DATA DRIVEN MAPPING OF THE DROSOPHILA LARVAL CENTRAL NERVOUS SYSTEM

DAVID GEORGE WOOD

Doctor of Philosophy
School of Informatics
University of Edinburgh

2018

David George Wood:
Data driven mapping of the Drosophila larval central nervous system
Doctor of Philosophy, 2018

SUPERVISORS:
J Douglas Armstrong
Barbara Webb

LAY SUMMARY

Fruit fly larva are studied extensively due to the extensive knowledge of their genetics, low costs and range of abilities including learning. Most of the 10 000 nerve cells in the central nervous system (CNS) of the larva can be uniquely identified, with each individual possessing one of each. The vast quantities of imaging data produced by many different laboratories can be difficult to navigate and requires specialised tools to work with. In particular, researchers require a common framework for communicating about the larval brain anatomy and also need to be able to find cells that are the same in 3D images of cells from many different individuals.

My work solves these problems by first creating a 3D reference atlas of the larval CNS by incorporating from the published literature brain compartments and navigation landmarks of nerve cell bundles. I then added thousands of 3D images of nerve cells to this atlas and made it publicly available online through a web browser. In order to ensure the points in these images were aligned appropriately to landmarks in the larval CNS atlas (as if they were in the same individual brain), I employed an automated procedure to warp these individual images. I also developed an algorithm that evaluated the success of this procedure.

To enable biologists to find similar cells, I successfully implemented an established technique to enable fast searching of the database. I also developed a novel algorithm using recent advances in computer vision that sped up the search by around 30 times. This new technique also allowed me to cluster the cells by similarity to help identify the different cell types in the larval CNS. Using the database I had created of cell type examples, I devised an algorithm to create average representations of cells of the same type to counter for individual variations. Finally, I found evidence for most of the brain compartments I had annotated by combining the structural information from all the cells in the database.

My work helps biologists to find a particular type of cell in fruit fly larva from images of individuals with different, known genetic mutations. These mutations can then be combined in order to genetically probe the functions of the cell under study.

ABSTRACT

The Central Nervous System (CNS) of the larval *Drosophila* model organism is extensively studied partly due to its small size and short generation times but also due to its ability to learn and the availability of genetic tools to investigate individual cell function. Unfortunately, it is very difficult to pool data from different studies: There is a lack of a standardised reference atlas and inference among separate 3D image stacks from different individual larvae is slow and error-prone. If, however, identical cells from images of different genetic lines can be found, this cell type can be isolated and probed for function via the Split-GAL4 method. The principal aims of my work were to find, implement and test methods that can be used to automate this process and analyse combined cell imaging data for information about the gross neuroanatomy of the larva.

I annotated a template larval Central Nervous System with neuropile domains and lineage tracts from the literature and compiled the most complete textual domain descriptions to date for the FlyBase database. To develop a registration pipeline for the whole-CNS channel of over 22 000 image stacks with a signal channel sparsely populated with neurons, I evaluated non-rigid registration parameters by measuring overlap of registered identical neurons. B-Spline Free-Form Deformations with a Correlation Ratio similarity metric were performed and candidate cell volumes extracted using adaptive thresholding. I evaluated registration accuracy with a novel local-intensity difference algorithm implemented with dynamic programming, yielding over 6 500 satisfactory individual whole-cell images.

I applied Machine Learning to identify neuron somas in semi-automatic cell annotation. To find similar neurons, I implemented and evaluated the established nBLAST method and developed a new approach: This condenses the representation of neurons with computer vision Artificial Intelligence (Convolutional Neural Networks within a triplet network architecture). These methods successfully allow biologists to rank cells by similarity, with the novel method demonstrating similar accuracy but executing 30 times faster. I validated this new method further by hierarchical clustering of cell examples to attempt to find cell type clusters. To create an average representation of a cell type from many examples, I developed a novel algorithm.

Finally, I have shown that voxel clustering on cell expression patterns supports the existence of most larval neuropil domains, with the notable exception of the Clamp. The registered cell examples have been made available as part of the freely accessible and actively used larval Virtual Fly Brain atlas.

ACKNOWLEDGEMENTS

I wish to thank my parents and sister, fellow PhD student Caroline, for your selfless support during my time in Edinburgh and throughout my education. Thank you to Dr. Emily Turner for your love and support, especially during the less happy moments of the past 3 years, and inspiring me to finish this project.

Thank you to my fellow Doctoral Training Centre colleagues and other Informatics post-graduates for your friendship, especially Dr. Tom Stone, Dr. Scott Lowe, Dr. Maciej Pajak, Dr. Sander Keemink, Dr. Matthew Graham, Dr. Katharina Heil, Dr. Agamemnon Krasoulis, Dr. Tom Mayo, Dr. Valentina Ferlito and Dr. Aleks Stolicyn with whom frequent discussions and your assistance often yielded useful insights or tips that hastened completion of my work. Thank you to my principal supervisor Prof. Douglas Armstrong for devising with me ideas for the project. Also thank you for passing on your links to the world-leading researchers in the field of developmental Drosophila neuroscience, providing feedback on written work and supporting my funding applications whilst generally guiding me through University bureaucracy. Thank you to my co-supervisor Prof. Barbara Webb for inviting me to be part of your lab group and the Minimal project, assistance in growing my academic network and providing feedback on building neural models that sadly did not make it into this thesis. A tremendous thank you to post-doc Dr. Robbie Court for your professional technical advice, multiple instances of out-of-hours computing support and all-round relaxed manner. Thanks to you and your co-researchers Dr. David Osumi-Sutherland and Dr. Marta Costa of the Virtual Fly Brain Project and FlyBase I was able to integrate my work with those systems.

Thanks to Prof. Jim Truman for providing me with years worth of your data, discussing with me the design of the project and hosting me at Janelia Farm Research Campus. An immense thank you to Prof. Volker Hartenstein for your efforts to not only annotate the standard Drosophila Larval Brain with me but teach me and discuss the anatomy and carefully review my written definitions. Thank you for also visiting and hosting me in this process and your inspirational curiosity and interesting discussions. Thank you also to your former post-doc Dr. Albert Cardona, you and your current post-doc Dr. Michael Wingding's assistance in integrating your EM data with my work is much appreciated. Also thanks to Professor Andreas Thum of Leipzig University and your assistance in defining the standard larval brain. I am indebted to many others who provided assistance throughout these years in the direction of my work and also to the support staff of the University without whom none of this would have been possible. Thank you also to Dr. Ian Simpson and Dr. Arnim Jenett for their comprehensive examination of this thesis.

Finally, I wish to thank all those others with whom I enjoyed my time in Edinburgh and Scotland, specifically I will fondly remember walks in Skye with Dr. Ilona Suojanen, producing many a business idea with Dragoş Stanciu, playing board games with Sam Laing, discussing politics over Korean food with Sophia Schneidewinde and exploring this brilliant city with Han Hao.

DECLARATION

I declare that this thesis was composed by myself, that the work contained herein is my own except where explicitly stated otherwise in the text, and that this work has not been submitted for any other degree or professional qualification except as specified.

Edinburgh, 2018

David George Wood,
17th May 2019

CONTENTS

Abbreviations xii

1 INTRODUCTION 1

- 1.1 Motivation . . . 1
- 1.2 The Larval Drosophila . . . 2
- 1.3 Drosophila larva Neuroanatomy . . . 3
- 1.4 Larval Drosophila as a Model Organism . . . 7
- 1.5 Genetic Labelling Tools . . . 9
- 1.6 The Virtual Fly Brain Project . . . 10
- 1.7 Motivational Problems, Goals and Thesis Outline . . . 11
  - 1.7.1 Thesis Outline . . . 14

2 DATASET AND NEUROANATOMY STANDARDISATION 15

- 2.1 Jim Truman's Dataset . . . 15
- 2.2 A Drosophila Larva Standard Brain . . . 17
  - 2.2.1 Central Brain . . . 19
  - 2.2.2 Ventral Nerve Cord . . . 23
  - 2.2.3 Lineages and Fascicles . . . 23
- 2.3 Chapter Summary . . . 24
- 2.4 Chapter Discussion . . . 24

3 REGISTRATION 26

- 3.1 Aims . . . 26
- 3.2 Introduction To Registration . . . 28
- 3.3 Fully Automated Registration . . . 29
  - 3.3.1 Computational Morphology Toolkit . . . 30
- 3.4 Template Choice And Registration Parameters . . . 34
  - 3.4.1 Initial Evaluation Tests . . . 35
  - 3.4.2 Detailed Evaluation Score . . . 37
  - 3.4.3 Template Choosing . . . 40
  - 3.4.4 Contrast Adjustment . . . 46
  - 3.4.5 Smoothing . . . 46
  - 3.4.6 CMTK Similarity Metric . . . 48
- 3.5 VNC Dorso-Ventral Orientation . . . 50
- 3.6 Neural Volume Segmentation . . . 51
- 3.7 Registration Accuracy Evaluation . . . 55
  - 3.7.1 A New Accuracy Score . . . 58

3.8 Mirroring . . . 69
3.9 Skeletonisation . . . 69
3.10 Computational Facilities And Technical Considerations . . . 72
3.11 Chapter Summary . . . 76
3.12 Chapter Discussion . . . 76
4 SINGLE NEURON ANALYSIS AND COMPARISONS 80
4.1 Introduction to Stereotyping And Comparing Neurons . . . 80
4.2 Cell Body Detection . . . 81
4.2.1 Data Preparation . . . 81
4.2.2 Gaussian Convolutions . . . 82
4.2.3 Linear Model . . . 84
4.2.4 Convolutional Neural Network . . . 85
4.2.5 Validation . . . 89
4.3 Pruning . . . 90
4.4 Tubeness . . . 93
4.5 Matching Neurons . . . 95
4.5.1 Fixed Point Boolean Comparison Matching . . . 96
4.5.2 Matching Neurons with BLAST Scores . . . 98
4.5.3 Matching Neurons with Machine Learning . . . 107
4.6 Lineage Classification . . . 114
4.7 Matching To The EM Dataset . . . 114
4.8 Averaging Neurons . . . 118
4.9 Chapter Summary . . . 121
4.10 Chapter Discussion . . . 121
5 BRAIN MAPPING AND CLUSTERING 127
5.1 Manual Selection Of Neural Volumes . . . 127
5.2 Clustering of Cell Types . . . 129
5.2.1 Hierarchical Clustering Technique . . . 129
5.2.2 Clustering Evaluation . . . 130
5.3 Arborisation Region Representation . . . 135
5.4 Voxel-Based Neuropil Domain Clustering . . . 138
5.4.1 Introduction to Voxel-Based Clustering . . . 138
5.4.2 Principal Component Analysis . . . 139
5.4.3 t-distributed Stochastic Neighbour Embedding . . . 141
5.4.4 k-means Clustering . . . 144
5.4.5 Gaussian Mixture Model . . . 144
5.4.6 Choosing the Best Model . . . 145
5.4.7 Evaluation of Automated Domains . . . 147

5.5 Integration With The Virtual Fly Brain Website 150
5.6 Chapter Summary 152
5.7 Chapter Discussion 153
6 GENERAL DISCUSSION 156
6.0.1 Deficiencies and Potential Improvements 156
6.0.2 Scope and Limitations 158
6.0.3 Theoretical Implications 159
6.0.4 Future Directions 161
A APPENDIX 163
A.1 Larval Drosophila Neuropil Domain Definitions 163
A.1.1 Mushroom Bodies 163
A.1.2 Inferior Protocerebral (Centro Central Brain) Domains 169
A.1.3 Superior Protocerebral (Dorsal Central Brain) Domains 171
A.1.4 Ventral Central Brain Domains 172
A.1.5 Central Brain Protocerebral Primordia 174
A.1.6 Ventral Nerve Cord (VNC) and Tritocerebrum 175
A.2 B-Splines 180
A.3 Inital Registration Scoring Guide 186
A.4 CMTK Parameters 187
A.5 Cell Type Training And Test Sets Summary 189
A.6 Neural Matching Convolutional Neural Network Parameters 190
A.7 Average Neuron Algorithm 192
List of Figures 195
BIBLIOGRAPHY 198

ABBREVIATIONS

1D	one dimensional
2D	two dimensional
3D	three dimensional
4D	four dimensional
ANTs	Advanced Normalization Tools
API	Application Programming Interface
AT	Andreas Thum
BLAST	Basic Local Alignment Search Tool
BO	Bolwig Organ
BP104	BP 104 anti-Neuroglian
CB	Central Brain
CMTK	Computational Morphometry ToolKit
CNN	Convolutional Neural Network
CNS	Central Nervous System
CPG	Control-Point Grid
CPU	Central Processing Unit
CR	Correlation Ratio
DNA	DeoxyriboNucleic Acid
DO	Dorsal Organ
E-M	Expectation-Maximisation
EM	Electron Microscopy
FDR	False Discovery Rate
FFD	Free-Form Deformation

FIJI	FIJI Is Just ImageJ
FRT	Flippase Recognition Target
GFP	Green Fluorescent Protein
GJ	Greg Jefferis
GMC	Ganglion Mother Cell
GMM	Gaussian Mixture Model
GMR	Gerry M Rubin
GPU	Graphics Processing Unit
JT	Jim Truman
KL	Kullback Leibler
LM	Fluorescence Light Microscopy
M/L	Machine Learning
MCFO	MultiColor FlpOut
MLS	Moving Least Squares
MNI	Montreal Neurological Institute
MI	Mutual Information
MW	Michael Winding
nBLAST	neuron BLAST
NMI	Normalized Mutual Information
OWL	Web Ontology Language
PAT	Primary Axon Tract
pBLAST	positional BLAST
PCA	Principal Components Analysis
RBF	Radial Basis Functions
RC	Rob Court
RNA	RiboNucleic Acid

REST	REpresentational State Transfer
ROI	Region Of Interest
SAT	Secondary Axon Tract
smGFP	spaghetti monster Green Fluorescent Protein
SOG	SubOesophageal Ganglion
SVD	Singular Value Decomposition
SVM	Support Vector Machine
SyN	Symmetric Normalization
TB	Terabyte
TO	Terminal Organ
tSNE	t-distributed Stochastic Neighbour Embedding
UAS	Upstream Activation Sequence
VFB	Virtual Fly Brain
VH	Volker Hartenstein
VNC	Ventral Nerve Cord
VO	Ventral Organ
XML	eXtensible Markup Language
	Synaptic Neuropil Domains
A	Abdominal
AL	Antennal Lobe
prAOTU	Anterior Optic TUbercle primordium
c	Central
cl	centro-lateral
CLA	Clamp
CLAl	lateral Clamp
CLAm	medial Clamp

cm	centro-medial
CRE	Crepine
CX	Calyx
dl	dorso-lateral
dm	dorso-medial
prFB	Fan-shaped Body primordium
IP	Inferior Protocerebrum (Clamp)
IPa	anterior Inferior Protocerebrum (Crepine)
IPp	posterior Inferior Protocerebrum
LAL	Lateral Accessory Lobe
LAML	Lateral Appendix of the Medial Lobe
LB	Labium
LON	Larval Optic Neuropil
LT	Lower Toe
MAML	Medial Appendix of the Medial Lobe
MB	Mushroom Body
MD	Mandibula
ML	Medial Lobe
MX	Maxilla
prNO	NOduli primordium
prPB	Protocerebral Bridge
PED	Peduncle
PLP	Posterior Lateral Protocerebrum
SIP	Superior Intermediate Protocerebrum
SLP	Superior Lateral Protocerebrum
SMPa	anterior Superior Medial Protocerebrum

SMPp	posterior Superior Medial Protocerebrum
SPU	Spur
T	Thoracic
TR	Tritocerebrum
TRc	Tritocerebrum central
TRcm	Tritocerebrum centromedial
TRd	Tritocerebrum dorsal
TRv	Tritocerebrum ventral
VL	Vertical Lobe
vl	ventro-lateral
vm	ventro-medial
VLP	Ventro Lateral Protocerebrum
VMCa	anterior Ventro Medial Cerebrum
VMCp	posterior Ventro Medial Cerebrum

1

INTRODUCTION

There is an information explosion, much like the population explosion, how on earth are you going to scan all that information? Of course you can get computers to help you...
- Alan Watts

1.1 MOTIVATION

In order to understand what enables our intelligent life, we must understand nervous systems. To do this we build mathematical models of the cells from which these systems are made that demonstrate how we believe these cells function and interact to produce actual observed behaviour. We must continually advance and iterate these models as we discover more biological evidence. The models can then be used to identify methods to repair and enhance our nervous systems and, ultimately, intelligent life.

The biological evidence we need comes partly from identifying and charting the morphologies of the neurons and supporting cells present in the system and identifying other cells with which they synapse, a field known as *connectomics* Silvestri et al. (2013). Building a complete map of cells, or "connectome", for a species or individual is only part of the solution, however. This is because the precise activity patterns of the cells and of many of the connections between them will depend on other factors, both genetic and external, such as *neuromodulators* Zeng and Sanes (2017) Bargmann (2012). Hence analysis of how a cell's varied activity influences the behaviour of those around it, and the organism overall - both simulated and in actuality - provide further evaluation measures for our models Zhang et al. (2007) Webb (2006).

Motivated by the idea of understanding how large networks of cells work together to produce functional "circuits", I have applied myself and my skills to enable better understanding of a heavily studied nervous system: the larval *Drosophila*. This animal provides a simplified yet highly capable nervous system with which to tackle these questions and crucially, through its genetic tools, allows the function of cells in hypothetical circuit models to be probed.

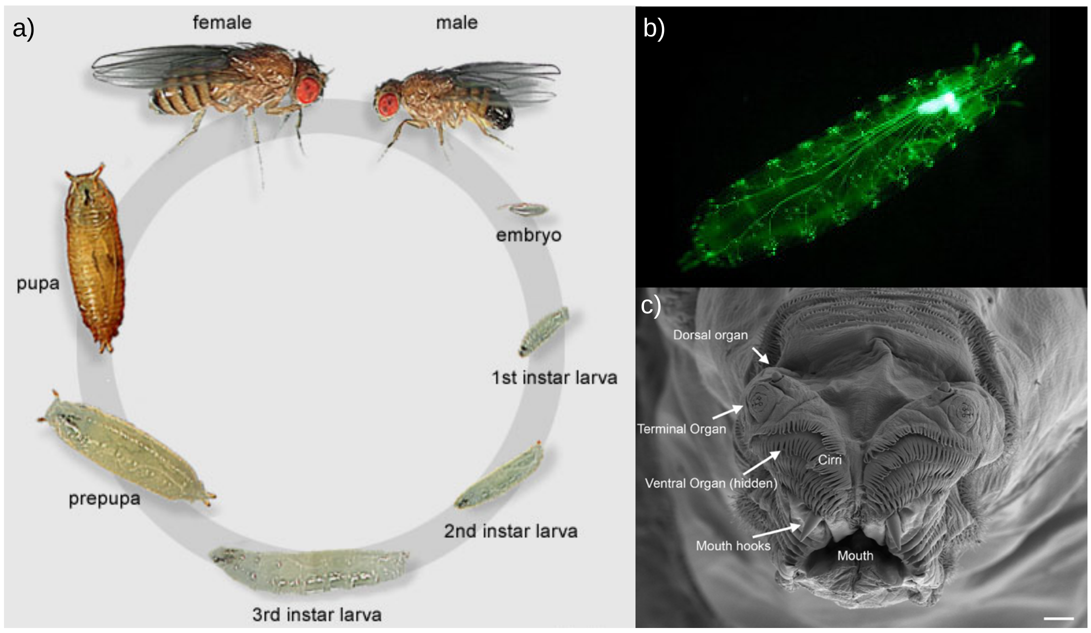


Figure 1.1: *Drosophila* Melanogater Life Cycle and Anatomy
a) Illustration of the life cycle of a *Drosophila* larva, from University of Muenster (2002). b) Selection of nervous system labelled with Green Fluorescent Protein with the Central Nervous System clearly visible, from Iyengar (2008). c) Scanning electron micrograph of the head of the larva, from Apostolopoulou et al. (2015). Scale bar 20μm.

1.2 THE LARVAL DROSOPHILA

Drosophila melanogaster, a species of the *Drosophila* genus of small fruit flies, is utilised extensively in scientific research. Herein I will equate the name of the genus to the species as is common in the literature. Under standard conditions (25°C) about 22 hours after egg laying a *Drosophila* embryo hatches into a freely crawling larva from an egg about 0.5mm long and 0.2mm wide Markow et al. (2009) (see figure 1.1a,b). The larva's principal activity is feeding in order to grow its mass by approximately 200 times Church and Robertson (1966) (mainly by increasing cell size rather than number). During this time it passes through three distinct stages, or *instars*; these instars are demarcated by moulting, each one after approximately one day Hales et al. (2015). Once reaching about 3-4mm in length, after approximately 2 days, the third instar larva pupate for a further 4-5 days Hales et al. (2015) before emerging as adult flies that can live for up to 7-8 weeks Linford et al. (2013).

During embroygenesis, patterns of gene expression differentiate regions of the embryo into segments, or *blastoderm anlage*, that will be responsible for specific regions of anatomy Campos-Ortega and Hartenstein (1997). In particular, the central segments

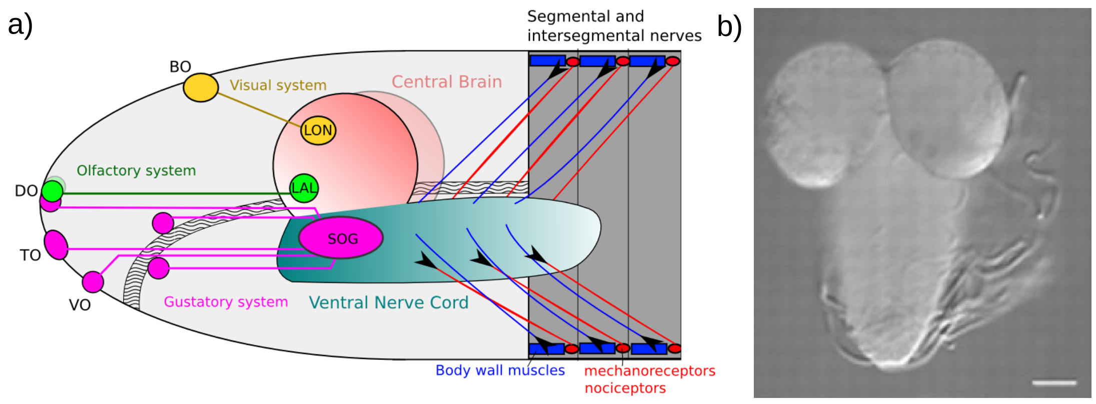


Figure 1.2: Overview of Larval Central Nervous System
a) Sagittal view of the *Drosophila* larva CNS and relationship to peripheral nervous system. The external organs (DO,VO,TO and BO) are paired on each side. Own work based on Stocker (2008). b) Dissected CNS, scale bar 100µm, from Wang et al. (2009).

correspond to repeated units in the central, elongated part of the body of the larva. The body is translucent white in colour, with a head easily identifiable by two dark mouth hooks (see figure 1.1c) including teeth used for mastication and sometimes attack Vijendravarma et al. (2013). In addition, the head contains olfactory, gustatory and very primitive light receptors. The bulk of the larva - its gnathal, thoracic and abdominal segments - are dedicated to movement with the inner sides of the body walls of these segments covered with a system of muscles Bate (1990). This enables the larva to crawl via the method of *peristalsis*, that is by contracting these abdominal wall muscles in sequence such that waves flow along the length of their body - either forwards or backwards Berrigan and Pepin (1995). In addition to crawling they may also burrow into soft substrates Narasimha et al. (2015) or roll to escape pain inflicted by wasps (or scientific researchers) Hwang et al. (2007). A gut passing through the centre of its body and associated digestive organs enable the larva to feed immediately after hatching Murakami et al. (1994), crawling at between 40 and 120 μms^{-1} Heckscher et al. (2012). Larvae will head towards sources of food based on smell Gerber and Stocker (2007), frequently stopping to cast their heads about to sense their environment Berrigan and Pepin (1995). Finally, the posterior three abdominal anlage form a small tail end region with anus Campos-Ortega and Hartenstein (1997).

1.3 DROSOPHILA LARVA NEUROANATOMY

The larval *Drosophila* nervous system is heavily stereotyped such that there is very little inter-individual variability on a structural and even single cell morphological level. This is to the extent that most neurons can be uniquely identified across individuals with light microscopy imaging. In this thesis, groups of cells from different individuals that are indistinguishable from being the equivalent cell in each other individual's nervous system are referred to as having the same *cell type*. This definition is discussed in section 6.0.3

The Central Nervous System (CNS), symmetrical about the mid-line, consists anteriorly of two spherical lobes that contain the two *neuromeres* of the protocerebrum (developed from the anterior most blastoderm anlage, the *acron* and *labrum*) and the deutocerebrum (developed from the antennal segment) Campos-Ortega and Hartenstein (1997). They are connected across the mid-line by large commissural neural fibres. The ventral-posterior sides of these lobes are fused to a common oblong mass (the Ventral Nerve Cord (VNC), see figure 1.2) via the tritocerebrum (developed from the intercalary segment). The CNS is located in the head of the larva as it's body grows out behind it. In the third instar the CNS is typically approximately 500μm long and 300μm across at the lobes and contains approximately 10 000 neurons Li et al. (2014).

The CNS itself is encased in a blood-brain barrier of surface glia cells Hartenstein (2011), within which a *cortex* of varying thickness made up of neural cell bodies surrounds a dense synaptic neuropil core containing the dendritic branches of neurons Hartenstein et al. (2008). Neurons project axons from the cortex into the neuropil, where they may form bundles - known as *fascicles* - with other axonal fibres which may include fibres projecting from opposite directions. The axons branch and form dendrites with synaptic connections to other cells over a stereotyped, genetically determined volume. A network of trachea support the CNS neuropil functions Pereanu et al. (2007) along with specific cortex and neuropil glia Hartenstein (2011). The neuropil glia also help divide the neuropil synaptic density up into distinct compartments or *domains*, which relate to functional compartmentalisation of the CNS Younossi-Hartenstein et al. (2003).

The proto-, deuto- and tritocerebrum together make up the Central Brain (CB) (Supraesophageal Ganglion), containing many domains that are related to sensory processing: In addition to possessing light sensing cells over their bodies Xiang et al. (2010), a pair of *Bolwig Organs*, each of 12 specialised photoreceptive neurons, project to the Larval Optic Neuropil (LON), a small synaptic neuropil domain in the protocerebrum Sprecher et al. (2011). *Drosophila* larvae sense odours through 21 olfactory receptor neurons located in the head's *Dorsal Organ*, which project to the deutocerebrum (Antennal Lobe (AL) synaptic neuropil domain) Python and Stocker (2002). Gustatory receptor neurons are also located in the dorsal organ and two further ex-

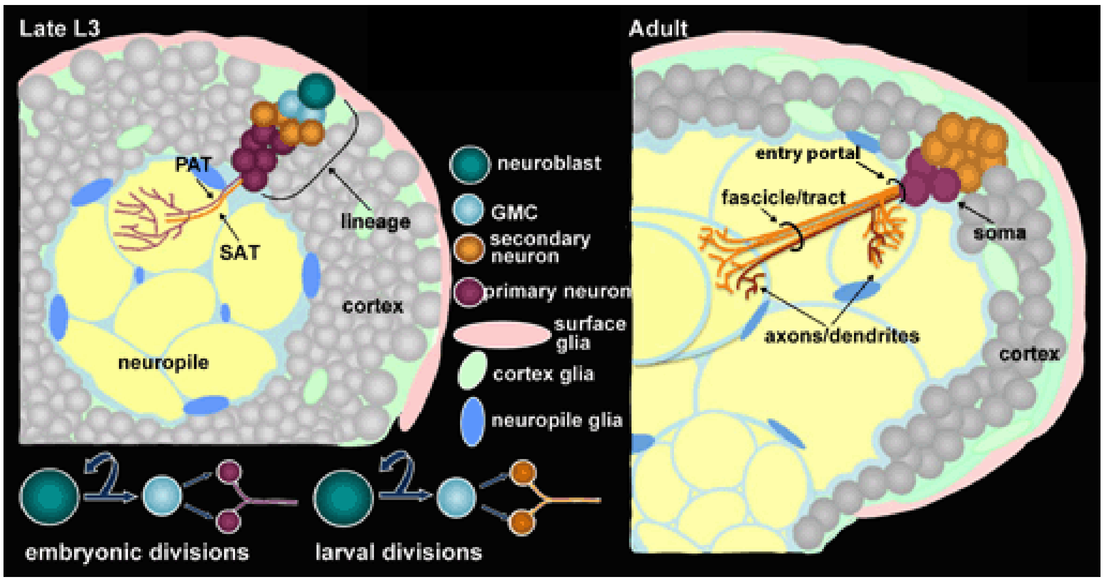


Figure 1.3: Overview of Neuropil of the CNS and Development
Cross sections through the CNS. From Hartenstein (2008).

ternal organs, the *Ventral Organ* and *Terminal Organ*, nearer to the mouth, as well as in three pharyngeal sense organs inside the gut. In total there are 68 gustatory receptor neurons which all project to the SubOesophageal Ganglion (SOG) synaptic neuropil domain of the CNS Kwon et al. (2011). Finally, larvae possess various somatosensory neurons throughout their bodies including nociceptors Hwang et al. (2007) and stretch mechanoreceptors Schrader and Merritt (2007), most of which project into the VNC.

The VNC itself has a metameric structure which develops initially in line with the segmental patterns found in the blastoderm of the embryo (when the CNS extends most of the length of the larval volume), with cell types being repeated across the segments. It consists of the SOG (composed of 3 fused ganglia, from which neurons controlling mouth and digestive movements emanate Schoofs et al. (2010)), 3 thoracic ganglia and 9 abdominal. These abdominal ganglia are chiefly responsible for the peristaltic motor output Kohsaka et al. (2012), with motor neuron dendrites receiving input mostly from the dorsal side of the VNC, with arbours to body wall muscles in corresponding segments for controlling the compression thereof. The VNC also receives somatosensory input, mostly to its ventral side. Finally, perhaps the least studied part of the CNS is that of the final abdominal segment, from which cells controlling defecation emanate Zhang et al. (2014). For more detailed neuroanatomy regarding the CNS, see section 2.2.

All neurons in the CNS develop from stem cells known as a *neuroblasts*. During embryogenesis, a neuroblast divides in one of two ways and its progeny are neurons that are related to each other genetically and said to be from the same *lineage* (see figure 1.3). Type I lineage neurons form from neuroblasts that divide 5-10 times, each time producing two cells, one the retained neuroblast and the second a *Ganglion Mother Cell (GMC)* Hartenstein et al. (1987). Each GMC undergoes a further subdivision into two neurons. Type II lineages form from more prolific neuroblasts that divide into multiple other cells that are all then capable of producing neurons in a way similar to type I neuroblasts Bello et al. (2008). Neuroblasts progress outwards in the cortex as they divide, leaving behind a centripetal stack of their progeny. As a result of this process, cells from the same lineage have cell bodies grouped together at reasonably well defined locations in the cortex and the axons project in bundles towards the neuropil and enter at well defined *entry points*. The initial path inside the neuropil, where all axonal fibres from a particular lineage remain grouped together, is known as a *lineage tract*. Some lineages, however, can be further split into two *hemilineages*, so that each GMC produces a cell from each. These each have their own tracts which will share a common lineage tract but then branch separately Lovick et al. (2016). A lineage tract formed by neurons that develop during embryogenesis is known as a Primary Axon Tract (PAT).

There are approximately 100 neuroblasts producing cells for each CB hemisphere, giving rise to about 1 500 neurons in each lobe of the first instar Hartenstein et al. (2015) Cardona et al. (2009). The VNC is produced from 30 pairs of neuroblasts and 1 unpaired for each segment giving rise to approximately 540 cells per segment (out of the approximately 700 produced including non-interneurons) Bossing et al. (1996) Truman (2004) Rickert et al. (2011). Following a period of quiescence during the early-mid larval period, during the late larval period neuroblasts divide further and secondary neurons are produced that send axons into the neuropil along the corresponding PAT but often extending further into the enlarging larval brain. Due to the increased size of the brain, some PATs that were grouped together are now visibly separate Hartenstein et al. (2015). A full tract produced by a secondary neuron is known as a Secondary Axon Tract (SAT). These secondary neurons, however, do not form synaptic branches until during metamorphosis. Secondary cells from the same lineage typically eventually form similar morophologies, innervating a select few domains, however primary neurons from the same lineage are more diverse in their structures Lovick et al. (2016).

1.4 LARVAL DROSOPHILA AS A MODEL ORGANISM

Drosophila is a well established scientific model organism, with its small size, lack of ethical considerations and short life cycles making the generation of large numbers quick and cost effective Hales et al. (2015). This has enabled the analysis of a large number of *Drosophila* genes by studying mutations, such as the *hedgehog* gene that is responsible for the polarity of segments in the larval body plan, mutations of which result in short and "spiked" cuticle Nusslein-Volhard and Wieschaus (1980). Further, Rubin and Spradling (1982) demonstrated a technique allowing one to insert new DeoxyriboNucleic Acid (DNA) sequences into the *Drosophila* genome, enabling the production of an even larger number of genetic mutants with a wide range of phenotypes. This in turn allowed the development of the revolutionary genetic tool, namely the GAL4-UAS system Brand and Perrimon (1993), that enables cell-type specific expression of desired *reporter* genes. Briefly, the GAL4 transcription activator (originally from yeast) is inserted into the *Drosophila* genome randomly, and in some cases will be inserted downstream of an *enhancer* sequence that will increase the likelihood that the GAL4 protein will be expressed. The expression of GAL4 is then specific to cells that express that enhancer. GAL4 binds to an Upstream Activation Sequence (UAS) in the DNA and this enhances the production of a gene inserted into the DNA following the UAS by recruiting further *transcription factors* that bind to the gene's promoter regions (at the beginning of the gene's DNA sequence) in order to recruit a further protein, RiboNucleic Acid (RNA) polymerase, that will finally produce the gene-encoded protein.

As a result there has been considerable effort to create collections of flies with a range of genetic *lines* which each express GAL4 in known, specific cells. When a fly from a GAL4 line of interest is crossed with a fly with a UAS promoting a reporter gene of interest, their progeny can express the protein encoded by that gene exclusively in the target cells. That protein may be, for example, Green Fluorescent Protein (GFP) that converts blue light to green light and thus allowing the morphology of the cell to be imaged via Fluorescence Light Microscopy (LM) without contamination from other cells Hales et al. (2015). In addition, the system can be used to probe the function of neurons by adding a UAS to produce photosensitive proteins that, when light is shone through the translucent larva, activate them Honjo et al. (2012) (a technique known as *optogenetics*). Alternatively, the *shibire* gene could be promoted that inactivates cells above a certain temperature by disrupting the recycling of neurotransmitters at the synapses van der Bliek and Meyerowrtz (1991).

In order to produce a full connectome of neurons in the larval CNS one requires high image resolution to identify synapses between cells. This pursuit has involved the production of a serial section Electron Microscopy (EM) of a single brain with slice pixels of size 4.4nm × 4.4nm at a slice thickness of 45nm Schneider-Mizell et al. (2016). This is being annotated manually by a large team of researchers who have developed software to assist in tracing the axonal and dendritic skeletons of all the neurons in the CNS and annotating the points of the synapses.

The strong genetics research backbone coupled with powerful computational tools has enabled a large amount of experiments into all systems of the larva, especially those dissecting neural circuits to determine the functional roles of specific cells. I will briefly introduce three exemplary studies: Firstly, larvae have been shown to learn to associate an odour with food and use this "memory" to express a preference for inhabiting spaces containing that odour Gerber and Stocker (2007). By utilising the GAL4-UAS system to selectively block neurons using *shibire* in the Mushroom Body (MB), these memories are also blocked Pauls et al. (2010), leading to the proposal that the odour and food reward signal are integrated into memories in the MB and therefore *stored* hereSchleyer et al. (2011), similar to the adult Davis (2011). In the motor system, a group of neurons was discovered - by using optogenetics to activate them - that influence the duration of motor output in a segment during production of the peristaltic wave of activity Kohsaka et al. (2014). Finally, the above tools have been applied to discovering a complete neural circuit that passes through several levels of cells to integrate mechanosensory and nociceptive inputs that enable the larva to escape attack by wasps Ohyama et al. (2015). Following an *in vivo* screen for neurons involved in this process performed by selective blocking via the GAL4-UAS system, the neural circuit containing the positive cells was reconstructed from the EM image stack and their functional connectivity validated using calcium signal imaging *in vitro* to detect post-synaptic activation.

By focusing on the larval brain rather than the adult, one gains tractability from simplicity, but still retains a system that is capable of learning and decision making, as well as the ease of visualising *in vivo* experiments through their lack of opacity. Finally, many homologs exist in *Drosophila* for human disease genes, and these are studied to screen for proteins (and eventually drugs) that affect these associated diseases Pandey and Nichols (2011). For example, Shulman and Feany (2003) utilised adult *Drosophila* to discover proteins that influence the production of toxic *tau* proteins linked to Alzheimers disease in humans.

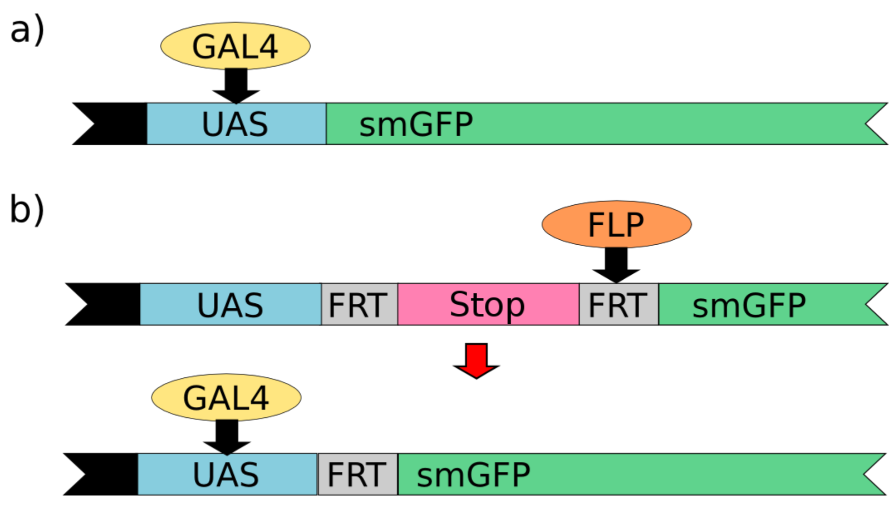


Figure 1.4: Explanation of Genetic Tools
a) GAL4-UAS System, b) Standard FLP-Out System. See text for description.

1.5 GENETIC LABELLING TOOLS

As described above, the GAL4-UAS system provides a method of selectively labelling cells, typically a group, with a fluorescent protein so that they can be imaged using LM. However, as most GAL4 lines contain many neurons where axons occupy the same fascicles and dendrites innervate the same neuropil domains, they appear to overlap at the resolution of these three dimensional (3D) images. In order to address this problem, several methods have been developed to produce image stacks that are much sparser so that individual cell morphologies can be determined in isolation. I will detail that of Nern et al. (2015), known as MultiColor FlpOut (MCFO), since this is the method that has been applied by Jim Truman (JT) to produce the dataset that I will work with and detail in section 2.1. It *stochastically* enables the expression of a subset of cells from a GAL4 line.

Key to this method is utilising the *Flp* (flippase) recombinase protein (originally from yeast) in a technique known as *Flp-out* Golic and Lindquist (1989): Between the UAS sequence and the gene's promoter region in the DNA sequence a *stop* sequence surrounded on either side by Flippase Recognition Target (FRT) sequences is added (see figure 1.4). The stop sequence blocks the recruitment of transcription factors, and so prevents expression of the gene that follows. This stop sequence can be removed, however, if the Flp protein is present and finds the FRT sequence to bind to and remove the DNA in between from the full sequence. The level of Flp present (and hence chance of cleavage) can be controlled by using a weak temperature-sensitive promoter region where Flp is itself encoded in the DNA.

In an extension of the Flp-out method, Nern et al. (2015) utilised the spaghetti monster Green Fluorescent Protein (smGFP) Viswanathan et al. (2015) which, although not actually fluorescent itself, contains within its large protein backbone a particular *epitope tag* (amino acid sequences onto which antibodies will bind). Three different sequences for three different smGFP (each with a different epitope tag) were inserted into the reporter *Drosophila* DNA, each proceeded by a UAS and Flp-out blocking stop cassette. This enabled the cells to produce one or more of three different smGFP depending on the stochastic behaviour of the flippase. One or more of three different antibodies could thus concentrate in a cell upon dissection - depending on which smGFP were present - by attaching themselves to the corresponding epitope tags. Three different fluorescent dye secondary antibodies (red, blue and green) were then applied, one attaching to each primary. This produced a total of up to 7 different coloured cells (with combinations of secondary antibodies giving yellow, magenta, cyan and white).

1.6 THE VIRTUAL FLY BRAIN PROJECT

Devised as a framework to enable worldwide data sharing of adult *Drosophila* brain confocal microscopy gene *expression pattern* image stacks (where groups of - or single - cells are labelled with a protein if they express a certain gene), the *Virtual Fly Brain* provides a popular and vital service for the *Drosophila* research community Milyaev et al. (2012). It is a web-based tool that also includes anatomical annotations to enable it to also behave as an interactive atlas. It utilises a database of standardised annotated domains and expression patterns that are submitted by global collaborators and *registered* to the common template to enable integration of these different data sets.

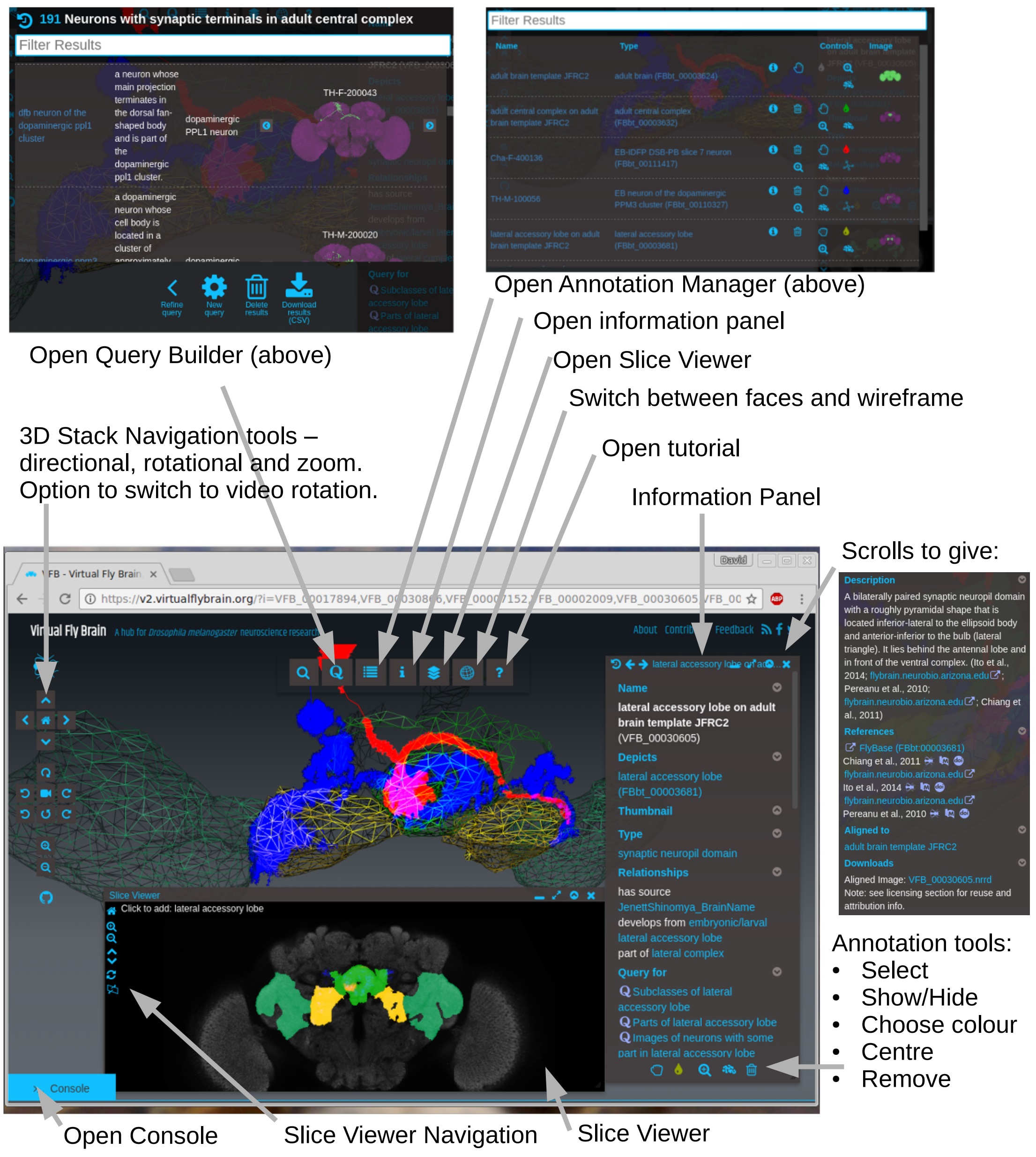
The database uses an ontology based on the Web Ontology Language (OWL) standard W3C OWL Working Group (2012) that stores relationships between the entities such as *part_of* (e.g. the Medial Lobe (ML) is part of the MB) or *has_synaptic_terminals_in* (for neurons to neuropil domains) Costa et al. (2013). Fast reasoning algorithms then enable queries to be performed on the database in real-time and enables website visitors to relate expression patterns to anatomical landmarks (e.g. finding neuropil domains that a neuron has synaptic terminals in) and infer relationships between the patterns (e.g. finding a neuron that might synapse with a query neuron as they have synaptic terminals that overlap). Implicit relationships are found by the reasoner using the transitive rules of the ontology (e.g. as the Lower Toe (LT) is specified as *part_of* the ML then the LT is also *part_of* the MB). The Virtual Fly Brain (VFB) database is closely linked to the database of the high-quality FlyBase Tweedie et al. (2008), which is curated and updated frequently and from which VFB obtains its descriptions, synonyms and references.

