that reoccurs in lots of various brain areas, bundles of axons, which can be generalized to the concept of collections of similar components and make the method therefore applicable to domains outside the neuroscientific context. As the graphs constructed are representations of the physical bundles in the form of sets of feature nodes that are interconnected (and the connections represent the respective neighbors of a process), the FNG approach is of semantic character and transforms the alignment problem to a matching (or assignment) problem. A pipeline was introduced that reaches from data collection over preprocessing and feature extraction to the matching itself. Evaluation and visualization methods for debugging and improvement purposes were introduced as well. A series of experiments that span across three different scenarios showed that aspects like feature selection, an optimization strategy and the choice of a proper cost/error function are crucial to achieving a good matching. The case where matching is done across different datasets that are influenced highly by distortions and noise artifacts during the initial recording, however, showed that there is still a lot of improvement possible. The configurations that involved learning were in all cases able to improve the performance drastically, however, there has been no further testing on another separate bundle of axons (given data scarcity) i.e., the training and test sets are (mostly) the same therefore it is not shown how the method performs on new unseen data. To conclude, this work has developed methods that offer partial solution to the local alignment problem answering the questions concerning how to efficiently handle and match data across neighboring datasets. In comparison to the presented methods that were about matching specific structures, a brief Appendix Chapter follows that covers the identification of such specific structures in the first place.

As mentioned already on various occasions, the importance of features is obvious and the alteration of this 'parameter' has great influence on the resulting predictions of the matching procedure. Therefore the addition of more and discriminatively stronger features is a source of improvement that should be considered in future work, especially features that may be automatically extractable. Furthermore, the addition of a framework that allows for utilization of even deeper knowledge about the neuronal processes i.e., knowledge like 'a neuron has only one axon while many dendrites', could improve the developed method as it already utilizes semantic information (but only geometry-like properties so far). Incorporating knowledge on the specific class of a given process (i.e., biological type like dendrite, axon, etc.) could further boost performance. Additionally, to the improvement of the features and the utilization of biological knowledge, proper training of the models that are used for establishing the cost of the combinatorial matching possibilities (like has been shown with learned transformation matrices with large margin nearest neighbor or compatibility scores of a neural network) using greater training sets that are specially designed with the matching of this neuronal data in mind could further improve the matching drastically. Another interesting aspect regarding future work based on this project may be the development of a hybrid solution of the two presented methods i.e., a combination of the developed approaches that utilizes the first to get a first coarse alignment of the data that covers especially easier regions. These 'easy' regions could be assessed by a confidence/uncertainty measurement that then if the hybrid method proposes that a given region is considered as uncertain the second part of the method is applied that extracts the region and considers a matching of the components contained within that region of interest. Practically this would utilize the best of both concepts, as the first approach is (relatively) computationally inexpensive and performs sufficiently well for most of the neuronal data and the second approach would step in when an uncertain region requires further investigation. As the appendix discusses, the identification of reoccuring neuronal 'themes' is an important prerequisite to the subsequent step of matching, therefore the automatic identification of axonal bundles through maybe the training of a 3D convolutional neural network that traverses the dataset and predicts for a given observed space whether a bundle is contained or not (and possibly where exactly within that specific frame) would be of high interest and value. While most of the CNN approaches in the past focussed on $2D$ images, neuronal tissue data being used in the context of this work, just like medical data in general, is $3D$ and there has been recent progress in the development of volumetric CNNs that could lay the foundation for the

presented idea [27]. With the methods, potential improvements and extensions proposed in this work, Connectomics and in more general the study of the brain has been extended by insights that in future may have an impact by either direct application or by laying the foundation for future work.

Bibliography

[1] J. Olesen, A. Gustavsson, M. Svensson, H.-U. Wittchen, and B. Jönsson, "The economic cost of brain disorders in europe," *European journal of neurology*, vol. 19, no. 1, pp. 155–162, 2012.

[2] F. A. Azevedo, L. R. Carvalho, L. T. Grinberg, J. M. Farfel, R. E. Ferretti, R. E. Leite, R. Lent, S. Herculano-Houzel, *et al.*, "Equal numbers of neuronal and nonneuronal cells make the human brain an isometrically scaled-up primate brain," *Journal of Comparative Neurology*, vol. 513, no. 5, pp. 532–541, 2009.

[3] H.-U. Wittchen, F. Jacobi, J. Rehm, A. Gustavsson, M. Svensson, B. Jönsson, J. Olesen, C. Allgulander, J. Alonso, C. Faravelli, *et al.*, "The size and burden of mental disorders and other disorders of the brain in europe 2010," *European Neuropsychopharmacology*, vol. 21, no. 9, pp. 655–679, 2011.