The visitor's interaction with VFB takes place around a 3D viewer with icon links to panels that provide the ability to search for anatomy that can be displayed as well as open the *Slice Viewer* and *Information Panel* (see figure 1.5). The visitor can first run a text-based search to find any synaptic neuropil domains or neurons of interest, including checking synonyms. Alternatively, a query builder leads the visitor through running a query of the ontology by, after having searched for and found an anatomical target, giving various query options such as finding subclasses or domains/neurons that physically overlap with this target. Items matching the query are returned and can then be added to the viewers.

The two dimensional (2D) Slice Viewer allows exploration of the anatomy as if it was a 3D microscopy stack, with coronal, axial and sagittal views available. The names of the neuropil domain at the present cursor position are displayed and they may be added as annotations by mouse clicking the position. The neuropil domain annotations take the form of meshes in 3D, which may be requested as wire frames or translucent faces, and in 2D they are displayed as semi-transparent coloured overlays. In 3D the expression patterns and cells are displayed as point clouds and the latter alternatively as traced, tubular skeletons if available. They are simply overlaid on the 2D image stack, mimicking a LM stack view. An Annotation Manager can be opened to control the visible annotations and overlays. Finally, an information panel for the currently selected item (from mouse clicking in the viewers) provides metadata. Besides the name and description, this panel has links to suggested queries, source references and a download of the item in a registered image stack.

1.7 MOTIVATIONAL PROBLEMS, GOALS AND THESIS OUTLINE

Laboratories studying *Drosophila* larvae worldwide are producing vast quantities of data that are relevant for the purpose of charting its neural circuits, in particular imaging data of gene expression patterns. Unfortunately, these often remain internal to the laboratory that produced them or available to others only as separate images, often only in 2D, e.g. Li et al. (2014). This also encourages multiple forms of anatomical terminology which can hinder communication between researchers. Rather than unwillingness to share, more commonly this data is not made available due to a lack of an appropriate, adapted system that possesses methods of linking between the



Name
lateral accessory lobe on adult brain template JFRC2 (VFB_00030605)
Depicts
lateral accessory lobe (FBbt_00003681)
Thumbnail
Type
synaptic neuropil domain
Relationships
has source JenettShinomya_BrainName
develops from embryonic/larval lateral accessory lobe
part of lateral complex
Query for
Subclasses of lateral accessory lobe
Parts of lateral accessory lobe
Images of neurons with some part in lateral accessory lobe

Description
A bilaterally paired synaptic neuropil domain with a roughly pyramidal shape that is located inferior-lateral to the ellipsoid body and anterior-inferior to the bulb (lateral triangle). It lies behind the antennal lobe and in front of the ventral complex. (Ito et al., 2014; flybrain.neurobio.arizona.edu; Pereanu et al., 2010; flybrain.neurobio.arizona.edu; Chiang et al., 2011)
References
FlyBase (FBbt:00003681)
Chiang et al., 2011
flybrain.neurobio.arizona.edu
Ito et al., 2014
flybrain.neurobio.arizona.edu
Pereanu et al., 2010
Aligned to
adult brain template JFRC2
Downloads
Aligned Image: VFB_00030605.nrrd
Note: see licensing section for reuse and attribution info.

Figure 1.5: The Virtual Fly Brain (Adult)
Adult brain displayed in Virtual Fly Brain version 2.0 Alpha release. The 3D Viewer is the centre of the browser window. The icons above the 3D Viewer link as annotated, and the left-most icon opens a search box for a text-based search. Neurons can be displayed as point clouds (shown here) or skeletons if available. The Query Builder panel shown above-left (overlaid over the main viewers when present) demonstrates the results of a query for neurons in the central complex, two of which are displayed in red and blue (from Chiang et al. (2011)) in the Viewers. The neuropil domains of the Lateral Accessory Lobe (LAL) (yellow), central complex (green) and anterior Ventro Medial Cerebrum (VMCa) (teal) are also displayed. The Annotation Manager is top right (also overlaid over the main viewers when present).

data sets.

Firstly, promoting a standardised anatomical framework - such as the BrainName Consortium has achieved for the adult *Drosophila* Ito et al. (2014) - will ensure co-ordination and understanding between researchers. Furthermore, integrating their various disparate imaging resources within this framework will accelerate performing inferences between them and thus speed up a range of further research. At its most simple, a system similar to VFB for the larva would allow a researcher to quickly look-up and view a gene expression pattern or a cell structure online rather than directly requesting the data or performing their own experiments. In addition, little data-driven analysis of the global structure of the larval CNS is possible without this integration and has not been performed before. This could provide further evidence for the synaptic neuropil domains of the CB and the number of different cell types present.

A specific problem relating to the larval *Drosophila* that illustrates the need for a system of data integration is that of finding a GAL4 genetic line that is reliably expressed in a single target cell. For example, this occurs when one desires to manipulate this cell to assess its functional purpose in a circuit, as was achieved by Ohyama et al. (2015). The GAL4 lines are generally not sparse enough for this; the Flp-out technique too stochastic. A method known as *Split-GAL4* achieves this by first splitting each GAL4 line into two further lines by breaking the GAL4 gene into two parts with a part in each new line, rendering it inactive in both Luan et al. (2006). It can then be reactivated in the progeny resulting from breeding (or *crossing*) two flies from each new line together. Sparse and single cell lines can be created when sublines originating from two different GAL4 lines are combined since expression only occurs at where the two lines intersect, i.e. both parts are present. Finding a GAL4 line with a target cell can be time consuming, but even more challenging is finding two lines that enable the Split-Gal4 to be performed. These large data sets warrant the tools of computation and automation to feasibly tackle the above problems.

When I began this project it was an open question as to whether a database could be produced for the larval *Drosophila*, a task perceived by many in the field as more difficult than in the adult due to the larva being a developing (and therefore changing) system. My first aim was to create this database and second to develop useful tools for researchers to find cells of interest. Finally, my goal was to analyse the data in the database to draw conclusions about the structure of the whole CNS.

1.7.1 *Thesis Outline*

I worked with one of the largest datasets of images containing larval neurons from a broad range of Gal4 lines, detailed in chapter 2. Also in that chapter, I explain how I produced a standardised larval *Drosophila* brain anatomy from a different data source. Following this I devote chapter 3 to detailing how I performed the registration of the 3D CNS image stacks to bring them into a common space from which I extracted the neural morphologies in various representations. Registration of the larval CNS had not been achieved before and so required a method of validation and judgement of accuracy, which is explained.

Chapter 4 reviews my methods for analysing the data, predominantly concerned with addressing the Split-Gal4 problem. I implement and evaluate several methods where a known cell is given as a query and I am able to return similar cells from different lines. In the final results chapter - chapter 5 - I organise the extracted cells by employing clustering techniques in an attempt to determine the number of types. I also evaluated the standard brain I had produced by applying techniques from computational neuroanatomy. Finally, also in chapter 5, I explain how I made the above images and system accessible to the community by extending VFB to the larval *Drosophila*.

2

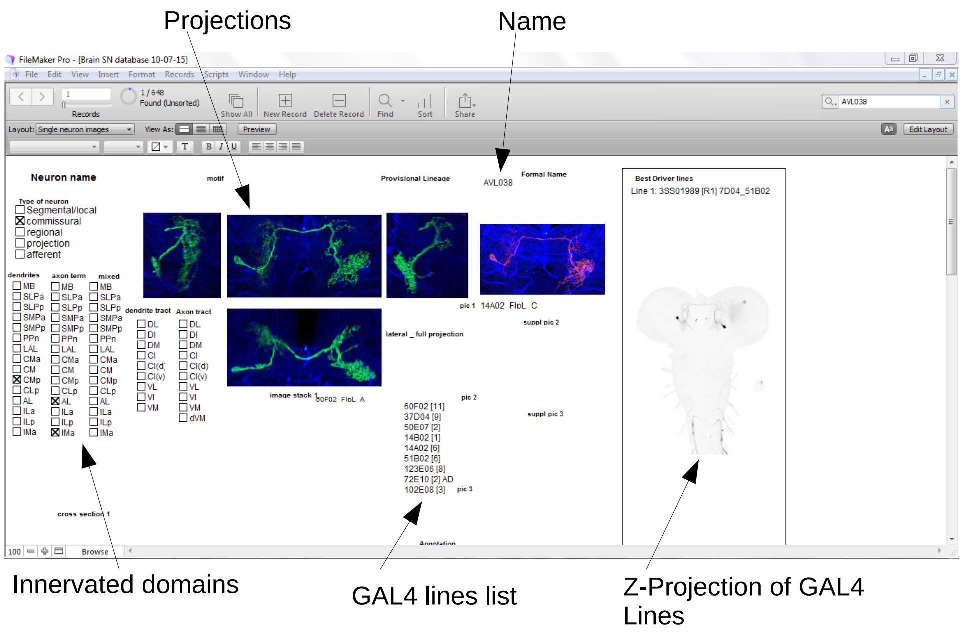
DATASET AND NEUROANATOMY STANDARDISATION

2.1 JIM TRUMAN'S DATASET

In order to build up a comprehensive atlas of neurons present in the larval *Drosophila* brain, I required images where each cell can be clearly separated from its surroundings and other cells but also interpreted with respect to them. I was fortunate enough to be able to work with what is accepted in the community as one of the largest and highest quality data sets in *Drosophila* larval neuroanatomy. This is a collection of image stacks each of which shows (*labels*) only a small number of neurons in an individual larval brain and, as a whole, covers almost all uniquely identified neurons in the CNS (JT, personal communication).

This imaging dataset was provided and produced by JT as part of the *FlyLight* project at Janelia Research Campus, Ashburn, VA, USA. He had employed the MCFO technique with two smGFPs and red and green reporter antibodies to visualise cells in a very comprehensive and sparse set of GAL4 genetic lines of flies maintained at the campus. These lines are known as the Gerry M Rubin (GMR) lines after the principal investigator who developed them. In addition, since larvae express the neuroglian protein in neural tracts reasonably consistently across the CNS, these tracts can be labelled by using the antibody *BP104* (also known as anti-Neuroglian). This allowed the tracts to be used as a reference, *reference* channel with which to relate the stochastically labelled red, green and yellow neurons to their surrounding neuroanatomy.

Aided by others and robotic automation, JT raised and dissected the CNS from tens of thousands of third instar larvae and imaged sets in a low resolution microscope. From these images he chose highly sparse expression patterns to image using a stronger 60x microscope. The distribution over the GMR lines was good with most lines having at least 10 image stacks and less than 10% more than 30 (mean 15.3, median 13). JT's aim was to identify all the *cell types* in the larval *Drosophila* brain where cells were classed as the same type when their gross morphology were indistinguishable from one another based upon his experience.



FileMaker Pro - [Brain SN database 10-07-15]

1 / 648 Found (Unsorted)

Layout: Single neuron images

Neuron name

Type of neuron
- ☐ Segmental/local
- ☒ commissural
- ☐ regional
- ☐ projection
- ☐ afferent

dendrites	axon term	mixed
☐ MB	☐ MB	☐ MB
☐ SLPa	☐ SLPa	☐ SLPa
☐ SLPp	☐ SLPp	☐ SLPp
☐ SMPa	☐ SMPa	☐ SMPa
☐ SMPp	☐ SMPp	☐ SMPp
☐ PPn	☐ PPn	☐ PPn
☐ LAL	☐ LAL	☐ LAL
☐ CMa	☐ CMa	☐ CMa
☐ CM	☐ CM	☐ CM
☒ CMp	☐ CMp	☐ CMp
☐ CLp	☐ CLp	☐ CLp
☐ AL	☒ AL	☐ AL
☐ ILa	☐ ILa	☐ ILa
☐ ILp	☐ ILp	☐ ILp
☐ IMa	☒ IMa	☐ IMa

dendrite tract	Axon tract
☐ DL	☐ DL
☐ DI	☐ DI
☐ DM	☐ DM
☐ CI	☐ CI
☐ CI(d)	☐ CI(d)
☐ CI(v)	☐ CI(v)
☐ VL	☐ VL
☐ VI	☐ VI
☐ VM	☐ VM
	☐ dVM

Provisional Lineage: AVL038

Formal Name

pic 1 14A02 FlpL C

image stack 1 60F02 FlpL A

lateral _ full projection

60F02 [11]
37D04 [9]
50E07 [2]
14B02 [1]
14A02 [6]
51B02 [6]
123E06 [8]
72E10 [2] AD
102E08 [3]

Best Driver lines
Line 1: 3SS01989 [R1] 7D04_51B02

Figure 2.1: Previous FileMaker Database
A screenshot of the private FileMaker Database of JT with a record for a single cell type shown. This was for a considerable time the world's premier database for information regarding the cell types of the *Drosophila* larva.

The image data that I received consisted of 3D confocal microscopy image stacks of either the CB or VNC. The individual slices of the images were made up of approximately 150µm × 150µm tiles that had been stitched together using the *stitchstack* routine, part of neuTube, itself part of Vaa3D Peng et al. (2014). The slice pixel sizes were 0.293µm × 0.293µm and the slices were taken every 0.5µm. The CB stack slices were a row of 2 of these tiles and the VNC a column of 3. Although, owing to their different heights, there were typically around twice as many slices for the CB compared with the VNC so the CB stacks were usually slightly larger in memory usage. The images were provided as 3D TIFF format images Aldus Developers Desk (1992) with 3 colour channels (the red and green *signal* channels and the blue BP 104 anti-Neuroglian (BP104) reference channel), with sizes of the order 250-350 MB. There were 22 404 stacks from 1 469 different GMR lines totalling approximately 8.8 Terabyte (TB). Some of the VNC and CB stacks were from the same CNS but there was no unique identifier matching the two, meaning these stacks could only be matched using the image data itself in stacks from the same GAL4 line.

In addition to the raw image data, I was provided with JT's computerised database. This was implemented using the proprietary *FileMaker* software (see figure 2.1) where each record was a cell type. Along with exemplary images of the cells, information about the regions of the CB the cell projected to were recorded if appropriate (based on JT's parcelisation that was similar to that of Volker Hartenstein (VH) - see section 2.2 - but not well defined). Of particular interest, however, was the record of GMR GAL4 lines from which this cell type has been "flipped out" - kept in a list for each cell type. I was able to export the data from *FileMaker* but unfortunately the stacks used to determine membership of a cell type to a particular GAL4 line were not recorded. In fact no link between stack filename and cell type was available in any form other than the unwritten experienced judgement of JT and possibly a few other researchers.

2.2 A DROSOPHILA LARVA STANDARD BRAIN

The neuoropil domains in the larval *Drosophila* brain have been characterised chiefly by the work of VH and colleagues over a number of years, the structures of which have remained broadly consistent throughout, although the names have changed as it has become clearer how they relate to the adult brain neuropil domains. Regarding the CB, Younossi-Hartenstein et al. (2003) and Younossi-Hartenstein et al. (2006) initially describe the neuropil domains but with little detailed description of the boundaries. Pereanu et al. (2010) gives some more detailed boundary definitions and renames them to relate better to the adult CNS. Kumar et al. (2009) discusses the tritocerebrum boundaries and Pereanu (2006), Lovick et al. (2013) and Hartenstein et al. (2015) use the neuropil domains as landmarks for lineage descriptions, enabling myself to do the opposite. For the VNC, the only published material is that of Hartenstein et al. (2018). These publications, along with an accompanying online atlas that utilises weblinks to ease navigation, provide non-interactive images of 3D compartments and tracts and LM maximum projections. Despite these it is not straightforward for researchers to annotate their own images and, in addition, many boundaries are not described. More definitive details are required to ensure that these are consistent among researchers as the wealth of larval neuroanatomical literature increases with the advance in tools. Other researchers, such as Andreas Thum (AT), are already defining subdomains in the MB Saumweber et al. (2018). It is also vital that these domain definitions are made accessible in the leading databases of the field, specifically FlyBase. Prior to my work this database contained only very basic definitions and was missing some domains entirely.

Given the above limitations, I tasked myself with improving the definitions of the neuropil domains to make them well defined and easy to apply to new imaging data. In doing so I made sure that the key people involved in defining these regions and

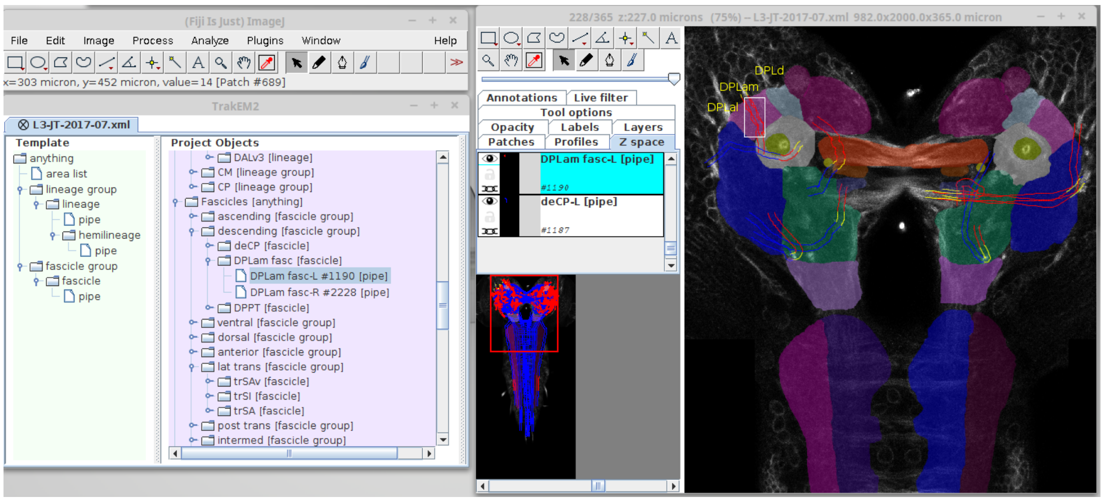


Figure 2.2: TrackEM Viewer
The working copy for annotating the larval CNS using TrackEM (part of the FIJI extension of ImageJ). The lower left window shows in the left "Template" panel the simple ontology I have created to store the pipes and area lists. The panel to the right shows the structure of the annotations and the right window the viewer, with options to add text labels and manipulate points of pipes and shade area lists by mouse. Pipes are colour coded by relative z-slice position - The right panel overlays pipe projections with annotated points in this Z-slice, while the lower left panel overlays all of them.

maintaining definitions, namely VH, AT and FlyBase, agreed upon them with me so that the definitions would represent a standardised brain that can be universally accepted by the community. I worked chiefly with VH in order to annotate the template image stack (see section 3.4.3) by manually "painting" *area lists* (shaded overlays) in the TrackEM Java-based software Cardona et al. (2012) (see figure 2.2). I made sure to match my work to the reference stacks and publications of VH and discussed with him any inconsistencies among these (of which there were very few, mostly only due to updated understandings over the course of research). As part of the process I also smoothed all domains by convolving binary domain volumes with a normalised Gaussian filter with standard deviation 4 x-y voxel widths and assigning voxels to the highest valued domain at its point. This smoothing reduced small variations in painting, especially between z-slices and additionally made the domains more visually appealing.

Having done the above I wrote what are the most detailed to date textual definitions describing these domains and agreed them with VH and others. These descriptions combined in one place for the first time, for each neuropil domain, the details of bordering domains and their respective boundary definitions relating to structural landmarks including cortical shapes, reference staining, lineage tracts and glial cells. Where possible, these details cited previous publications, otherwise were for first

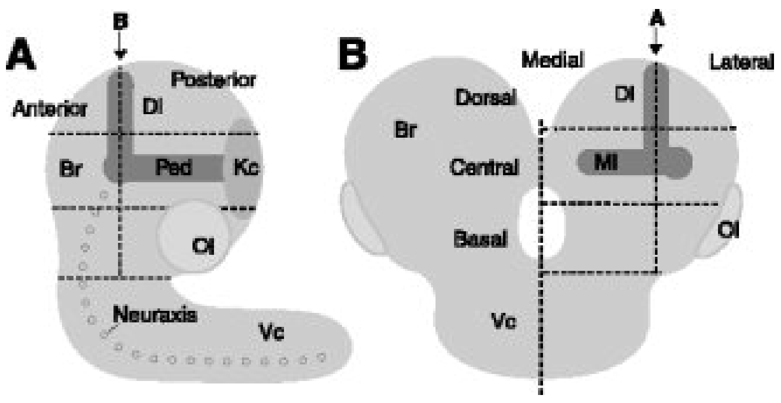


Figure 2.3: Brain Axes Orientation
Illustration of the axes of the *Drosophila* Larval Brain. Note the curved neuraxis in a). From Nassif et al. (2003).

publication as part of my own research paper. These textual definitions were used to update definitions in the FlyBase database, and in some CB cases - e.g. Superior Intermediate Protocerebrum (SIP) and for all of the VNC - *add* new ones.

Specifically, the names of the adult brain domains were the subject of protracted discussions for the *Insect Brain Name* publication of the adult *Drosophila* Ito et al. (2014). In an attempt to avoid these here I worked closely with both VH and AT and met with FlyBase to agree which would be the prominent names given to the neuropil domain, and which would be synonyms. We decided to follow the Insect Brain Name paper as closely as possible which mostly matched the positional names published by VH, with the exceptions being the Inferior Protocerebrum (IP) becoming the *larval Clamp* and anterior Inferior Protocerebrum (IPa) becoming the *larval Crepine*. The FlyBase ontology was also updated to reflect some minor changes, most notably incorporating the LT - formerly Medial Appendix of the Medial Lobe (MAML) - into the ML.

In what follows I will summarise the key structures in the larval brain, since they will be referred to again in section 5.4.7. The detailed, to be published definitions referring to named lineages and fascicles are reserved for appendix A.1. See also figures 2.4 and 2.5.

2.2.1 *Central Brain*

The CB (or *supraesophageal ganglion*) includes proto-, deutero- and trito- cerebrum regions. The following domains are part of the protocerebrum, unless stated otherwise. Here the neuraxis for each of the symmetrical, spherical sides is about the centre

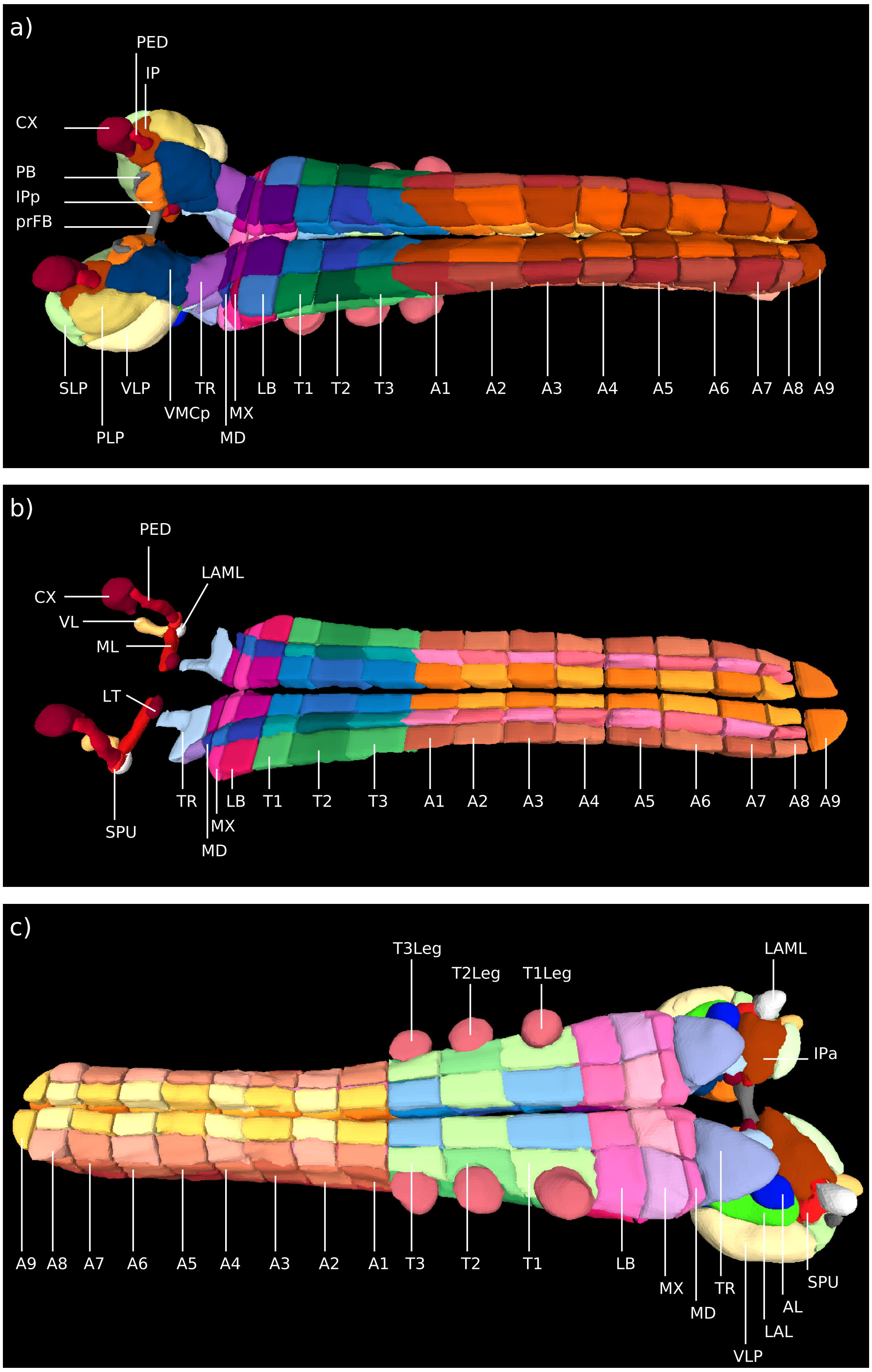


Figure 2.4: 3D Neuropil Domains
3D views of a), b) Dorsal side of the VNC, posterior side of the CB a) with all domains, b) central domains only and MB. c) Ventral side of the VNC, anterior side of the CB. The abbreviations list gives a list of the names corresponding to the displayed labels.

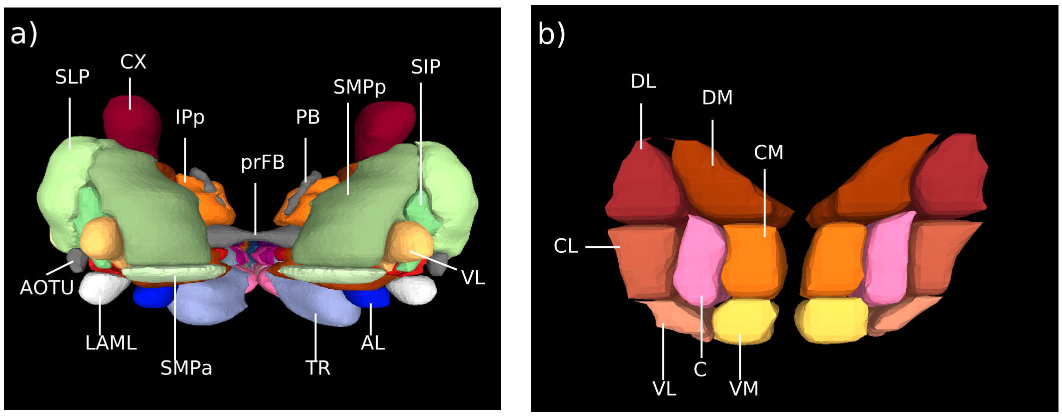


Figure 2.5: 3D Neuropil Domain Cross Sections
3D views of a) front (dorsal) side of the CB, b) coronal cross section through the VNC. The abbreviations list gives the names corresponding to the displayed labels.

(Spur) of the MB. This is because it is easy to identify, especially in BP104, due to the high concentration of cell tracts and arborisation and its three lobes being approximately mutually perpendicular; the ML is directed from the centre medially towards the commissures, and includes a ventrally directed appendage at its tip (the LT). See figure 2.3. The Vertical Lobe (VL) is directed dorsally and the Peduncle posterior towards the spherical protruding Calyx. A much smaller, spherical appendage is lateral and anterior of the spur; this is the Lateral Appendix of the Medial Lobe (LAML).

Surrounding the Peduncle, the cylindrically shaped larval Clamp is itself surrounded by further domains that prevent it from reaching the cortex apart from at its very posterior end. The Clamp's boundaries with these other domains, that do reach the cortex, can be delineated based on both subtle changes in expression of BP104 and lineage tract and fascicle landmarks. Starting above the VL and moving clockwise *on the left side of the CB*, one encounters the posterior Superior Medial Protocerebrum (SMPp). Next, in line with the bulging shape of the neuropil-commissure boundary, posterior of the ML, one reaches the Fan-shaped Body primordium (prFB), a high BP104 intensity, oblong neuropil domain surrounded by a lower intensity domain, that crosses the commisure. Instead, more posteriorly one reaches the posterior Inferior Protocerebrum (IPp). The IPp's ventral boundary is a *virtual* (i.e. no clear delineations in synaptic density) axial plane. The next two clockwise domains are ventral of the Clamp and IPp and are also split coronally at the commissural level, giving four domains in total. These are the VMCa (anterior) and the posterior Ventro Medial Cerebrum (VMCp) (posterior, both deuterocerebrum). These are followed clockwise respectively by the Ventro Lateral Protocerebrum (VLP) (anterior) and Posterior Lateral Protocerebrum (PLP) (posterior). These pairs are split from each other by a virtual plane drawn along the sagittal axis. Next, lineage entry points mark the start of the next clockwise domain, the dorso-laterally located Superior Lateral Protocerebrum (SLP), that has notably higher BP104 intensity and synaptic density. In the posterior of the CB, the SLP meets the SMPp posterior of the VL, though more anteriorily it is interrupted by the small SIP domain that is slightly lateral of the VL and also of high BP104 intensity.

Just posterior of the coronal level of the MB VL and ML, the Clamp is replaced around the Peduncle by the Crepine, a domain that also surrounds the ML anterior of the prFB, replacing the IPp. Around the Peduncle, the Crepine itself is surrounded by the same domains as the Clamp, although as one moves anterior they soon end as cortex is reached, with the exception of the IPp, VMCa and SMPp, the latter of which is replaced at the coronal level of the VL with the much smaller anterior Superior Medial Protocerebrum (SMPa) domain. The VMCa meets the LAL which, moving anteriorly grows between the VMCa and the VLP as the VMCa shrinks between the dorsal protrusion from the tritocerebrum alongside the foramen and the LAL. The LAL forms a clear bulge in the neuropil on the ventro-anterior side of the CB. The deuterocerebrum includes one further domain in addition to the VMCa and VMCp, the AL which is a spherical domain located to the anterior of the VMCa and LAL boundary, just dorsal of the tritocerebrum, mostly surrounded by cortex.

The prFB is classed as a primordium since the domain is developing in the larva from SATs but devoid of synapses. There are three other primordia that can be identified in the stack: Firstly, the Anterior Optic TUbercle primordium (prAOTU), which is a small appendage to the doroslateral Spur. Secondly the Noduli, a very small spherical appendage to the prFB on the ventro-lateral side and finally the Protocerebral Bridge (prPB), a tubular structure that runs over the posterior, then dorso-medial surfaces of the IPp towards the commissure for later fusion with its opposite counterpart.

The tritocerebrum is somewhat distinct from the remaining two regions, and is the least studied part of the CNS. Although it is part of the CB I will describe it in the following section in its relationship to the VNC. The protocerebrum also includes the optic lobe, a region I do not consider in this project and therefore have not labelled or defined.

2.2.2 *Ventral Nerve Cord*

The VNC is demarcated from the tritocerebrum and then into segments with boundaries of repeated, approximate coronal slices defined by the entry points of lineages as commissures. This gives the *subesophageal ganglion* domains (Mandibula, Maxilla, Labium), Thoracic domains (T1,T2,T3) and the Abdominal domains (A1-A9). The whole VNC and tritocerebrum can also be divided axially into dorsal, central and ventral tiers, with the central tier surrounding the large longitudinal fascicles. From the Mandibula to A8, the ventral and dorsal tiers are then finally divided into lateral and medial domains by sagittal planes approximately through their mid-lines, with the central tier split into 3 either side of the longitudinal fascicular tract regions (lateral, central, medial). The tritocerebrum is the least well defined region and lacks continuation from the VNC of lateral domains as the longitudinal fascicular tracts curve laterally, retaining only "dorsal","central","centrolateral" and "ventral" subdomains.

2.2.3 *Lineages and Fascicles*

Again, a number of key papers exist that describe the lineages and fascicles of the *Drosophila* larva CB. Hartenstein et al. (2015) details the PATs in the first instar larva and Pereanu (2006) the SATs with Lovick et al. (2013) providing further details of the SATs and linking them to the adult. As the larval data I am working with is third instar, I was able to identify the secondary lineage tracts based on these sources and also personal communications with VH. The cells of the image dataset I worked with developed during the embryonic stage of the larva and followed PATs that correspond to at least the initial paths of the SATs in the neuropil followed by the later born cells from the same lineages. The coherent SATs are likely longer than the PATs since the larval brain has grown in size, but by precisely how much it is not clear (VH, personal communication).

For my template stack, I have annotated the CB with both SATs and fascicles and the VNC with only fascicles. The annotations took the form of *pipes* in the TrackEM software with radii adjusted to be broadly surrounding the tracts. The pipes began at the entry point into the neuropil or further into the cortex such that all cells in the lineage should pass axons along the entirety of the annotated lineage tract. Some lineage annotations required multiple pipes for splits and some required multiple annotations for different hemilineages. When it was not possible to differentiate two lineages from each other, for example *DALcm1* and *DALcm2*, a single tract was denoted as for both by "DALcm1/2". I annotated 96 tracts for each side of the CB at least as long as they (or their merged-into fascicles) could be seen in the template reference channel keeping them consistent with the published descriptions and reference stacks. The BP104 labelled tracts well enough for the vast majority of the CB fascicles to be completely labelled (35 fascicles were labelled in each CB hemisphere and in the VNC 5 cordal and 28 commissural).

2.3 CHAPTER SUMMARY

In this chapter I have introduced the data set that I will be working with and detailed my initial impressions and highlighted some limitations. I have also detailed the extensive work I undertook to define a standardised larval neuroanatomy that has substantially improved the world-leading database FlyBase regarding *Drosophila* larva. These textual descriptions are the most comprehensive that I know to exist today.

2.4 CHAPTER DISCUSSION

Whilst the current database is well curated, its access is mostly restricted to JT himself. It is difficult to search for a particular cell type in the CB, other than by domain innervation or GAL4 line if known. It is even more problematic in the VNC where there are no innervated domain annotations. These factors mean that to locate cells often expert knowledge of the dataset is required, which means this is dependent on a very limited resource: JT's time. This delays research requiring sparse or split GAL4 lines as the required cells must be found in the database by JT himself. It also limits the scope of information able to be gained about global structures and morphologies in the neuroanatomy. In addition, the database is likely to suffer from some form of human error: For example, it is suspected that there may be some duplicates of cells or two or more treated as one as some are very similar (JT, personal communication).

Considering my project, it is unfortunate that there is no mapping of cell type to the original stacks the type was observed in, since this would simplify the task of matching the cell type names in this database to the examples I will extract from the stacks. As it stands, I am able to suggest candidate types from all those specified in the database to have been observed in the GAL4 line the stack represents. It is not clear, however, how complete these lists would be as it is not known which stacks have been used to produce the database. As a result, grouping the observed cells into the types of the database will require input from JT and his wealth of experience in identifying unique cell types. The above points do, however, all add to the justification for my project in working to better organise this dataset and discover ways to work with it.

Regarding the neuropil domains, some synaptic neuropil boundaries are classed as "virtual" due to a lack of any clear boundary in the reference staining. This means that labelling these domains must be based on visible landmarks, usually lineage tracts. This is helpful for locating the region of the boundary but may bias its exact location, contributing to justification for experiments with automated segmentation based on a large collection of neural morphologies. By agreeing these domain definitions with the major researchers in the field (AT, JT and VH) and FlyBase I hope that their usage will be adopted by the community, particularly the EM project, and will avert the requirement of much time and energy to do this in the future (as was the case for the adult fly domains).

3 REGISTRATION

3.1 AIMS

In order to make it possible to relate cells from different image stacks to each other and the annotated template, it is necessary to display cells from different individual larvae together in a single, annotated image stack. All points in the cells must be located at the corresponding positions in the template and different individual brains from which they came. Due to the varying image positions, sizes and orientations of the different individual brains, I first needed to transform, or *register*, the cells into the template space. Secondly, I needed to calculate representative data about the cells that was brief enough but also descriptive enough to enable the database to be searchable in a reasonable amount of time (the images are far too big to be searched directly).

I agreed with JT that a suitable target would be for at least 25% of the image stacks to be judged by him as "well registered" so that (assuming no bias) there would be reasonable coverage of all the cell types across the CNS and from a range of GAL4 lines. As discussed, however, a key project goal was to allow researchers to quickly discover examples of the same cell type from different GAL4 lines. As this latter goal required the investigation and development of further tools and I had limited time, I decided to take an Agile Software Development approach Beck et al. (2001). This meant that I prioritised a system that worked well enough end-to-end; one that proved that it was possible to deliver for the researchers what they needed most (i.e. the lookup tool) but also one that could be iterated and improved with time. This is in contrast to perfecting each component progressively (the *waterfall* approach).

Finally, it is important to keep in mind that the following was carried out prior to any published successful work in larval registration, against a backdrop of some scepticism in the community.

Figure 3.1 gives an overview of the pipeline described in this chapter.

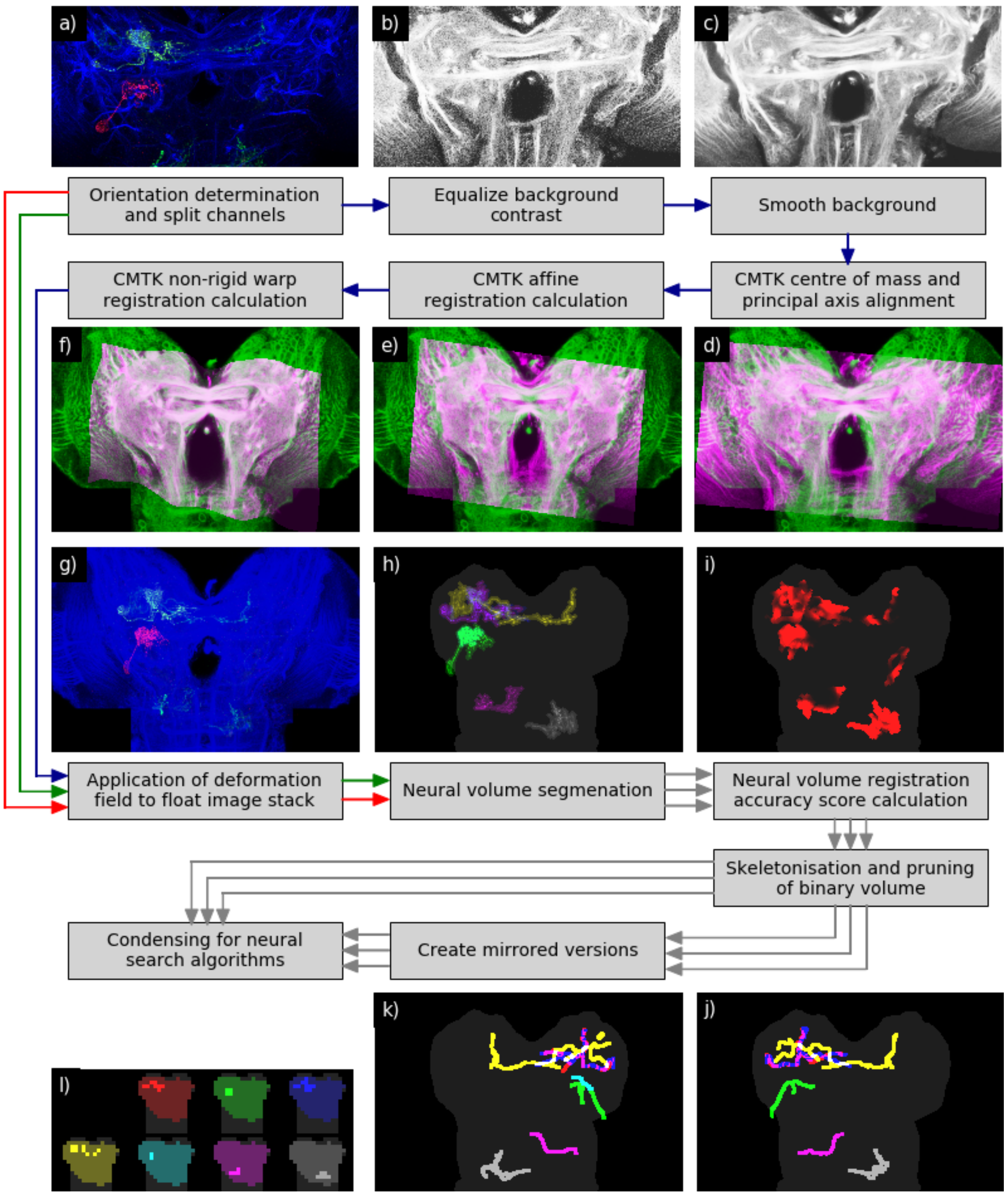

a)
b)
c)
Orientation determination and split channels
Equalize background contrast
Smooth background
CMTK non-rigid warp registration calculation
CMTK affine registration calculation
CMTK centre of mass and principal axis alignment
f)
e)
d)
g)
h)
i)
Application of deformation field to float image stack
Neural volume segmenation
Neural volume registration accuracy score calculation
Skeletonisation and pruning of binary volume
Condensing for neural search algorithms
Create mirrored versions
k)
j)
l)

Figure 3.1: Overview of the image registration pipeline
The coloured arrows represent the different channels of the original image stack (background as blue, signal as red and green). The grey arrows represent the neural volumes. a) Maximum z-projection of raw image stack, b), c) z-slices of background channel, d), e), f) z-slices of overlays of registered float background channel (magenta) on template (green). g), h), i), j), k) maximum z-projections of: g) registered float signal (red, green) in template background (blue), h) different coloured signal and volume occupied for each neural volume i) neural volume registration accuracy map, j) and k) pruned neural volume skeletons. l) z-slices of condensed representation (fixed points).

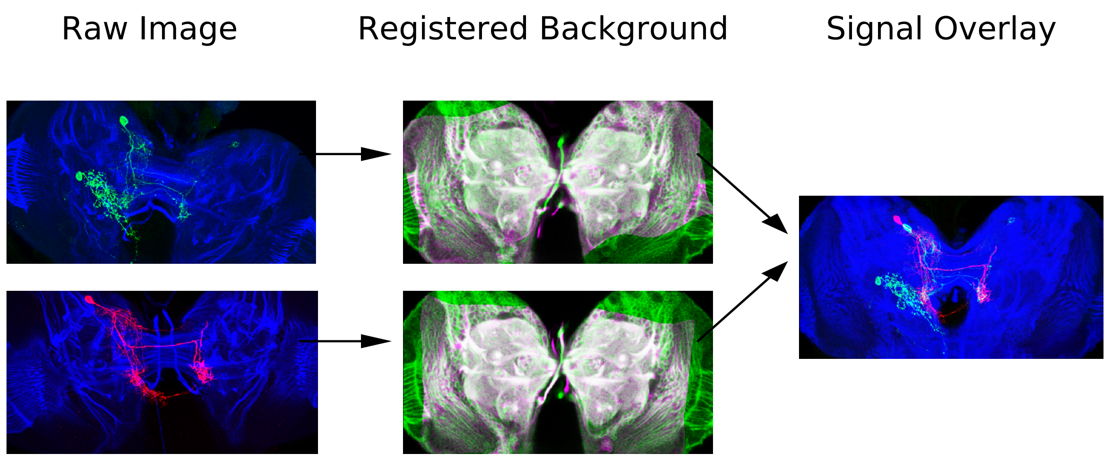


Figure 3.2: Overview of registration process
Left: Maximum projections of sample images of the Central Brain to illustrate the power of registration. Middle: Coronal slices of the same level through the template image stack (green) with the registered float reference channel superimposed (magenta). Right: Maximum projection of the overlay of the two signals which enables one to identify a common type of cell between the two signals.

3.2 INTRODUCTION TO REGISTRATION

For an individual's 3D brain image, which is commonly termed the "float", a warping of coordinates of voxels into their corresponding positions in the common, template, brain space is required such that the anatomical features are in the same locations in the common space (see figure 3.2). It is important to note that this is not an exactly solvable problem: There will be some inter-subject variability among the brain specimens, due to subject-specific environmental and biological factors and particularly due to the human-induced factor of dissection and preparation of such small specimens. In any case, most computational algorithms will converge to a local best registration in the space of possible registrations between a float and template. As a result the quality of registrations can vary significantly so it is important to employ a method to judge when a registration system performs satisfactorily and when not.

In general, image registration is the process of calculating a map of one topological space to another, one that should be a *diffeomorphism*, i.e. one that is one-to-one continuous, differentiable and inversely differentiable Encyclopedia of Mathematics (2017). This one-to-one (bijection) property is important as it means that the map function can be reversed and that no parts of the image can become hidden. The mapping between the float and template spaces could be based on a global function such as a simple linear transformation Goshtasby (2012) or something more complex e.g. a Fourier transform Luce et al. (2014). Alternatively, an optimized individual voxel-level ("non-parametric") map Vercauteren et al. (2007) could be employed. Commonly, a mixture of these two methods is applied, matching up corresponding points between the float and template and generating functions to interpolate voxels between the points Goshtasby (2012).