[4] W. Denk, K. L. Briggman, and M. Helmstaedter, "Structural neurobiology: missing link to a mechanistic understanding of neural computation," *Nature Reviews Neuroscience*, vol. 13, no. 5, pp. 351–358, 2012.

[5] J. W. Lichtman, H. Pfister, and N. Shavit, "The big data challenges of connectomics," *Nature neuroscience*, vol. 17, no. 11, pp. 1448–1454, 2014.

[6] W. Denk and H. Horstmann, "Serial block-face scanning electron microscopy to reconstruct three-dimensional tissue nanostructure," *PLoS biology*, vol. 2, no. 11, p. e329, 2004.

[7] G. Binnig and H. Rohrer, "Scanning tunneling microscopy," *Surface science*, vol. 126, no. 1-3, pp. 236–244, 1983.

[8] M. E. Diamond, M. Von Heimendahl, P. M. Knutsen, D. Kleinfeld, and E. Ahissar, "'where'and'what'in the whisker sensorimotor system," *Nature reviews. Neuroscience*, vol. 9, no. 8, p. 601, 2008.

[9] K. M. Boergens, M. Berning, T. Bocklisch, D. Bräunlein, F. Drawitsch, J. Frohnhofen, T. Herold, P. Otto, N. Rzepka, T. Werkmeister, *et al.*, "webknossos: efficient online 3d data annotation for connectomics," *nature methods*, vol. 14, no. 7, pp. 691–694, 2017.

[10] R. Szeliski, "Image alignment and stitching: A tutorial," *Foundations and Trends® in Computer Graphics and Vision*, vol. 2, no. 1, pp. 1–104, 2006.

[11] T. S. Caetano, J. J. McAuley, L. Cheng, Q. V. Le, and A. J. Smola, "Learning graph matching," *IEEE transactions on pattern analysis and machine intelligence*, vol. 31, no. 6, pp. 1048–1058, 2009.

[12] J. Munkres, "Algorithms for the assignment and transportation problems," *Journal of the society for industrial and applied mathematics*, vol. 5, no. 1, pp. 32–38, 1957.

[13] K. Pearson, "Liii. on lines and planes of closest fit to systems of points in space," *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, vol. 2, no. 11, pp. 559–572, 1901.

[14] H. Hotelling, "Analysis of a complex of statistical variables into principal components.," *Journal of educational psychology*, vol. 24, no. 6, p. 417, 1933.

[15] Y. Bengio, Y. LeCun, *et al.*, "Scaling learning algorithms towards ai," *Large-scale kernel machines*, vol. 34, no. 5, pp. 1–41, 2007.

[16] K. Q. Weinberger and L. K. Saul, "Distance metric learning for large margin nearest neighbor classification," *Journal of Machine Learning Research*, vol. 10, no. Feb, pp. 207–244, 2009.

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

[18] F. Rosenblatt, "The perceptron: A probabilistic model for information storage and organization in the brain.," *Psychological review*, vol. 65, no. 6, p. 386, 1958.

[19] M. Minsky and S. Papert, "Perceptrons.," 1969.

[20] Y. LeCun, Y. Bengio, *et al.*, "Convolutional networks for images, speech, and time series," *The handbook of brain theory and neural networks*, vol. 3361, no. 10, p. 1995, 1995.

[21] K. Hornik, M. Stinchcombe, and H. White, "Multilayer feedforward networks are universal approximators," *Neural networks*, vol. 2, no. 5, pp. 359–366, 1989.

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

[23] D. G. Lowe, "Object recognition from local scale-invariant features," in *Computer vision, 1999. The proceedings of the seventh IEEE international conference on*, vol. 2, pp. 1150–1157, Ieee, 1999.

[24] H. Bay, A. Ess, T. Tuytelaars, and L. Van Gool, "Speeded-up robust features (surf)," *Computer vision and image understanding*, vol. 110, no. 3, pp. 346–359, 2008.

[25] M. Brown, D. G. Lowe, *et al.*, "Recognising panoramas.," in *ICCV*, vol. 3, p. 1218, 2003.

[26] M. Berning, K. M. Boergens, and M. Helmstaedter, "Segem: efficient image analysis for high-resolution connectomics," *Neuron*, vol. 87, no. 6, pp. 1193–1206, 2015.

[27] F. Milletari, N. Navab, and S.-A. Ahmadi, "V-net: Fully convolutional neural networks for volumetric medical image segmentation," in *3D Vision (3DV), 2016 Fourth International Conference on*, pp. 565–571, IEEE, 2016.

[28] B. Zonta, A. Desmazieres, A. Rinaldi, S. Tait, D. L. Sherman, M. F. Nolan, and P. J. Brophy, "A critical role for neurofascin in regulating action potential initiation through maintenance of the axon initial segment," *Neuron*, vol. 69, no. 5, pp. 945–956, 2011.