Examples of registration in neuroscience tend to focus on the human brain where MRI scans are frequently registered to various *standard* brains. In a study seeking to target the same part of the human brain with a broad-based tool such as Transcranial Magnetic Stimulation, an adjustable, bounding 3D box containing a subject's MRI was overlaid onto the standard Montreal Neurological Institute (MNI) brain and manually adjusted to fit Wood (2013). Other studies that attempt to segment subvolumes and make inferences across populations require more accurate methods, such as in Kanai et al. (2011). Registration is frequently applied to the CNS of mice Richardson et al. (2014) and adult *Drosophila* Armstrong and van Hemert (2009), usually in order to make across-subject inferences about genetic-based expression patterns. Sometimes a manually registered brain can be optimum, but with large variations and the human eye being distracted by matching up well-known anatomical landmarks and disregarding less obvious inconsistencies there are likely to be many systematic errors - ones that also vary between registrar. In general, fully automated methods outperform the manual (e.g. Sarkar et al. (2005)). A range of algorithms and tools exist to automate this process and improving registration algorithms is an active area of research with computational time a major limiting factor in possibilities.

3.3 FULLY AUTOMATED REGISTRATION

Automated registration methods work to optimize a similarity function of both float and template image intensities Goshtasby (2005). This could be calculated locally, at points in the image using voxels around it, or globally for the whole image. Parameters can then be adjusted using gradient descent.

Some of the most popular and well-maintained tools for registration form toolkits, for example the popular Advanced Normalization Tools (ANTs) Avants et al. (2008), *elastix* Klein et al. (2010) and Computational Morphometry ToolKit (CMTK). All three of these open-source toolkits allow registration to be implemented using popular "elastic" algorithms to produce a Free-Form Deformation (FFD). A FFD optimizes the matching of a 3D grid of control points throughout the image. These control points are then used to construct *B Splines* for interpolating between them. The ANTs toolkit also includes a non-parametric algorithm called Symmetric Normalization (SyN), which is part of a family of non-parametric algorithms based on the Demons Algorithm for diffusion Thirion (1998). Briefly, these algorithms find shifts for each and every voxel from the float to the template and optimize these with respect to a similarity metric whilst ensuring the shifts are "continuous" (there are no overlaps) by using a regularisation technique such as smoothing. Although the more recently produced SyN algorithm has shown to be promising, B Spline methods are still "competitive" with it in comprehensive human brain registration tests for accuracy and do not demonstrate significant speed increases Klein et al. (2010). Finally, a popular tool for *Drosophila* adult brains, namely BrainAligner Peng et al. (2011), seeks to find in the float image manually defined "corners" or edge points in the template (high entropy points). The deformation is then interpolated from these paired points as in elastic registration.

I decided to initially utilise CMTK for registration in this project for a variety of reasons: Firstly, it has performed impressive registrations for the brains of bees Rohlfing and Maurer (2003) and adult *Dropsphila* Knowles-Barley (2012), including with the same reference labelling (anti-neuroglian) Court (2017). Specifically, Court (2017) reports that CMTK performs better registrations than BrainAligner with the adult *Drosophila* ventral nerve system due to a lack of clear landmarks in this system. Given that I require registration of the larval VNC, I decided not to investigate BrainAligner further. CMTK also does not require manually choosing landmarks which would likely be time consuming. Furthermore, Rob Court (RC) (a post-doc with the VFB project) was able to provide support by suggesting parameter combinations to explore as well as experience in utilising the software within the university computer system. In addition, CMTK was also being integrated with the VFB website to allow users to upload their own stacks for registration, which could be a useful tool for the larval version of VFB too. I believe, therefore, that the slightly better performance of non-parametric methods on other data sets did not justify their investigation at this point (though it and alternate toolkits should be kept in mind for later tests as part of the Agile workflow).

3.3.1 *Computational Morphology Toolkit*

Below I discuss the robust method for image registration I applied from CMTK Rohlfing and Maurer (2003), summarised in figure 3.3.

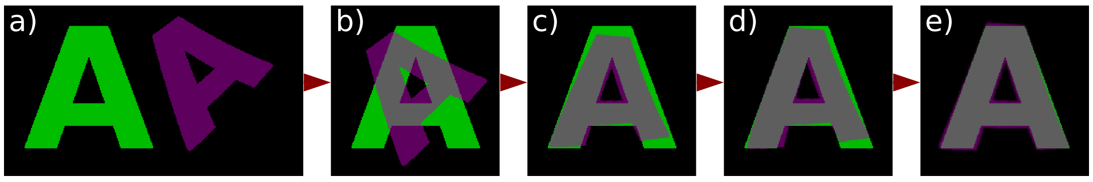


Figure 3.3: Simple 2D summary of CMTK image registration steps
a) The magenta float is to be mapped to the green template with results displayed following: b) centre of mass alignment, c) principal axis alignment, d) affine, rigid registration (notice scaling), e) non-rigid (warp) registration.

First, I applied the CMTK **`make initial affine`** method Rohlfing (2011) to each image stack. This calculates a rigid transform (i.e. one of translation and rotation), with 6 degrees of freedom. It treats the intensities of the image as weights in an inertia matrix and translates the float image such that its centre of mass matches that of the template and rotates it such that the orthogonal principal axes Alpert et al. (1990) are aligned along the same directions.

The initial transform is used as the starting point for an affine transform which has 9 degrees of freedom, with an anisotropic scale-factor added to the translation and rotation. These are independently and iteratively adjusted to optimize a global similarity measure. I calculated the affine transform with the CMTK **`registration`** method Rohlfing (2011) that down-samples the image to lower resolutions for prior, less accurate adjustments, increasing it in stages up to the full-resolution image, similar to Studholme et al. (1997).

Finally, a non-rigid deformable FFD is built based upon points in a 3D, uniform Control-Point Grid (CPG) that has been constructed across the template image space Rueckert et al. (1999). Each of these points has a non-parametric, independent shift (in each dimension). To find the shift in each dimension at voxels between the CPG points one simply sums the surrounding CPG point shifts, $\boldsymbol{\phi}$, weighted by the third-order B-Spline functions Lee et al. (1997). These B-Spline functions (B) for a voxel at a point of interest, (x, y, z), are calculated from the proportional position (represented by (u, v, w)) of this point between the uniform CPG points in the template space (see figure 3.4). The shift (transform, $\mathbf{T}$) at the point (x, y, z) is given by:

$$\mathbf{T}(x, y, z) = \sum_{l,m,n=0}^{3} B_l(u) B_m(v) B_n(w) \boldsymbol{\phi}_{i+l, j+m, k+n} \tag{3.1}$$

where B_0 to B_3 are the B-Spline functions' third order polynomials (see appendix A.2 for definition and explanation). (i, j, k) are the co-ordinates of the lowest indexed

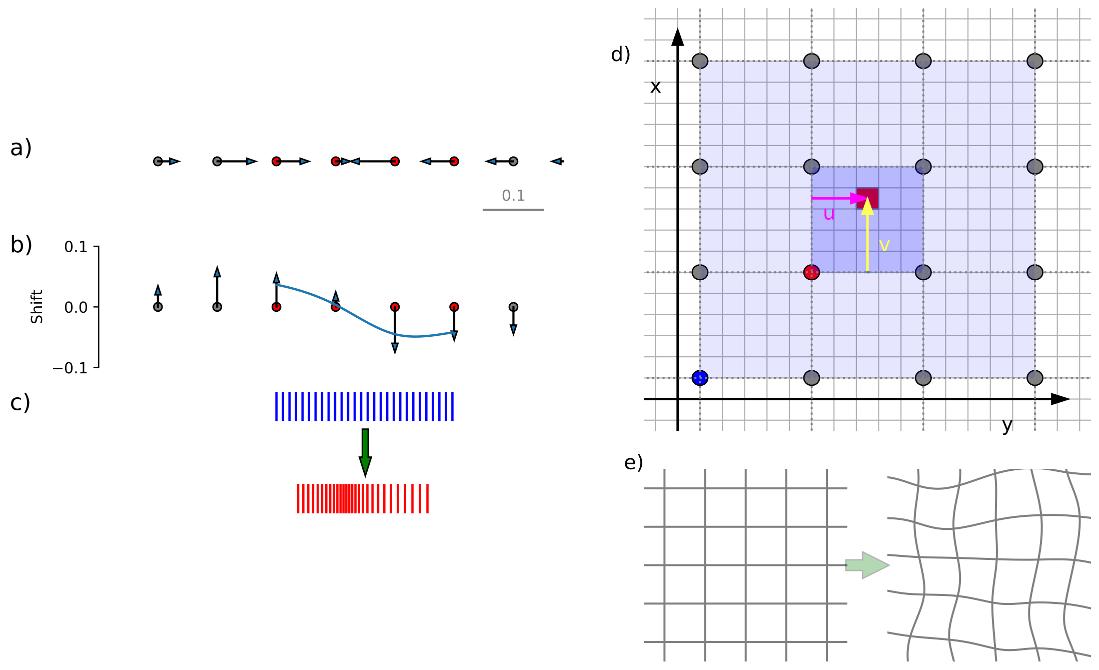


Figure 3.4: B-Splines Explanation
B splines are a generalisation of the more commonly found Bezier curves. a) One dimensional example of control points (knots) with their shifts indicated by attached arrows. b) A third order B-Spline is constructed to evaluate the shift at any point surrounded by at least 2 grid points on both sides. Note that the spline does not necessarily pass through the shifts of the control points. c) The top blue lines represent regularly spaced points between the knots and the red lines below demonstrate the spacing after applying the shifts to bunch up the lines. d) Diagram of Control-Point Grid (circles) in two dimensions imposed on the pixel grid. The displacement, (u, v), from the red grid point $(\phi_{i+1,j+1})$ to the red pixel of interest, at (x, y), is referred to in equations 3.1-3.3. The blue shaded box inclusively marks the region of the Control-Point Grid used for the third-order B-Spline at the red pixel, with the blue bottom-left point equating to the lowest index corner in two dimensions $(\phi_{i,j})$. e) An example of a third order B Spline deformation applied to a 2D grid.

point, in all dimensions, considered for the splines constructed for this point of interest. They are given by:

$$i = \left\lfloor \frac{x}{\delta_x} \right\rfloor - 1, \quad j = \left\lfloor \frac{y}{\delta_y} \right\rfloor - 1, \quad k = \left\lfloor \frac{z}{\delta_z} \right\rfloor - 1 \tag{3.2}$$

with δ_x, δ_y and δ_z being the distances between the CPG points in each dimension ($\lfloor\rfloor$ represents the floor function). Finally, the fractional distance from the point of interest to the lowest corner of it's containing CPG "cell" is:

$$u = \frac{x}{\delta_x} - (i+1), \quad v = \frac{y}{\delta_y} - (j+1), \quad w = \frac{z}{\delta_z} - (k+1) \tag{3.3}$$

This method, calculated with CMTK **warp** Rohlfing (2011), is in common with creating free-form deformations Sederberg and Parry (1986). The only degrees of freedom are thus the CPG shift vectors and these are moved in directions that improve a global similarity metric calculated from applying the FFD to the template space. As with most other registration algorithms, CMTK uses a gradient descent method to calculate the direction to further shift each control point, and then finds the optimum distance to move it in that direction. The gradient and optimisation is calculated for each direction at each point (independently over the whole grid) prior to updating the shifts and recalculating the similarity metric between the shifted template voxels and the float. Trilinear interpolation is used to calculate intensity values at the shifted points. The process is iterated to gradually improve similarity as the step sizes in the gradient shifts decrease.

Furthermore, CMTK applies an adaptive grid-spacing and resolution approach that decreases grid spacing and increases resolution and number of control points as it refines the registration Rohlfing and Maurer (2003). This allows large deformations to be optimised quickly first before smaller perturbations are added in. In addition, for the non-rigid calculation, CMTK introduces options to help prevent large decompressions or folding of the resulting image over itself, to ensure a bijection Rohlfing and Maurer (2001). I applied the option of adding a bending energy introduced by Wahba Rueckert et al. (1999) Rohlfing and Maurer (2003) Wahba (1990). This adds the following regularisation penalty term to the similarity measure, with a weighting given by λ and the sign dependent on the desire to minimise or maximise the similarity measure:

$$E_{\text{smooth}} = \lambda \int_D \left(\frac{\partial^2 \mathbf{T}}{\partial x^2}\right)^2 + \left(\frac{\partial^2 \mathbf{T}}{\partial y^2}\right)^2 + \left(\frac{\partial^2 \mathbf{T}}{\partial z^2}\right)^2 + 2\left[\left(\frac{\partial^2 \mathbf{T}}{\partial x \partial y}\right)^2 \left(\frac{\partial^2 \mathbf{T}}{\partial y \partial z}\right)^2 \left(\frac{\partial^2 \mathbf{T}}{\partial z \partial x}\right)^2\right] dx \tag{3.4}$$

Where D is the set of reference image voxel coordinates. These values are easily computed from the B Spline equations (see Rohlfing and Maurer (2003)) and have the effect of smoothing the deformation field. CMTK also fixes control points that occupy zones of little variation (such as background) to speed up the calculation.

Following its calculation, the FFD was applied to find the position in the float space of each voxel in the template space. For each colour channel, the intensity value for each template-space voxel was assigned from the float image using trilinear interpolation of the intensity values of voxels surrounding the non-discrete position in the float. I employed the CMTK **`reformatx`** method Rohlfing (2011) for this. The template space was usually larger than the registered stack space and these vacant volumes were filled with zero value intensities. In order to keep a record of the boundaries of the warped float image (its *bounding box*), the **`reformatx`** method was also applied on a 100% full intensity (255) monochrome float image. This allowed identification of registered cells in the template space that were clipped by the original image and likely missing parts that would be contained in the template stack space had the original image covered them.

3.4 TEMPLATE CHOICE AND REGISTRATION PARAMETERS

CMTK runs in parallel and was able to complete the registration of a single image stack of the image data I had within the range of 2-15 hours using 3 Central Processing Unit (CPU) cores. The accuracy and length of time, however, depended upon the parameters set. I needed to balance the compute time I had available with the requirement to find a combination of template and registration parameters that produced consistently highly accurate registrations. I expected there to be a great degree of variation in registration quality among different templates as well as certain parameters and this was confirmed once I started testing. Key CMTK parameters are the similarity metric used, the CPG spacing and its number of refinements, the size of the steps used in the gradient descent and, finally, the weight of the regularisation. I also investigated pre-processing the images with contrast normalisation and smoothing.

No numerical accuracy target was agreed with collaborators, but I aimed for within 5µm within the neuropil as this region should be near consistent between individuals at this level. I did not set any aims for the much more variable cortical regions. Templates and parameters were tested in a two step process involving two sets of image stacks from the data set. This was motivated by the time constraints, since the first step was faster than the second for each combination. The first was sufficient to identify good parameter and template combinations but relied on visual inspection and hence was somewhat subjective and not precise. Therefore the second, slower test enabled these good combinations to be separated and ranked objectively and mathematically. As I would need to register around 22 000 image stacks, and I had full-time access to approximately 150 CPU cores, I required each registration to take on average up to 8 hours with 3 CPUs for the whole set to be processed in 2 months (my target to allow time for analysis work). I decided, therefore, to limit investigation of parameters to those that satisfied this requirement.

3.4.1 *Initial Evaluation Tests*

The first evaluation of a template and combination of parameters consisted of initially registering a common set of randomly chosen 5 CB or 5 VNC stacks to the corresponding stack to be used for the template. I visually inspected the alignment of the tracts in each of the 5 registered float stacks' reference channels to those of the template. To do this I overlaid the float over the template using a different colour as in figure 3.5 and rated the alignment from A (good) to F (bad) by visual inspection across the stack. (See examples in figure 3.5 and section A.3 for an explanation of the rating system). The initial 5 registrations gave enough varied results that I could rule out over half of the combinations (where there were two or less A or B rated stacks). For the rest I performed and inspected a further 5 registrations from the other region of the CNS and for the similarly high performing stacks from this region, I carried out up to 4 further registrations (from both regions) to separate these results further.

I performed this test on all 128 potential templates using the registration parameters successful in the adult *Drosophila* since initial tests showed some promising registrations. As the templates were similar in their relationship to the registration parameters (in terms of size, contrast etc.), the parameters were then adjusted using just one template that performed well rather than with many. I adjusted most CMTK registration parameters independently about those for the adult *Drosophila* until I met values that produced notably worse results or took too long to complete. From within the explored range I noted the best performing values to use and evaluate further. I similarly investigated the pre-processing options. For groups of the parameters that were related, such as grid spacing and initial gradient step, I performed a *grid search*. This meant systematically combining each value for one parameter from a list (usually 3-4 values, sometimes more) with each value from a list for the other parameter (forming a "grid" of parameters in 2D parameter space).

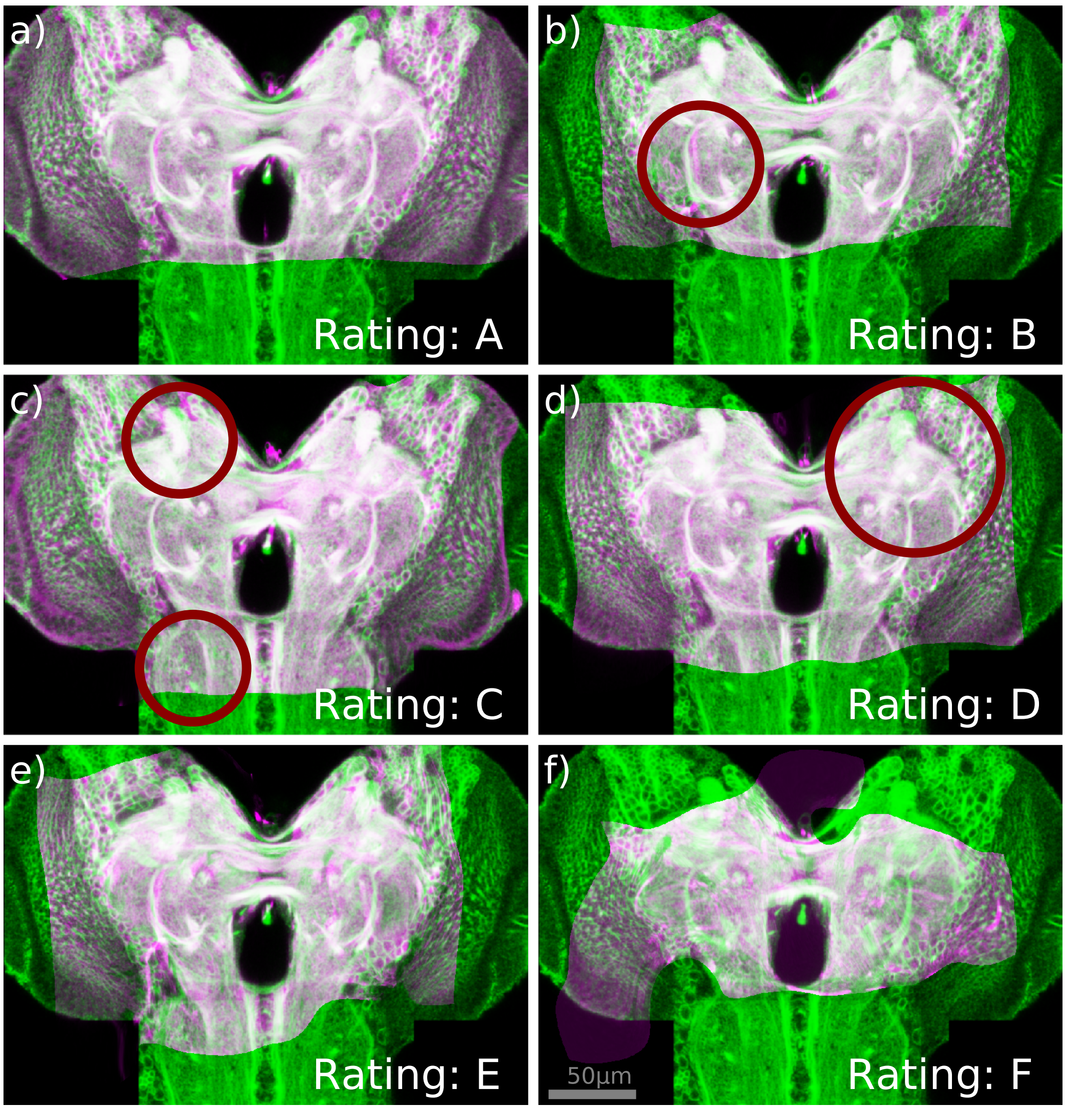


Figure 3.5: Examples of overlays of registered reference channels to a template
These examples illustrate some of the variability and the ratings that would be applied if the image was a single slice. Coronal slices of the template reference are shown in green and the registered float in magenta with well-registered regions displaying white as to indicate good overlap. Images have matched contrast and are smoothed for display as per sections 3.4.4 and 3.4.5. a) A very well registered image stack. b) Well registered apart from the circled fascicle where there are portions of no overlap, although they are within 5µm. c) is registered well apart from in the circled regions: The left MB dorsal lobe and the left SOG. These registration errors are still within 10µm, however. d) This stack suffers from a serious (more than 10µm) registration error in the circled region, meaning it scores lower than c). e) There are serious registration errors in the dorsal part of this image stack slice, but the ventral parts are reasonable, meaning it is not a complete failure. f) This is a terrible registration of score F. These rarely occurred across the dataset using my final parameters and template and usually indicated that the stack was orientated incorrectly in the z-direction. Myself and my colleagues felt that scores of A and B would be accurate enough for most uses of an atlas with C acceptable depending upon the location of the labelled neurons avoiding the badly registered regions.

As the registration performance varied strongly between templates, the above allowed me to clearly identify template stacks that performed better than others. In line with the subjective nature of the ranking, I manually identified 6 templates that performed well based on my experience in visual inspection of all the stacks. These template's scores had high averages and good consistency (low variance). I tested these templates further with the highest scoring parameters using the detailed evaluation metric explained below.

3.4.2 *Detailed Evaluation Score*

This more thorough, objective and numerical test involved registering 134 image stacks. These stacks contained subgroups of single stained neurons that were indistinguishable across multiple stacks (from different specimens) and thus believed to be the same type of cell based on homologous morphology. The individual cells were segmented from the rest of the signal prior to registration using the segmentation method discussed later in section 3.6 to create stacks that contained no or very little (estimated at less than 1% by volume) staining signal from other cells. After warping into the template space, stacks for each given neuron type subgroup (of which there were 18) should ideally contain signal expression in the same locations. Due to biological variability, however, this will not be exact, particularly in arborisation regions, but better registrations will have more overlap of the labelled cells.

I developed a pairwise overlap score, O, similar to that used in Klein et al. (2009) to evaluate registration of manually labelled brain regions. As the neurons include many thin tubular parts where a slight misalignment will cause no overlap (see examples in figure 3.7), I modified the overlap score of Klein et al. (2009) to include a tolerance in distance. Hence the overlap score between a specified query neuron, a, and other neurons, b, within the same subgroup established how much of the signal of the query neuron was within 2µm of the signal of the same type of neuron from another stack. Hence the higher the proportion within 2µm, the better the registration.

This was easily calculated by dilating the "other" neuron signal by 2µm in all direction so that the overlap score was the volume occupied by the query neuron and the expanded other neuron volume as a fraction of the total volume of the query neuron. The binary neuron volume occupied was found by thresholding the signal to take all above zero. The dilation was performed by convolving the positive binary neuron volume with a positive sphere and again thresholding at zero. This allowed

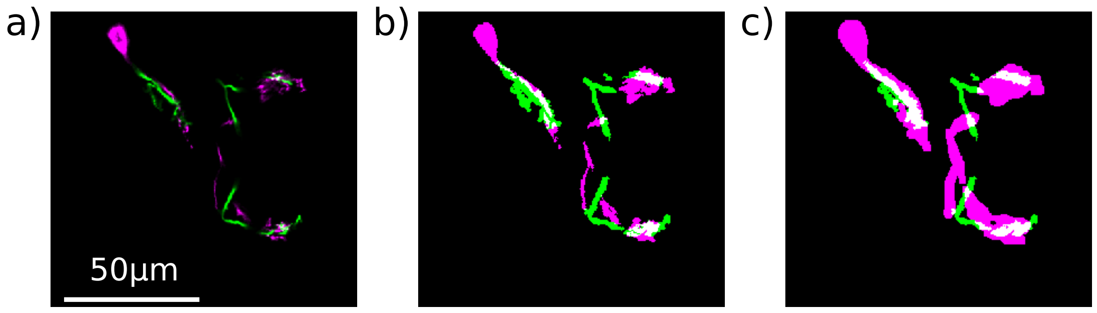


Figure 3.6: Illustration of the overlap score calculation method
a) Single, coronal slices of two cells of the same subgroup (same type) where the green cell is the query neuron and the magenta cell the other. Overlap is shown for a) raw signal, b) thresholded signal and c) following the magenta cell expansion. The overlap score in this 2D slice only is 61.3%.

me to utilise efficient python library routines. As I am only interested in registration quality in the more consistent neuropil, for all neurons, only the volume within the neuropil mask, M, was considered, with the rest removed before dilation.

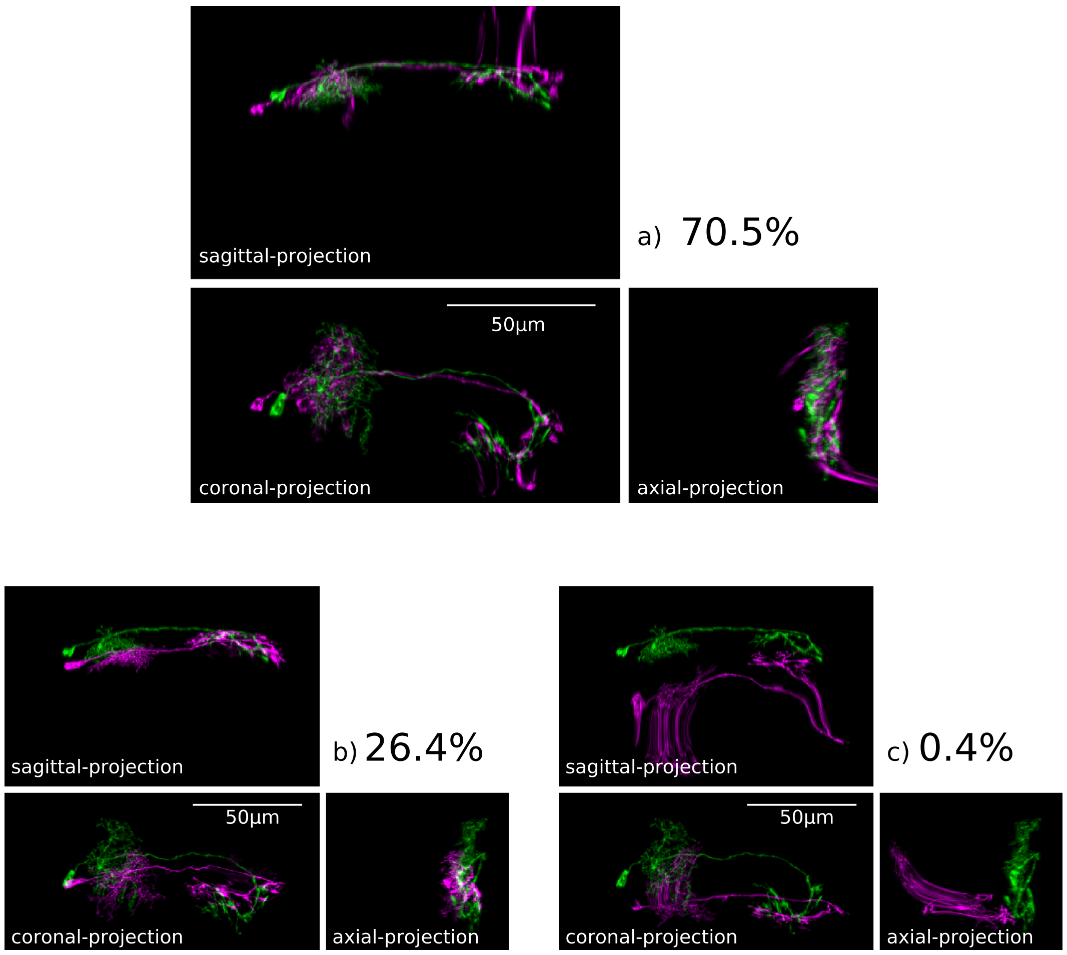
$$O(a,b) = \frac{((S_a > 0) \cap M) \cap (((S_b \cap M) \circledast C) > 0)}{(S_a > 0) \cap M)} \qquad (3.5)$$

Where S_a is the query neuron signal channel and S_b other neuron signal. The method is illustrated in figure 3.6 with examples of scores in figure 3.7.

The neurons for this second evaluation set were chosen prior to registration to be from across the CNS and represent a range of typical neuron morphologies with a minimum of 4 neurons per cell type, maximum 11 (mean 7.4). I desired a single overall overlap score for each template / parameter combination with which to make comparisons. The number of neurons per type varied and I wanted to give equal weight to each neuron type rather than each registration as this would give equal weight to the included morphologies and regions of the CNS. To do this I used a two-step averaging approach: I calculated a mean score for each neuron type from all possible pair-wise combinations of all examples of that cell type. The overall score for a template and parameter combination was then given as the mean average across all neuron types of these individual cell type mean scores.

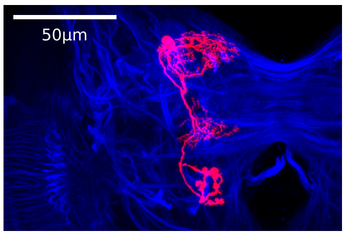
I used this overlap score to make some of the parameter choices (specifically the energy weighting and the similarity metric, the latter of which is discussed below in more detail). I also used the overlap score to quantify the effect of some of the other parameter choices from the initial screening with respect to my chosen template, namely the contrast normalisation and smoothing (also discussed in more detail be-

Figure 3.7: Overlap scores for example cell type

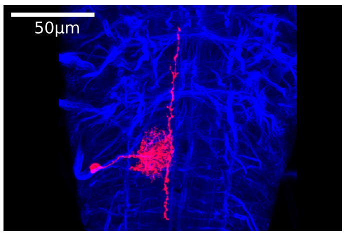


A green, query cell from the SOG which appears to have registered well is compared with three other, magenta registered cells of the same type. The severe stretching evident in c) is typical of very bad registrations. Mean overlap scores are displayed as percentages. The overlap in a) was considered acceptable by myself and colleagues, whereas b) and c) were not.

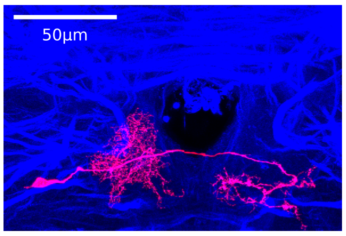
Table 3.1: Registration Parameter Overlap Scores



Example Cell A



Example Cell B



Example Cell C

Cells	Example A	Example B	Example C	All 18 Cell Types	
Overlap	Mean	Mean	Mean	Mean mean	Mean std. dev.
My Solution	72.7	72.9	42.2	53.9	20.3
Non-rotated template	73.2	52.9	0.23	49.8	22.4
Without Contrast Normalisation	68.2	70.1	0.0	31.0	19.0
Without Smoothing	72.5	71.6	23.3	52.1	20.5
MI instead of CR	71.6	60.7	42.2	52.5	22.4
NMI instead of CR	64.0	57.2	38.1	44.6	22.4

std. dev. = Standard Deviation. Overlap scores are expressed as percentages. Here the registration parameters are kept the same as those used in the pipeline unless changed as stated in the left-hand column. It is important to note that when the mean overlap scores for each cell type were compared between my solution and those of any one of the other combinations above, it outperformed them for the majority of cell types. MI = Mutual Information, NMI = Normalised Mutual Information, CR = Correlation Ratio (see section 3.4.6).

low). The results are summarised in table 3.1.

As a result of these tests I chose to register the data set with a grid spacing of 80 voxel widths (giving 1 950 CB and 1 188 VNC initial control points) with a single refinement to grid spacing to 40 voxel widths. The initial optimization step size was 30 voxel widths and the regularisation bending energy weight was 0.1. A full list of parameters and commands used for CMTK is available in appendix A.4.

3.4.3 *Template Choosing*

In this project I decided with the VFB team to see if I could find a template that was a single individual brain that performed good registrations. This was in contrast to using an *average brain*, i.e. one calculated by registering a set of CNS template stacks

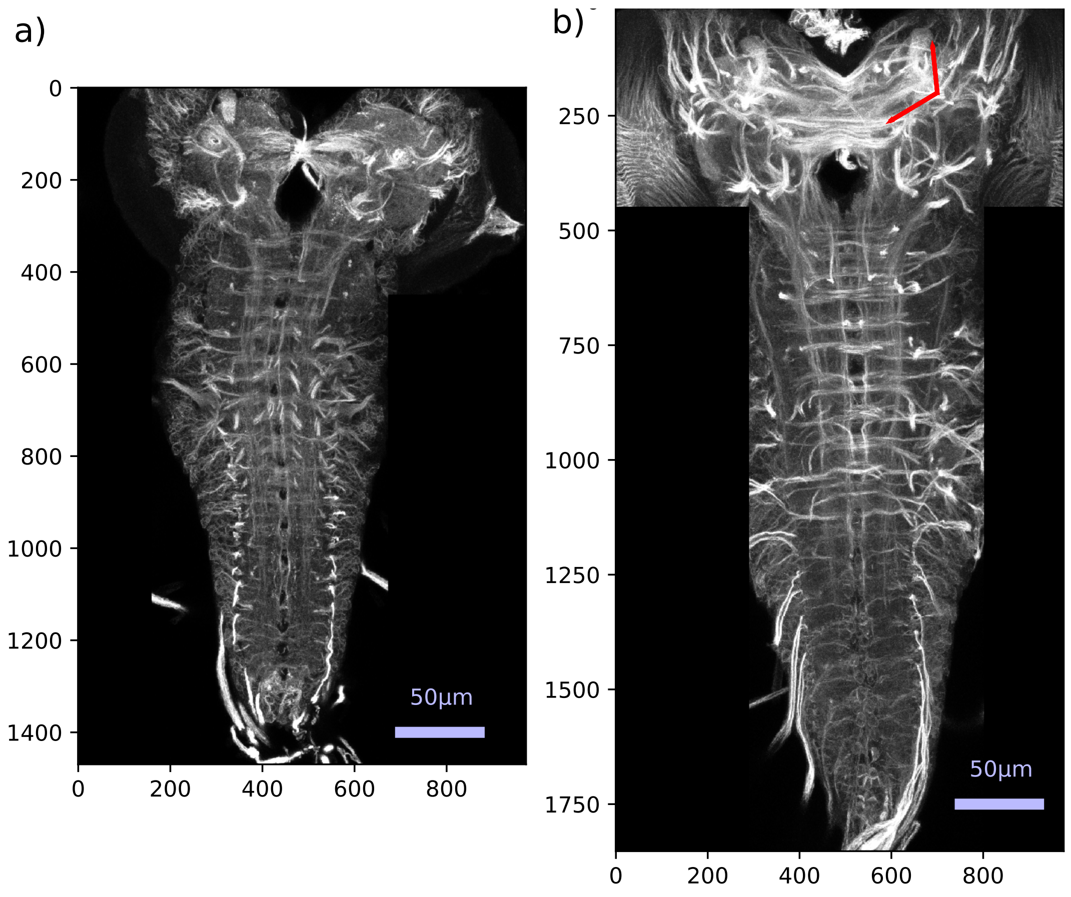


Figure 3.8: Illustration of some unsuitable features in a template
Maximum coronal projections of a) 10 z-slices, b) 35 z-slices. Note in a) that the mushroom bodies are not present at the same coronal level, the VNC is bent and the top of the left MB is at the very edge of the image. The overall length of the CNS is also small in comparison with b) which is more typical of the image stacks. Any one of these imperfections would make this stack unsuitable. b) demonstrates an obtuse angle (red arrows) between MB dorsal and medial lobes that are also bent. In fact, the lobes in many specimens were not perpendicular due to being pulled in various directions during dissection.

Table 3.2: Relationship between biological and template axis

Template axis	CB axis	VNC axis
x	medial-lateral-medial	medial-lateral-medial
y	dorsal-ventral	anterior-posterior
z	anterior-posterior	dorsal-ventral

All are indicated in the increasing direction (top left corner in figure 3.9).

to each other and finding a space that minimises the total amount of deformation required to reach it by each of the template set members Joshi et al. (2004). Using an individual brain has the advantages of being in common with the adult VFB approach and using the original textures and smaller details in the template that are not available in an average (the intensity of which appears as in figure 3.10) and may be helpful to the similarity metric. Despite this, the average brain approach has been shown to enable better registrations in published work on bee brain images Rohlfing et al. (2004) - significantly better than some individual brains, though only modestly better than the best performing individual brains. An average brain does not represent a specimen that existed and therefore may not be representative of any of the actual brain specimens Lancaster et al. (2007). For example, if the template set contained two types of stacks with opposing features, they may cancel out in an average template, producing one that is not optimised for either set.

Since I was able to find individual template candidates that registered a large proportion of the data well and I needed to make progress with other aspects of the project, I did not explore the average template further. My work will, however, enable a future study to compare performance of an average template with a well annotated single larval brain, as has occurred recently for the adult *Drosophila* brain Arganda-Carreras et al. (2018).

To create candidate whole-CNS template image stacks, I searched for VNC and CB image stacks from the same specimen that had considerable overlap between the two imaged CNS zones and were also considered aesthetically of good quality. I had to match VNC and CB stacks from within the same GAL4 line, though not both portions of most specimens were imaged. I decided to perform this search manually as I could use the signal and reference channel coronal projections to do this relatively quicker than automating it using a registration algorithm on the boundary and I had no other motivation to match the images. In addition, I could exclude some stacks when manually searching. In finding a suitable template, particular attention was paid to only choosing images that satisfied the following priorities:

- Contain the entire CNS and as much surrounding cortex as possible (nearly all images miss some cortex around the SOG).
- High contrast ratio to add to visibility of structures and more information for registration.
- Good stitching between tiles within the image stacks (no offsets).

- Similar intensities across the entire image (especially between VNC and CB component stacks) to enable whole-image contrast adjustment.
- Specimens oriented such that the rostral-caudal axis of the VNC and the dorso-ventral axis of the CB were parallel to the Y-axis and medio-lateral axis parallel to the X such that the left and right sides of Z-slices were anatomically correspondent to one-another. These requirements enable easy left-right comparisons in eventual use by researchers.
- Mushroom body lobes approximately perpendicular.

Figure 3.8 gives some bad examples of templates based on the above criteria.

The most restrictive constraint was finding specimens with both CB and VNC regions imaged that also overlapped to enable the two components to be combined into a whole-CNS template. After manually searching over 10 000 stacks I found 128 potential templates, used in the initial screen detailed above. The best performing of the six used in the detailed evaluation had a mean overlap score of 53.9%.

Although I took care to only test template images that were oriented such that the medio-lateral axis was approximately parallel to the x-axis, in my chosen stack the MB ML were not quite parallel to the y-axis. I decided to attempt to improve the visual appeal of this template by rotating it by just 0.9 degrees along the central CB to VNC directed y axis such that the left side moved towards the viewer in a coronal slice. This was performed by TransformJ in FIJI Is Just ImageJ (FIJI) Schindelin et al. (2012) and the intensity values were recalculated using nearest neighbour interpolation to maintain most of the original intensity values. The rotation actually improved the overlap score by almost 4 percentage points and reduced the variance. It is not clear why this was the case, since the initial CMTK registration transformation is expected to ensure the registration is independent of rotation. There was also little loss to image resolution and accuracy due to the small angle of rotation. I estimated this at less than 0.9% of non-background voxels (and hence information). I calculated this figure by reverse rotating the rotated template by 0.9 degrees and noting the proportion of voxels different from their original value out of all non-background voxels.

The template is shown in figure 3.9 and figure 3.10 shows an average reference intensity pattern from registered stacks. I also hand drew a mask of the neuropil with TrackEM (and smoothed it by applying a Gaussian filter and thresholding) in order to identify signal inside and outside the neuropil. This gave a neuropil volume of $1.8 \times 10^6 (\mu m)^3$. As a guide to the reader, and especially due to the previously

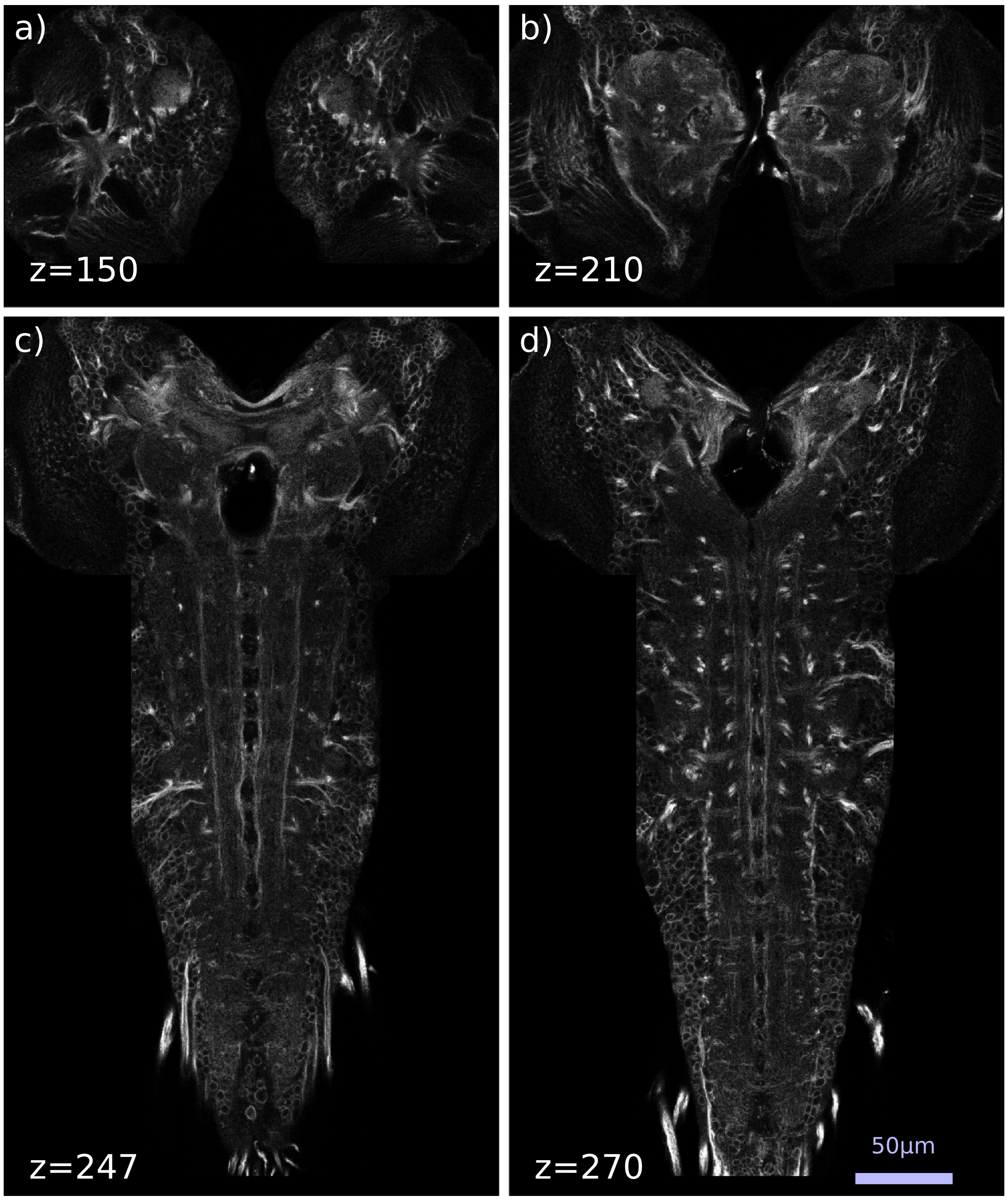


Figure 3.9: The template
z-slices through the template (a) and b) did not contain VNC neuropil). Note that the caudal end of the VNC just fits inside the imaged volume, so some extra empty volume was added in order to keep any extra signal here in the registered image (as CMTK will restrict registered images to the size of the template space). Note the lateral limits of the imaged VNC stack given by the black regions in the cortex either side of the thoracic region.

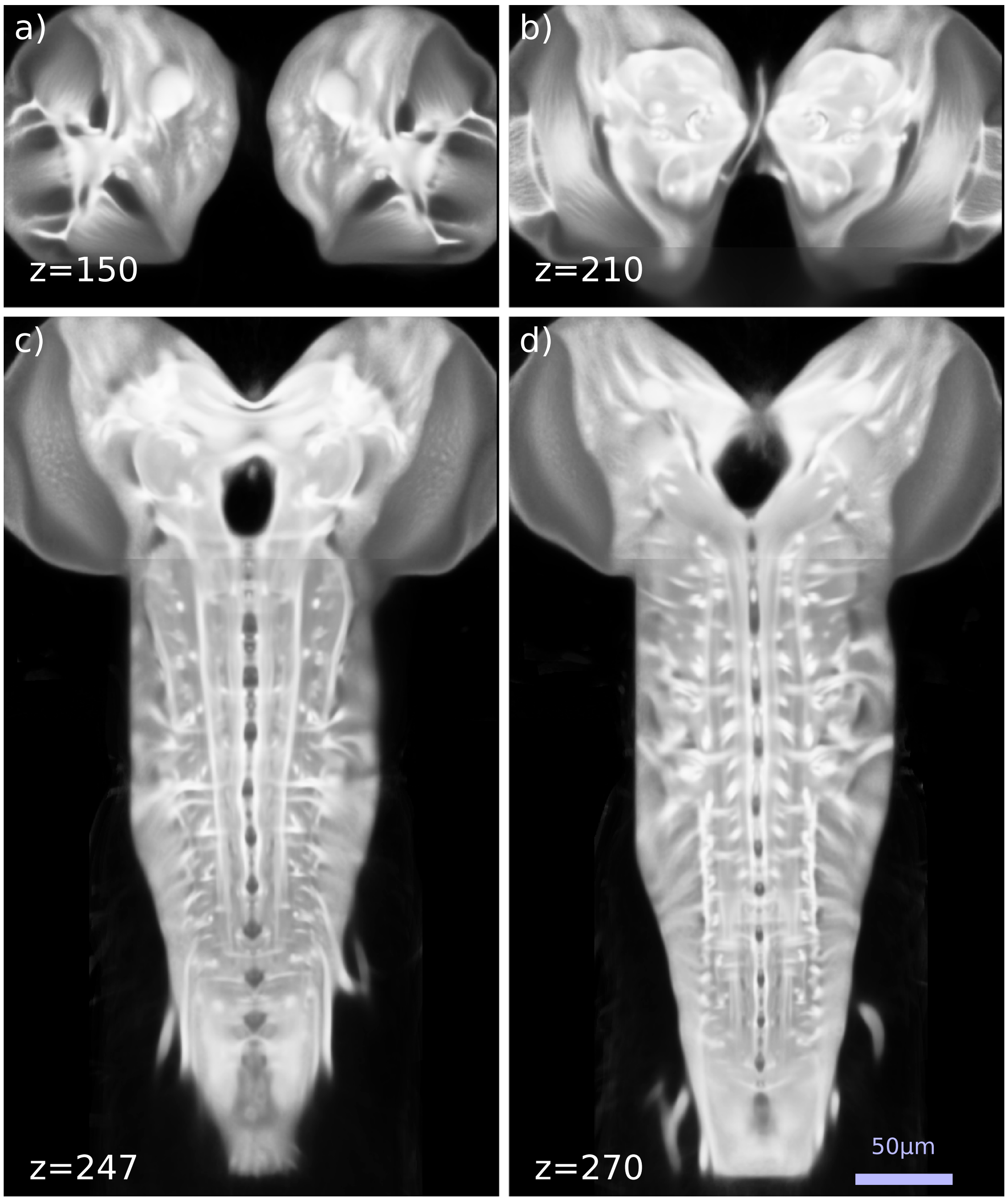


Figure 3.10: Average of registered image stacks
z-slices of average intensity of 1000 registrations in the template space. Registrations had an average neural volume registration accuracy score (see section 3.7) over 0.6.

mentioned distortion of the CNS *in vitro*, the relationships between the biological axes and those of the template image space are summarised in table 3.2. The total template size was $(982 \times 1760 \times 365)$ voxels. I split the template into CB and VNC with the following parameters:

- CB template bounding box: $[0:977, 0:801, 5:341]$, size: $(977 \times 801 \times 336)$ voxels.
- VNC template bounding box: $[170:830, 480:1760, 193:351]$, size: $(660 \times 1280 \times 158)$ voxels.

This enabled quicker registrations as the template image space was reduced and also improved the initial alignments along the principal axis, particularly for stacks of the CB as this would otherwise be rotated 90 degrees out of place.

3.4.4 *Contrast Adjustment*

Image contrast was highly variable among the source images and, since it has been reported that contrast normalisation can improve the quality of the registrations when this is the case Foley et al. (2016), I adjusted the contrast of the whole image reference channel. To do this I implemented histogram equalisation by creating a monotonic function to map the intensity values in the reference channels of an image to new values such that the resulting cumulative intensity histogram would be close to linear. More precisely, this function was the normalised cumulative histogram of the image background above zero, linearly scaled to range from zero to 255 (see figure 3.11). This relied on the background being the lowest intensity value which was almost always the case. The signal intensities were not adjusted. The contrast normalisation was found to improve the overlap score by a very significant 22.9%. I also explored a gamma-correction Jähne (2004) but this did not improve on these results.

3.4.5 *Smoothing*

It has been reported that smoothing of the reference channel can also improve the registration Court (2017), as it reduces the local minima effect of fine-grain but high intensity perturbations found at high image resolutions. I therefore smoothed the reference (see figure 3.12). I found that using a float-precision Gaussian filter with standard deviation of one voxel width in each dimension and rounding the result to the nearest 8-bit value, improved the overlap score over non-smoothing by 1.8%.

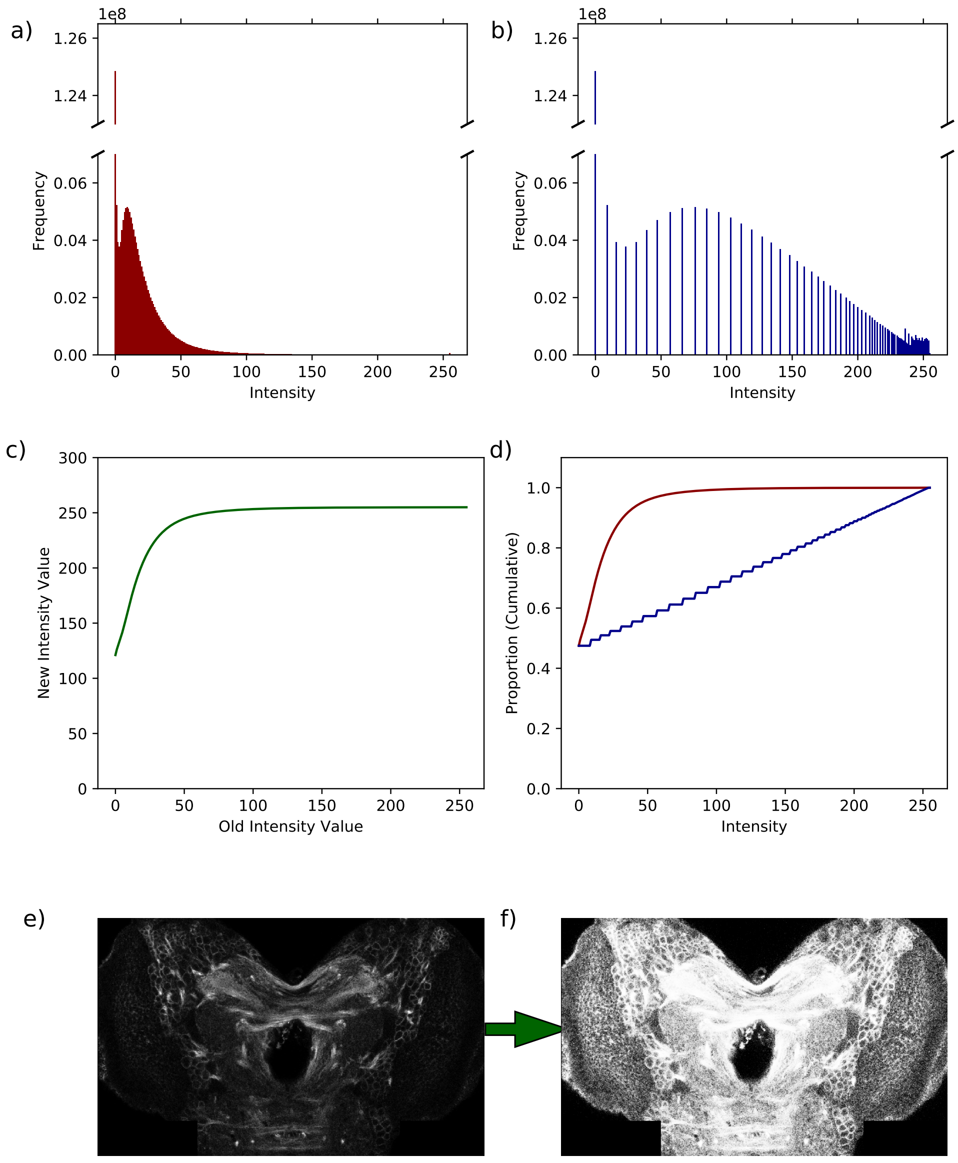


Figure 3.11: Illustration of contrast normalisation including image intensity histograms
Intensity histogram a) prior to and b) following normalisation. c) Mapping function from old to new values (red curve in d) multiplied by 255). d) normalised cumulative intensity frequency graphs pre (red) and post (blue) normalisation. Coronal CB slice of template e) pre- and f) post-normalisation.

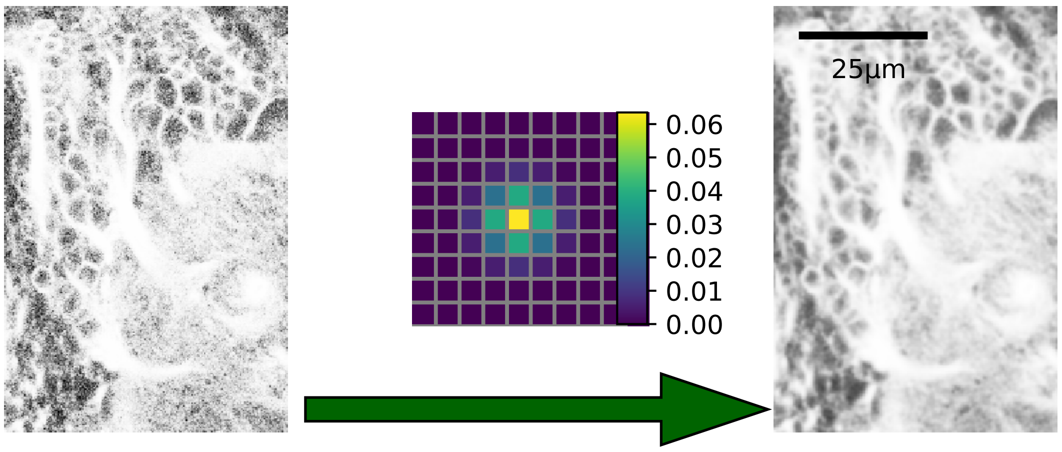


Figure 3.12: Gaussian smoothing of image pre-registration
Demonstration of the effect of applying the Gaussian kernel in the centre to the left section of coronal slice of CB, to produce that on the right. The grid on the Gaussian represents individual voxels.

3.4.6 *CMTK Similarity Metric*

The similarity metrics in CMTK do not rely on absolute differences in intensities at corresponding points x in the template image (X) and y in the float image (Y). Instead the metrics calculate how determinable the intensity at y is given its value at x. This is in order to match up the same image pattern and features, despite global shifts in intensity values.

I varied the metric of the two images that was optimised with CMTK using the options available in the software. Firstly I evaluated Mutual Information (MI), $I(X, Y)$. This is the information about intensity values of one image that can be inferred from those of another image Maes et al. (1997) (and *vice-versa*) as calculated by the standard formula:

$$I(X,Y) = \sum_{x,y} p(x,y) \log_2 \left(\frac{p(x,y)}{p(x)p(y)} \right) \tag{3.6}$$

The probabilities above are calculated empirically by taking the histograms of the intensity values and dividing them by the total number of points, with a 2D histogram for $p(x, y)$. Considering the logarithm, MI is thus higher if there is more variation of the 2D histogram from what would be expected from the individual histograms - those that give $p(x)$ and $p(y)$ - were they completely independent (and not similar).

Hence CMTK aims to maximise the MI.

Normalized Mutual Information (NMI) Studholme et al. (1998) is similar to MI except for that it divides the MI by the Joint Entropy, $H(X,Y)$, or the total amount of information about intensity values contained in the two images:

$$H(X,Y) = \sum_{x,y} -p(x,y)\log_2(p(x,y)) \tag{3.7}$$

$$NMI(X,Y) = \frac{I(X,Y)}{H(X,Y)} \tag{3.8}$$

Maximising NMI has been shown to be better than MI in cases where partial image overlap is the case (such as in my project), since it reduces the tendency to increase the total mutual information by simply increasing the size of the overlap Studholme et al. (1999). This was not the result for my study, however, where the overlap score for MNI was 7.9% worse than for MI.

Finally, the Correlation Ratio (CR) Roche et al. (1998) works by *minimizing* the total, over all x intensity values in X, of the dispersion among the values in image Y for all voxels with the corresponding x intensity value. This is achieved by minimising the variance of $y - \bar{y}_x$ across all y and x where $\bar{y}_x$ is the mean (or *expected value*) of intensity in Y of voxels with corresponding intensities x in X. This is equivalent to *maximising* the correlation ratio, η, such that the sum across all x intensity values of the variances (with respect to the mean intensity in Y) of the *expected* intensity of Y given an intensity in X:

$$\eta^2 = \frac{\sum_x n_x(\bar{y}_x - \bar{y})^2}{\sigma_y^2} \tag{3.9}$$

where n_x is the number of values of intensity x in image X and σ_y^2 is the overall variance of intensity values in Y.

This technique contrasts to MI and NMI in that they do not consider dispersion explicitly, instead just pairwise correspondence between intensity values in each image. The division by the variance in Y prevents a solution that minimizes the variance by reducing the overlap region. Note that the value is different depending on direction, i.e. which image (template or float) is assigned to which letter (X or Y). CMTK appears to implement the CR by summing the two directions as there is no reason one should

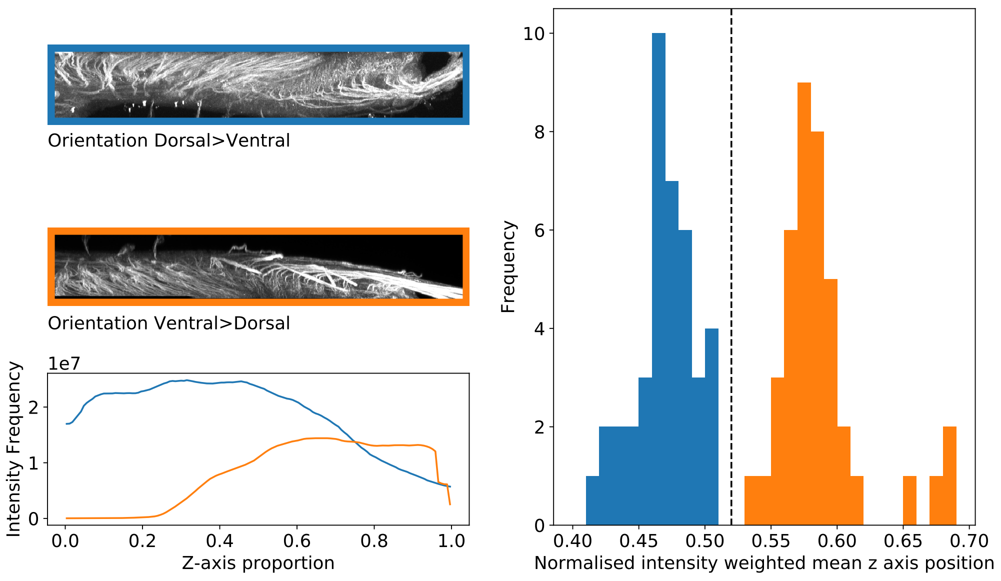


Figure 3.13: VNC dorso-ventral orientation determination
Left: Example maximum sagittal projections of VNC stacks for the two orientations and plots of their axial intensity sums as a function of z-axis position. Right: Histograms of the test statistic for the training set.

predominate in image registration.

As shown in table 3.1, the correlation ratio performed the best of these metrics in the detailed evaluation and was used in the registration pipeline. This was consistent with what I expected from my visual inspection of overlays of the registered reference channels to the template in the initial tests.

3.5 VNC DORSO-VENTRAL ORIENTATION

The orientation of the ventral-dorsal (z) axis in many VNC image stacks in the dataset (approximately 25%) was inverted with respect to the template. These stacks required inverting to enable registration since the principal axis alignment would not be able to rotate the image through 180 degrees, leading to failed registrations. In any case, rotation rather than inversion would be incorrect since this would invert the left and right sides of the brain, but the stacks were consistently imaged with the ventral side of the CNS on the base of the microscope slide without x-axis inversion.

To automate this process, I developed a simple algorithm to attempt to separate the two sets, utilising a training set of VNCs with 40 oriented such that increasing z moved from the dorsal side towards the ventral side (denoted Dorsal>Ventral) and 40 with the converse (Ventral>Dorsal). The algorithm calculated the sum of intensity for each z-slice. The mean z-position, weighted by these intensity sums, was normalised by total z length to give a test statistic, t:

$$t = \frac{\sum_z z \sum_{x,y} I(x,y,z)}{n_z \sum_{x,y,z} I(x,y,z)} \tag{3.10}$$

where I(x,y,z) is the intensity at point (x, y, z) and n_z is the number of z-slices. Those stacks with values of t below 0.52 were inverted in the z-axis (not rotated) to ensure all were consistently Dorsal>Ventral in the z direction. Figure 3.13 shows that for the training set it was possible to completely separate the two groups. I validated my method using a test set of stacks randomly selected from the remainder of the data set (n=30,Ventral>Dorsal:4,Dorsal>Ventral:26), with 28/30 orientations correctly identified, with one false positive for required inversion and one false negative.

A much smaller number of CB stacks were not consistent (approximately 5%) and, as no similar simple algorithm was forthcoming and it was observed that the resultant registrations were usually easily identifiable as very bad, I decided to identify these abnormally oriented stacks following registration and re-run them through the pipeline inverted.

3.6 NEURAL VOLUME SEGMENTATION

The signal channels commonly contained clearly separate neurons and multiple neurons that were intertwined with one another where it was difficult to determine visually which signal belonged to which cell, especially in regions of high arborisation. For the database I required images to be available containing just a single cell of a specified type so I needed to extract these from the registered images. A single threshold was able to create a binary image of components that represented each neuron or intertwined group of cells. Due to differences in image contrast, additional dim regions in the signal and dim neuropil background that was often not zero, it was necessary to vary between image stacks the intensity threshold deployed to separate the signal from background. To automate this process I found, based on experiments with the test group of stacks, a method to calculate the threshold as follows (bearing in mind the sparse nature of the signal):

Signal Segmentation Threshold: *The highest intensity value at or above 10, such that higher would leave less than 0.1% of the total registered image stack voxels.*

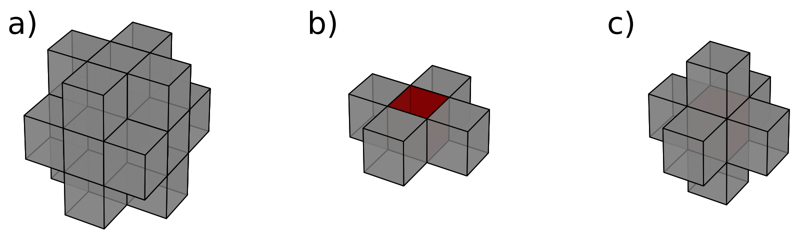


Figure 3.14: Voxel connectivity
Illustration of connectivity based on neighbours of the central, red voxel: a) 18-nearest neighbours, b) 4 immediate neighbours in x-y plane only, c) 6 immediate nearest neighbours.

A binary image was thus created with all values at or above the threshold set to 1 and all below 0. Next, an initial cleaning step was applied: All binary signal voxels were assigned to a connected component where a voxel is in the same component as all of its 18 nearest neighbours that also contain signal ("18-connected" - the $3 \times 3 \times 3$ cube surrounding a signal voxel minus the corner voxels, see figure 3.14a). Small components, 100 voxels or less, were removed as they mostly represented speckles of signal not part of the stained cells. The remaining binary signal was then dilated such that if one of a voxel's 4-nearest neighbours in x-y is signal (figure 3.14b), it becomes signal. This was followed by a dilation based on the 6-nearest neighbours in all dimensions (figure 3.14c). These two dilations were iterated over three times, with the discrepancy designed to account for the approximate difference in voxel sizes in each dimension without the high computational costs of a Gaussian filter for each connected component. This was done to include any nearby low intensity neural signal as well as fill in small holes and connect signal across small gaps that might have occurred due to weak staining - especially in thin neuronal fibres.

Finally, *neural volumes* were identified: separate volumes in the image stack that contained part of or an entire cell or combinations of instances of these. The signal was split into these neural volumes by finding connected components as above but for the expanded binary image (see figure 3.15). Only components of 50 000 voxels or over were designated as neural volumes in order to remove cells only partially in the CNS or image space, extraneous material or other small artefacts such as those in figure 3.16.

The average size of these neural volumes over 50 000 voxels was 317 000 voxels (see histogram in figure 3.18a) and the average number of neural volumes per image

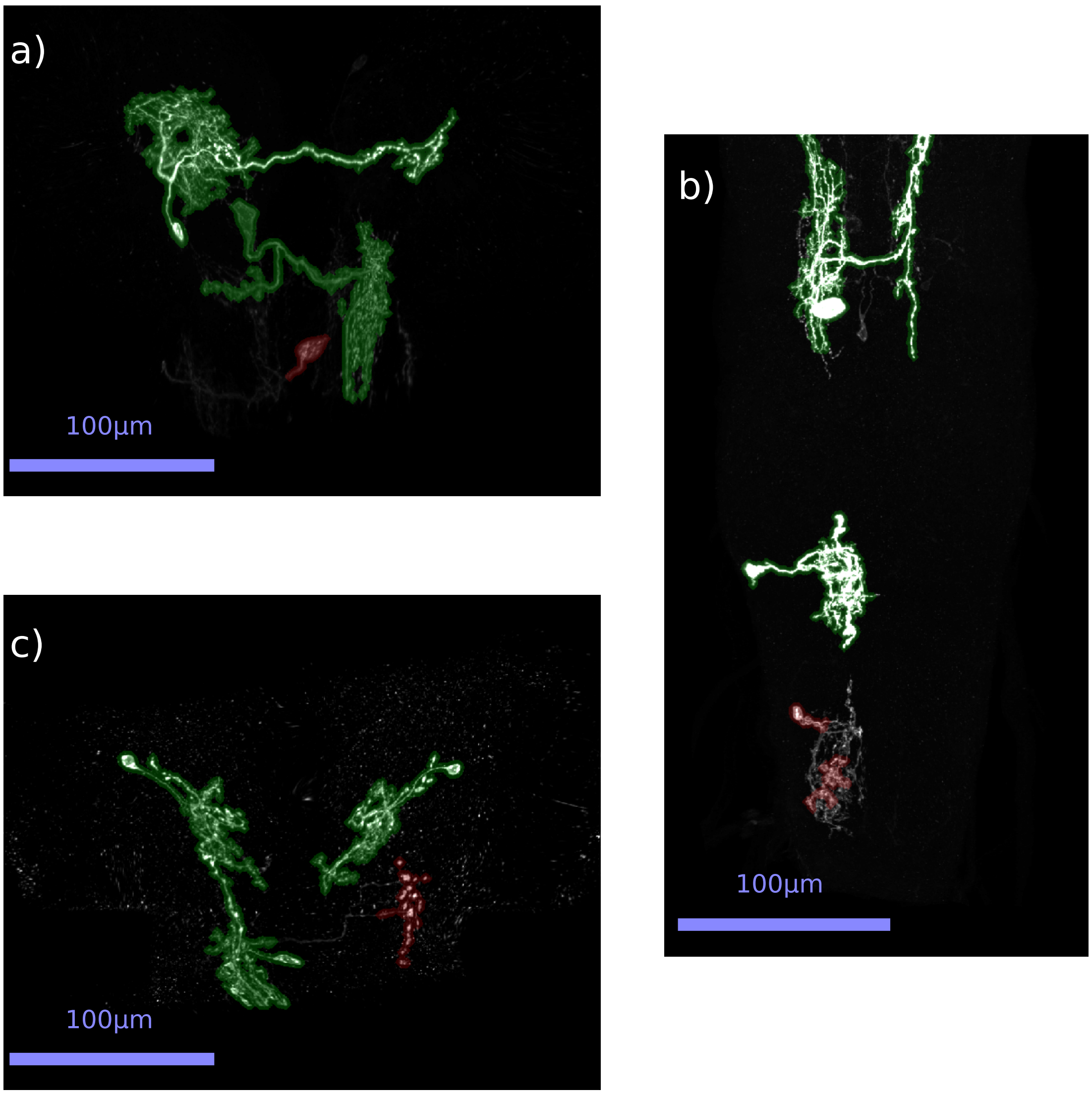


Figure 3.15: Successful segmentation and neural volume examples
Maximum z-projections of contrast-enhanced signal stacks identifying neural volumes imposed on signal that were either kept (green, over 50 000 voxels) or not kept (red, here over 20 000 voxels but under 50 000). Note that in c) the left neural volume includes two cells with the lower side truncated by the original image (the range of which is noticeable by the speckled background), and in b) the top cell is cut off by the template. The remaining green volumes are whole, single cells.

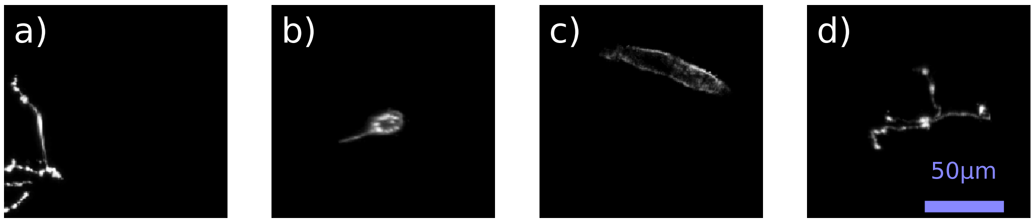


Figure 3.16: Examples of removed small signal volumes
Maximum z-projections of contrast-enhanced neural volumes: a) partial track cut off by image edge b) partial staining of cell body (these often occur in cells outside the neuropil) c) part of non-neural structure emanating from CNS d) small amount of neural arborization.

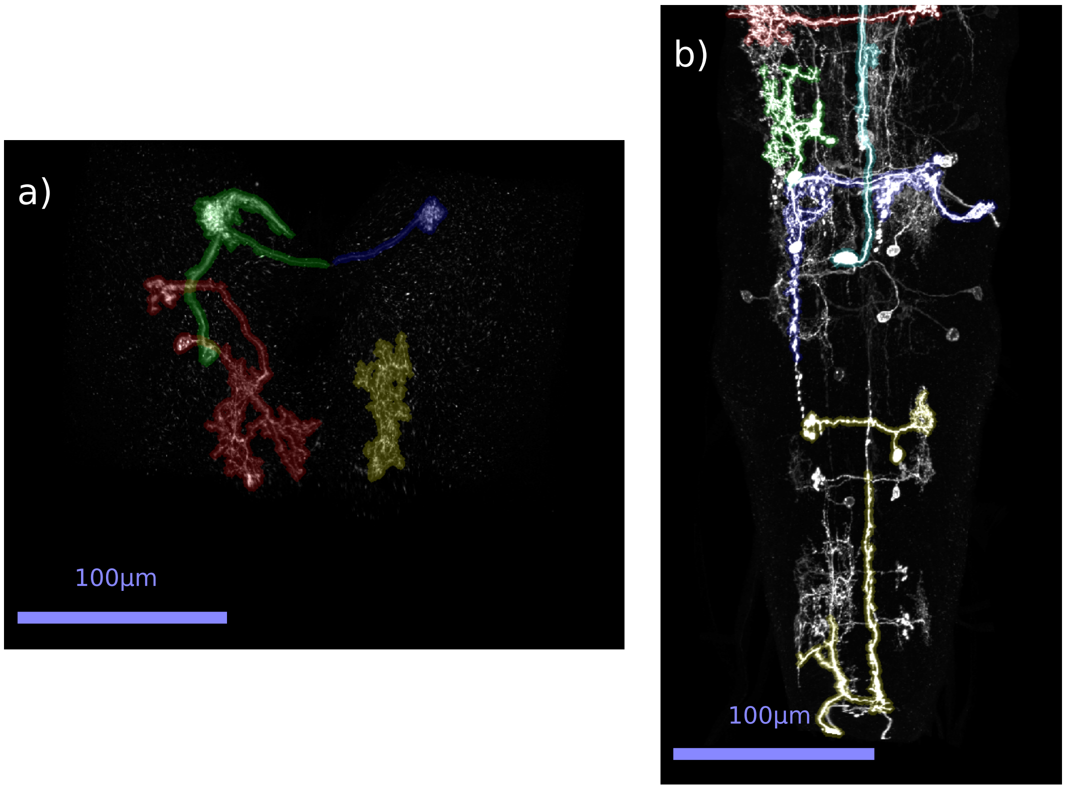


Figure 3.17: Limitations of segmentation process
Maximum z-projections of contrast-enhanced signal stacks identifying neural volumes with different colours imposed on signal to differentiate volumes. a) this CB segmentation fails due to the missing signal in the commissure splitting a single cell into two components. b) This VNC segmentation is typical of VNC failures due to the large amount of signal meaning that this stack is not suitable for segmenting into single cell volumes (the threshold intensity was 81).

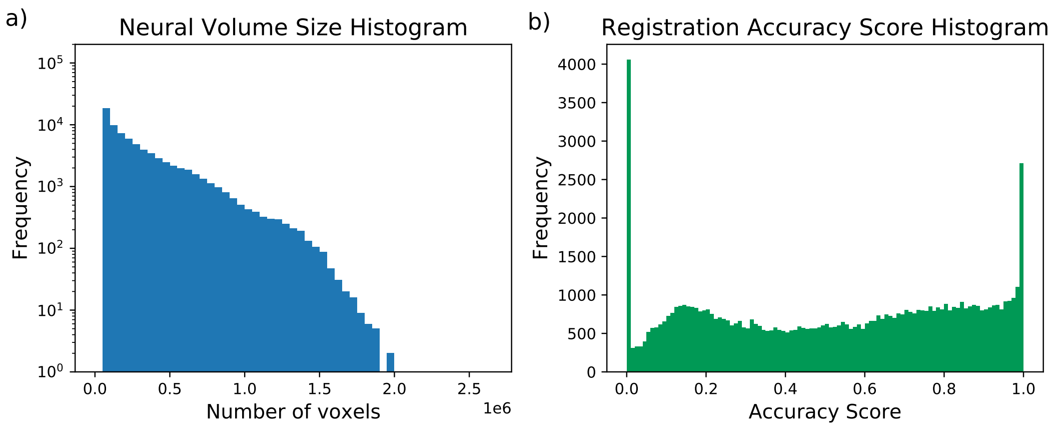


Figure 3.18: Histograms of number of voxels and accuracy scores per neural volume Only neural volumes over 50 000 are shown. Note the logarithmic y axis for a). The accuracy score is detailed in section 3.7.1.

stack was 3.56. The segmentation did not always work well, for example weak or missing staining meant cells could sometimes become split, as in figure 3.17. Some stacks were very busy and meant that the threshold was calculated to be far too high meaning many parts of cells were missing. As a result, and because I am mostly interested in sparse, single-cell neural volumes, all neural volumes in stack channels with thresholds over 75 were excluded from subsequent analysis (1500 stack channels or 3.4%, over 90% of these VNC).

3.7 REGISTRATION ACCURACY EVALUATION

I have already demonstrated how I assessed the success of individual registrations by comparing the float and template reference stacks by visual inspection. Due to the large number of stacks and subjective nature of this method, I aimed to automate the selection of well registered cells for inclusion in the database. CMTK does not output an estimate of the degree of error in the registration result so I thus needed to implement and evaluate my own metric to do this.

As the registration quality often varied across image stacks, and neural volumes usually only inhabit a small subregion of the stack, it was desirable to calculate a registration accuracy score for each neural volume rather than the whole signal volume. It was also desirable that each neural volume would receive an easy to interpret map that indicated how well it was registered at each point, in case researchers using the registered neural volumes wanted to judge accuracy of their own deductions that relied on the quality of the registration of certain parts of the cells.

I evaluated different scoring methods by comparing the different registration accuracy scores to accuracy estimates for neurons from the test set used in section 3.4.2 (for which I calculated the overlap scores). The accuracy estimate for a single cell was the mean pair-wise overlap score (in both directions, denoted by $\bar{O}_i$) between this cell (i) and all other cells of the same type (j):

$$\bar{O}_i = \frac{\sum_{j \neq i} (O(i,j) + O(j,i))}{2(n-1)} \tag{3.11}$$

where n is the size of this cell type's subgroup. I judged the accuracy scores by plotting them against the mean overlap score for each neuron and correlation strength was measured using the Pearson Product-Moment Correlation Coefficient, r (see figure 3.19). I did this for all cells together and also individually for each cell type subgroup. Three cells were excluded from the analysis, each from different subgroups, as they registered so badly that no signal was present within the neuropil mask which meant that it was not possible to calculate an accuracy score for them.

As an initial baseline and for simplicity, I first calculated the magnitude of the difference between each registered float reference voxel intensity, I_F, and its corresponding voxel intensity in the template, I_T. For each cell, the accuracy score was the average across all the voxels in the binary neural volume (n_v) that were also in the binary neuropil mask, M:

$$\frac{\sum_{x,y,z} n_v(x,y,z) M(x,y,z) \left| I_T(x,y,z) - I_F(x,y,z) \right|}{\sum_{x,y,z} n_v(x,y,z) M(x,y,z)} \tag{3.12}$$

There was a weak negative correlation in the overall data ($r = -0.43$) and the mean of the within-subgroup correlations was -0.59 (figure 3.19a).

Next I calculated a Correlation Ratio for each cell using the voxels within the neural volume and the neuropil mask, taking the sum of the individual ratios in each direction (template > float, float > template). There was no correlation between the two scores, which implies that this measure has already been maximised to a similar degree across the image stacks (figure 3.19b). This is not clear from the design of CMTK, which should optimise metrics to local minima, whatever they are. It is perhaps a reality of including a bending energy.

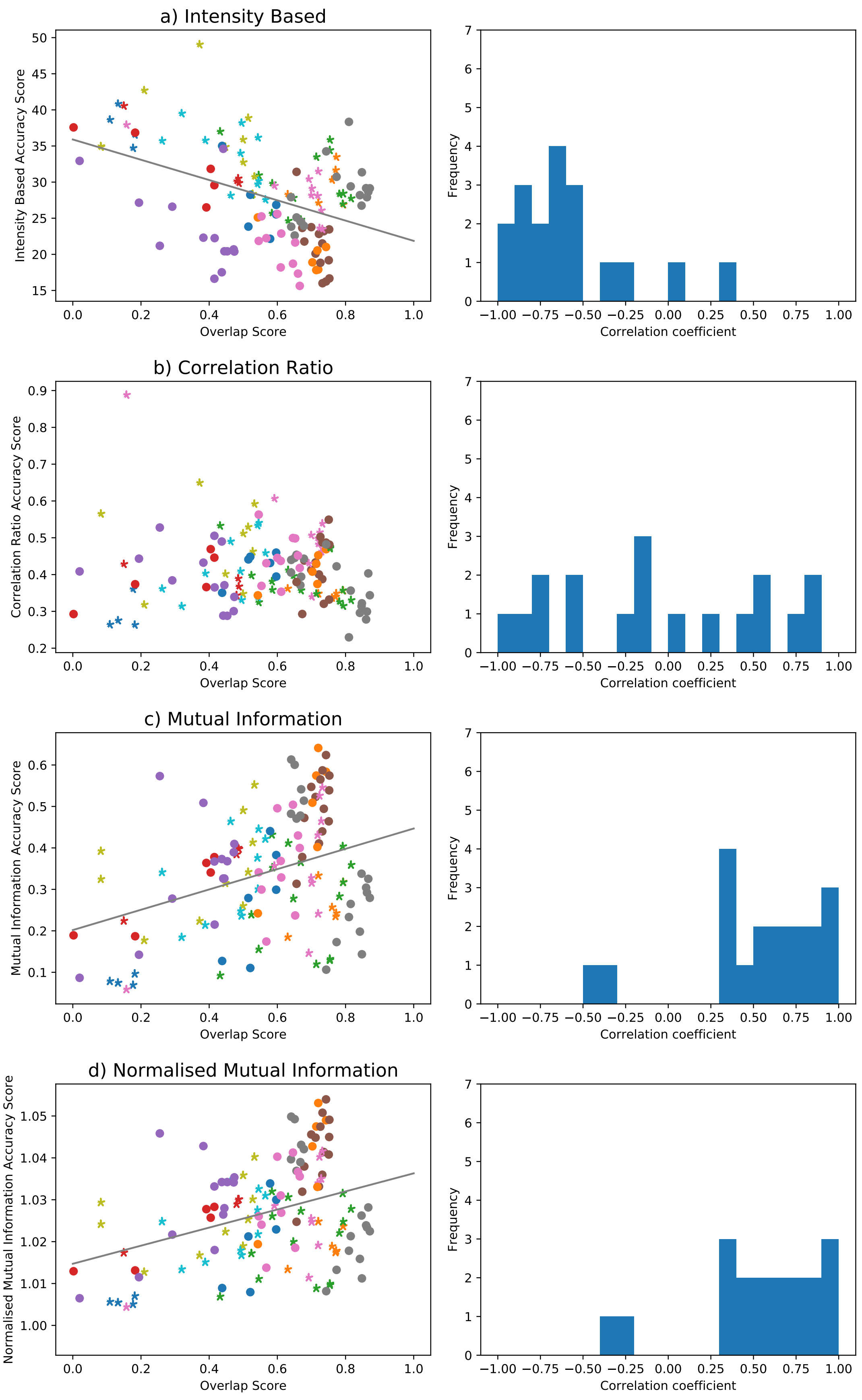


Figure 3.19: Correlations between overlap and unsatisfactory accuracy scores Right: Scatter graph of mean overlap score with similar cells ($\bar{O}_i$, equation 3.11) vs. specified accuracy score. The Correlation Coefficients, r are as follows: a) -0.43, b) -0.12, c) 0.35 and d) 0.36. Colours and shapes represent cells of the same type (sub-group). Stars represent VNC cell types, circles CB cell types. Left: Histograms of individual sub-group Correlation Coefficients (n=18).

Following this, I investigated the other CMTK metrics, mutual information and the normalised mutual information (see figures 3.19c and 3.19d). Here I calculated the probability distributions from only the voxels in the neural volume in the neuropil. I then took as the metric the mean values across these voxels to normalise the volumes by size. This improved the number of cell subgroups with strong correlations over the CR, (mean of subgroup correlations 0.53 and 0.55 for MI and NMI respectively), but many lines of best fit were very steep and thus very dependent on the cell type under investigation. This meant a universal cut-off could not be applied easily and indeed this was reflected in that the overall correlation was weak, at 0.40 for both MI and MNI respectively. I also experimented with using the probability distributions for CR, MI and NMI created from voxels across the entire image but the results were no better. These results therefore serve as a starting point for further research.

3.7.1 *A New Accuracy Score*

I attempted to devise a new metric that I could apply to the problem of determining the accuracy of the registrations. Inspired by the good performance of the intensity differences above, I investigated looking at the differences in *normalised* intensity between two *blocks* of voxels centred on a single voxel (see figure 3.20b). Rather than include all voxels in the block however, I sampled regularly from a 3D grid within the block (see figure 3.20d). The normalisation of the samples was carried out by subtracting the mean and dividing by the standard deviation. I then found the total squared difference in normalised intensity between voxels at corresponding positions in the identically sized block grids surrounding the specified central voxels in the float and template. As such the pairwise block similarity metric, S_B, between two voxels at positions (x_T, y_T, z_T) in the template and (x_F, y_F, z_F) in the registered float were given as follows:

$$S_B = \sum_{i=-c_x}^{c_x} \sum_{j=-c_y}^{c_y} \sum_{k=-c_z}^{c_z} \left(\frac{I_T(x_T + d_x i, y_T + d_y j, z_T + d_z k) - \mu_T}{\sigma_T} - \frac{I_F(x_F + d_x i, y_F + d_y j, z_F + d_z k) - \mu_F}{\sigma_F} \right)^2 \quad (3.13)$$

where $d_x, d_y, d_z \in \mathbb{Z}$ represent the sampling intervals of the grid and $c_x, c_y, c_z \in \mathbb{Z}$, the multiples of grid samples the block extends out from the central, given point in each direction. σ_T and μ_T represent the mean and standard deviation of the intensities in the template block respectively (σ_F and μ_F likewise for the registered float block). Lower scores indicate better matching and hence registration. To test this met-

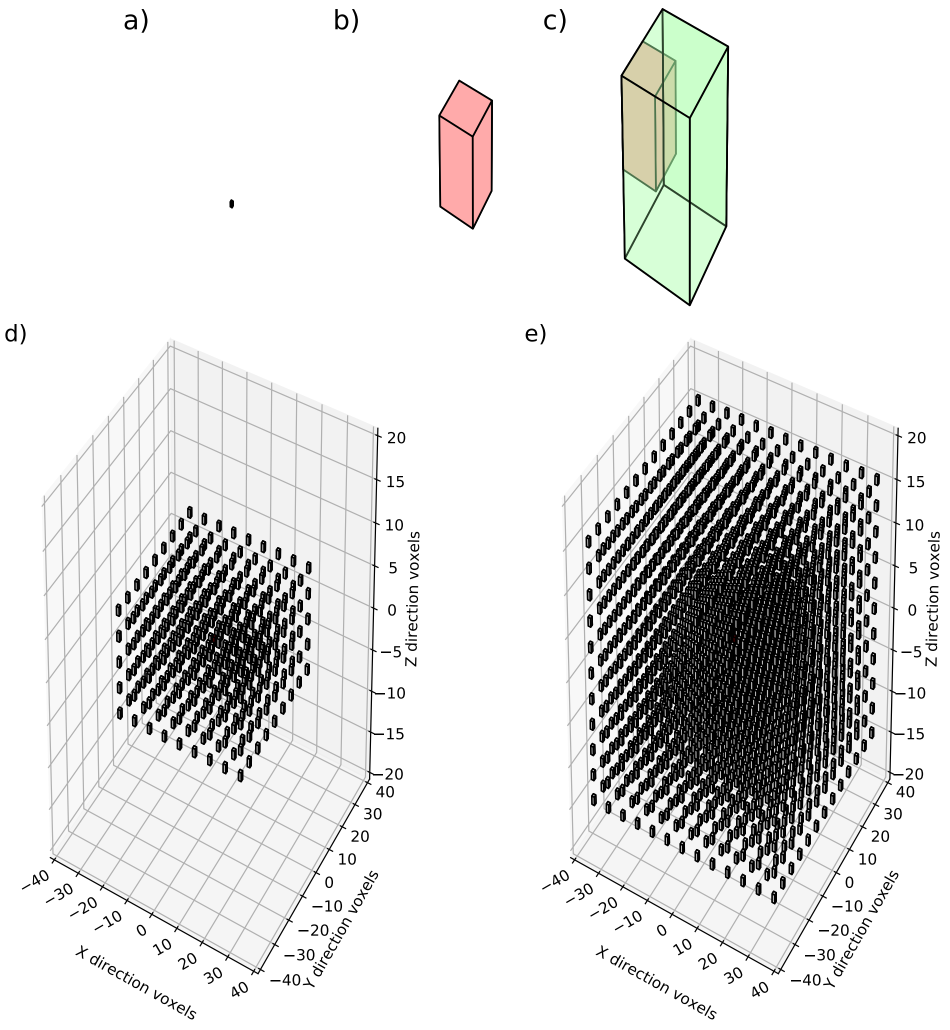


Figure 3.20: Explanation of implemented registration accuracy scoring method a)-c) to-scale representations showing a) a single voxel, b) and c) the red surrounding block of voxels used for pair-wise comparisons and c) the green exploration region in the template space for optimum pair-wise comparison score. d) and e) to-scale representations of the grids of voxels sampled for d) the pair-wise block comparison score between two voxel points - the red cuboid in b) and c) - and e) the neighbourhood exploration in template used in finding the shift from float voxel position - the green cuboid in c).

ric I calculated it between corresponding voxels (i.e. $x_T = x_F$, $y_T = t_F$ and $z_T = z_F$) in the neuropil mask of the registered neuron volume and found the mean average score. These results appeared promising and are summarised later in figure 3.22a for comparison to the final implemented method.

This work also further inspired the eventual method I employed, where it is important to recall that I also ideally wanted a clear way to judge the degree of inaccuracy that was informative to researchers. The previously discussed metrics may be able to give a relative, most likely non-linear indication of the degree of error (e.g. the magnitude of the intensity difference) but this is not clear and the values not easy to interpret. I decided, therefore, to investigate another method, detailed below, that estimates the distance required to shift each point to its optimum, independent of smoothness constraints or other regularisation, based on the block similarity metric S_B. This is predicated on the idea that the broad surroundings of matching voxels should still be more similar than non-corresponding ones even despite deformations in the surrounding structure.

Rather than just calculating the S_B score between each float voxel and its positionally corresponding template voxel, I calculated the S_B score between a float voxel and all template voxels in a surrounding, sparse 3D grid around the corresponding template voxel (similar to the block grid - see figure 3.20d). The final *shift score* for a single point in the registered float image is then given as the square of the Euclidean distance (in the original template space) from the current corresponding template voxel's position to that in the surrounding grid that has the lowest S_B score.

The downside of using the shift score is that this makes the computation much more expensive. As a result, I decided to only calculate scores themselves for voxels in this sparse grid, with Gaussian smoothing used to interpolate between these points. Setting the sparse grids applied to both the surrounding block and the search space to be equal can speed up calculations as it makes intermediate calculations reusable. This effectively replaced the template x_T, y_T, z_T with a varying position of the form $x_F + s_x d_x, y_F + s_y d_y, z_F + s_z d_z$ where $s_x, s_y, s_z \in \mathbb{Z}$

By employing the dynamic programming ethos of re-using as many sub-calculations as possible, I was able to heavily speed up the calculation as follows: Firstly, formula 3.13 can be rearranged to give:

$$S_B = \frac{1}{\sigma_F \sigma_T} \left(\sum_{i=-c_x}^{c_x} \sum_{j=-c_y}^{c_y} \sum_{k=-c_z}^{c_z} \sigma_F^2 I_T^2 - 2\sigma_F \sigma_T I_T I_F + \sigma_T^2 I_F^2 \right) + \frac{n_c}{\sigma_F \sigma_T} \left(-\sigma_F^2 \mu_T^2 + 2\sigma_F \sigma_T \mu_F \mu_T - \sigma_T^2 \mu_F^2 \right) \quad (3.14)$$

with n_c the number of points in the block. For each comparison block around each grid point in the template I could pre-calculate the mean, standard deviation, and mean squared intensity. This left the sum of the multiple of the two intensities at each point (the $-2\sigma_F \sigma_T I_T I_F$ term) as the only one that would vary for each float-block and template-block combination and was hence the main time consuming item in the calculation. I was, nevertheless, still able to speed this up by calculating all shift scores in a cuboidal Region Of Interest (ROI) for each offset direction at a time. I performed a running calculation that I modified by adding the values for new points in the block and subtracting the old ones as the centre of the block was moved along the x and y axes. The cost of calculating scores for more points (a whole cuboidal ROI) rather than just the points in the neural volume was more than offset by reusing calculations in the spirit of dynamic programming.