A Identifying Other Brain Structures: AD and AIS

In order to match certain structures across dataset boundaries, as has been done with the developed methods throughout this work, it is necessary to have structures to be considered in the first place. In this Appendix, another part of additional work that has been done during the time invested in this project is being presented that is about identifying structures just as remarkable as bundles of axons. Collections like axonal bundles stand out because of their unique properties and appearance in neuronal tissue, two other interesting structures are apical dendrites (AD) and the axon initial segments (AIS). While the first structure reoccurs within various brain areas the other is an essential part of the basic composition of a neuron.

A.1 Extraction of Apical Dendrites

Apical dendrites (AD) describe dendrites that emerge from the apex/peak of a pyramidal soma. This dendritic component of pyramidal cells distinguishes these cell types from spiny stellate cells which are also encountered within the barrel cortex. The structure of pyramidal neurons although being stereotypical can vary between and within regions (e.g., different brain areas, different layers). Another category of dendrites, basal dendrites, occur at the base of pyramidal cells where also the outgoing axon is expected. An apical dendrite travels very straight and can extend for several hundred microns before branching to form the so-called apical tuft. These properties are what make this certain structure special and the following filtering steps, implemented in a method, make use of these criterions to extract these structures for a given state of dendrite agglomerates. The agglomeration is done automatically and based on *SegEM* as mentioned in other sections before.
The method for extracting apicals first introduces a contact site criterion. Because of their great lengths it is very probable for an apical, if it exists, that it then touches at least two planes of the dataset boundary as the datasets are usually a lot smaller than the pyramidal cells (in this case most probably the planes across the x-axis as that is the direction that leads towards the pia mater). This step already reduces the number of possibilites drastically. After the first criterion, a filtering based on calculated path length is done i.e., processes that are either too long or too short (based on heuristics) are removed. The next criterion is about directionality. PCA is applied on the point structure of the agglomerates to get the variance explained by the first principal component and all processes that are below a certain threshold are again filtered out as the first principal component describes the 'straightness' of a given process. After that, the mean axis is calculated and based on that all agglomerates filtered out that deviate too much from that mean axis (which is calculated with the scalar product). The results of this filtering procedure are shown in Figure A.1. As can be seen, a reasonable amount of apical dendrites were filtered out of the whole dataset.

A.2 Extraction of AIS Candidates

Axons, rather collections of these forming bundles, have been the focus of the matching throughout this project. Axons, as introduced in Section 2.1.3, play a key role in the neuronal networks of the brain. The axon initial segment (AIS) as part of this structure is therefore of high interest. AIS is considered as critical for the initiation and propagation of action potentials [28]. These action potentials are fundamental to signaling in vertebrate nervous systems. The AIS is considered to be located between the axon hillock (the part that connects the axon to the soma) and the axon itself (or the myelin sheath in myelinated axons).
AIS, similar to ADs, has properties that distinguish this structure from other, which are also described by filters applied that are implemented within a method to extract AIS candidates. Again a dendrite agglomeration is assumed and additionally the knowledge about attached spine heads onto the agglomerates as this information will be used in a later step of the procedure. Furthermore, the locations and agglomeration of somata need to be available. The first step is to create a new agglomeration state where the somata have been removed in order to have all components originally reaching out of the somata because AIS would otherwise not be identifiable within the scope of this method. Then all 'straight' processes are being extracted by applying the directionality criterion, just as in the section above for ADs. After that, all processes that touch the border in pia-direction at least once are filtered out. These steps give a set of processes that contain myelinated axons, apicals and AIS. To then filter for potential AIS candidates, spine head densities are measured by counting the spine heads attached to a component and then dividing this value by the respective path length. Myelinated axons usually have a density of 0, while apicals have a relatively low density but not as low as AIS which is usually close to 0 (while not exactly 0 even though it should be because spines are actually a dendritic structure). Figure A.2 shows an example of an AIS identified using the described method.