Finally, the calculation was sped up by a factor of over 100 when I implemented the above in the C programming language over python, due to python being an interpreted language and the many loop levels required in the above algorithm which slows python down significantly. Although the code was written such that it was easily parallelisable by splitting the ROI into subsections, it was not in practice run in parallel since I was able to run the calculation on many different registered image stacks in parallel.

The grid spacing and sizes were optimized using a training set of artificially warped template image stacks. This was created by using a known warp field to map each voxel in the new float space to a voxel in the template space to take that voxel's intensity value. The warp field was generated by summing various sizes of random Gaussian shifts centred across the template space (see figure 3.21) with the highest shifts being around 20μm. This gave me artificial float images that looked similar to the registered stacks that I used to optimise the d_x, d_y, d_z spacing parameters, block size c_x, c_y, c_z parameters and exploration shift parameters (s_x, s_y, s_z ranges). For each float image, I calculated the shift score vector to optimum location for each voxel in

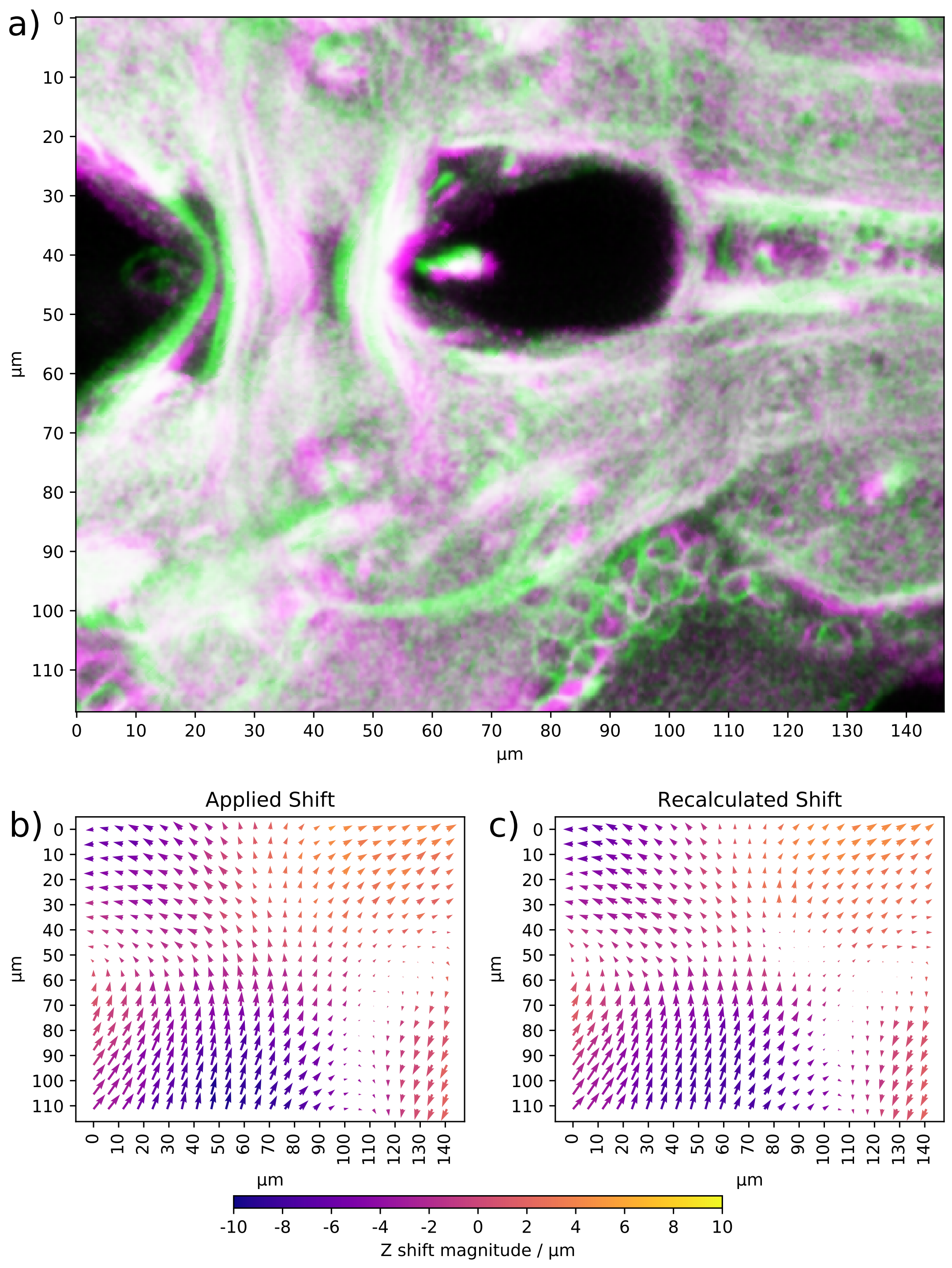


Figure 3.21: Random, known warp applied to template
a) Shows a coronal z-slice through the CB of overlapping positions of the template (green) and artificially warped template image (magenta). In b) the arrows indicate the direction and magnitude of the field in the X-Y directions applied to float positions from which to extract template intensity values to create the artificially warped image above. The colour of the arrows corresponds to the shift in the Z-direction. c) is the re-calculated field using the shift scores for each point. Note the strong similarities between b) and c) except in the low-varying intensity region of the foramen.

the neuropil mask and then the magnitude of the distance between this shift score vector and the actual shift vector. I averaged this distance over all within-neuropil voxels and then calculated the mean and standard deviation for all 7 artificial floats to produce final test statistics for choosing parameters. I chose the $x : y$ ratio $1 : 1$ for all parameters and the grid spacing $x : z$ ratio roughly $1.707 : 1$, in line with the spatial ratios. This effectively reduced the number of parameters to 5.

I chose the parameters with both the lowest mean and standard deviation that also completed in a reasonable amount of time (about 1 hour for each stack). These parameters were to work on a grid of voxels with spacing of $[d_x, d_y, d_z] = [6, 6, 3]$ in each dimension, where blocks of half-width $[c_x, c_y, c_z] = [4, 4, 2]$ (on this grid) were compared in a local cuboidal range about the central, correspondingly positioned template voxel of -6 to 6 grid points on the $x - y$ grid and -5 to 5 on the z. This effectively gave an exploration space of [73,73,31] voxels.

The shift accuracy score *for each neural volume* was calculated as the proportion of voxels it contained that had a square rooted shift score below 2.5μm. I chose this measure and value as it gave a broad distribution of accuracy scores and the results correlated well with the mean overlap scores (see below). It thought to be reasonable to require a set percentage of the registration to be within this tolerance to enable the cells to be identified by type. Ultimately, however, this had to be verified by a successful search system (the subject of chapter 4). Figure 3.18a shows the range of the neural volume accuracy scores, for which the mean was 0.70.

The results demonstrated an overall, strong correlation of 0.52 with the overlap score (figure 3.22b), with the average correlation for each cell 0.56. 12/18 were over 0.6 i.e. strong or very strong correlations. The outlier with a negative correlation (-0.78) contained only badly registered cells with low accuracy and low overlap scores meaning the overlap scores were probably not relatively representative, giving a meaningless trend line. Figure 3.23 gives further examples of the strong trends within subsets. The shift map for some cells in an example type subgroup is illustrated and visually inspected in figure 3.24 whereas figure 3.25 investigates two of the outliers in the general trend curve. This relatively strong performing accuracy metric, coupled with a meaningful error signal map was what caused me to use it to automate the selection of registrations for inclusion in the database of cells.

Based on the distribution of the scores seen in figure 3.22b and analysing the reference channel overlap by eye, I decided in conjunction with collaborators (JT, RC) that

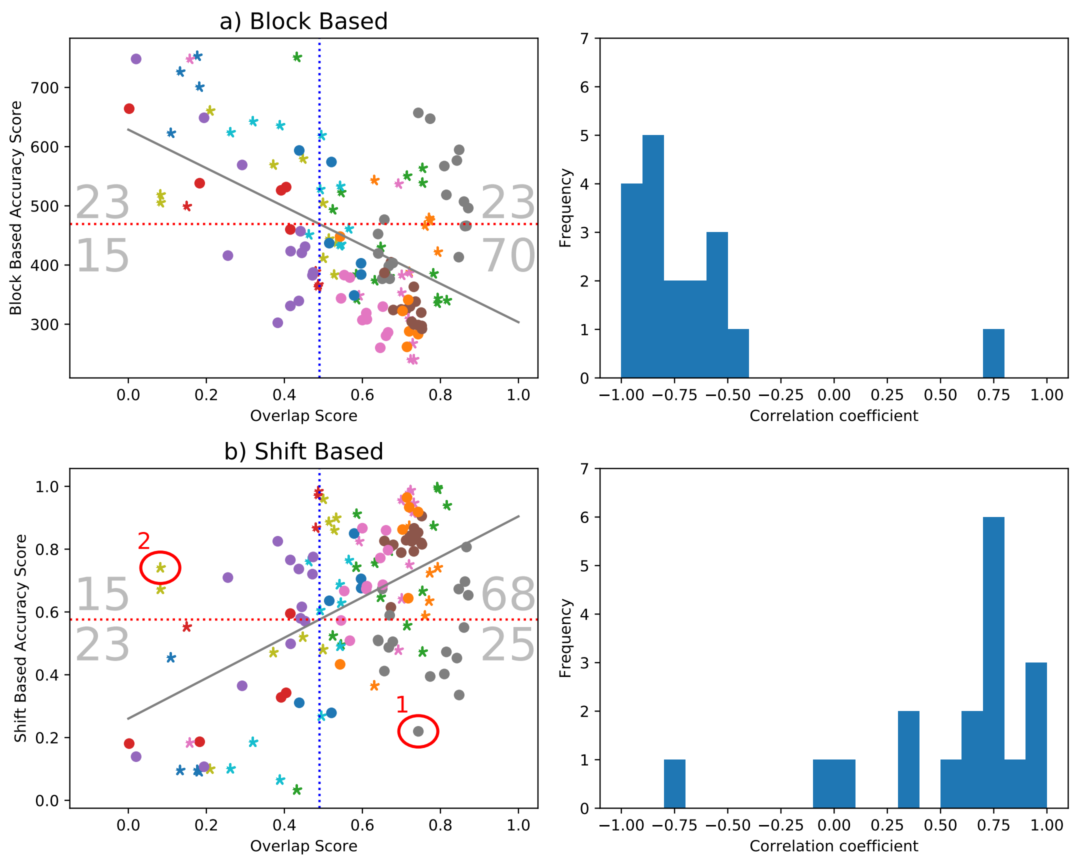


Figure 3.22: Correlations between overlap and implemented accuracy score Right: Scatter graph of mean overlap score with similar cells ($\bar{O}_i$, equation 3.11) vs. specified accuracy score. The Correlation Coefficients, r are as follows: a) -0.53 and b) 0.52. Colours and shapes represent cells of the same type (subgroup). Stars represent VNC cell types, circles CB cell types. Left: Histograms of individual sub-group Correlation Coefficients (n=18). In a) the parameters for the grid sampling for the blocks are as in b) i.e. ($d_x = 6, d_y = 6, d_z = 3$ and $c_x = 4, c_y = 4, c_z = 2$). Outliers shown in figure 3.25 are circled in b).

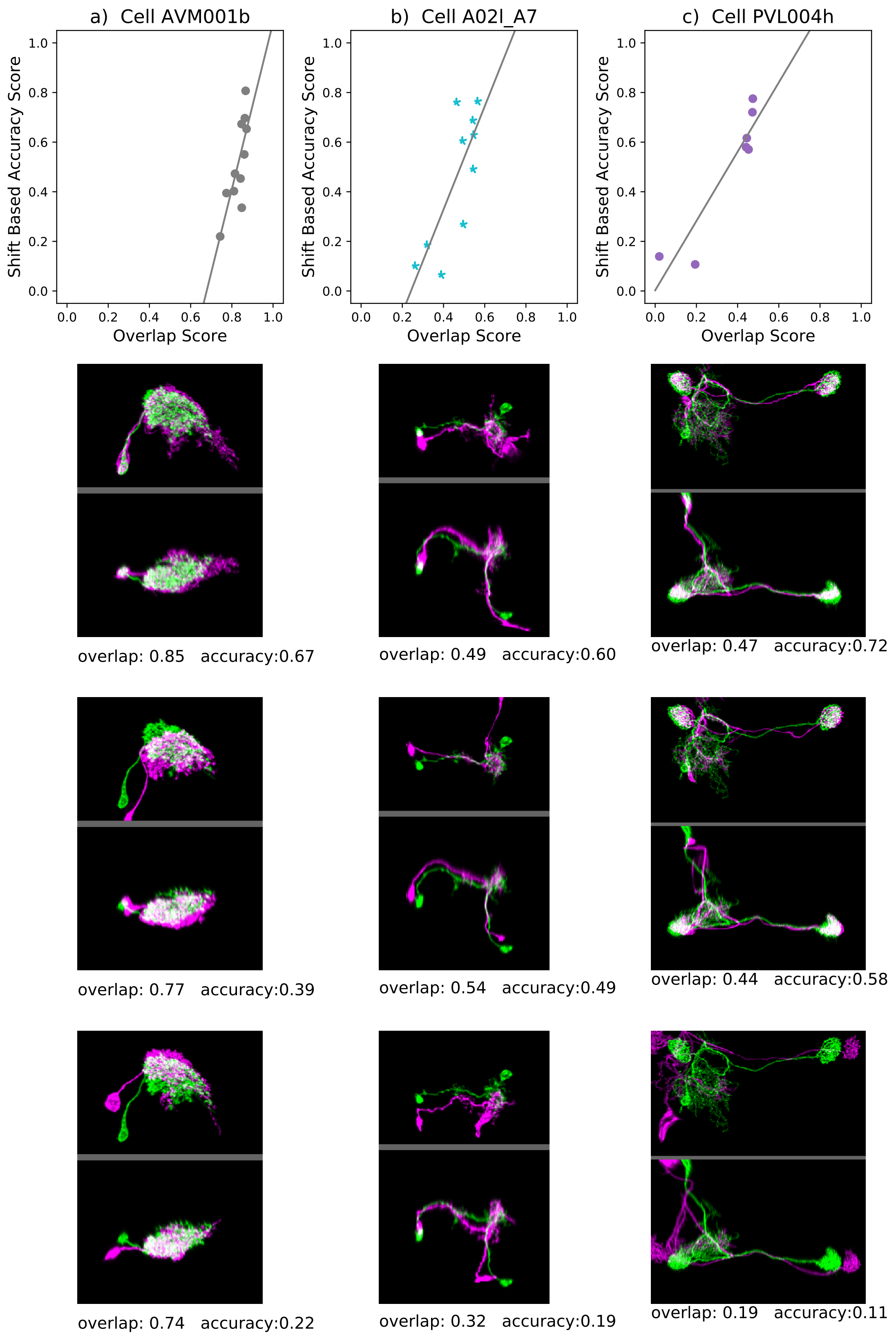


Figure 3.23: Examples of correlations in the subgroups
For three subgroups, plots of correlation between overlap and shift accuracy score, with three examples of cells (magenta) compared with the cell scoring the best overlap score in the subgroup (green), above: maximum z-projection, below: maximum y-projection. Correlation coefficients were a) 0.78, b) 0.79 and c) 0.93.

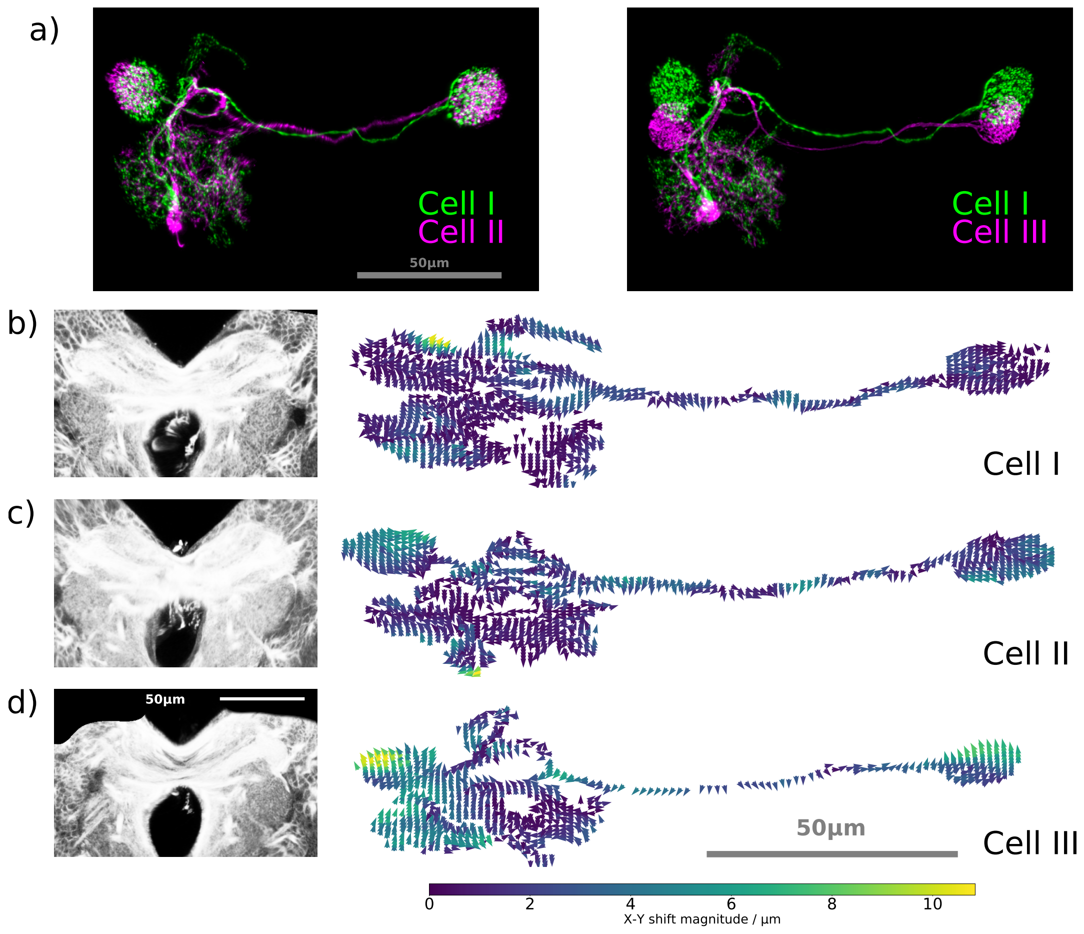


Figure 3.24: Example accuracy score maps within a neural volume
a): Overlap of two cells of the same type registered into the template space. The green cell is identical in both with a good alignment score (cell I, score 0.86), with the left frame containing a good alignment of another, magenta cell (cell II, score 0.80) and right frame containing a worse alignment (cell III, score 0.37). b)-d) Left: corresponding coronal slices through the registered reference image stack at the level of the signal in the right of the CB. Right: The coronal maximum (by image signal) projection of the warp field at points in the cells according to the evaluation metric, with arrows indicating the direction of the shift score and colours indicating magnitude of shift in microns (both confined to x-y direction only). The worse registration of cell III compared to the other cells is clear from the bad overlap - note the missing neuropil on the left of the reference channel and the high inaccuracy scores in d). Note also that the direction of the correction arrows mostly matches those required to further warp the signal to the correct locations.

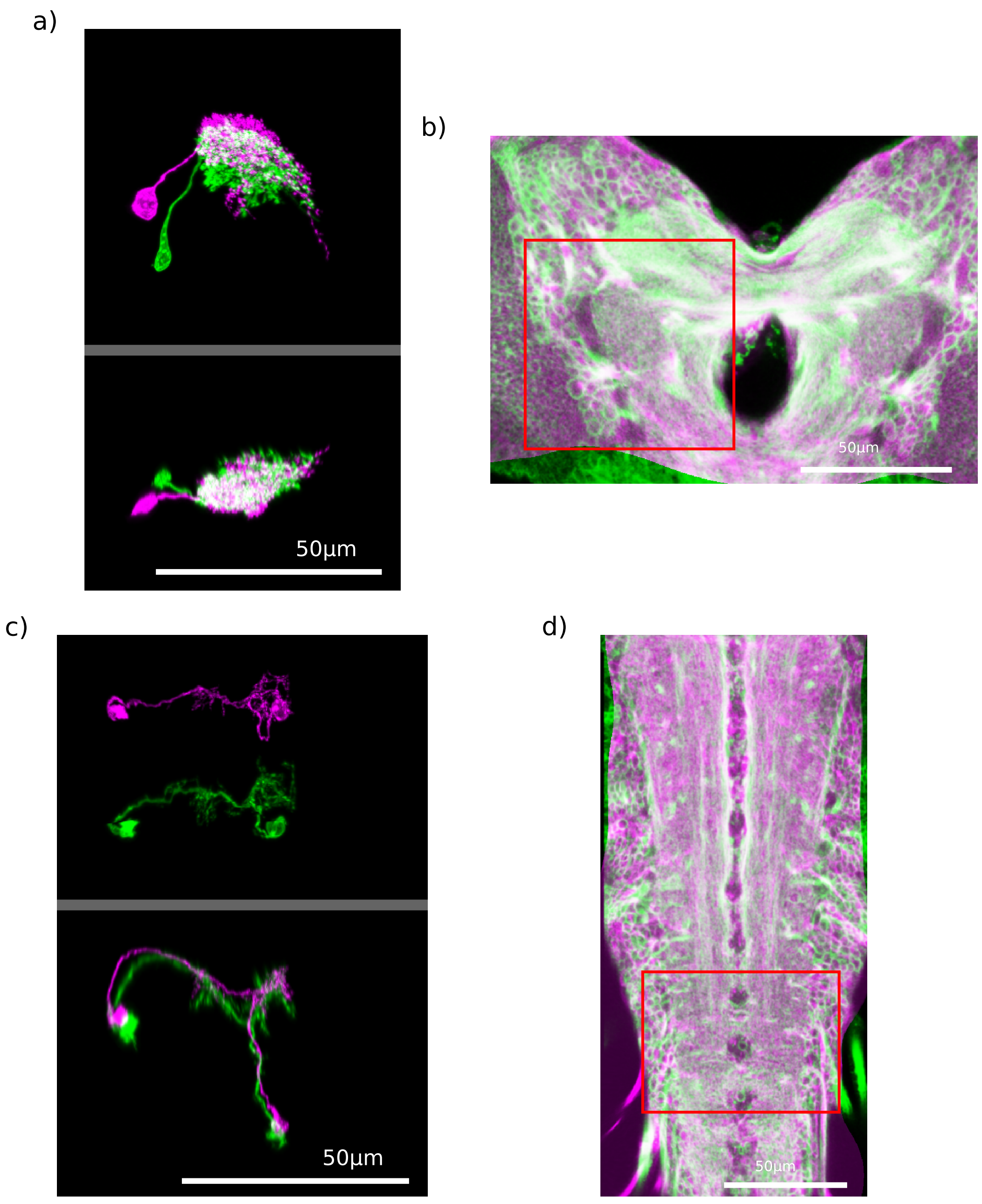


Figure 3.25: Outliers of overlap and accuracy scores
a) and b) refer to outlier 1 (false negative) in the scatter graph of figure 3.22b, and c) and d) outlier 2 (false positive). a) and c) show maximum z- (upper) and maximum y- (lower) projections demonstrating the overlap between the outlier cell (magenta) and the cell in the same subgroup that has the best overlap score (green), for outlier 1 and 2 respectively. b) and d) show z-slices through the image stack with the red boxes representing the volume used in the projections in a) and c). It is clear here that there is some mismatch of the tracts in b) at the ventral side of the CB (which is one of the most common regions displaying bad registration). Thus the accuracy score appears to be performing well here. d) demonstrates a cause of bad registration that is harder to detect: an uneven stretch in the anterior-posterior direction of the VNC, such that the shift at the location of the neural volume is approximately an entire segment. The similarity between two segments of the local segment intensity map makes this so difficult to spot with this method.

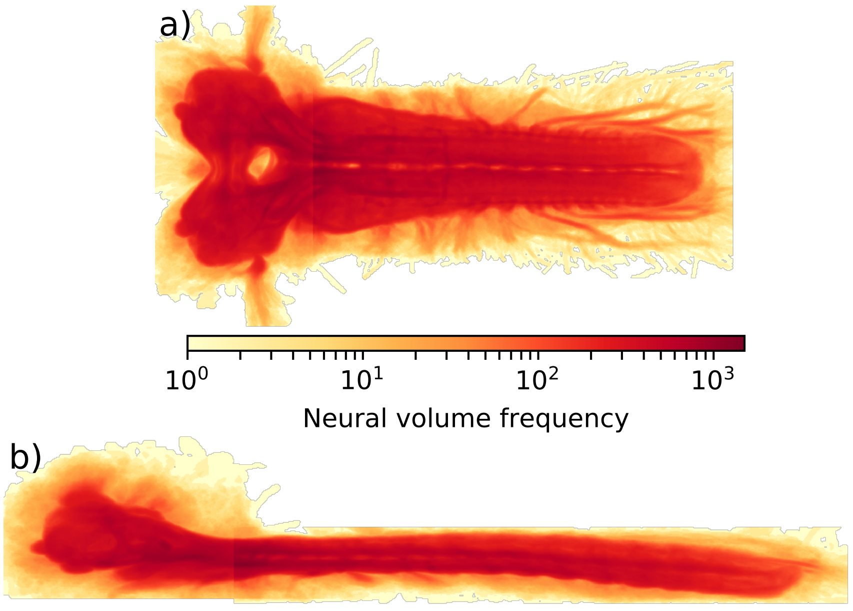


Figure 3.26: Density map of successfully registered neural volumes
Maximum a) z- b) y-projections of the 3D density map of neural volumes. Each voxel was an integer representing the number of neural volumes in the successfully registered dataset that occupied that point (n =35 715). Note the logarithmic scale.

registrations with an overlap score above 0.49 would be good enough for inclusion in the database of registered cells to enable precise enough inferences whilst the accuracy was not too high to exclude so much data as to significantly restrict inferences. From the line of best fit, the accuracy score threshold was thus 0.576, giving a low amount (15, 11%) of volumes with good accuracy scores but bad overlap (false positives), although excluding slightly more (25, 19%) false negatives. This meant that 47.7% of neural volumes (35 715) were kept, determined as satisfactorily registered for further analysis. A density map of the kept neural volumes showed little bias to particular regions (see figure 3.26), although there appeared to be less coverage towards the caudal end of the VNC. There was still, however, a good number of cells imaged in this region (as in all), suggesting there is good coverage of the cells expressed in the dataset.

The shift score predicted a correction that could be applied to the final image, and I attempted to do this, though the results were found to often be of poor quality. Due to this warp being from the float to template space, it meant some template voxels were missing assigned float voxels and thus required interpolation. The lack of rigidity constraint meant some regions were greatly expanded or disappeared altogether.

As a result I did not pursue this further.

3.8 MIRRORING

As discussed in section 1.3, the larval brain is highly symmetrical about the medial axis. Asymmetry in individual cells is rare in the adult *Drosophila* Jenett et al. (2012) and also in the larva Schneider-Mizell et al. (2016). In fact, this fact is used in the larva to compare and validate EM reconstructions Eichler et al. (2017). The multi-coloured flip-out technique more commonly than not stains just one of the two corresponding cells on either side of the CNS, despite the GMR lines from which they originate containing both in the vast majority of cases (Li et al. (2014) and JT, personal communication). Therefore it is justified to "mirror" all cells to their contra-lateral side to see where a contra-lateral cell of the same type would be and use this representation to identify cells of the same type on the contra-lateral side. This effectively allows the database to cover cells of which there are not isolated, satisfactorily registered examples or which do not appear in the original data. This will allow researchers to find more lines for their experiments, however, it must be clearly indicated to researchers when they are viewing mirrored cells.

I simply inverted the y-axis of the template and registered it to the template using the same parameters as above. I could then apply this registration to y-inverted signal (see figure 3.27). Although this does allow error compounding due to two successive registrations, the *bridging* registration between the inverted and template brains was checked thoroughly by eye and was of very high quality throughout. I therefore did not consider it a significant issue.

3.9 SKELETONISATION

In order to perform fast computational comparisons between cells, I needed to extract the morphology of each neural volume into a simple, low-memory graph-like representation. Semi-automated methods, such as available with the *Vaa3D* software Peng et al. (2010), are still time consuming, requiring some key points to be input manually. It is still difficult for a human operator to identify these points in images of the resolution I have. There are, however, a number of fully automated algorithms used for *tracing* neurons to produce a *skeleton* representation of the neurons in confocal microscopy images, including adult *Drosophila* (see Acciai et al. (2016) for a full review). This skeleton representation consists of lines (or *branches*) between points in

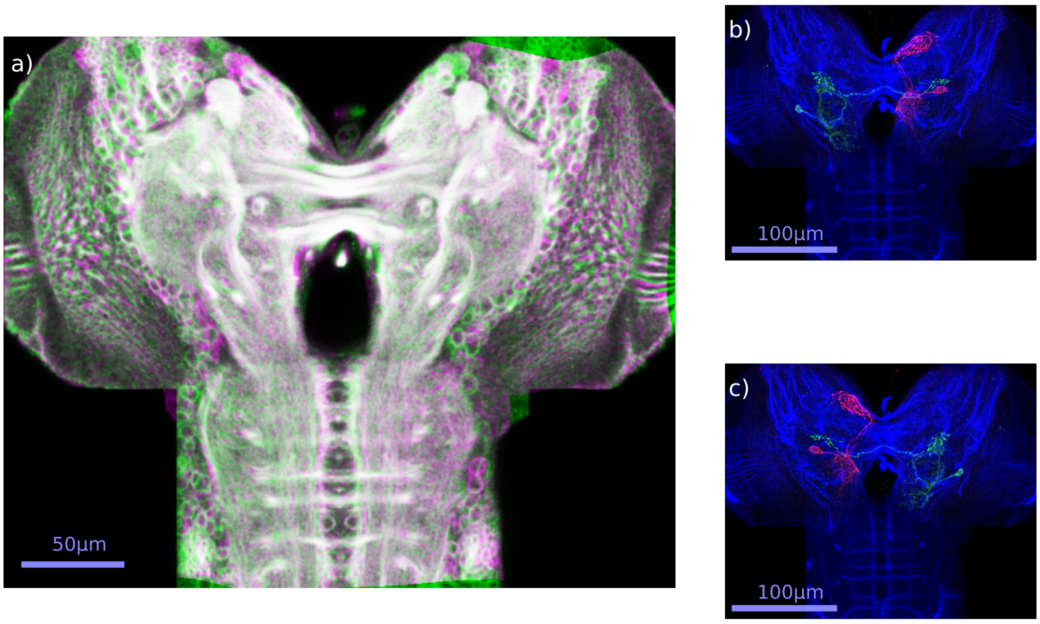


Figure 3.27: Example of mirrored stack
a) Overlap of the (very good) registration of the mirrored template (magenta) to the original template (green). Maximum z-projection of b) original registered stack on template and c) following application of the registration for mirrored template to the inverted b) stack.

space that are either end or branch points where several branches meet. The lines representing branches may be defined by other spatial parameters such as intermediate points or curve control points.

To find the skeleton, most algorithms first identify specific points in the cell that relate to the skeleton. These may be end / branch points or *seed* points that represent good starting points for tracing. This is performed in a number of ways, such as fitting template cylinders to the data to find the most cylindrical parts of branches Zhao et al. (2011), calculating the gradient vector field of the intensity to identify key saddle and attracting points Yuan et al. (2009) or employing a *sliding volume filter* to find key points in curves Sui et al. (2014).

If the identified points are expected end / branch points, then some algorithms build tubular paths between them using a weighted shortest path algorithm (e.g. Dijkstra or minimum spanning tree). Weighting might be simply via intensity Xiao and Peng (2013) or a more complex function e.g. of closeness to a branch-point Lee et al. (2012) or local tubeness score Turetken et al. (2012) (see section 4.4). Alternatively, tubular branch traces can be built up by starting tracing individual branches from the seed points and connecting branches when they meet. Traces can follow gradient field eigenvectors Yuan et al. (2009), centre lines of active contours Chothani et al. (2011) or minimum energy snakes Sui et al. (2014) among others.

In order to skeletonise the neuron morphologies, I began with a relatively simple approach. This was essentially a minimalistic implementation of the successful algorithm of Chothani et al. (2011). I took a binary threshold, as is common with other skeletonisation techniques Lee et al. (2012), and found the centre lines of the binary structures. In contrast to Chothani et al. (2011), that uses the voxel coding algorithm, I employed the more established FIJI plugin "Skeletonize" that applies the algorithm of Lee et al. (1994). I chose this since it was readily available and recommended as working well with the matching techniques discussed in chapter 4 by the author of one of these key methods (Greg Jefferis (GJ), personal communication).

Therefore the segmented neural volume was skeletonised by reducing it to its medial axis. This thins objects represented in digital topology iteratively by identifying voxels to remove that satisfy the following criteria: Voxels that, if removed, do not change the Euler characteristic for the signal, S:

$$E(S) = O(S) - H(S) + C(S) \tag{3.15}$$

where O represents the number of connected neural volume objects, H the number of torus-like holes in the neural volume and C the number of cavities (background objects entirely surrounded by neural volume. Here a "connected" signal object contains points that are connected to each other such that each point has a neighbour in the same object within its nearest 26 neighbours. The algorithm considers each direction at a time to erode the neural volume to its medial axis. Voxels are not removed if doing so would reduce the remaining voxels' connectivity (number of connected neighbours) to less than 2, thus preserving points of connectivity equal to 1 as end points once they are identified. The use of the Euler characteristic, which can be summed from components, enables a fast, dynamic memory approach to be applied.

Upon testing this with the registration test set, using the binary neural volumes extracted above, I decided to proceed with this method. See the example in figure 3.28. This choice was based on reviewing image stacks, such as in figure 3.29, where it appeared to extract most of the morphological structures that were common in cells of the same type. Indeed, I only require a morphology for the type and for the variation in the high arborisation regions that I am not interested in I would prefer a simplified representation. In the spirit of an Agile approach, I could complicate the method further and assess performance impact on the end goal once I had implemented the whole system.

The disadvantage of the Lee et al. (1994) method, however, is that sometimes there are many loops in the resulting skeleton, which I suspect is due to the Euler characteristic requiring the object to preserve the number of holes and cavities in the segmented volume, which seem to occur frequently in arborisation zones. Also, there were many short branches that were not consistent between cells of the same type. To address these concerns, and in order to ensure that I had the most important sections of the skeleton, I pruned these skeleton representations (see section 4.3).

3.10 COMPUTATIONAL FACILITIES AND TECHNICAL CONSIDERATIONS

I implemented the above pipeline using the open source programming language Python with packages numpy Oliphant (2006) and scipy Jones et al. (2001) with some extra custom scripts written myself in C and for the Linux Shell.

The CMTK algorithm requires significant amounts of compute time with the parameters that I specified (see figure 3.30), and the skeletonisation and scoring components also non-trivial amounts (see figure 3.31). As part of the University of Edinburgh Neu-

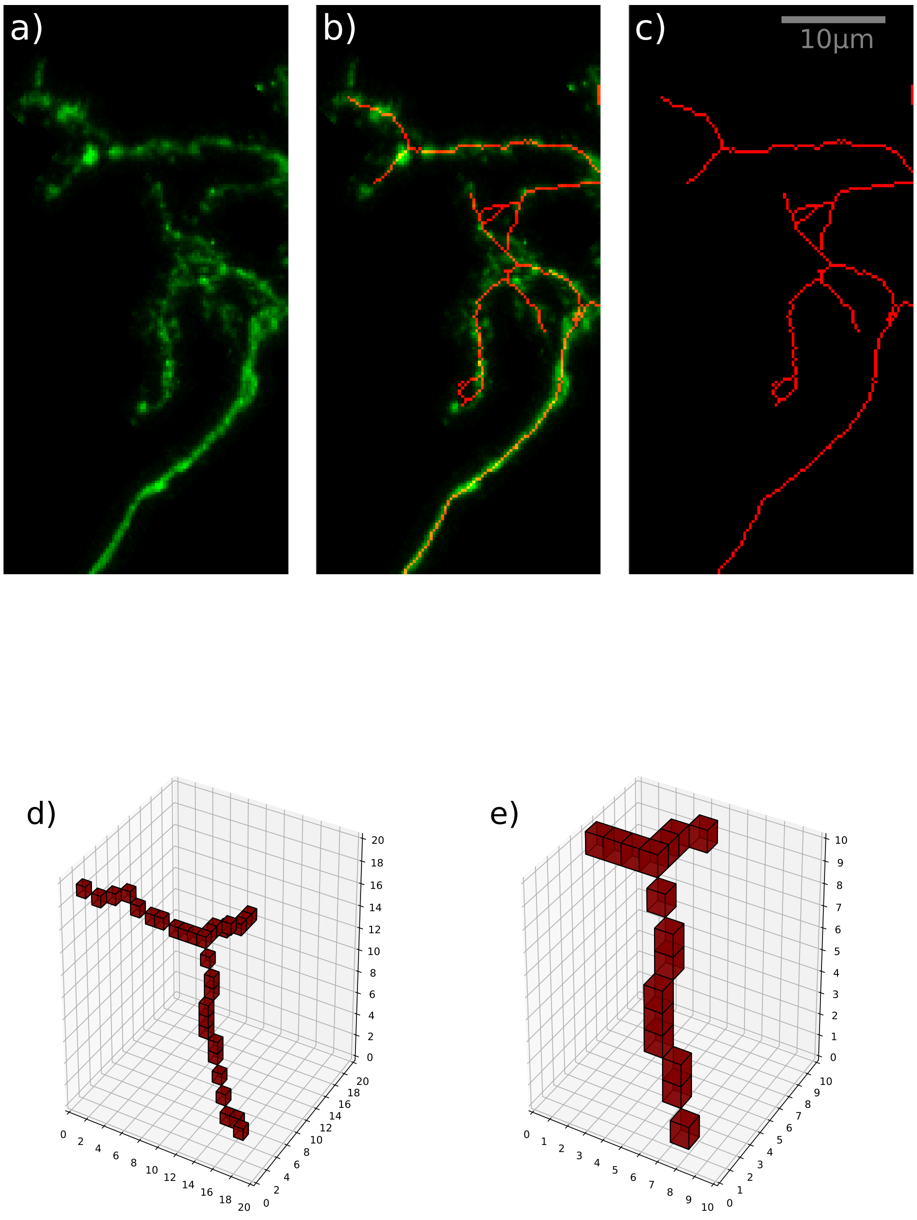


Figure 3.28: Skeletonisation example
z-projections of a) signal of neural volume, c) FIJI Skeletonisation output and b) overlap of a) and c). d) Partial 3D representation of skeleton with e) zoomed to centre to demonstrate the connectivity. Axes units are in voxel widths / heights.

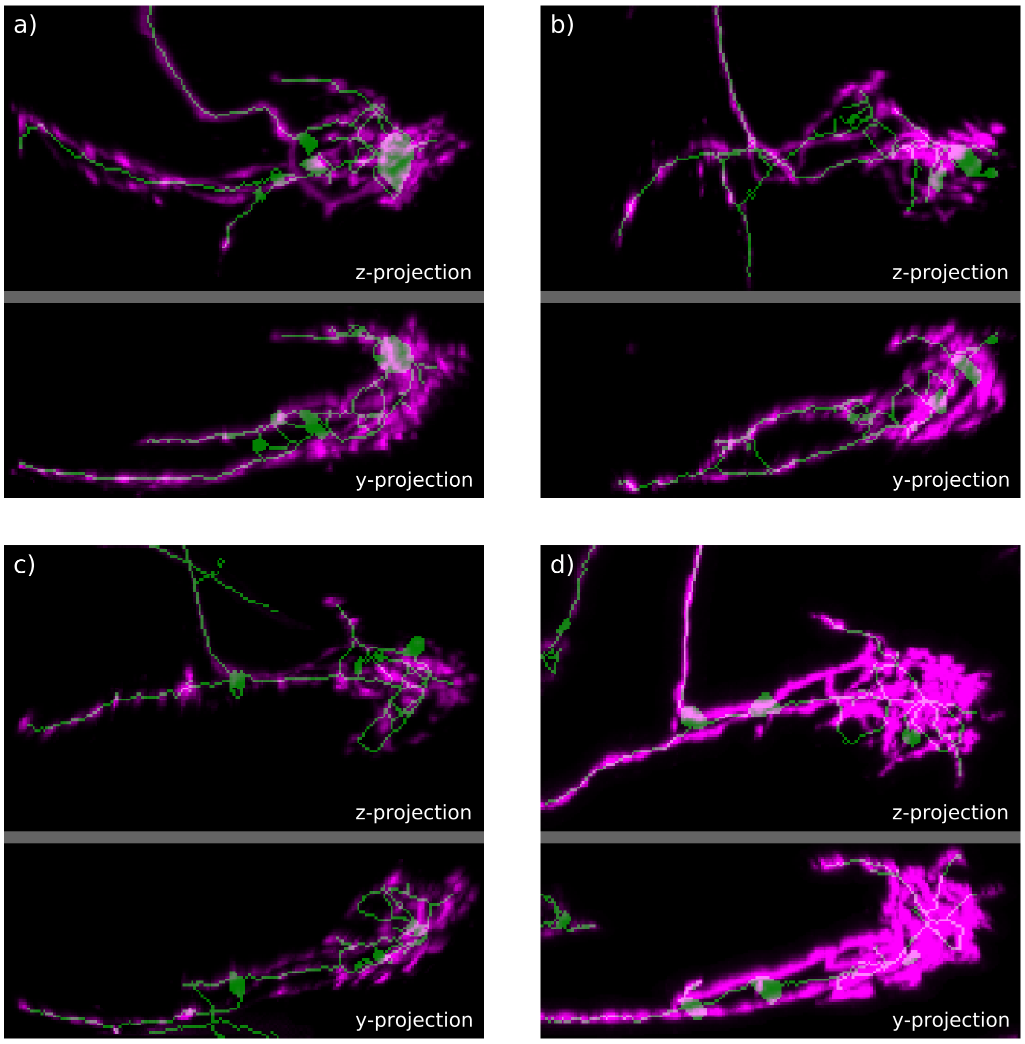


Figure 3.29: Skeletonisation and biological cell variation
Illustration of cell biological variation for well-registered cells of the same cell type. Maximum intensity projections of: registered signal (magenta), medium axis skeleton (green). Note, in particular, the high degree of variation in the position of the left dendrite, the presence or lack of lower protruding branch in the z-projections and apparent difference in size of the arborisation region. This highly arborised region on the right demonstrates how difficult the problem of tracing becomes in these regions with the small larval neuron dendrites. The skeleton trace is able to identify the broad structure clearly, outside the arborisation region, for example picking up the consistent upper projection from the arborisation region. The green globules in the skeleton are artefacts of enclosed cavities in the binary representation and will be removed in chapter 4. Clearly there are also some limitations to the tracing such as in a) and d) two parallel branches are merged. This will also need to be taken account of when choosing the comparison algorithm.

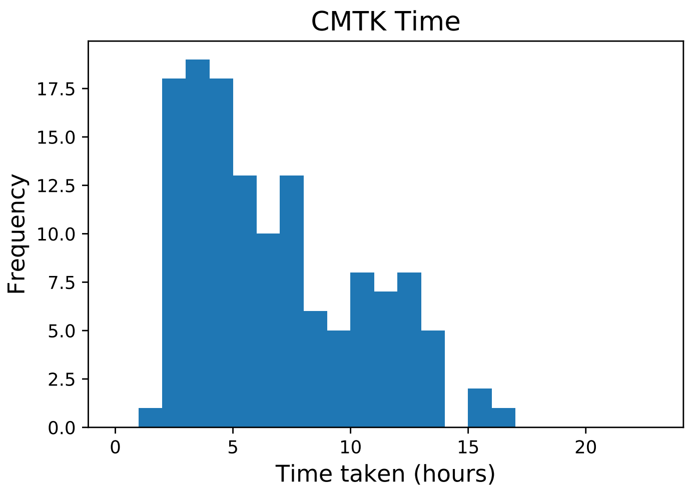


Figure 3.30: Time taken to run CMTK registration for the test data set Times are for performance using 3-CPUs at 2.4 GHz. n=134.

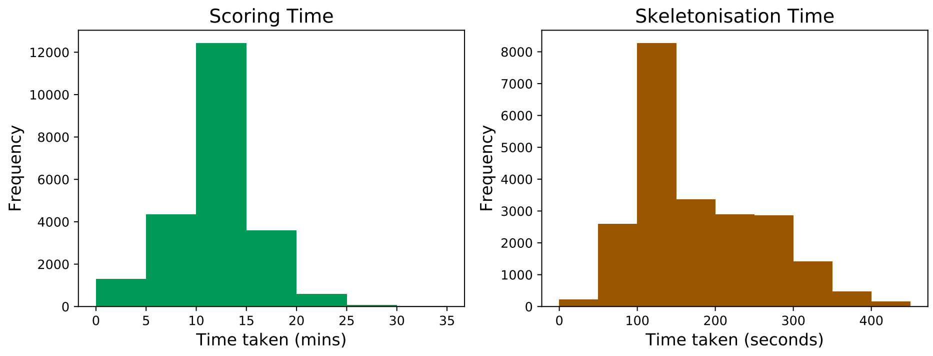


Figure 3.31: Time taken to run scoring and skeletonisation for all stacks Performance using a single core at 2.4 GHz. n=22 308.

roinformatics Doctoral Training Centre, I was able to utilise the Edinburgh Compute and Data Facility's Compute Cluster (Eddie) effectively without budget constraints, although with a limited quota. The large size of the raw data meant that I was required to both move the data from my local network hard disk array to the compute cluster and execute it in batches. I automated this process by writing a server script that I ran on my University desktop computer that also checked the completeness of the registration pipeline for each image stack. The returned data had a mean size of 31.8MB per stack (excluding the registered reference channel). In order to utilise compute time effectively, the CMTK part was separated from that of the scoring as the former utilised multi-core processing (I used 3-cores) and the latter did not.

3.11 CHAPTER SUMMARY

In this chapter I have outlined my successful method of registration of a large amount of 3D image data to a common template. I have detailed how I identified a suitable template from a limited data set, made the raw data more consistent and tuned registration parameters using my own test set of image stacks. I explained how I extracted neural volumes from the registered image stacks and assessed their registration accuracy by developing and validating an effective scoring algorithm. This is the first fundamental step in allowing researchers to make inferences across many larval CNS specimens.

3.12 CHAPTER DISCUSSION

Although more accurate registration is always desirable, it is also clear that bad registrations were the major cause in data being cut from the atlas (52.3% of neural volumes lost). This performance, however, was considered better than expected and sufficient by collaborators. Registration of the *Drosophila* third instar larval CNS is more difficult than that of the adult brain partly due to the larva being a developing system, so that the precise time of dissection will result in noticeably different brain structures and sizes (such as the short VNC example in figure 3.8). The dissection procedure itself also results in significantly variable angles of rotation of the CB lobes with respect to one another. The region of the join between the CB and VNC (anterior SOG) was often noted to have issues in registration, and this is likely due to two reasons: Firstly, due to the CB rotating forwards when taken *in vitro*, this might distort this region. Secondly, this region was often near the boundary of the raw data stacks I registered, thus the automated registration sometimes attempted to stretch or compress the image to fit the excess or reduced neuropil volume of the partial template to which it was registered (this might explain figure 3.25d). Nevertheless almost half of the registered neural volumes were determined to be registered well enough for a public database. In addition the system I have implemented gives additional feedback to the researcher in the form of a map of accuracy of registration (something novel in the field).

The rotated template stack produced significantly better results than I expected without much variation in intensity and despite the principal axes alignment of CMTK. Having analysed the source code of CMTK, I suggest that the reason for this could be the increased empty background region that this stack gained in the x- and z-directions. These blank regions increase the region used in the similarity metric, especially in the z-direction, and as a result might discourage signal in these regions. This is opposed to when no template information was available for them, meaning that CMTK did not assess the similarity metric here and therefore erroneous float reference channel signal here was not penalised. In addition, in retrospect and for future usage, I would test a VNC portion of the template that was increased in the y-direction towards the CB as I noticed on visual inspection that many VNC registrations were cut off by this edge.

The small gender differences in the *Drosophila* larva CNS are one possible source of error that should be noted but I do not think it contributed significantly in this case. This is principally because these differences are so small, specifically relating to a small group of neurons in the terminal abdominal lineages Li et al. (2014). Ideally, one would produce a separate template for each gender and larval stage with enough data from each type to create a complete map of cells in each so that researchers could analyse the differences and work to their desired stage. This was not possible with the data set I used as the gender of the larvae was not known.

It was particularly noticeable that, for registration similarity metric, NMI (CMTK's default) performed considerably worse than MI despite both having similar results for scoring metric. As NMI is designed to reduce the effect of MI increasing due to increasing the amount of signal overlap, it was interesting to see that, in general, stacks registered using MI were warped more than those of NMI, particularly along the edges of the float that were within the CNS and not background. It is not clear to me, however, why this might be the case.

There has also been successful registration reported (several years after I finalised my registration parameters) in the whole-CNS, lower resolution larval registrations of Muenzing et al. (2018) that utilised the B-Spline registrations of the elastix toolkit Klein et al. (2010). The main interesting aspect of this work in relation to my research is that the registrations were performed in the range of tens of minutes as opposed to the hours that CMTK employed (although the size of the data was a few factors smaller). This was mostly due to utilisation of random sampling of the image intensities to calculate the similarity indices. In addition, a median average template was used due to the availability of many whole-CNS stacks and demonstrating that this could also be a valid approach.

The overall accuracy was reported using a "gold standard" method of measuring distance between landmarks in the registered and template image stacks. The average error was 5.4µm, which suggests that it performs worse than my method, where the registration accuracy score gave the median shift of all voxels within the neuropil for the registration test set of neural volumes as 1.55µm. In any case this is a good validation of my overall approach; it was especially reassuring to note that Muenzing et al. (2018) showed that the ANTs SyN method performed similarly to the elastix B-Spline method. Clearly it would be worth investigating elastix if registration duration became an issue. The use of high-information landmarks, however, could bias the results quoted in Muenzing et al. (2018) as these could be easier to match up due to their higher local information. There were also very few landmarks in the central section of the VNC meaning that registration accuracy was not measured here.

Now that I have a working system, I would also investigate creating a statistical template (similar to Muenzing et al. (2018)). This aesthetically pleasing template would look similar to the average intensity CNS in figure 3.10, removing many textures but emphasising the tracts. Ideally this would be produced from a series of high quality, very carefully dissected whole CNS stacks. Arranging to produce these in time for this project was however not something I pursued, unfortunately, as I expected it would delay my project significantly.

The neural volume segmentation method took advantage of the stacks being sparse and was therefore very simple. For stacks where there was a large amount of neuron staining (especially in the VNC) or very weak staining, the segmentation threshold was often estimated as too high, removing large amounts of signal. In this project this was not a large problem since I am mainly interested in clearly separate cells, generally found in sparse stacks. However, it would be worth investigating not only a more intelligent method of determining a threshold, but also methods to separate cells or their skeletons that are within the same neural volume. Some tracing methods already consider this by using trained algorithms to separate cells. Chothani et al. (2011) uses a trained cost function based on angles of crossing branches and average intensities and Gala et al. (2014) a Support Vector Machine (SVM) on a large number of features.

The accuracy scoring method works well despite the registered stack voxels in theory being at local minima. I suggest that the reason there are still shifts suggesting better minima is due to the smoothing regularisation and smoothness of the B-Splines This is since applying the calculated shifts to the float images produced highly warped images with notable points of attraction and repulsion. There are still, in theory, better overall registrations for the low accuracy scoring stacks, but producing them in reasonable time with current methods remains to be discovered. Despite being mostly successful, the scoring method clearly fails in some instances - such as the false positive in figure 3.25. Possibly combining the neural volume accuracy score with a whole-image score could help address issues such as a unit segmental shift in the VNC. The accuracy score as defined - being a proportion of the volume under a threshold - would also, in theory, allow a minority of the cell to be very badly registered. In practice this was very rarely the case, but does illustrate the need for accuracy maps. A more complex accuracy score could take into account the distribution in space of the accuracy scores to ensure that there are not very badly registered regions.

Whilst the total time taken to run the registration pipeline was significant, I kept in mind the resources I, and my collaborators, had available as I adjusted parameters to ensure that I had the best performance for what I defined as reasonable run-times (i.e. of the order 10 hours per stack). As I was able to run the pipeline on many (over 50) stacks at a time, this enabled registration for all data to be performed over the course of several weeks.

Throughout this chapter I have emphasised the agile approach to development that I attempted to apply. I kept in mind my end goal of creating a database of registered cells that can be searched quickly for similar cells rather than producing the best possible performance at all stages. This was in order to ensure I reached this goal in good time. Now that I have a working pipeline for a good proportion of the data, there are a few areas I would pay attention to developing further in the future in order to increase the percentage of usable data. In particular I would investigate creating a statistical, average template and evaluate the effect of this on the registration performance in the context of the registration accuracy scores. I also would improve the combined segmentation and skeletonisation method (this is discussed further in section 6.0.1 in light of the requirements and developments of the rest of the project).

4

SINGLE NEURON ANALYSIS AND COMPARISONS

4.1 INTRODUCTION TO STEREOTYPING AND COMPARING NEURONS

As mentioned in chapter 3, I desired identification of cells of similar morphology (and hence same type) across different GMR lines. Therefore, I needed to identify properties and/or representations of the cells that allowed classification by type or at least the ranking of cells by a probability of being from the same type as a query cell. I needed to extract and store this information since the registered images themselves were very large and searching them would have required a large amount of computer memory and time. In this way I could build up a similarity metric to judge how similar two cells are.

Indeed, cell-type classification is not well defined and requires expert visual inspection of the raw image stacks. Hence I expected that it would not be possible to fully automate the classification. Therefore, I decided to prioritise producing a system that would allow real-time searching of the database of neural volumes by researchers looking for similar cell morphologies to a query cell, with candidates returned listed in order of similarity.

I worked with both the registered stained signal channel and skeleton trace of neural volumes to produce key identifying features of the cell types and condense their representations into a low memory, cell-type dependent representation. Key features that I attempted to identify are discussed towards the beginning of this chapter: cell bodies in the images and tubular regions in the skeleton. I also simplified the skeleton trace by pruning it. Following this I detail how I implemented an established cell-cell similarity metric algorithm on these simplified skeletons. I also developed other metrics by judging them against the performance of classified cells, including a Machine Learning (M/L) representation. Finally, I produce an algorithm for creating an average representation of many neurons of the same type.

Once again, I stress here that the focus was to produce a usable (i.e. fast) system that was accurate enough for its intended use, as opposed to the most accurate system. It is also important to keep in mind that, while registered cells of the same type will

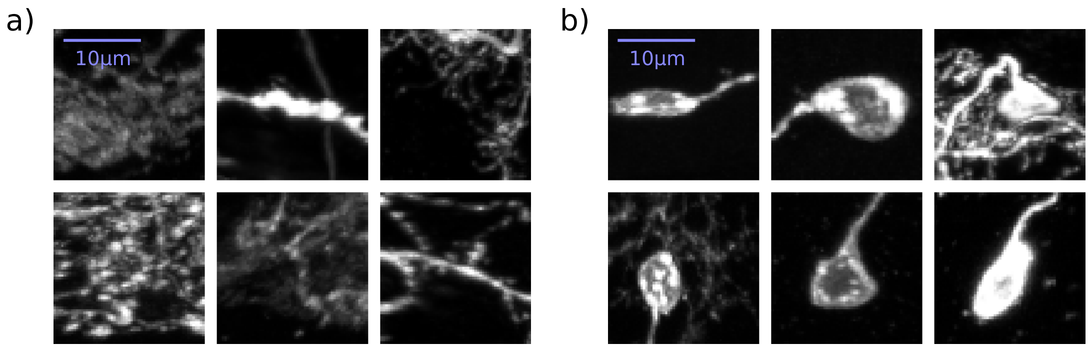


Figure 4.1: Examples of the Cell Body Detection ROIs
Z-projection windows of the 3D ROIs, with a) Non-cell body containing labels and b) Cell body containing.

have a similar global morphology, their will be biological variations as well as those that arise from any deficient registration. As a result there is often less overlap than might be initially expected: For example, a small, radial shift of a thin, tubular neurite could mean this neural section no longer overlaps with a cell of the same type (see the central column of figure 3.23). Alternatively, if this tubular neurite passed through a region of high arborisation of another cell type there would be a false positive raw signal overlap, though the local morphology is very different.

4.2 CELL BODY DETECTION

There were two key motivations for finding the cell bodies of the neurons - firstly the location of the cell body is a key identifier in the cell type and is often used as the *root* node of neural traces that are represented as trees Li et al. (2017b). Secondly it would help - ideally automate - the process of determining if a neural volume contains just one cell such that it can be used as a clean example of a *single cell volume* in the database. I started my approach by first taking a random sample of 388 image stack signal channels from the whole registered data set and split them into a training set (305 stacks) and test set (83 stacks).

4.2.1 *Data Preparation*

I wrote a simple plug-in for the FIJI Schindelin et al. (2012) software, that enabled me to quickly manually iterate through maximum z-projections of signal channels for each set of the images and use the paintbrush tool to mark (generally circular) ROI overlays on top of the image such that they contained the cell bodies and little else.

Table 4.1: Cell Body Detection Training And Test Set Statistics

Numbers of	Training Set	Test Set
Image stacks	305	83
Cell body markers	330	89
Cell body ROIs	7482	825
Non-cell body ROIs	26570	3189

Table 4.2: Cell Body Detection Method Results (Test Set)

Method	Correct CBs (%)	Correct Non-CBs (%)	Mean Correct (%)
One Gaussian	85.6	73.8	79.7
Difference of Gaussians	88.6	73.3	81.0
Linear Model	87.5	83.1	85.3
CNN	86.5	99.2	92.9
CNN (with rotations)	91.2	99.3	95.2

Points in the skeletons of each image were then classified as being part of cell bodies if their x-y position was within an overlay region and also within 10 voxel widths of the centre of mass (in the x-y direction) of the overlay. All cell body points and a random sample of approximately 5% of the non-cell body points from the remaining skeleton were selected for further analysis. A 3D ROI from the signal channel of the image stack around each selected point was copied to a four dimensional (4D) data array for each set with a corresponding one dimensional (1D) label array along the extra dimension. The cuboidal ROI size was: $31 \times 31 \times 19$ voxels centred on each point. The label was binary: "containing cell body at centre" or "containing no cell-bodies" (see figure 4.1 for examples). The labels were checked and corrected if required by manually reviewing Z-projections of the ROIs myself. Boundary cases, e.g. cell bodies near edges, were removed. Statistics on the training and test sets are given in table 4.1.

4.2.2 *Gaussian Convolutions*

A common method to detect cell bodies is to convolve the image with Gaussian filters and find a threshold to determine each image voxel as being part of a cell body Shuvaev et al. (2017). I convolved each ROI with a normalised 3D Gaussian distribution (figure 4.2a) to produce a test statistic. I iteratively searched for a Gaussian standard deviation that optimised the the separation of cell body and non-cell body ROI test statistics. I settled on 6 x-y voxel widths (1.76µm), and applied a threshold of

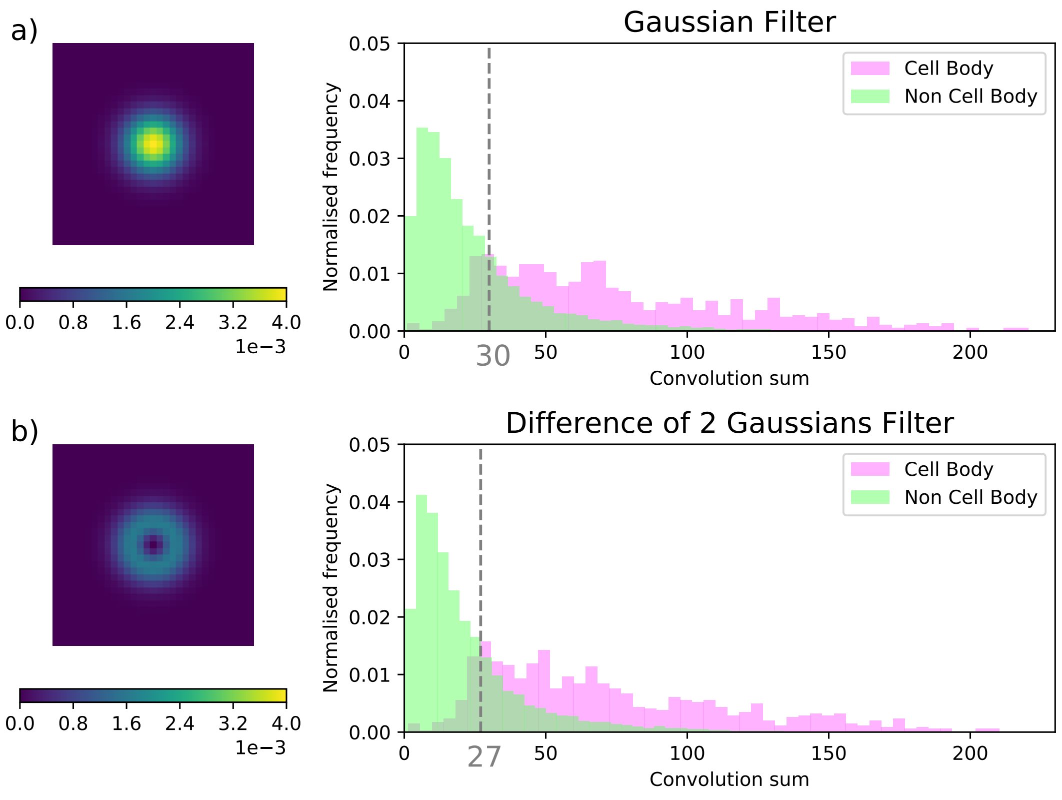


Figure 4.2: Cell Body Results for Gaussian Filters
a), c) The median x-y slice in the z direction of the filter. b), d) Histograms of separation of the scores of the test set. The dashed line represents the threshold at or above which the ROI was classified as a cell body.

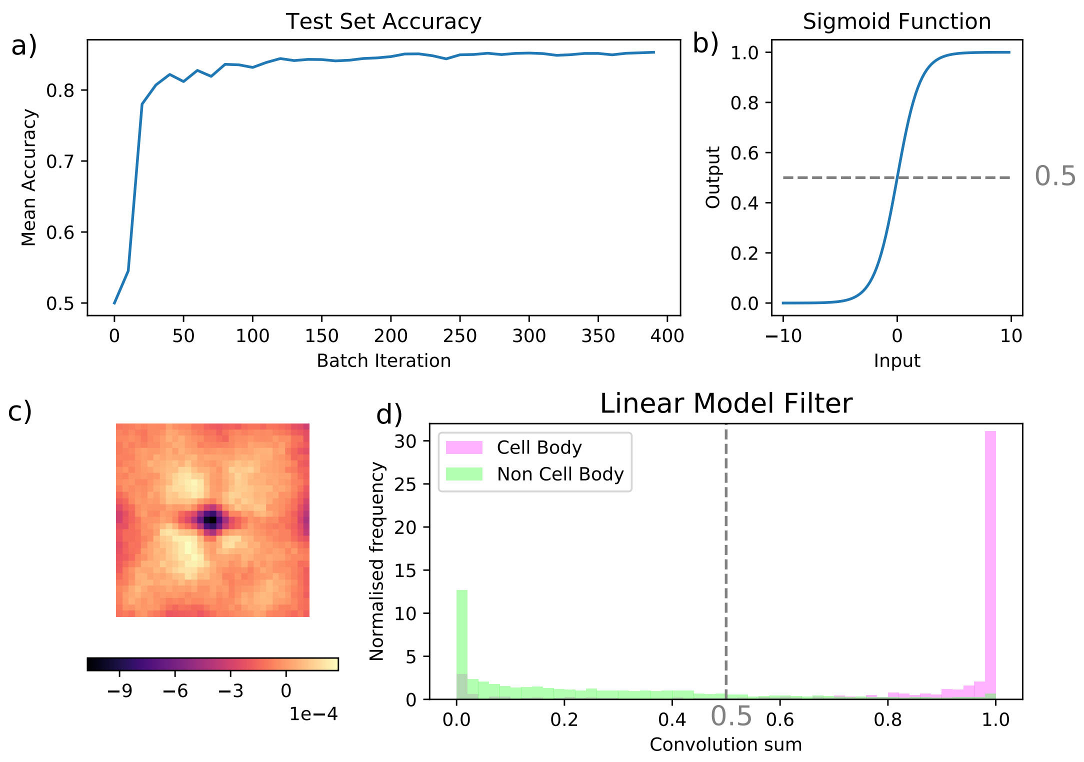


Figure 4.3: Cell Body Results for Linear Model Filter
a) Test set accuracy during training showing saturation around 85%. b) Sigmoid function illustration. c) The median x-y slice in the z direction of the filter. d) Histogram of separation of the scores of the test set. The dashed line represents the threshold at or above which the ROI was classified as a cell body.

30 on the test statistic at and above which ROIs were classed as containing cell bodies. This was reasonably effective, achieving an accuracy of 85.6% for actual Cell Bodies and 73.8% for Non Cell Bodies on the test set, giving a "mean score" (i.e. the average of these two numbers) of 79.7% (see table 4.2). I aimed to improve this mean accuracy figure.

To reflect the fact that the intensity values in the centre of cells was usually less than the borders, I successfully improved the accuracy of this method by changing the convolution filter above by subtracting a narrow central normalised 3D Gaussian (standard deviation 0.59µm, figure 4.2b). For this "Difference of Gaussians" method the threshold was set at 27 and this increased the mean accuracy score by 1.3%.

4.2.3 *Linear Model*

I decided to use the training set to optimize the applied filter. This is equivalent to a linear model, whereby at each position in the 3D ROI surrounding the point of calculation, (i,j,k), the intensity value, I, is multiplied by its corresponding filter value, W, and the results summed together for the whole ROI:

$$\vec{I} \cdot \vec{W} = \sum_i \sum_j \sum_k I(i,j,k) W(i,j,k) \tag{4.1}$$

where $\vec{I}$ and $\vec{W}$ are the 1D vector representations of the intensity for the whole ROI and filter respectively. The result of this sum was passed into a sigmoid function, S, a differentiable function that converts an infinite range monotonically into one between zero and one, with extreme values being mapped to a value very close to zero or one (see also figure 4.3b):

$$S(\vec{I} \cdot \vec{W}) = \frac{1}{1 + \exp^{-\vec{I} \cdot \vec{W}}} \tag{4.2}$$

By designating labels, $L(x)$, for a cell x with cell body ROIs having value 1 and the non-cell body values being 0, it was possible to calculate an error function, E, as being the squared difference between label values and sigmoid output i.e.:

$$E = \left(L(x) - S(I(\vec{x}) \cdot \vec{W}) \right)^2 \tag{4.3}$$

This function is differentiable with respect to the values of W and as a result this error could be minimised with respect to $\vec{W}$. I did this by using batches of examples from the training set to evaluate the sum of the error terms for each batch and applied back propagation Nielsen (2015) to adjust the weights using gradient descent.

I implemented this using the TensorFlow Python package Abadi et al. (2015) and trained over 400 iterations with a batch size of 200. The test set showed that this improved results by a further 4.3% to 85.3%. Note how the filter (shown in figure 4.3c) resembles somewhat the 2 Gaussians, but note also the more steady reduction towards the edges, perhaps to encompass a wide variety of cell body sizes.

4.2.4 *Convolutional Neural Network*

I attempted to improve these results using a more complex model, namely that of a Convolutional Neural Network (CNN). CNNs have been reported to have been used successfully to detect cell bodies Sanborn (2015). CNNs are highly useful in image processing, although can take long durations to train and require much data to do so, especially those working on three dimensional input data. I therefore decided to work with only the z-projection "windows" of ROIs. Using CNNs on 2D input data mimics human-brain processing methods of images LeCun et al. (1998) - those that I used to inspect the ROIs to create the training and test sets earlier - and thus I expected them to be successful.

Figure 4.4 shows the structure of the CNN that I implemented following trials of different sized networks trained on the training set and evaluated with the test set. I specified the output activities, **a**, to be 2 dimensions which I converted to probabilities exploiting the exponential function as a positive, monotonic function (soft max. method):

$$p_i = \frac{e^{a_i}}{\sum_{i=0}^{1} e^{a_i}} \tag{4.4}$$

I was then able to apply the cross entropy as an error function, L, as this has been shown to perform better than the mean squared error with neural networks Golik et al. (2013). Since this scales errors by the logarithm function to compensate for the exponentials in equation 4.4 it reduces the effect of large differences being hidden in the output activities for neurons when both are well within a specific class:

$$L = \frac{1}{N} \sum_{n=1}^{N} \left[- \sum_{i=0}^{1} y_i(n) \log_2(p_i(n)) \right] \tag{4.5}$$

where N is the number of samples (ROIs) in a batch and $\mathbf{y}(n)$ is a 1D vector for each ROI specifying the label (as a *one-hot* representation) such that here $[0, 1]$ contains a cell body and $[1, 0]$ does not. To reduce over-fitting, I used the drop-out method Srivastava et al. (2014) (rate of 0.5) and optimised the weights using gradient descent. Equal numbers of Cell Body and Non-Cell Body ROIs were presented in each batch. To speed up the calculations this was run on a Graphics Processing Unit (GPU). The results were significantly better than with the linear model, with a 92.9% mean accuracy score. See figure 4.5.

Finally, during training, I applied a random rotation to ROI projection windows (which had been created slightly wider and cut to appropriate size after rotation to prevent missing signal) before adding them to each training batch. This was in order to artificially increase the training set size and I thought would be valid as the detection should be independent of rotation. This indeed added over 2% to mean accuracy or, more importantly, reduced the error by almost a third.

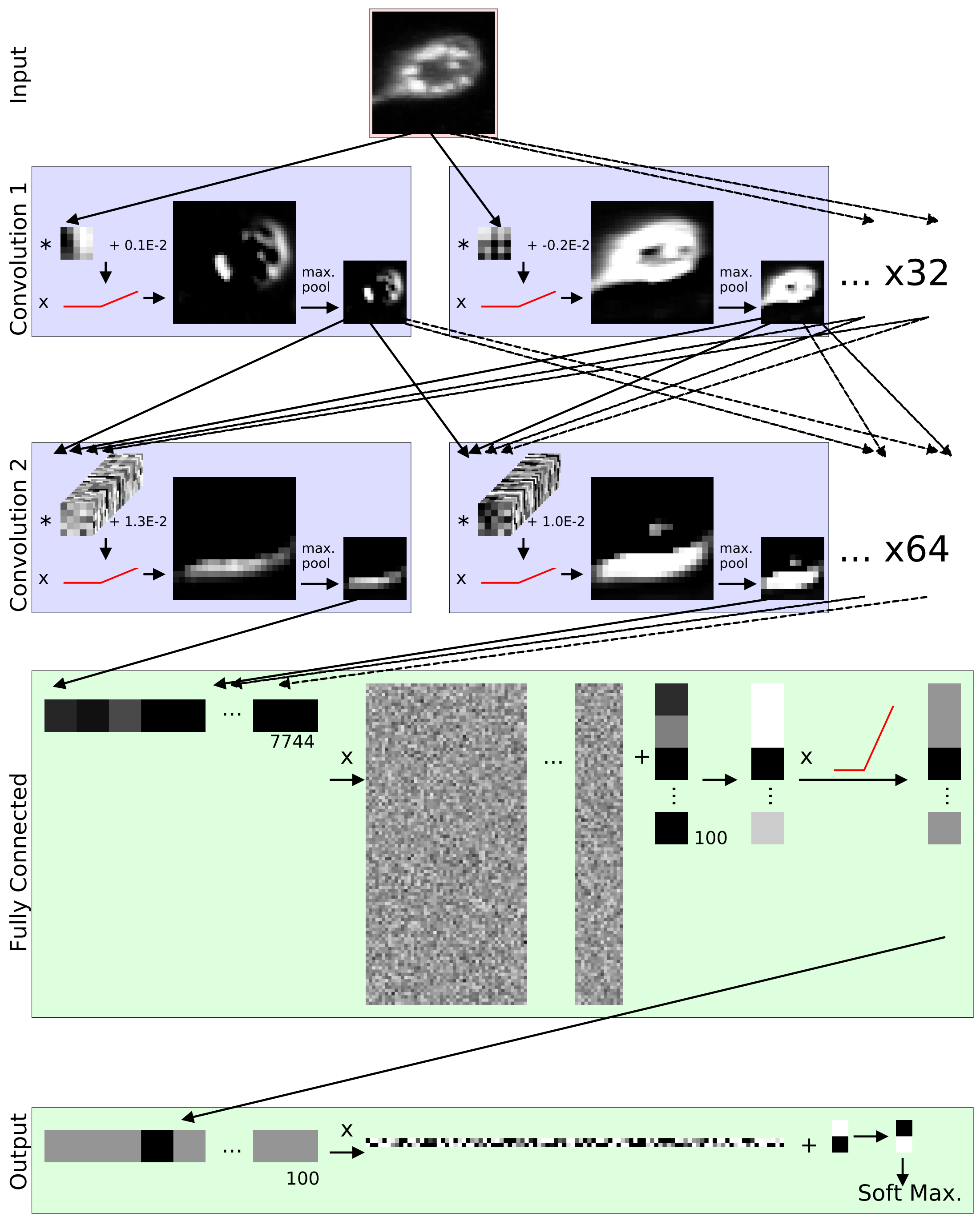


Figure 4.4: Layer Diagram of Convolutional Neural Network For Cell Body Detection Example of processing a projection window containing a cell body following training. The neuron weights and outputs are represented by the different shades of grey and intermediate images shown following the convolutions respectfully (scale varies between images). The red line graph represents the rectifier linear neurons used and the arrows the flow of information between components. There were 32 convolutional dimensions in the first layer and 64 in the second, with each layer taking the max. pool of a 2x2 grid, giving 7 744 inputs to the fully connected layer that reduced to 100 output neurons and, through the final linear layer, 2 values for a soft max. approach (see text). Note that familiar edge detection convolutions appear to be involved for the first layer.

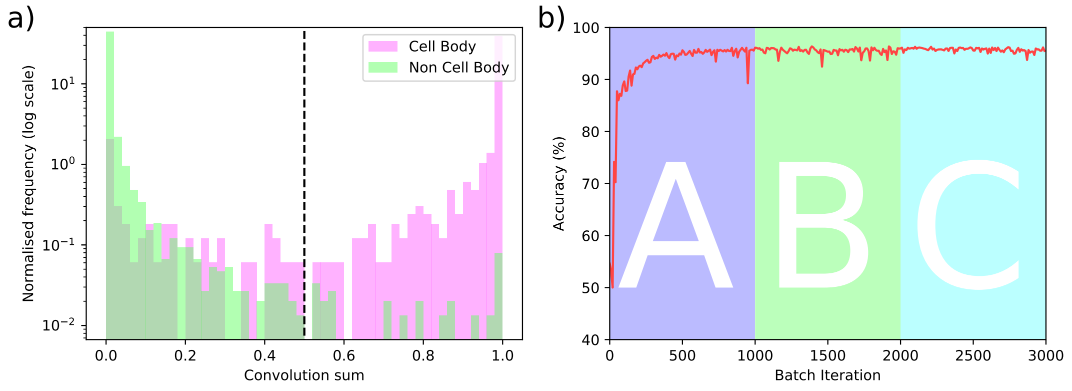


Figure 4.5: Convolutional Neural Network Results
a) Histograms of separation of the scores of the test set (note the logarithmic y-axis). b) Test set accuracy during training. The training rate was reduced by a factor of 2.5 between A and B, and 4 between B and C.

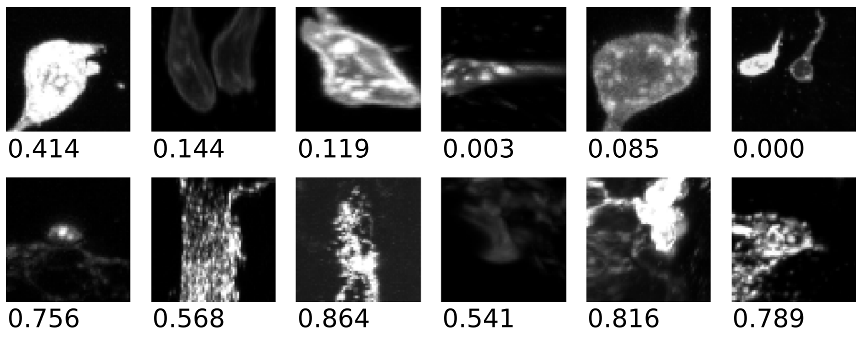


Figure 4.6: Errors Of Convolutional Neural Network
Examples of incorrect predictions from the CNN. Top row: false negatives (cell body ROI classed as not containing cell bodies by the classifier). Bottom row: false positives (*vice versa*). The predicted probability of containing a cell body is shown below each image. Note the double cell bodies and the very large cell bodies in the false negatives and the large high intensity regions in the false positives.

Table 4.3: Cell Body Detection Validation Results

Prediction \ Actual	0	1	2	>=3	Total
0	27	5	0	0	32
1	3	28	3	3	37
2	3	6	8	2	19
>=3	2	0	3	7	12
Total	35	39	14	12	100

4.2.5 *Validation*

It is important to use a "hold-out", validation set to report the accuracy of a method to account for any over-fitting to the test data set since the test set was used to judge the performance of varied hyper parameters (e.g. the model used). I decided to do this by applying the CNN as part of the process I required it for, i.e. detecting cell bodies in entire images rather than ROIs. Therefore, I created a further set of ROIs from a validation set of image stacks. For the neural volumes in these stacks ROIs centred on all points in the entire skeleton that were not more than 10μm inside the neuropil were fed into the CNN (as cell bodies should not be inside the neuropil unless the registration is very bad). As there were still many false positives, I decided to increase the threshold at which to class voxels as cell body voxels from 0.5 to 0.9. I expected to lose some of a particular cell body's ROIs but maintain enough higher scoring ROIs. A single cell body was then classed as a collection of 4 or more predicted cell body ROIs that were all centred on voxels that, if cubes of width 6 voxels (in each dimension) were centred on them, the cubes would all be touching or overlapping as a single object. The centre of mass of these points in a collection was given as a cell body centre.

The validation results show this technique is not especially accurate at predicting the number of cell bodies in a neural volume, at 70%, although it can give a rough estimate that corresponds approximately with the actual distribution. See figure 4.6 for error examples. I did not judge this accuracy to be high enough for complete automation of detection of number of cell bodies in a neural volume. Given that cells sometimes include a cell body that is cut off the scan (and hence neural volume) or the segmentation algorithm may miss a cell body, even if this method was 100% accurate it would make some mistakes. For these reasons I did not explore this method further and I determined that the tool would be best used to bring human attention to parts of the scan where there were likely to be cell bodies, such as in the image in figure 4.7a.

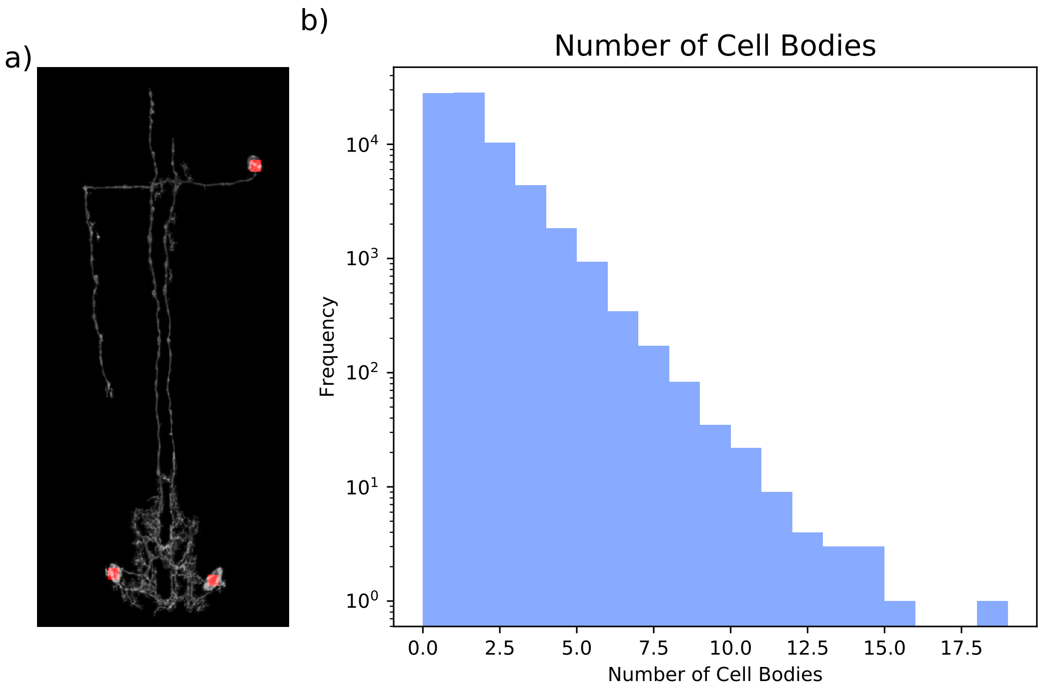


Figure 4.7: Validation of Cell Body Detection
a) Example of successful cell body detection on a VNC neural volume image, labelled in red. b) Histogram of number of cell bodies detected for all satisfactorily registered neural volumes over 50 000 voxels (n=74 655.)

This was indeed the case when manually reviewing neural volumes in the following chapter. Estimates for the entire data set are nevertheless given in figure 4.7b.

4.3 PRUNING

As noted in the previous chapter, inspecting neuron morphology reveals that many of the smaller non-arborising branches are not consistent among neurons of the same cell type. Much of the highly arborising (and highly branching) dendritic regions also posses greatly variable local morphologies (and hence traces) among specimens of the same cell type. These are also the regions where the skeletonisation tracing was most oversimplified and smaller branches were often unrepresentative (see figure 3.29). As a result of these considerations, I decided to simplify the skeletons. I did this with the aim of creating more consistent traces among cells of the same type as well as reducing the overall size in computer memory of the skeletons by removing many short (and hence low information) branches. This branch removing process is known as *pruning*.

A form of pruning is employed by some published tracing methods to deal with over-representative skeletons of neurons Acciai et al. (2016). Neural morphologies tend to be less stereotyped (i.e. less similar to others of the same type) the more branch points one traverses along the arbours away from the cell body. This idea has been used in applications which have simplified skeleton representations of cells based on Horton-Strahler number Horton (1945) where points are ignored after a specified number of branch points away from the cell body. I experimented briefly with this method but, due to the lack of inclusion of some important sections following regions of high branching and also difficulty identifying a unique cell body for a neural volume, I decided to implement an alternative pruning algorithm.

My chosen method would keep more information about the structure, specifically skeleton paths that represented the overall structure across all regions the skeleton was present in, although not necessarily local structure. I desired enough information about the variation across the image stack of the overall structure of a cell to be able to differentiate between stereotyped cell type rather than information about local variations. I therefore intended the algorithm to preserve long paths and branches to distant end points but remove shorter ones emanating from them. As I believe I had calculated whole cell skeleton representations for most of the approximately 5 000 different cell type pairs in the larval CNS and these features appeared to vary significantly between them, I was confident that this would be a good way to separate them.

Despite satisfactorily-appearing and fast 2D skeleton pruning algorithms, e.g. Solís Montero and Lang (2012), I was not able to come by a satisfactory similar, independent 3D implementation. For example, the pruning method of FIJI's Analyze Skeleton plug-in Arganda-Carreras et al. (2010), removed long branches to end points which I wanted to maintain. I hence devised the following method to identify skeleton voxels to keep in the pruned skeleton, which relates strongly to some of the tracing method pruning algorithms such as Xiao and Peng (2013) and Lee et al. (2012). I implemented the below algorithm by writing it in the Python language myself, giving myself more control over tuning the algorithm for my purposes.

First, in order to speed up the pruning calculations, I represented the skeleton as a graph with nodes and weighted edges. The nodes were any voxel in the skeleton that had exactly one or more than two skeleton voxels in their 26-nearest neighbour voxels ($3 \times 3 \times 3$ cube). An edge represented any direct path between these nodes along only skeleton voxels with exactly two 26-nearest neighbour skeleton voxels. The weights were sums of the Euclidean distances from voxel-centre to voxel-centre travelled along these paths.

Next, I attempted to find one of the (potentially many) cell bodies in the neural volume - a good predictor of a cell body location was the node in the cortex part of the skeleton furthest in straight-line distance (in voxel widths) from the neuropil-cortex boundary. In the validation set above for cell bodies, out of 65 neural volumes containing cell bodies, only 4 of these predicted cell bodies were not either in or very close to - and on the same axonal tube as - a cell body. I considered this sufficient to identify an "origin" point from which, for cells of the same type, a skeleton path would emanate that entered the neuropil at a reasonably consistent point across specimens. Using a detected cell body from the previous section did not improve these results. For skeletons without a point outside the neuropil, the point closest to the neuropil boundary was designated the origin.

I used the edge weights as the distances between nodes in order to find the shortest path from the origin node to every other node in the skeleton graph (using Dijkstra's algorithm). This enabled me to rank all nodes by length of shortest path from the origin. I then identified which node was furthest from the origin in this ranking and all voxels and nodes in the "longest shortest" path, to this furthest node, were then designated to be kept as part of the pruned skeleton. Next, these voxels marked to keep were removed from all remaining shortest paths and their path lengths updated in order to give the shortest paths from the kept skeleton to the remaining unkept nodes. The new longest shortest path was kept and the process of updating the paths and finding the next longest iterated until there were no paths remaining over length 28.3µm (100 x-y voxel widths). This value was chosen after careful visual comparisons of the pruned representations for cells of the same type and comparisons of the pruned skeletons to the raw staining such that the vast majority of the pruned skeleton accurately traced the centre of neural arbours that were responsible for the broad structure of the neural volumes. The level of pruning, however, will be discussed later in relation to the end goal of enabling the discovery of groups of cells from the same type.

The above algorithm enforced a tree structure without any loops (i.e paths in the skeleton that connected back to previously visited nodes), which for most neurons is appropriate (See figure 4.8). Indeed, the algorithm successfully removed the many small loops in the structure as well the small cavities that were represented as globules in the maximum projections of the input skeletons (see figure 3.29). Cell arbours becoming very close again after a period of separation, however, created larger holes

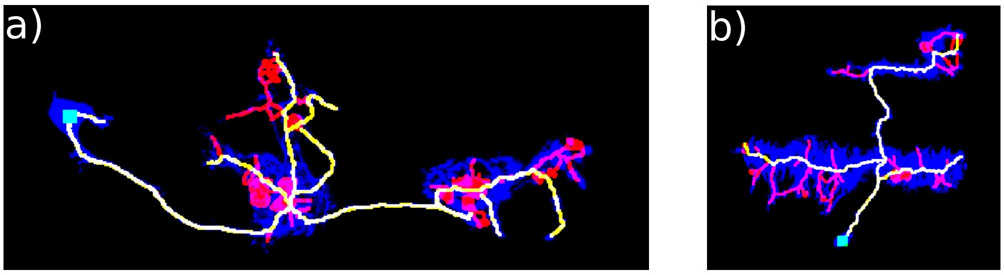


Figure 4.8: Skeleton Pruning Examples
Maximum z-projections of: a) CB neuron b) VNC neuron. The red and yellow skeleton is the original, the red removed and yellow kept. The green point is at the origin. The blue channel is the raw, registered signal. Note in a) there are some blotched red zones of the skeleton (representing enclosed holes from the original volume) that are removed upon pruning.

in the thresholded neural volumes and this resulted in the skeletonisation algorithm preserving them as larger loops. I therefore decided to include any large loops made by keeping long edges (over 14.15µm) that were not yet kept but whose two end nodes had been kept. This was to preserve more information about the cell morphology in a situation where it was not clear which end of the loop was a branch point and which an adjacent passage.

4.4 TUBENESS

As mentioned in the introduction, I wanted to classify parts of the skeleton as arborisation or non-arborisation, tubular regions. Partly this was since the accuracy of the skeleton tracing appeared much better in the tubular regions and also since the tubular regions demonstrate more consistent skeleton path direction vectors between specimens of the same cell type. Hence I could perhaps use these distinct regions in different ways when determining the similarity between two cells, such as requiring similar direction vectors in the tubular regions and focusing more on domain overlap for arborisation regions. In the following work, however, this was mostly useful in creating an average representation of a cell type.

I performed this using the "tubeness" metric Sato et al. (1998) that has also been used successfully for adult *Drosophila* neural data in Longair (2009) and Masse et al. (2012). One convolves a representative volume around each skeleton voxel with the derivatives of a normalised Gaussian, G - see figure 4.9a, in order to calculate a matrix,

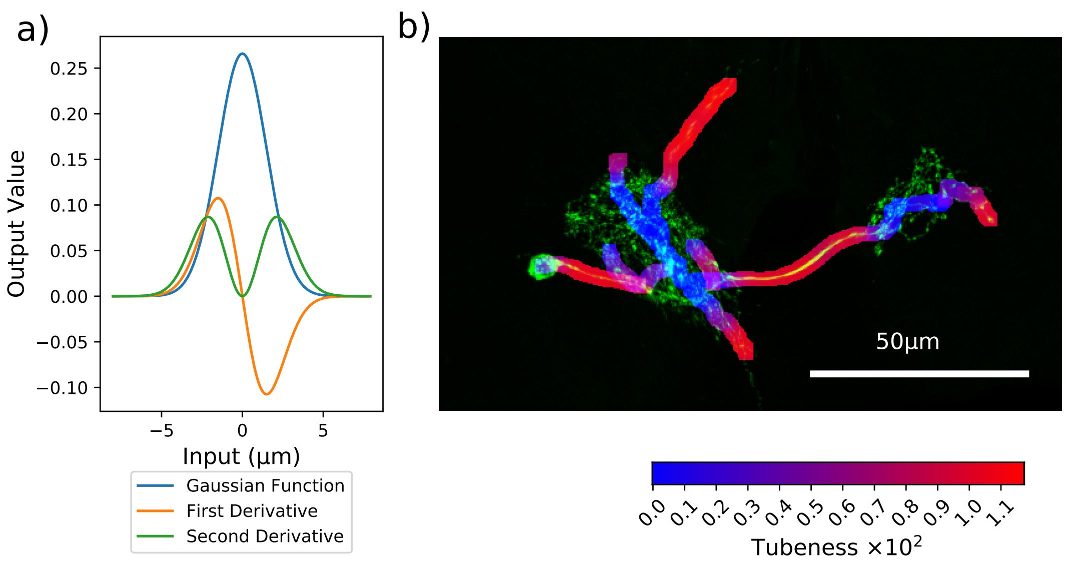


Figure 4.9: Tubeness Example
a) Gaussian functions, at scale used (standard deviation 1.5µm). b) Maximum z-projections of neural volume signal (green) with overlay of skeleton points (enlarged) shaded to tubeness score.

I, (similar to the Hessian matrix) that contains the values of all partial 2nd derivatives in the three dimensions:

$$I_{i,j} = \left\{ \frac{\partial^2}{\partial d_i \partial d_j} G(\mathbf{x}, \sigma) \right\} * I(\mathbf{x}) = \frac{\partial}{\partial d_j} G(x_j, \sigma) * \left\{ \frac{\partial}{\partial d_j} G(x_j, \sigma) * I(\mathbf{x}) \right\} \tag{4.6}$$

where i, j are matrix indices and d the set of dimensions x, y, z and where a higher σ, the Gaussian standard deviation, detects thicker tubes. The tube is directed along the least Gaussian-shaped direction, i.e. the direction of the highest second derivative, which is the eigenvector of the highest eigenvalue, λ_1, of the matrix. A tubular voxel will have highly negative second derivatives in these two perpendicular eigenvector directions and hence negative eigenvalues, λ_2 and λ_3. Therefore the tubeness score, T can be calculated for tubular voxels as:

$$T = \sqrt{\lambda_2 \lambda_3} \tag{4.7}$$

Non-tubular voxels received a score of zero.