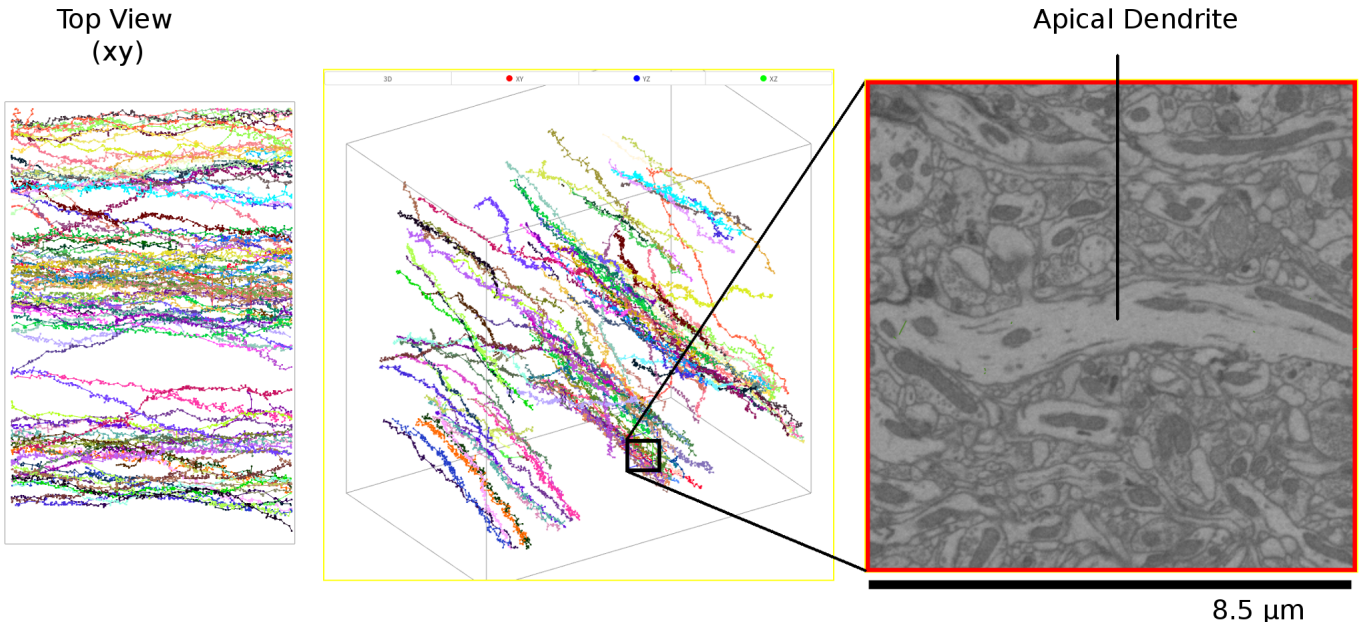

Figure A.1.: Apical Dendrites. The middle viewport (from *WebKnossos*) shows the $3D$ view with all apicals candidates extracted by the filtering method. The left shows the top view which illustrates the travel direction and straightness of these processes. The right shows an example of an AD extracted using the discussed filters. They are usually broader processes (similar to myelinated axons) that travel through the whole dataset.

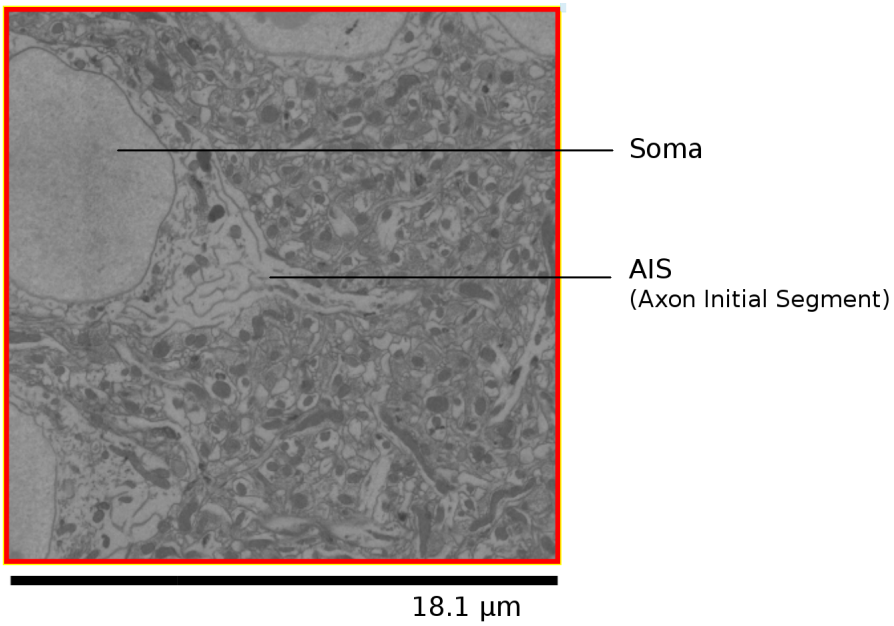

Figure A.2.: Axon Initial Segment. An example of an AIS identified using the discussed method. The density that led to the discovery of the given example is significantly lower than that of all other components (like dendrites or apicals). However not exactly 0 as for myelinated axons that are contained within the state of the agglomeration in use (even though spines are considered a dendritic structure).