I implemented the tubeness score in python myself, using scipy library functions. This was in order to account for the non isotropic voxel dimensions. As input I used the thresholded neural volume after smoothing it by a Gaussian of standard deviation 1µm to reduce the size of the Gaussians required in the algorithm (1.5µm) and hence also significantly the time required. These values were adjusted to optimise a visual inspection of examples such as in figure 4.9b. This figure illustrates a typical example showing that this method ran very satisfactorily on the numerous examples that I checked prior to applying it to all neural volumes.

4.5 MATCHING NEURONS

A range of neuron-neuron matching methods have been developed, mostly based on analysing skeletons to match them to one another entirely as one or in constituent parts: The skeletons can be condensed into a low dimensional representation for direct comparison, for example by creating vectors of global morphological features (e.g maximum path length, number of branches, etc.) Scorcioni et al. (2008) Wan et al. (2015) which does not use positional data (and hence does not even require registration). Alternatively, a condensed representation string can be trained on a sample set such that the binary distance between two cells represents their similarity Mesbah et al. (2015). These quick approaches can often act as an initial stage in the process of finding more cells as it is fast but often less accurate.

More detailed comparisons rely on optimising the pairing up of specific parts of the skeleton (e.g. curves in *Path2Path* Basu et al. (2011)) by calculating local morphological similarity metrics. The method of neuron BLAST (nBLAST) Costa et al. (2016) disregards optimising pairings and simply calculates positional and directional metrics for points in the query skeleton and their nearest neighbours in the comparison skeleton. Then robust statistical methods are applied to sum these up to a cell-cell similarity metric. More involved methods attempt to match the whole skeleton structures consistently, usually treating the skeletons as a tree graph, for example *Tree Edit Distance (TED)* Heumann and Wittum (2009). Other methods Gillette et al. (2015) look purely at tree branch matching (without taking account of positions and directions). Some methods combine tree and positional data such as *BlastNeuron* Wan et al. (2015) which performs a global search and then treats segments as tree branches that are matched using nearest-neighbour distance between segments to build up a near-optimal global tree matching. This is a similar strategy to Mottini et al. (2015) which uses a different similarity metric.

In order to assess the quality of the matching algorithms that I investigated, I created an evaluation set from the dataset of retained, satisfactorily high enough accuracy-scoring neural volumes from chapter 3. This set contained some mirrored neurons (as these should be returned as well) and was grouped into 13 subsets by neuron type such that each subset contained the majority of (though not necessarily

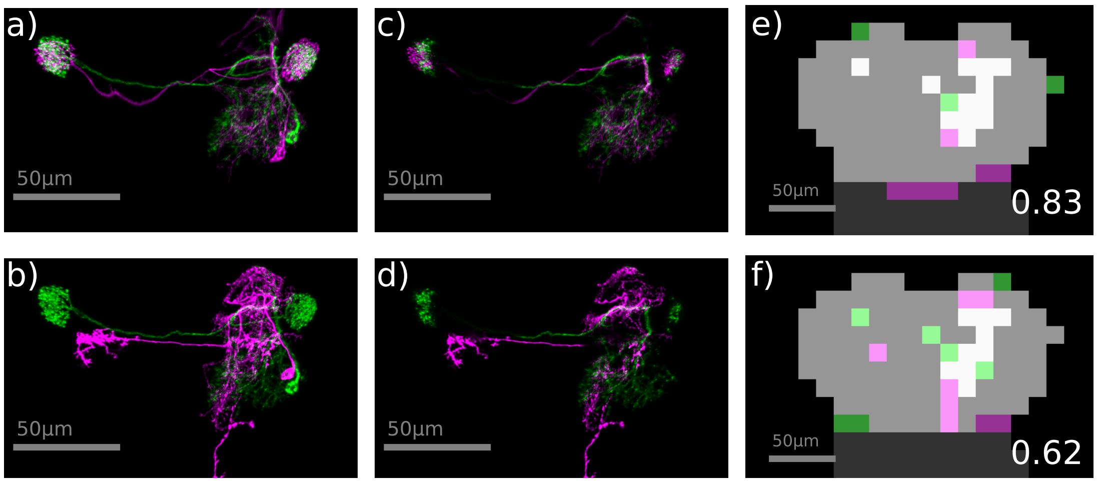


Figure 4.10: Fixed Point Boolean Comparison Example
a), b) Maximum z-projections of query neuron (subject of search, green) and candidate neuron (magenta). c), d) 20µm width maximum z-slice projections about the slices for points shown in e), f). e), f) Single z-slices of values of points in the fixed point boolean comparison method. The dark grey background indicates the interior of the neuropil mask (black is outside). The volume covered by each image is a darker green (query) or magenta (candidate) (when covered by both this becomes a light grey). The lighter green and magenta are the points containing signal for the query and candidate neurons respectively with overlap white. Pair-wise scores are shown in white text.

all) the cells of a type where the subset's members are indistinguishable in gross morphology in LM. Each neural type subset contained at least 3 examples. Each subset was further split into two: neural volumes containing single neurons and multiple neurons (with up to 10 single neuron cells per subset). Finally, in addition, some sets contained a related group of neurons - dubbed "unsure" - for which complete morphological indistinguishably was debatable.

The following methods were judged on their ability to take as input one *single neuron* volume within a type subset and find other neurons from within the entire retained dataset (16 339 CB and 19 376 VNC cells). Each pair-wise comparison for a query neural volume to all candidate neural volumes produces a single score that enables ranking of all candidate cells in the dataset.

4.5.1 *Fixed Point Boolean Comparison Matching*

I first developed a quick and simple similarity method by condensing the 3D volume of the template space into a smaller 1D array, in the spirit of Mesbah et al. (2015). Rather than train a representation on the overall morphology, however, due to the

limited training data I had, I decided to create a simple condensed representation as follows: For each neural volume, I aimed to reduce the number of dimensions to approximately 800 and so selected points on a regular grid (of spacing 50×40 voxels) within the neuropil mask and the following values at each grid point were calculated as:
0 for points outside the bounding box of the float image in the template space
1 for within the bounding box but not within the neuron volume
2 for exclusively within the neuron volume (in the case of expansion of neural signal on the edge of the bounding box)
3 for within both the neuron volume and the bounding box

See figure 4.10. For each volume, these values were stored in a row of one large 2D array for each of the CB and VNC separately with 834 and 739 points (columns) respectively.

The pairwise comparison score of a query neuron against each data-set neuron calculated the amount of corresponding 1D array entries that were 3 in both cells as a proportion of those that were 3 in the query neuron only. Hence higher comparison scores meant more similar neural volumes. This simple, fixed point comparison was fast as the positions in space that each array entry represented were the same for all of the neurons so no searching for matching positions was required. Thus each pair-wise comparison involved less than 850 comparisons and so over 30 000 of these comparisons took less than 0.15 seconds of CPU time on a single thread of a typical desktop computer.

For each query cell in the test set I thus had a ranking of all cells in the dataset by this similarity metric. Prior to selection of grid points, the binary neural volumes were twice dilated using a 3D Gaussian filter of standard deviation 0.88µm and a threshold of 1E-6 was applied following this. This equated to an approximate perpendicular expansion of the volume surfaces by 6µm. This was to ensure that thin and small regions of the volumes were more likely to be picked up by the grid points. I tuned the Gaussian smoothing width using the similarity measure test set, optimising the amount of the (same region) data set that I could exclude by setting a universal threshold in the similarity metric that would also not exclude many of the same cell matches for any of the query cells (excluded matching cells were weighted 10 times more important an error than included non-matching ones). Hence I was able to quickly exclude, on average over each query neuron, just over 91.5% of the volumes in the dataset as not containing a matching cell (see figure 4.11). This was achieved

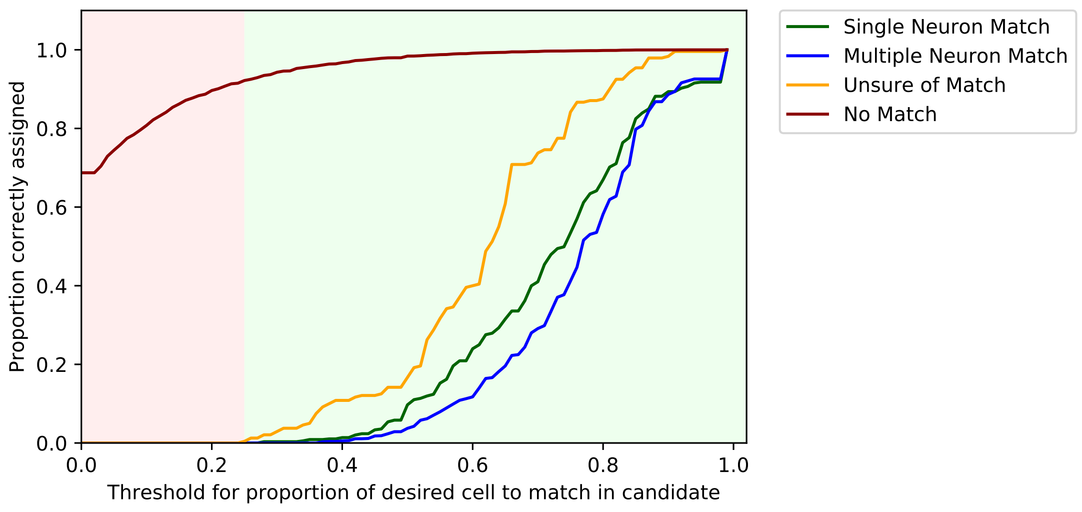


Figure 4.11: Fixed Point Boolean Comparison Matching Results
Averaged across each query neuron in the evaluation dataset, plots for the proportion of cell-cell pairwise comparisons for a group of candidate neurons that score above a specific threshold (x axis). The threshold at 0.25 represents the proportion of points occupied by the desired, query cell that must also be occupied by the candidate. Note that some query neurons match completely with candidates and receive scores of 1.0, which leads to the steep jump in the single and multiple neuron match lines at 1.0.

by applying a comparison score threshold of 0.25, above which, all of the single and multiple matching cells were kept for the more detailed analysis that follows.

4.5.2 *Matching Neurons with BLAST Scores*

At this stage, due to the large amount of neurons still remaining (of the order of thousands) and considering that it was desirable to have a system that could be run with new data in real-time, the requirement for a fast system outweighed the desire for coherently paired points across the whole of the skeletons. This meant I did not consider consensus skeletons produced by tree matching algorithms that take of the order minutes to compare two skeletons to one another Costa et al. (2016). These tree-matching algorithms are also generally designed for higher quality (usually EM), *single*-cell *tree* graphs rather than coarser skeletons and *multiple*-cell neural volumes that potentially contained loops and have been heavily pruned. I therefore opted instead to investigate further a method based on the work of Costa et al. (2016), namely nBLAST, that was itself inspired by the Basic Local Alignment Search Tool (BLAST) method of gene sequence alignment Altschul et al. (1990). The BLAST methods, as will be explained, can easily be trained with a small test set to suit the dataset I have.

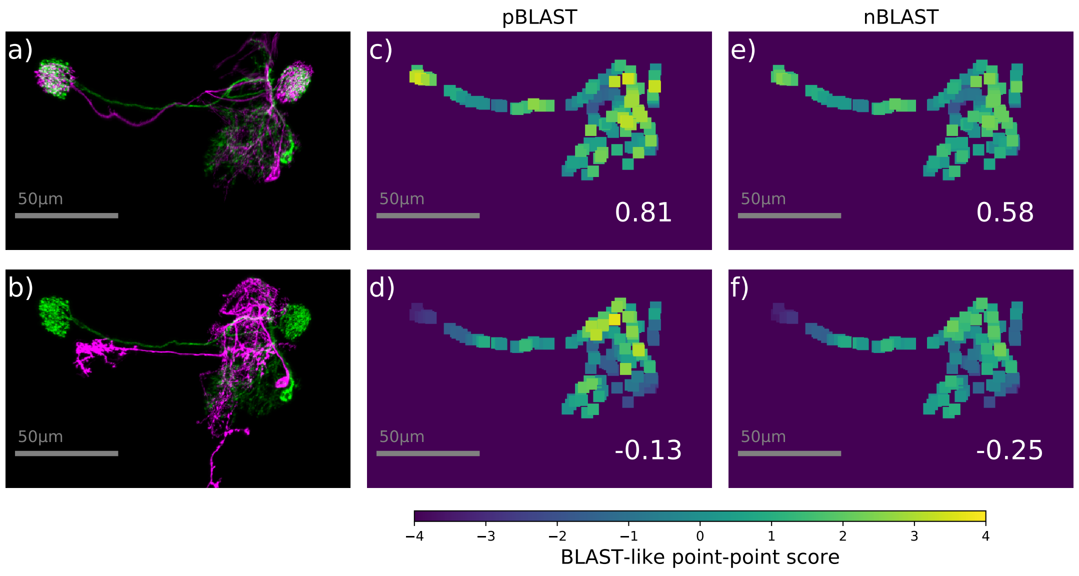


Figure 4.12: BLAST Score Examples
a), b) Maximum z-projections of query neuron (subject of search, green) and candidate neuron (magenta). c) - f) Maximum projection heatmaps of c), d) pBLAST scores and e), f) nBLAST scores with final comparison scores shown in white.

For my BLAST method investigations, I looked at several local comparison metric BLAST scores between two points in the query and candidate neural volumes. I averaged these scores over all the points I calculated this metric for in the query cell to give a total skeleton-skeleton score to judge the similarity of two skeletons, and hence that of two neural volumes. First I randomly selected a specified number, n, of points from each pruned skeleton. For each of these selected points from a given query skeleton, I found the nearest selected point in each candidate skeleton and calculated metrics based on these two points. Restricting the number of points for each skeleton to a limited number (n) of randomly selected points enabled more efficient data storage and quicker skeleton-skeleton calculations as there were less point-point calculations performed.

I quantified the success of the BLAST methods by, for each query neuron entered into the search method, first ranking the remaining candidate cells by BLAST score. Then I recorded the proportion of other neurons within the subset that were found in the top Q number of results. Q was varied but I focussed on the value of 50 as I believed a researcher could explore 50 maximum-projection results in a reasonable amount of time.

The first metric I investigated was the distance between the two points (one from the query volume and its nearest neighbour from the candidate) in the registered, template space. For any *point-point* pairwise *metric*, however, it was required to convert them to a *score* that I could then use to find the average across all points. This was so that each point-point score would represent the significance of whether the points were from similar or non-similar cells rather than just the actual metric value. For example, for the distance metric a separation of 100µm is probably as useful an indicator of bad similarity as 150µm, but 0µm suggests much better similarity than 50µm, despite the same difference. Following the method of BLAST, each metric was converted to a point-point BLAST similarity score, B, from probabilities, p, as follows:

$$B = \log_2 \left(\frac{p(\text{points from same cell type})}{p(\text{points from different cell types})} \right) \tag{4.8}$$

The only way to calculate these probabilities was to do so empirically based on the data I had. For this I created a training set that was taken from the test set for registration accuracy in chapter 3 where, for 15 of the cell types, there were from 3 to 5 single-neuron skeletons which matched well. For the matching cells, I produced a histogram of the point-point metric of interest for pair-wise comparisons within the subset, and similarly a histogram for non-matching cells was made for the same metric when these skeletons were queried against all other skeletons in the data set with a fixed point boolean comparison score above 0.25. Normalising the histograms gives estimates of the probability distributions in equation 4.8.

Considering the distance, D, between the query skeleton point and the nearest available candidate skeleton point, I constructed a 1D histogram (see figure 4.13) and fit a 4-degree polynomial on the separation distance to this histogram for finding a suitable cut-off point and limit for the curve in high distance separations where little data had produced a highly varying histogram. The fit was performed using ridge regression (minimising ordinary least squares with L_2 regularisation). The resulting formula for this positional BLAST (pBLAST) score with $n = 150$ was:

$$B_p = \begin{cases} 4.22 - 0.95(\mu m)^{-1}D + 0.056(\mu m)^{-2}D^2 & \\ -1.7 \times 10^{-3}(\mu m)^{-3}D^3 + 1.7 \times 10^{-5}(\mu m)^{-4}D^4 & \text{if } D < 25\mu m \\ -3.25 & \text{otherwise} \end{cases} \tag{4.9}$$

I decided to use polynomial approximations such as above for calculating this and other BLAST scores in my future implementations due to similar performance to ac-

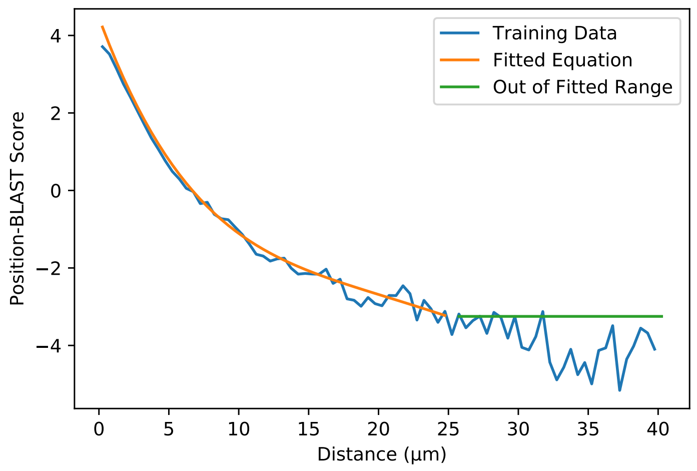


Figure 4.13: pBLAST Functional Mappings
Histogram of distance of nearest neighbour separations and approximating polynomial function used. Histogram bins were 0.5µm width and ranged to a separation of 40µm.

tual histograms, the ease of interpretation in the final calculations and the better interpolation in low density regions.

With the test set I experimented with a range of different values for the number of points, n, (see table 4.4) and found that my best average scores for finding matching cells in both multiple and singular neural volumes was 150. I also found using the function in equation 4.9 and the histogram value very slightly improved the results. Using 4.9 and ranking all candidates by average point-point B_p score in each pair-wise skeleton comparison successfully found, on average, 85.7% of the other single cells from the subset in the top 50 ranked candidates and 78.1% of the cells in multiple-neuron volumes. See figure 4.12 for an example of the scores between cells of the same and different types.

The single-cell performance could be improved by also calculating a reversed B_p score from candidate to query neural volume (i.e finding points in the query volume nearest to the candidate volume's points and calculating the average B_p score for these point-point combinations). Ranking was then performed on the sum of these average B_p scores from both "directions". This improved the average proportion in the top 50 by 6.8% to 92.5%. To explain this, some branches of skeletons in some of the query cell were not present or as long as those in a matching candidate, whereas a higher proportion of these candidate points would be represented in the query skeleton giving a reverse average B_p score that would be higher than the standard score. There was an opposite effect on finding the matching cells in larger, multiple

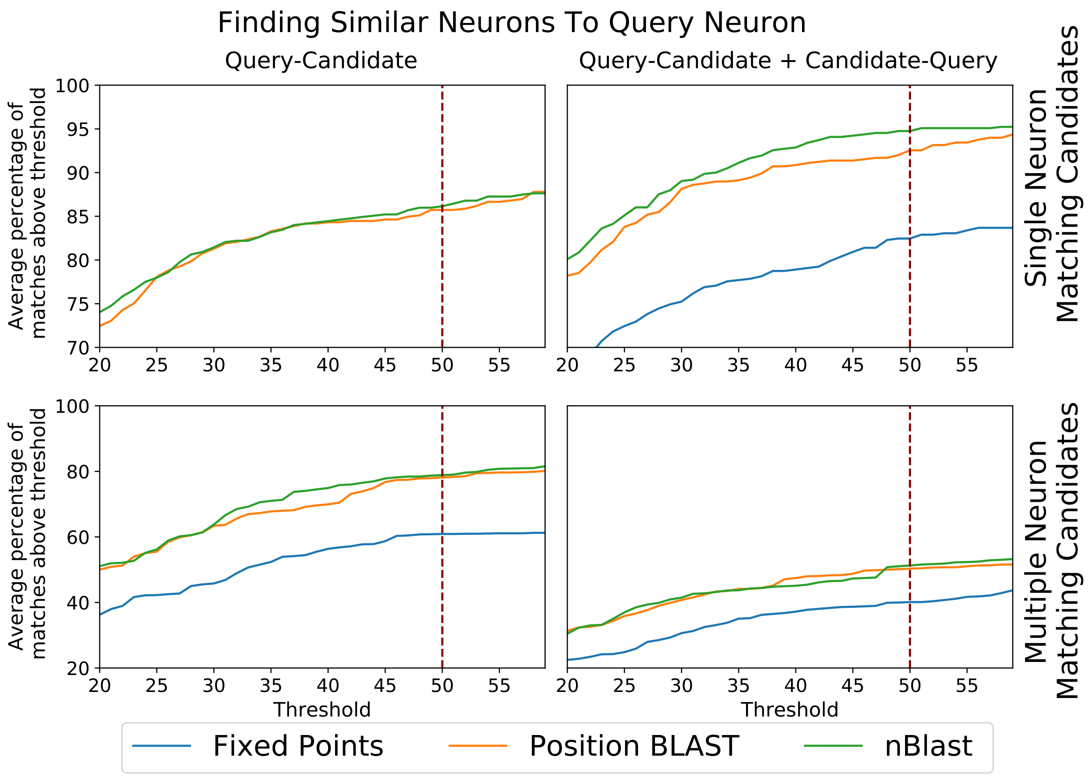


Figure 4.14: Percentage of Matching Cells Found Above Threshold for BLAST Methods
Top row: Single neuron volumes of same-type neurons, Bottom Row: Matching neural volumes contain multiple neurons, one of which is the cell of the same type as the query. Left column: Similarity scores are from query to candidate. Right Column: BLAST-like final ranking scores are the sum of scores in both directions. Fixed points scores are the multiple of both directions. Both BLAST scores use the fitted interpolation equations (equations 4.9 and 4.10). Note the marginal improvement of nBLAST over pBLAST but the significant improvement over the fixed points method of section 4.5.1. Also note the different y-scales between top and bottom graphs.

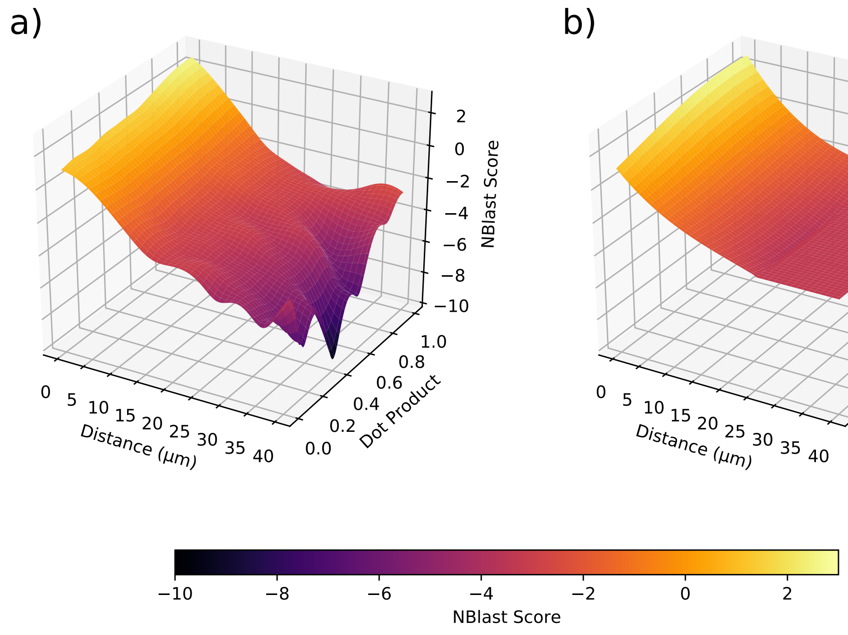


Figure 4.15: nBLAST Functional Mappings
a) Smoothed 2D Histogram for nBLAST score from distance of separation and direction dot-product of nearest neighbour points in the BLAST training set. b) Approximation of histogram as explained by formula 4.10.

neuron skeletons for the same, reversed reason (the query cell was missing much of the candidate).

I next supplemented the distance metric with the dot product of the local skeleton direction calculated in a similar way to that of Costa et al. (2016): I first tabulated a matrix, M, of the 3D positions of the η other points in the pruned skeleton that were within 3 point traversals from the point of interest. The mean position was then subtracted from each of the 3 dimensions of this $3 \times \eta$ sized matrix. The local direction was then calculated by finding the right singular vector with the largest singular value in the Singular Value Decomposition (SVD) of M Masse et al. (2012). This is equivalent to finding the principal axis of the moment of inertia about the centre of mass of the selected points, i.e. finding the eigenvector with the largest eigenvalue of the 3×3 matrix $\mathbf{M}^{\mathsf{T}}\mathbf{M}$ (which I implemented to perform the calculation).

Now I calculated from the training set normalised 2D histograms combining the distributions of both the separation distances and dot products of the local directions, one with matching and one with non-matching point-point comparisons (see figure 4.15). Dividing the matching histogram by the non-matching one produced the 2D nBLAST histogram with estimated probabilities. I used bins of separation 0.01

Table 4.4: Blast Neural Matching Results
Average percentage of known matching neural volumes ranked in top 50 candidates:

Metric	Volume Type	Number of points (n) and function type						
		50	100	150	300	1000	150P	150F
pBLAST QC	Single	86.4	86.9	85.6	84.8	81.5	78.1	85.7
	Multiple	62.0	71.1	78.1	78.3	81.1	64.8	78.1
pBLAST QC+CQ	Single	92.4	93.3	92.3	92.1	92.3	86.8	92.5
	Multiple	37.4	45.5	50.3	52.9	54.1	42.0	50.3
nBLAST QC	Single	87.4	86.8	86.5	84.9	82.5	78.1	86.1
	Multiple	67.5	76.6	79.3	79.9	77.9	65.5	78.8
nBLAST QC+CQ	Single	92.6	94.8	94.9	94.6	94.9	88.2	94.8
	Multiple	37.9	46.8	51.2	52.2	53.5	40.2	51.3

Average median position in candidate ranking of known matching neural volumes:

Metric	Volume Type	Number of points (n) and function type						
		50	100	150	300	1000	150P	150F
pBLAST QC	Single	24.6	22.4	24.8	30.8	36.4	53.2	24.7
	Multiple	63.8	57.2	48.7	47.1	40.9	111.2	50.1
pBLAST QC+CQ	Single	24.1	23.9	23.0	23.0	22.4	33.1	23.0
	Multiple	166.3	126.8	111.3	109.8	95.8	141.4	112.4
nBLAST QC	Single	23.2	21.7	23.8	28.2	33.0	49.9	24.1
	Multiple	55.9	48.9	42.7	43.2	38.1	102.6	41.8
nBLAST QC+CQ	Single	22.7	22.7	21.4	20.8	20.9	30.5	22.0
	Multiple	163.1	122.8	111.4	109.1	99.7	126.0	112.3

Median position refers to the rank position of the medianly ranked of the (single or multiple) candidate neural volumes that contain the same type cell as the query. For example, in a group of 5 single-cell neural volumes where for one query cell the 4 others have ranks 5,6,9,30, the median position is 7.5. The median position is averaged over all query cells. Here *QC* refers to only Query-Candidate average BLAST scores whereas *QC-CQ* refers to adding the reversed Candidate-Query average BLAST score for ranking. The distance histograms and smoothed 2D histograms were used directly (with varying thresholds on distance and values for the highly variable regions) other than for *150F*, where the function described in equations 4.9 and 4.10 have been used for 150 points. For *150P* I used skeletons that were pruned further with branches less than 70.75μm removed.

in direction dot product, V, and 0.5μm in distance, ranging up to 50 μm. This balanced accuracy with the need to reduce the number of empty cells in the histograms. Nevertheless, both the matching and non-matching histograms were smoothed by a Gaussian with standard deviation 5 bin-widths to fill in gaps in the sparse regions, and hence avoiding a division by zero. The approximating polynomial formula, fitted by the ridge regression method, was then given for 150 points as:

$$B_n = \begin{cases} 1.20 - 0.34(\mu m)^{-1}D + 2.61V + 0.0011(\mu m)^{-2}D^2 & \\ -0.36(\mu m)^{-1}DV + 0.82V^2 - 1.7 \times 10^{-4}(\mu m)^{-3}D^3 & \\ +4.1 \times 10^{-3}(\mu m)^{-2}D^2V + 0.23(\mu m)^{-1}DV^2 - 1.81V^3 & \text{if } D < 25\mu m \\ -3.25 & \text{otherwise} \end{cases} \tag{4.10}$$

Replacing the polynomial part with the values from the histogram bins gave near-identical results (see table 4.4). In general, there was a very modest improvement in using nBLAST over pBLAST, as shown in the table and figure 4.14. Specifically, for the 150 points implementation, on average only 0.4% more single and 0.7% more multiple matching cells were found in the top 50 using only query-candidate scores, which I did not consider a significant amount. When adding the reverse scores, however, 2.3% more were found, giving a total success percentage here of 94.8% and demonstrating a clear, albeit small, improvement.

Table 4.4 also demonstrates the effect of varying the numbers of points used, with 150 shown to be near-optimum for this data. It also records the average position of the median ranked cells from the same type as the query. These results are in line with those of the percentage in the top 50.

I also investigated using a mixture of the two BLAST scores for 150 points. For those points in the query neuron with tubeness scores over 0.005 I used an nBLAST score (as tubes have well defined directions), and those below I used a pBLAST score (with new functions recalculated as such from the training set to reflect this). The average point-point score was then calculated for each comparison, giving an average proportion of 80.1% (median position 30.9) for matching single and 65.6% (median position 50.4) for matching multiple cell volumes in the top 50 for the test set (without reversing). As this was less than with using nBLAST alone, I did not pursue this further but instead searched for a different approach to attempt to improve the scores, especially

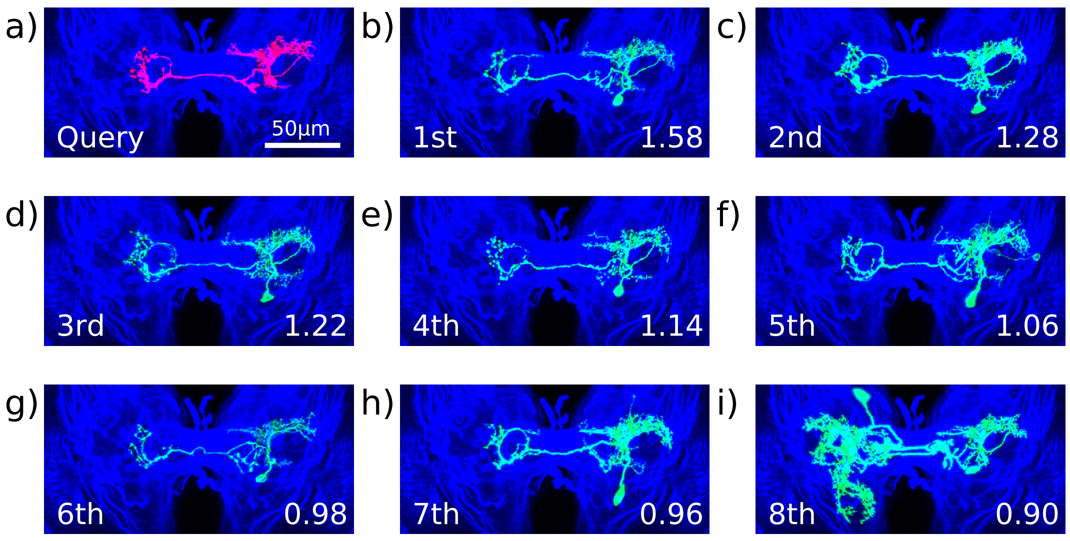


Figure 4.16: nBLAST Example Ranking
Maximum projections of a) The query cell and b) - i) the candidates in order of ranking by nBLAST QC. i) contains a multiple neuron volume.

for the mono-directional query-candidate search (see the next subsection).

I also investigated further pruning (and hence further simplification) of the skeletons but this did not help. Setting the minimum path length to 70.75µm in the pruning algorithm decreased substantially the percentage of matching cells in the top 50 query-candidate results when using 150 points - by 7.5% or more for both nBLAST and pBLAST (see table 4.4). Therefore, again, I did not focus my time further here but explored further similarity metrics.

Each skeleton-skeleton average BLAST-score calculation took approximately 1ms, with the bulk of the calculation time spent finding the nearest neighbours. I implemented this in a C-function and ordered the candidate volume points in the x direction so that only points within 25µm in the x-direction could easily be selected and considered, reducing the required number of point-point distance calculations without affecting the results. This ran at over ten times the speed of my simpler python implementation. I containerised the BLAST searches into a REpresentational State Transfer (REST) Application Programming Interface (API) (the *Software as a Service* model) and made it accessible to collaborators through a password protected web server. The results of queries were similar to those shown in figure 4.16, with additional information about the GMR line the neural volume was produced from.

4.5.3 *Matching Neurons with Machine Learning*

I attempted to improve the rankings of the previous subsection by replacing the directional metric component of the point-point BLAST score with a comparison of the condensed representation of the actual signal around each skeleton point. My aim was to create a low dimensional representation that would also take account of skeleton points in regions of high variation or branching where there are less consistent directions between samples (see figure 4.20k for example). This could be done by learning to represent points in these regions as separate classes of point. I employed several M/L methods for this.

I created a further, larger training set especially for training this metric consisting of 52 single cell type subgroups (184 single cells in total) without including any neural volumes from the *test* set used in 4.5.2 (and used later here as a validation set for the M/L model). As there are now several sets of neural volumes, Appendix A.5 summarises them for the reader.

Next, up to 1 000 random skeleton points from within each cell were selected. Around each of these points in the registered signal image stack a cube ROI of size $301 \times 301 \times 101$ with slices every $4 \times 4 \times 2$ voxels was taken giving cubes of size $76 \times 76 \times 51$. (Prior to this any values below a threshold were set to zero to remove the background, with the threshold calculated as in section 3.6 with 0.15% instead of 0.1%, max 25, min 5). The maximum z-projections of these sliced cubes were taken to give "projection windows", similar to those used in 4.2.4, and a histogram equalisation was applied to each of these windows (using the OpenCV Library Bradski (2000)). Finally, they were normalised by subtracting their individual means and dividing by their standard deviations. I chose these parameters in order to balance the requirements for low memory usage and maintaining reasonable coverage of local surroundings as well as to attempt to counter for variations in image contrast. The projections appeared to give most of the required information to identify a class a point as detailed above (more detailed and larger windows did not appear to aid the M/L accuracy in tests).

Firstly, I reduced dimensionality in the projections by converting them to a 10-dimensional Principal Components Analysis (PCA) representation, $\mathbf{c}$, calculated on the above described training set. I created a M/L test set where I took from the BLAST training set 21 016 triplets of 3 windows where, compared to the first window's centre point in the triplet, the second is "matching" and third is not. Here a matching

Table 4.5: Machine Learning Neural Matching Results

Metric Type	Vol. Type	Percentage in top 50					Median position				
		Non ML	ML Blast		ML Fixed		Non ML	ML Blast		ML Fixed	
			PCA	CNN	PCA	CNN		PCA	CNN	PCA	CNN
QC	Sing.	86.1	86.7	89.0	81.4	91.6	24.1	39.2	20.2	905.2	20.2
	Mult.	78.8	52.3	79.9	30.1	47.7	41.8	179.6	32.3	5680.4	208.4
QC +CQ	Sing.	94.8	88.6	91.3	81.6	92.7	22.0	29.9	18.4	808.3	19.6
	Mult.	51.3	44.1	68.0	33.1	42.7	112.3	210.6	48.8	5666.1	240.0

Average percentage of known matching neural volumes ranked in the top 50 candidates and median position of matches (using 150 points). Non-M/L nBLAST results used in production from table 4.4 (150F) are shown for convenience of comparison. As above, *QC* refers to ranking via only Query-Candidate average scores whereas *QC-CQ* refers to adding the reversed Candidate-Query average score for ranking.

point is one of a selection of up to 20 nearest points to the original example point that are from the registered skeletons of other cells of the same subgroup (with up to 4 top nearest points per other neural volume). They are also constrained to be within a normalised squared voxel grid distance of 1000. A non-matching point is a random point from any neural volume in the dataset.

The point-point (p_1-p_2) metric, M_{ML}, was then the sum of the squared differences between each point's PCA representation in each of the 10 dimensional components of **c**, i.e. simply:

$$M_{ML}(p_1, p_2) = |\mathbf{c}_1 - \mathbf{c}_2|^2 \tag{4.11}$$

In the test set, the PCA representations of their projection windows correctly gave lower scores to comparisons between the two matching windows over the two different ones in 94.1% of the triplets.

I then created projection windows for 150 points in the BLAST training and test sets from section 4.5.2 in order to calculate the PCA representations at these points. The BLAST training set produced a 2D histogram to convert the similarity metric, M_{ML}, to a BLAST score for the BLAST test set to enable ranking of query-candidate skeletons as in the previous subsection. Unfortunately, these results were mostly notably worse or very similar to nBLAST (See M/L BLAST PCA in table 4.5 and figure 4.17).

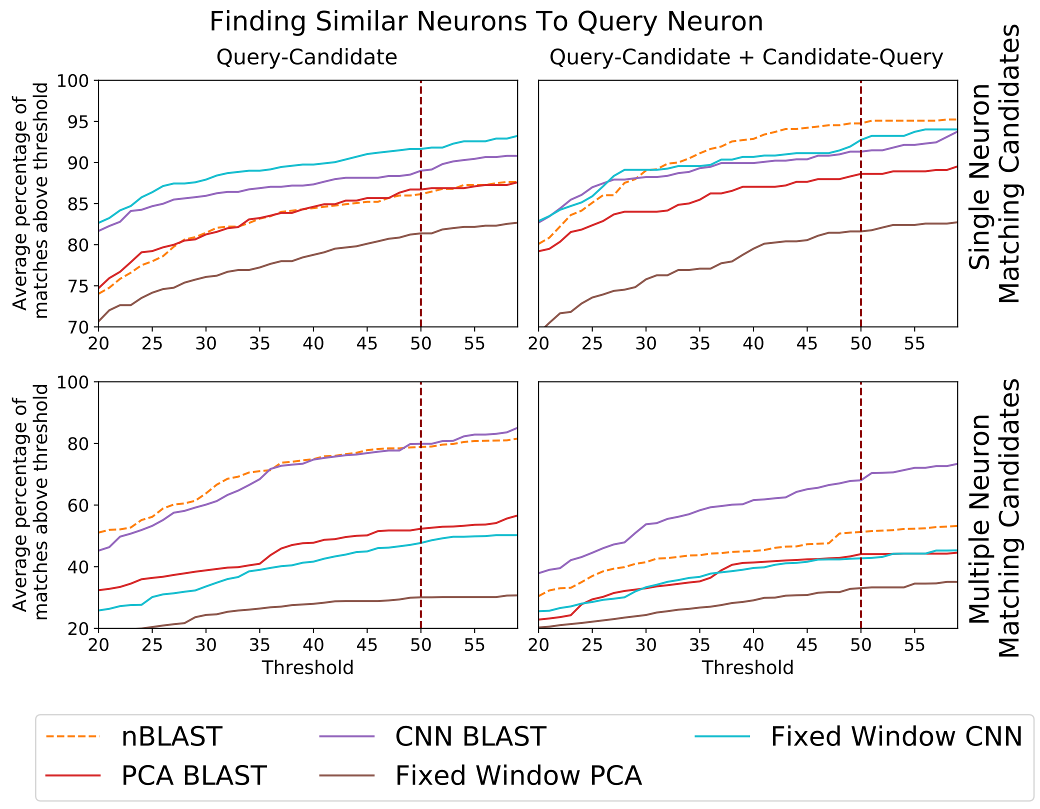


Figure 4.17: Percentage of Matching Cells Found Above Threshold for Machine Learning Methods
The percentage of cells found above the varied threshold for the machine learning methods compared to those for nBLAST. Top row: Single neuron volumes of same-type neurons, Bottom Row: Matching neural volumes contain multiple neurons, one of which is the cell of the same type as the query. Left column: Similarity scores are from query to candidate. Right Column: Ranking scores are the sum of scores in both directions. BLAST score use fitted equations. Note the different y-scales between top and bottom graphs.

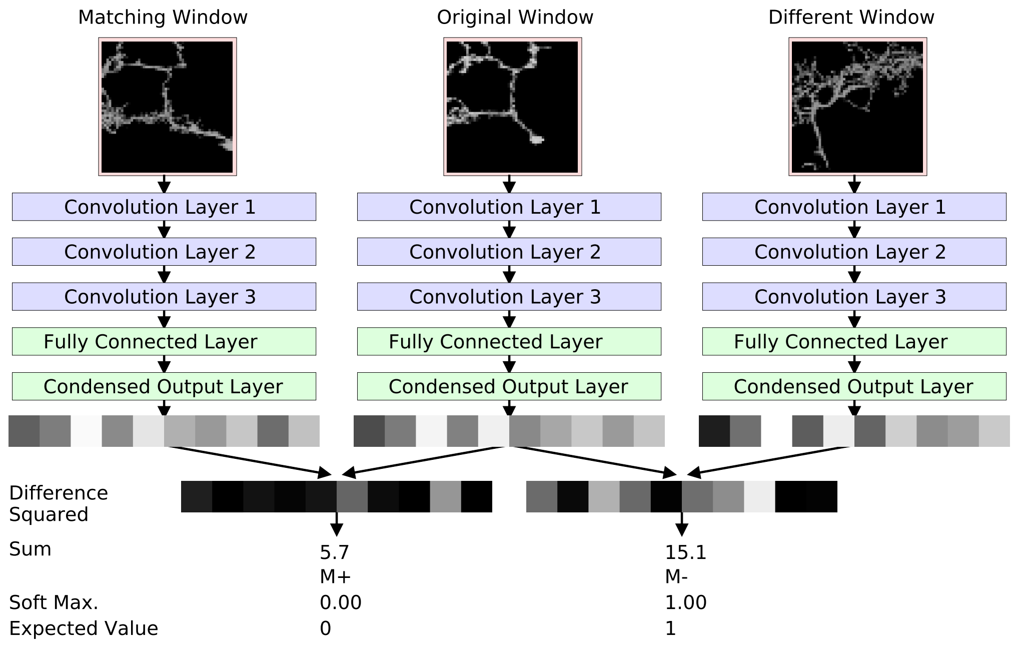


Figure 4.18: Training Convolutional Neural Network For Matching Neurons
Triplet network training paradigm involving simultaneously running the network over three inputs. Grey scales consistent between images within layers.

I improved on these results by training a CNN with the structure shown in figure A.9 to convert the projection windows into a 10-dimensional representation. I hoped that a CNN would extract features that represented common neural shapes, e.g. tubes in specific directions or compact dendritic arborisation, as they do in object recognition LeCun and Bengio (1995). The training was performed using a triplet network Hoffer and Ailon (2015) (see figure 4.18) which, for a single training set example input, calculates the reduced representations additionally for a matching point's projection window and a non-matching point's window, as was the case with the M/L test set. The metrics are converted to a predicted probability of being the non-matching pair as follows:

$$p_{\pm} = \frac{\exp^{M_{\pm}}}{\exp^{M_{-}} + \exp^{M_{+}}} = \frac{\exp^{|c-c_{\pm}|^2}}{\exp^{|c-c_{+}|^2} + \exp^{|c-c_{-}|^2}} \tag{4.12}$$

where M_+ is the difference squared metric with the matching point and M_- with the non-matching point and p probabilities for these accordingly. The exponentials ensure positive values. The system then attempts to increase the separation between the two 10-dimensional representations by using the Adam optimizer Kingma and Ba (2014) (variation on stochastic gradient descent) to reduce the mean squared error, e_{ms}, from the expected values:

$$e_{ms} = (p_+)^2 + (1 - p_-)^2 \tag{4.13}$$

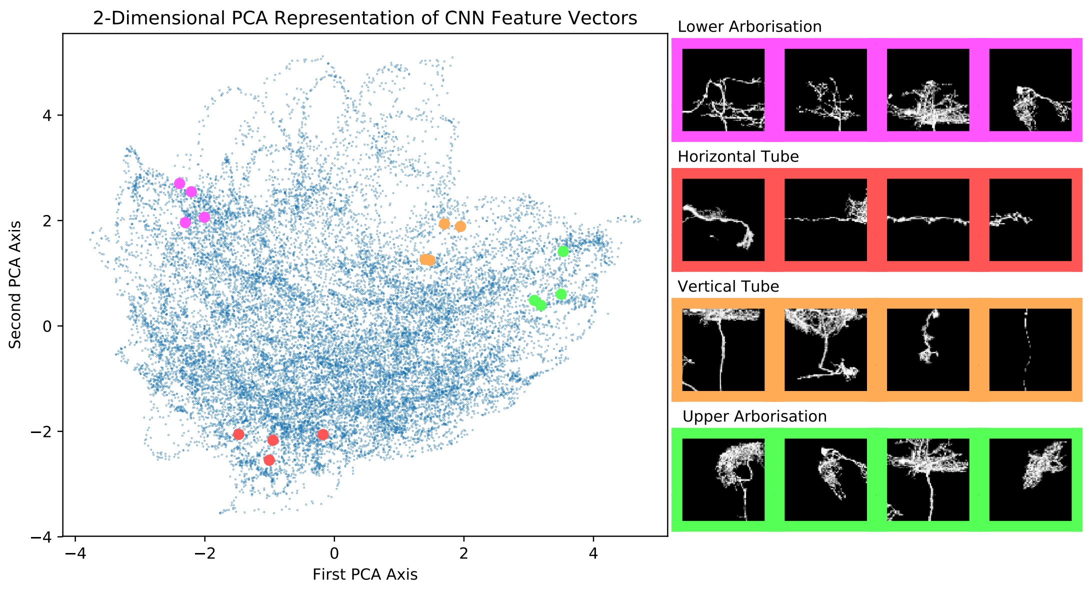


Figure 4.19: Convolutional Neural Network Clusters For Neural Projection Windows Left: Plot of the CNN 10-dimensional vectors for the point windows in the M/L test set mapped to a 2D space using PCA (i.e. further condensation) to illustrate distributions. Right: Example projection windows corresponding to coloured points in various regions of the plot. Note the similarities between the local projections of cells from the same region. All projection windows of the same type (colour) are from different specimens. Note also that some specimens show multiple times but are centred on different points.

This training function was averaged over batches of size 50 with the model implemented using the TensorFlow library Abadi et al. (2015) and drop-out Srivastava et al. (2014) to protect against over-fitting. The input was the central square (54×54 voxels) to allow for some data augmentation, namely random rotation and small x-y shifts to apply to each input in order to artificially increase the size of the training set. Appendix A.6 gives the full parameters of the CNN.

After adjusting parameters, the chosen, best performing CNN model's 10 dimensional representation squared difference sums (M_{ML}) were lower for the matching pair than the non-matching pair in 98.5% of cases for the test set, more than halving the error of the PCA representation on this set. Figure 4.19 demonstrates the separation of the different projection windows in the test set when the CNN representation distribution is projected into a 2D space, verifying the principles of the triplet network for sorting data.

I used the BLAST training set to create the 2D histogram to determine a mapping from the squared difference sums of the CNN representations (M_{ML} values) and separation distances to a BLAST score (figure 4.20m). This verified the M/L representation

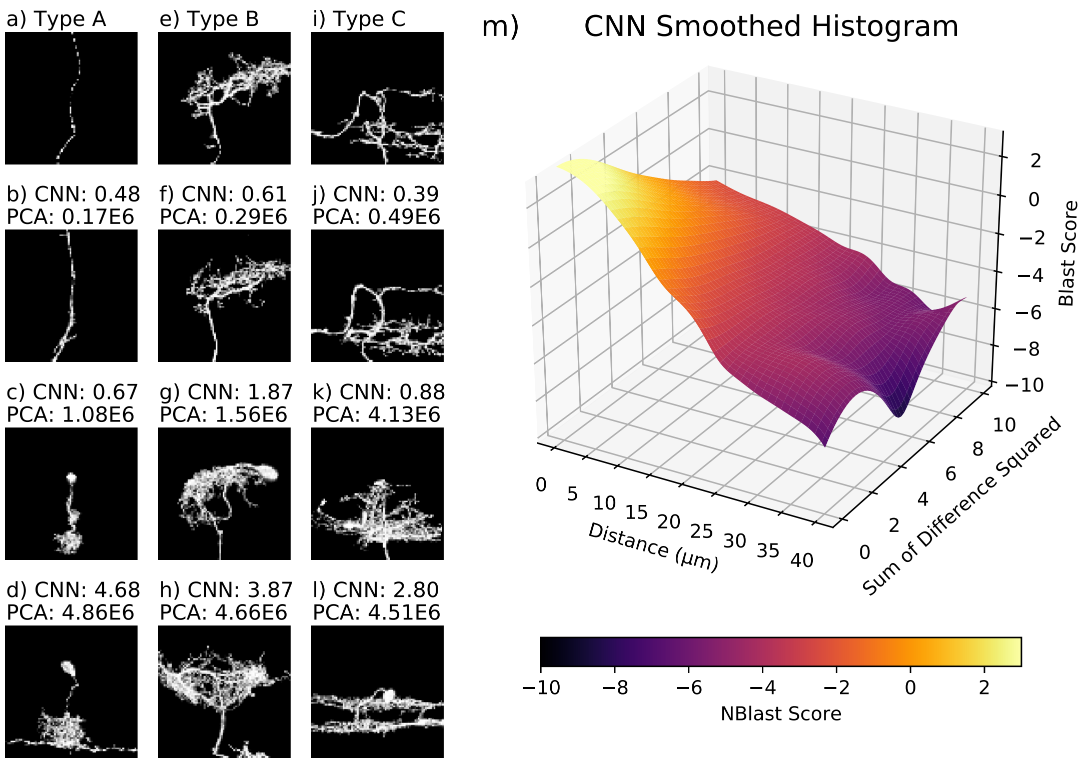


Figure 4.20: Convolutional Neural Network Projection Window Representation a)-l) Maximum projection input windows. a), e) and i) are compared to their following three windows, producing their corresponding results shown. m) 2D histogram converting distance of separation and sum of differences squared for two points to a BLAST score (smoothed with Gaussian standard deviation 5 bin widths).

with notably higher BLAST scores for lower M_{ML} values, as expected. Using this histogram to extract the scores, the BLAST *test* set was able to act as a validation set for the chosen best-performing model. The results (table 4.5, figure 4.17) demonstrate that this method is slightly better than nBLAST for finding matching cells without including the reverse comparison, especially for single cells (an improvement of 2.9% for the top 50). Interestingly, when the reverse comparison is added, this method greatly improves in finding multiple volumes over using the reverse in nBLAST (by 28.6% for top 50), however this is still no better than without including it and therefore not particularly useful.

Finally, inspired by the above results, I applied the PCA and neural network condensation techniques to projection windows taken as above but centred about the fixed points in section 4.5.1. Instead of comparing the nearest neighbours to produce the squared difference M_{ML} score, the corresponding position points were compared, removing the hefty requirement to find nearest neighbours. I recalculated the PCA transforms and trained the CNN using as input projection windows at the fixed points in the registered image stacks used for the M/L training set above. For the CNN, point pairs from corresponding positions in space in the same cell type were designated as matching and random pairs designated as not. Again, the BLAST training set was then used to calculate the mapping of the metric M_{ML} to another BLAST score, with the query-volume to candidate-volume score the average value found across all fixed point-point pairs for which the query cell's 10-dimensional representation was not that for an empty projection window.

Whilst the performance of PCA in this case was very bad, that of the CNN was impressive, performing the best of all above methods (without involving reversed comparisons) for finding single cell matching volumes (91.6% in the top 50, median position 20.2). The performance with reverse comparisons was only slightly worse than that of nBLAST (92.7% vs. 94.8% found). However, it was no better for finding multiple cell volumes (only 47.7% in the top 50, median position 208.4). The main advantage of this method, however, is that, as the points are fixed, no point pairing is required and thus each cell-cell comparison takes approximately 30μs, or less than a 30th the time of the nBLAST method. This is a significant improvement since, as databases become ever larger, the fixed point M/L method will allow rapid identification of similar neurons in a very large database of exclusively *single* cell neural volume confocal microscope images.

Table 4.6: Lineage Classification Results

Rank Position	1	2	3	4	...	7	...	17	...	21
Frequency	6	3	1	1		2		1		1

Ranked position using nBLAST of correct lineage for test set of cells.

4.6 LINEAGE CLASSIFICATION

As I had labelled the pipes of lineages in my template image stack using TrackEM, I was able to write scripts to extract, for each lineage, up to 150 points from the centre lines of these pipes with directions as I did for the neuron skeletons. I then used the trained nBLAST method from section 4.5.2 to find lineage pipes that matched to skeletons in a test set of 15 single-cell neural volumes from the well-registered set for which I manually assigned a lineage based on inspection of the morphology and the literature. The ranked position of the correct lineage out of all annotated 228 measured the success of assignment, with 6 cell lineages being ranked first and 3 second (see table 4.6). It was disappointing that some were badly ranked. I explored weighting the matching score by closeness to skeleton origin point (usually cell body), but unfortunately this did not help. I therefore concluded that the BLAST methods were not suitable for automating assignment of lineages to cells.

4.7 MATCHING TO THE EM DATASET

As discussed in section 1.4, *Drosophila* larval neurons have been traced manually to high-resolution in an EM image stack. These traces consist of a list of $[x, y, z]$ points and it was desirable for my collaborators to be able to upload a file of these points and return a list of image stacks containing neurons of similar morphologies in the LM data. As my collaborators were continually producing skeletons it was important this look-up was performed in "real time", for which the nBLAST system is highly suited (the M/L methods were not relevant since these relied on having confocal data, not EM skeleton points). I therefore needed to create a bridging registration to convert points in the EM space to that of my LM template. The EM representation I was given consisted of co-ordinates of points that had been marked in the centre of neurites of the cell but without connections to each other, meaning I also required a method to extract 150 points that represented as best as possible a random selection from a *pruned* skeleton.

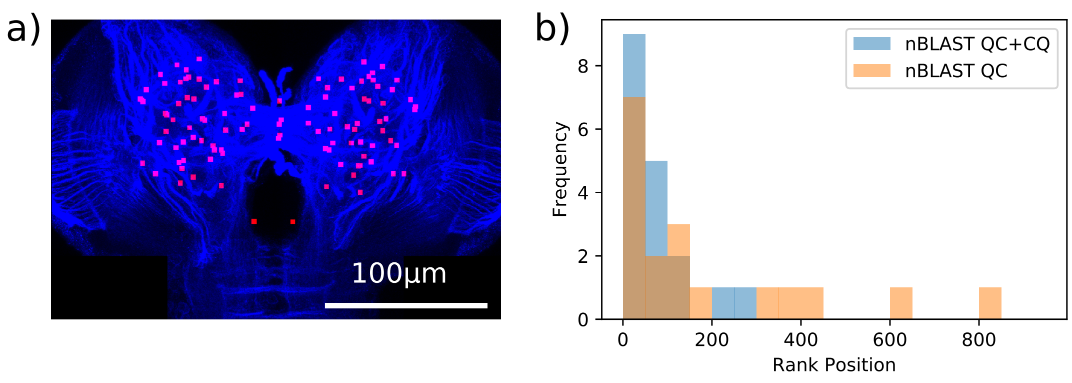


Figure 4.21: Landmarks and EM to LM Test Set Results
a) Maximum projection of landmark locations (red) on BP104 staining of template (blue). b) Histograms of the ranking positions of LM cells containing the query EM cell for the validation set.

Working in collaboration with an expert on the EM dataset, Michael Winding (MW), we matched up the 88 lineage neuropil entry points that had been annotated in both stacks and in addition identified 21 landmarks at locations inside the neuropil. Using these landmarks I evaluated 3 methods of creating a bridging registration, with parameters decided based on fit of the EM neuropil mask to the LM mask.

- Polynomial Function: Each dimension in the LM is a 3rd order polynomial of the EM dimensions. I fit the polynomial using ridge regression.
- Radial Basis Functions (RBF): Each point is interpolated by summing weighted contributions from a specified function of the distance to each landmark. The weights are adjusted such that the sum passes through each landmark Lazzaro and Montefusco (2002). I used linear basis functions.
- Moving Least Squares (MLS): Each point is mapped from the EM space to the LM space using an individual affine transformation that minimises the weighted total of the distance errors were this affine transformation applied to all landmark mappings Schaefer et al. (2006). Each weight is of an inversely proportional nature (to the distance from the corresponding landmark) allowing matrix algebra to be applied to find the transformations. Infinite weights at landmark points ensure the mapping passes through them.

As I was not able to use my pruning algorithm (reconstructing the skeleton would have taken too much time and required guessing connections), I instead approximated it as follows: I randomly sorted the points and then added each point progressively to a new list to use if each point was not within 10 voxel widths (in any dimension) of n points already added. I began with n as zero and if I reached the end of the list of candidate points to use, I incremented n and ran through the shorter

Table 4.7: EM to LM Matching Results

Mapping For Test Set	Matching Method	Top 50	Median
Polynomial Function	pBLAST QC	11/14	16.5
	pBLAST QC+CQ	11/14	18
	nBLAST QC	11/14	16.5
	nBLAST QC+CQ	12/14	14.5
Radial Basis Functions	pBLAST QC	11/14	13.5
	pBLAST QC+CQ	10/14	19.5
	nBLAST QC	11/14	18
	nBLAST QC+CQ	12/14	17.5
Moving Least Squares (MLS)	pBLAST QC	11/14	13.5
	pBLAST QC+CQ	9/14	9.5
	nBLAST QC	11/14	13
	nBLAST QC+CQ	11/14	8
Validation Set (MLS)	pBLAST QC	6/18	97
	pBLAST QC+CQ	9/18	47.5
	nBLAST QC	7/18	100.5
	nBLAST QC+CQ	9/18	42.5

"Top 50" refers to the proportion of ranks for the matching LM cell for each EM trace that were in the top 50 ranked cells. "Median" refers to the median value of all the matching ranks.

list of all still unused candidate points until I had 150 points. Figures 4.22c and 4.22d demonstrate the results of this method and show that it was successful in spreading points out with a similar distribution to those selected following pruning (personal visual inspection). I calculated the nBLAST directions for each of these points using the original set of points to find nearest neighbours for the matrix of positions. I then ran the searches through the neural volume data set using the same models for nBLAST and pBLAST as in the productionised system outlined in section 4.5.2, producing a ranking of all cells in the data set.

The test set consisted of 7 LM cells and their mirrors, that were matched by collaborators to both left and right cells of four types of EM cells (with up to 2 LM cells for each EM cell). A validation set was as the test set but contained 18 LM cells for six other EM cell types. I judged the success of each mapping on both of the BLAST score ranking positions of the LM cells given the EM input (see table 4.4).

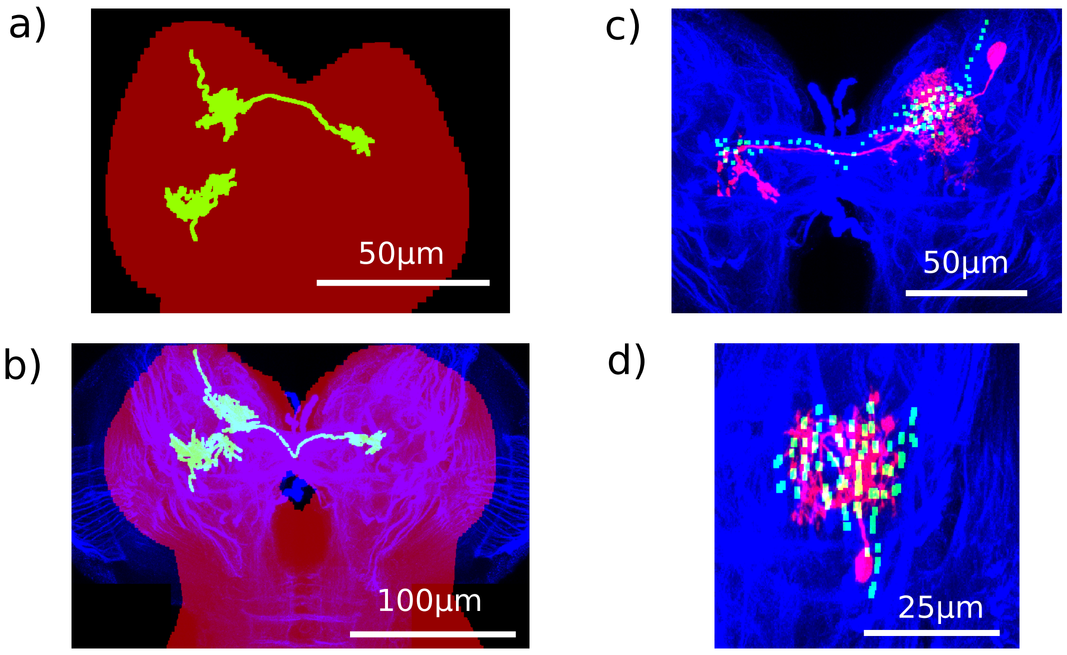


Figure 4.22: Examples of EM to LM Mapping
Maximum projections of examples of mapping two neurons (green) from the validation set from a) EM space to b) LM space. The red outline is that of the CNS cortex-air boundary from the EM specimen. b) demonstrates a reasonable mapping of this boundary and cells. c), d) Green: 150 points to represent each cell for the BLAST comparisons, Red: Overlay of example of cell of same type in the LM dataset. c) ranked 35th using nBLAST QC-CQ and d) ranked 11th.

I chose to implement the MLS method due to the better median rankings for the test set. Despite the good test set performance, the validation set (which mostly consisted of commisural cells) did not perform quite as well, perhaps suggesting some over fitting. Nevertheless, as more training data was not easy to come by, I created a simple, password protected web-based interface to allow my collaborators to upload a file of EM points and return a ranked list of candidate LM cells with their maximum projections and original image stack file names (and hence genetic lines from which they were produced). Figure 4.22 demonstrates this ability to find target cells.

4.8 AVERAGING NEURONS

A cell type could be defined by recording the common morphological features of known examples of the type. I therefore was motivated to develop a system that extracted these common features, namely common principal tubular neurites and regions that contain many neurites but are highly variable and are usually regions of high arborisation.

Most research into matching neural skeletons to one another has been conducted in order to determine the degree of similarity between just two cells, rather than with the aim of creating a single average cell from a population. Progress has been made on building generalised *consensus* skeletons from more than two examples Zheng et al. (2010) and, more recently, consensus neural trees (not taking account of position) Gillette et al. (2015). My method is similar to (though not actually inspired by) elements of Mottini et al. (2015), in that I build common tubes and add branches to them, but rather than building the average skeleton following whole-tree matching, I perform the skeleton building alongside point-point matching. These established methods were not sufficient since I decided to represent an average neuron as a combination of averaged skeleton tubes and averaged regions of arborisation where the tubes were highly variable among neurons of the same type (and also difficult to segment from one another).

I was able to easily average the boolean signal expressed in the arborisation regions, but this was less successful for tubular regions as overlap was often not exact due to small variances in tracts and limits in registration accuracy. I therefore decided to write an algorithm that would first match parts of the tubular components to each other. This is summarised below and explained in detail in appendix A.7.

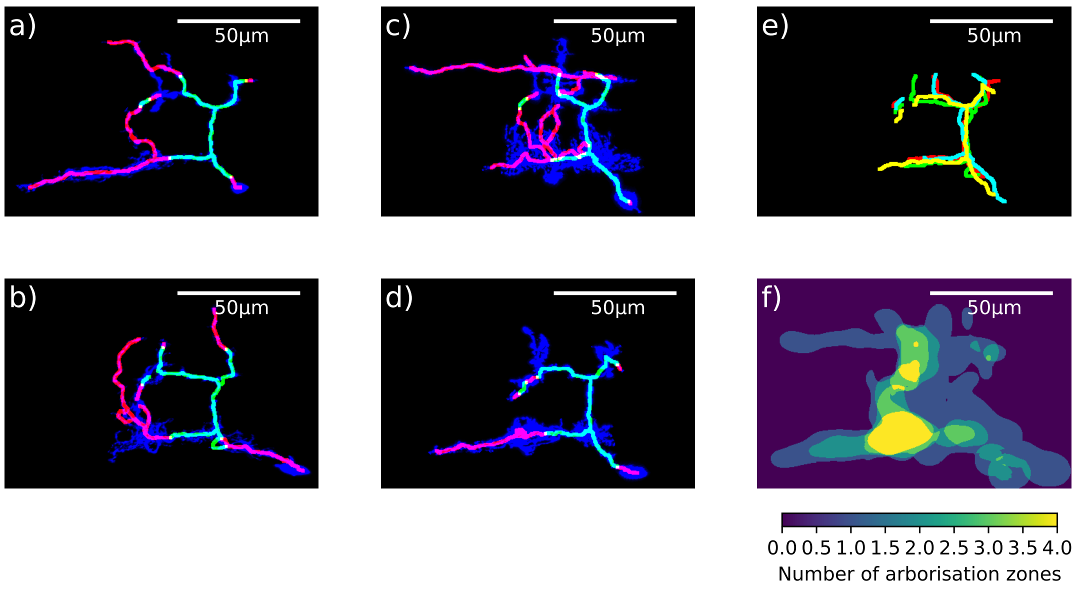


Figure 4.23: Example of average neuron skeleton matching
a) - f) maximum projections for cell of type "A31c" in the VNC Abdominal segment 1. a) - d) The four input neurons with the raw registered signal in blue and the pruned skeleton overlaid in green and red. Green points are the matched, "good" tubeness scoring points that are removed from signal images prior to averaging arborisation regions. The red are the "bad" tubeness and unmatched, arborisation points. Note that c) might be contaminated with a small part of another cell, but this is ignored in the average neuron. Most neurites match with corresponding neurites throughout the skeletons with the exception of the cell body axon in b). It is common in the VNC for cells of the same type to have somewhat varying lengths of projecting neurites, as seen here. e) shows overlays for the parts of the original skeletons matched up to an average tube with a different colour for each skeleton. f) is the projection for the number of remaining, non-tubed arborisation signal regions at each point.

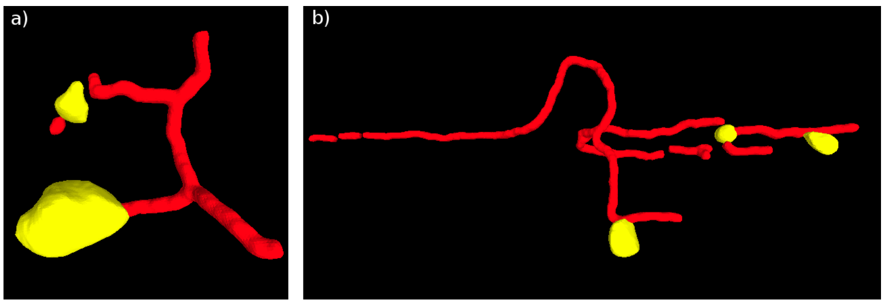


Figure 4.24: Average neuron 3D Representations
3D representations of a) the average cell from figure 4.23 and b) another, larger, averaged cell also from the VNC.

All possible emanating paths of length up to 100 voxels were constructed starting from each point in each skeleton of the matching cells to be averaged. An initial group of well matching paths, one from each skeleton, was found. First the path with the best average nBLAST score from its constituent points to those of the other skeletons was identified. This was then paired with the best matching path in each other skeleton (calculated using the average nBLAST score for points in the constituent path pairs only). The starting voxels of these paths were grouped as the first points in tubes in each skeleton. These tubes were extended with further corresponding points by using the approximate string matching algorithm Skiena (2008) to align the paths to one-another, with the average insertion/deletions required to match each path to the next voxels in the other paths used as the amount of incrementation for each path. This path-path matching is similar to the algorithm used in Cardona et al. (2010) (though that uses "the subtraction of two vectors" as the cost function). The group of best matching, unused, paths starting at this new group of points from each skeleton was then found using the same nBLAST method above and the tube extension process was iterated until there wasn't a satisfactory group of paths found.

Further tubes were added by finding well matching groups of paths connected to the tubes already found (using the above methods) and then the whole process repeated to find further regions of high consistency. The tubes enforce consistent morphology up to the point of being in the same direction but not necessarily same branching order, which is an important consideration if two different branches from a central spine are near to each other but their branch order is not conserved.

The average tubes were constructed with average positions and tubeness scores of their constituent points and, following some tubeness score smoothing, signal in the original stacks surrounding points matched to average points with high average tubeness scores was removed. The remaining signal above the threshold used in section 3.6 was dilated slightly using a Gaussian and those voxels that appeared in all signal examples kept as average arborisation regions to add to the average tubes (see figure 4.23f).

I adjusted the parameters and inspected the results visually on a range of cell types so that they appeared to extract the average representation, although further validation would require the input of experts in cell type (such as JT) for which there was not time in this project. The method ran quick enough to enable the average of 4 typically sized cells to be found in less than 10 minutes without parallelisation. I judged the results to be mostly satisfactory based on trials with the training and test sets for the BLAST methods.

4.9 CHAPTER SUMMARY

I have characterised cells using a range of condensed positional and directional representations to enable fast identification of similar neural volumes to a query cell. I added state-of-the-art machine learning methods to this process that demonstrated potential improvements in terms of speed and accuracy over the former. I have made several of the techniques accessible to collaborators and these methods have since been actively employed by them to quickly identify cells of the same type from their own dataset. In addition, I detected cell bodies of the cells mostly successfully which will help me classify the neural volumes as single or multiple cells. I also demonstrated techniques to identify regions of arborisation vs. regions that are highly tubular. Finally, I have provided a proof-of-concept to quickly produce an average representation of cells of the same type.

4.10 CHAPTER DISCUSSION

The detection of cell bodies was not accurate enough to permit full automation, mainly due to the large number of false positives. Even though this was a low percentage of all projection windows, the vastly higher number of non-cell body points meant this was a problem. Notable examples of failure were classifying highly spherical axonal arborisations as cell bodies but missing two adjacent cell bodies. Nevertheless, I demonstrated that the machine-learning method of CNNs improved the task and the low-level of undetected cell bodies were later found to be very useful in assisting in creating a highly-accurate and - compared to purely manual annotation - fast semi-automatic classification of cell bodies. Improving the accuracy further would have likely required a much larger training set, which would have required a large amount of time to create as well as further data for testing and verification.

Measuring the similarity between two cells for clustering neurons by type (recall section 2.1 regarding the meaning of *type*) requires a system as accurate as can be found, whereas ranking the neurons by similarity aids researchers by limiting the time spent searching databases, and does not therefore require this high level of accuracy. Researchers can always check accuracy by inspecting the original pre-registered CNS stacks. It was clear to me during my analysis that different methods were somewhat suited for different cells, and it would be advisable that a researcher looking for cells of a particular type should work through the various methods I have made available until they have found enough similar cells.

Although I demonstrated that Machine Learning can help with the similarity process I was impressed by the performance of the simple pBLAST matching. This is likely due to the high accuracy of the registrations, the limited number of cell type morphologies in the larval brain, and the success of the pruning algorithm in making sure shared morphological features are kept in the skeletons. The limitations in possible improvements of the other BLAST methods may be attributed to the axonal tracts of cells forming stereotyped bundles (tracts and fascicles) in the larval CNS. As these tracts often connect disparate regions of arborisation (particularly across the commissures) they are almost always included in pruned skeletons. This can mean that a large proportion of the skeleton points are part of these fascicles, leading to a potential problem illustrated in figure 4.25 and explained below:

Suppose a tubular axon representing a large proportion of two candidate cells is matched to a query cell and for one candidate, that is of a different type, the matching of this axon is very good but for the other candidate, that is of the same type, it is just satisfactory. For this region the average BLAST score will be higher for the cell of a different type. A bad matching of points in the remaining parts of the query cell for the cell of a different type and a satisfactory (but not great) matching to that of the same type will mean a higher BLAST score for the cell of the same type, but this may not be enough outweigh the difference in matching scores for the large number of points in the axonal tube. This is a result of the way the BLAST scores were calculated: from a histogram of all individual point-point pairs rather than looking at whole skeleton scores (which, unfortunately, I do not have enough training data for at this time).

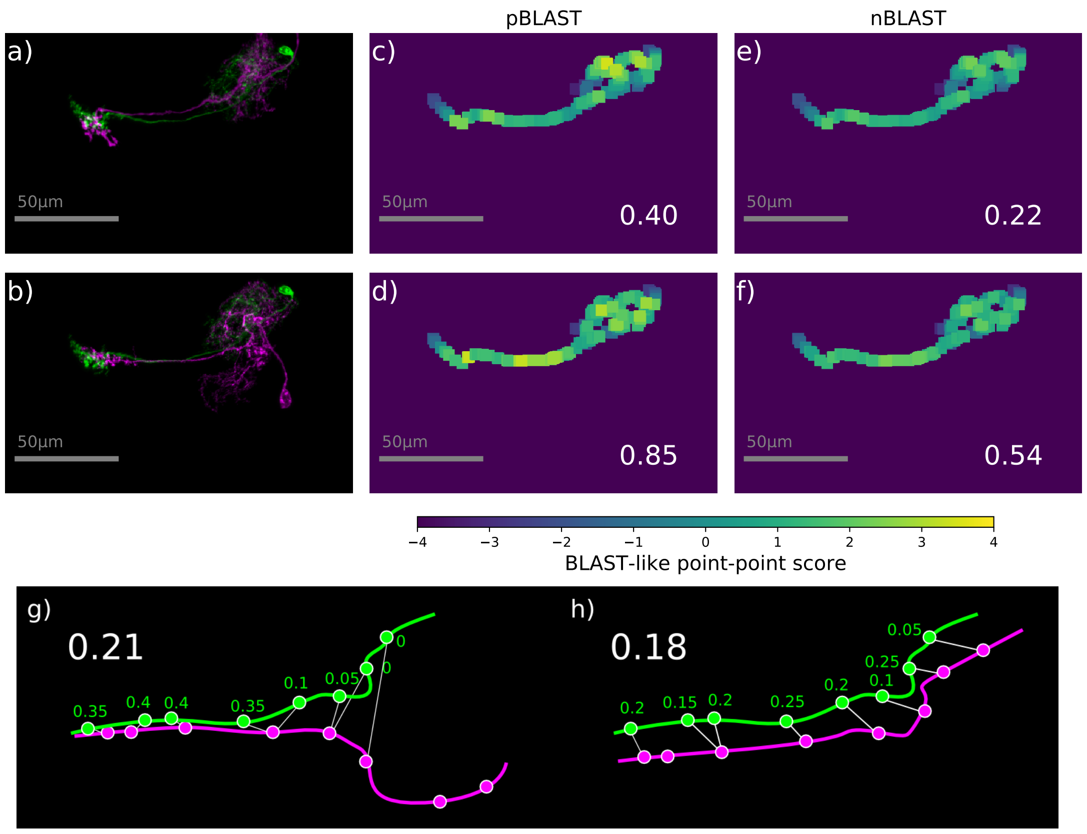


Figure 4.25: Explanation For Limited Search Result Perfomance
Illustration of potential problem in matching: The top row is for a candidate cell of the same type, the bottom row for one of a different type. The same cells match satisfactory overall, but the differing cells match very well in the commissural fascicle. As a result the differing cells receive better (higher) BLAST-like scores. Maximum coronal projections of a), b) Green cell query, magenta candidate. c), d) pBLAST (distance only) scores, e), f) nBLAST scores. g), h) Further illustration of how BLAST matching scores from green query to magenta candidate could result in a higher score for a g) non-matching cell pair over an h) matching pair. The 8 points are given estimated scores with the white score the overall average for these 8 points.

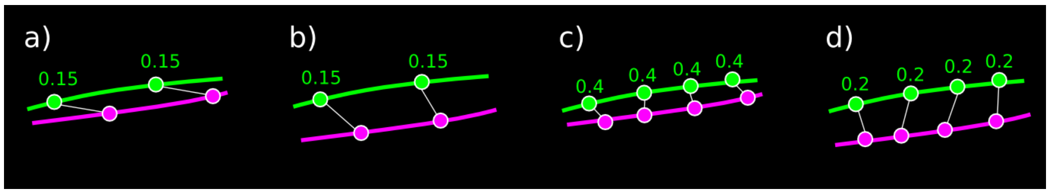


Figure 4.26: Explanation For More Than 150 Points Not Being Beneficial
Estimated BLAST scores for points in cells that are a), c) non-matching but close in this region and b), d) matching but not close in this region. Although both cells in a) and b) have the same score in this region at a lower number of points, c) (non-matching) has a much higher score than d) (matching) when more points are used, due to the axonal separation. This effect could outweigh the contribution of mediocre matching vs. non-matching in other parts of the cell.

This artefact might also help explain why adding more points over 150 for the BLAST scores does not improve the results - more points along highly matched fascicles of two non-matching cells will further reduce the average nearest neighbour distance between pairs in this region, as the point density increases, and increase the BLAST score here (see figure 4.26). This effect might outweigh the contribution of adding more only satisfactory matched points over the whole of a matching cell (whose average nearest neighbour distance must be at least the minimum physical neurite separation in template space). Since this effect will not apply to the fixed-point machine learning method, it might also explain some of its success.

Given this observation it might be interesting in future to measure cell similarity based on their proximity to fascicles, percentage of innervation of each neuropil domain etc. such that all skeleton points classed as part of an anatomical structure are grouped and then each structural contribution weighted by importance for the comparison metrics. That stated, determining the importance of each of the contributions of these, depending on size and position etc., is likely to require much tuning. This method would most likely be very well suited to determining cell type classification once these types are better defined. The average neuron representation could also specify the anatomical structures to consider for each cell type, for example. In addition, a larger known classification of cells by type would provide more training data to improve the Machine Learning methods, potentially allowing work on the full 3D window around points rather than projections.

As I demonstrated, the results of the nBLAST matching depend upon the degree of pruning (the shortest kept branches) such that further pruning gave worse results. These results are also affected by the tracing algorithm used in section 3.9 as well as the registration parameters. Given more time, I would have investigated together alternative tracing algorithms and keeping shorter pruned branches, since I believe that shorter pruned skeletons than I used would be less representative of the cells. This is relevant here as, ultimately, the results of pruning have to be assessed by the nBLAST matching performance. As I discovered a similarly accurate method to compare single cells, however, the effects of pruning were not relevant in the remainder of my project.

As emphasised previously, I set out to solve the problem of researches struggling to find cells of the same type from different genetic lines. I followed the AGILE approach and produced a baseline implementation where personal communications from researchers confirm that they are very happy with the results. As my dataset is broadly representative of cells in the larval nervous system my search methods are unlikely to degrade much in accuracy as more stacks are added (although search times will increase). Following achieving this, I chose to spend time investigating the application of M/L to improving the accuracy of matching methods without adjusting the tracing methods. This did not deliver a significant improvement in the accuracy, however it did improve the speed of a method that did not depend on tracing or pruning and is a novel contribution to the field.

The poor performance of the lineage classification for some cells could be attributed to some not being particularly close to lineage tracts, some of which were very short. Nevertheless, the Virtual Fly Brain will allow a search for nearby lineage tracts using nBLAST so that the researcher can display these on the viewer and then make any classification manually (see section 5.5). As with the rest of the Virtual Fly Brain site, it is important to keep in mind that I do not see anatomists as the foremost audience, instead I expect this to be of great use for the much larger group of researchers for whom the larval *Drosophila* is a neuroscience tool and the anatomy instead a framework for using it.

The mapping of the EM to the LM dataset is highly beneficial for researchers who have found cells in the EM dataset for which they need to find genetic lines that allow the study of their function. Although the bridging registrations showed some problems, such as bulges in the neuropil mask upon mapping across, and the pruning systems were different, it performed satisfactorily enough to enable researchers (MW and others) to find cells of interest substantially quicker than using the original dataset or having to consult its creator (JT). One should bear in mind that there is some structural difference between the 1st and 3rd instar of the larva, partly limiting registration performance. Once one cell has been found in the larval data set, however, it can then be used to find more using the better larval matching system.

The average neuron algorithm demonstrates a possible way to define cell type morphologies in the future by using the common features between examples classed as the same type. In my tests, visual inspection suggests this would work well most of the time though if further development of the algorithm is investigated it would require comparison with a human expert-produced set of average representations of some cell types (traced tubular skeletons and approximated arborisation volumes). As I encountered reluctance from most in the field to define cell types I did not pursue this here, though suggest the methods I produced would be complimentary to anatomists as they attempt to define cell types.

One instance where the average representation algorithm I used might struggle is if cells of the same type form subsets with alternative tubular paths. It is not clear if this is the case in the larva, but should be kept in mind (though the algorithm would still work within subtypes). Since the average neurons are produced from whole, single cell images for which cell bodies can be identified, they should lend themselves to averaging using the tree-matching methods detailed in Gillette et al. (2015) and Mottini et al. (2015), especially when combined with more advanced tracing techniques. In my case it was not practical to implement more than one system since the motivational reason at this point was not strong enough, however, with more data (particularly that of the EM variety) this will, I believe, likely become much more important.

5

BRAIN MAPPING AND CLUSTERING

5.1 MANUAL SELECTION OF NEURAL VOLUMES

In order to ensure that I would only use the highest quality data throughout this chapter, I applied rigorous selection criteria on the 35 775 neural volumes that I had prepared in chapter 3. As previously mentioned, neural volumes are segments of *gene* expression patterns and could contain one or more entire cells. They may alternatively or in addition contain parts of cells where some of the non-visible cell expression pattern is:

1. outside the registered template stack VNC or CB ROI.
2. outside the original imaged ROI of the microscope.
3. not inside the neural volume due to at least some of that part of the cell being below the dynamic neural volume intensity threshold (see section 3.6).

These deficiency classes will be referred to throughout this chapter.

Volumes may contain any number of components from above combined when these constituents were not physically separated enough for separate segmentation. Volumes containing only entire, single cells are ideal, particularly of interest being those containing just one cell. Volumes also, or exclusively, containing cells with deficiencies 1 or 2 are useful within their respective regions, although type 2 will require consideration that regions outside the imaged volume cannot be reasonably utilised in processing to determine cell similarities, innervation regions etc. Volumes containing cells with the partial cell expression of deficiency 3 were considered not "useful" for the remainder of this chapter due to their partial nature and the contaminating effect of the additional expression in volumes that contained cells without this deficiency. This was especially since many of the badly segmented cells did not include their cell bodies in the neural volume meaning it might be difficult for a researcher viewing only the neural volume (without the signal from the rest of the registered stack) to notice that there was expression data present from another cell.

To clean the data set, I manually labelled each neural volume to indicate if it was a useful segmentation or not (i.e. possessed a type 3 deficiency or not). For those

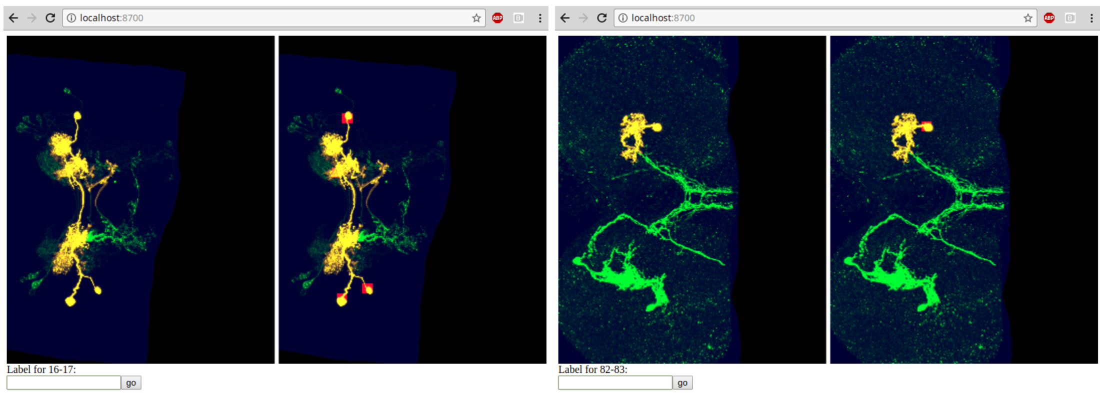


Figure 5.1: Example of Labelling Web App
Two example screen-shots for labelling two different neural volumes in the labelling web-app. Maximum z-projections of the registered image stacks where green signal is outside the volume, the yellow inside. The registered boundaries of the original stack are in dark blue. The right projection in each browser window is identical to the left except that it contains red marker boxes at the predicted cell body positions. Contrast could be adjusted and the label was simply typed below the image and saved on moving to the next. The left browser window exhibits a failed segmentation, since the stack is very busy with merging cells and has variable expression pattern intensity. Only parts of some cells are segmented (this is particularly noticeable near the centre). The right browser window demonstrates a successful segmentation of a single cell.

that did not, and for which there was a significant amount of neural volume in the template ROI, I indicated the number of cells present (as "1","2" or "more than 2") and whether or not the volume touched and passed out of the image or template ROI boundary. There were a large number (13.5% CB, 1.6% VNC) of small incursions into the ROI by cells mostly outside of it. This usually consisted of a single unbranching axon or, particularly in the VNC part of the template CB ROI, arborisation from a cell that was fully imaged elsewhere in a *VNC* stack of the same specimen. I removed these incursions from the data set. There were many more neural volumes that were excluded as they contained at least some expression of type 3 (41.2% CB, 69.0% VNC). There were also some neural volumes (5.0% CB, 0.2% VNC) that did not actually contain cell expression data, but instead other anatomy (e.g. ventricles). I manually applied the labelling myself by creating the private web-app shown in figure 5.1 in order to ensure my recordings were paired with the correct neural volumes as I iterated through viewing their maximum projections. The inclusion of predicted cell bodies from section 4.2 heavily speeded the task of identifying the number of cells, often bringing to attention locations that were not immediately obvious.

Table 5.1: Neural Volume Labelling Statistics.

Neural Volume Type	ROI	Number of cells present			
		1	2	> 2	Total
All component cells entirely visible	CB	3 090	513	445	4 048
	VNC	3 440	449	142	4 031
At least one component cell with type 1 and/or 2 deficiencies	CB	1 036	268	1 228	2 532
	VNC	1 073	196	361	1 630

The table does not include mirrored volumes.

5.2 CLUSTERING OF CELL TYPES

As the dataset contains similar cells I would like to identify cells of the same morphological type (as grouped earlier) to aid classification and also display only the best samples for each type to ease initially finding a cell of interest. As such it is necessary to use a similarity metric distance to cluster nearby cells such that each cluster contains just one cell type and a new distance similarity metric is given for this cluster to other cell type clusters. These clusters can then be usefully visualised using a dendogram tree where similar cell types are contained on branches that are near to each other in the tree space (ideally on the minimum sized sub-tree). This is frequently achieved using hierarchical clustering, for example Wan et al. (2015) and Costa et al. (2016) where it also serves as a method to validate the similarity metrics used in those studies. With this in mind, I will perform clustering using my new, fast fixed points M/L similarity method in order to further verify it and, since it is independent of pruning and tracing methods, is closer to the original data. I will use the dataset of single cell neural volumes as shown in table 5.1 (including with deficiencies 1 and 2). I will also include mirrored versions of these cells.

5.2.1 *Hierarchical Clustering Technique*

I used the standard agglomerative hierarchical clustering algorithm (implemented as part of the Scipy python package Jones et al. (2001)) that begins with each cell as its own individual cluster and iteratively combines the two clusters that have the lowest distance between them until one single cluster is left containing all cells in the dataset. For the cell A-cell B distance I used the sum of the squared differences of the 10-dimensional M/L representations of fixed-points from 4.5.3, averaged over all points with projection windows containing signal in cell A (query cell). The *distances* between two cells for the clustering algorithm must be the same in both directions in these algorithms, so I simply calculated these as the sum of both directions (query-candidate + candidate-query), an effective average.

The distance metrics, d, between subsequent clusters can be defined in a number of ways. I will detail and evaluate the effectiveness of four popular methods for this below. The distance between two clusters, s and t, can be defined as:

- **Single**: the *shortest* distance between all possible pairs of cells where one cell is from each cluster.
- **Complete**: the *longest* distance between all possible pairs of cells where one cell is from each cluster.
- **Average (UPGMA)**: the *mean* of the distance between possible pairs of cells where one cell is from each cluster. (UPGMA refers to *Unweighted Pair Group Method with Arithmetic mean*).
- **Ward**: when squared, the difference between the *total variance* (see below) if the two clusters are combined and the sum of each cluster's individual total variances.

Concerning Ward's variance minimisation technique, the total variance is the average difference of each cell's "position" (in Euclidean space) from the mean cluster position. As I only have pair-wise distances and the space is non-Euclidean I must calculate what would be the Euclidean total variance from the total within-cluster pair-wise distances, D. Since, in Euclidean space, $D(x) = n_x^2\sigma^2(x)$:

$$|d_{\text{ward}}(s,t)|^2 = n_{s\cap t}\sigma^2(s\cap t) - n_s\sigma^2(s) - n_t\sigma^2(t) = \frac{D(s\cap t)}{n_{s\cap t}} - \frac{D(s)}{n_s} - \frac{D(t)}{n_t} \quad (5.1)$$

where σ^2 is the standard statistical variance and n the number of cells in the cluster. Thus the ward metric still minimises a measure of the compactness of the clusters. These distances are calculated in practice by the efficient recursive Lance-Williams algorithm Lance and Williams (1967).

5.2.2 *Clustering Evaluation*

Starting with the neurons used in the M/L and BLAST training sets I created 23 comprehensive subsets (15 CB and 8 VNC) containing all neurons of the same morphological type that are present in the data set. This was defined as above, such that the cells appeared to be indistinguishable in gross morphology in the LM data. As a result I had a validation set that I could use to evaluate the different clustering methods. Firstly, as an indication of the error rate, I looked for methods that minimised the False Discovery Rate (FDR) for the *smallest clusters that contained all cells of a subtype*, averaged over all subtypes in the CB or VNC:

$$\text{False Discovery Rate (FDR)} = 1 - \text{Precision} = \frac{|\{\text{cells in cluster}\} \cap \{\text{cells of different type}\}|}{|\{\text{cells in cluster}\}|} = \frac{n_{\text{false positives}}}{n_{\text{true positives}} + n_{\text{false positives}}} \tag{5.2}$$

Whilst FDR is useful for seeing how well individual clusters form, it would be of use to find a threshold in the distance metric at which to stop merging clusters and consider each cluster, ideally, to be a different cell type. To attempt to find this value, I first also calculated the Recall (defined below) to see how well known similar cells are grouped together:

$$\text{Recall} = \text{Sensitivity} = \frac{|\{\text{cells in cluster}\} \cap \{\text{cells of same type}\}|}{|\{\text{cells of same type}\}|} = \frac{n_{\text{true positives}}}{n_{\text{true positives}} + n_{\text{false negatives}}} \tag{5.3}$$

I have, for each ROI and for each method and a range of thresholds, calculated the average FDR and Recall scores. These averages are means over all subsets of the mean within-subset scores for each of the subset's constituent cells and their correspondingly occupied cluster at the given trial threshold.

Figure 5.2a and 5.2b demonstrate how the two measures can be plotted against one another. To determine the optimum threshold I have combined both the Precision (to represent FDR) and Recall and maximised both using the F_1 score. It is defined to give equal weight to each quantity by calculating the harmonic mean as follows:

$$F_1 = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}} \tag{5.4}$$

I maximised the F_1 score by varying the cluster distance threshold.

Table 5.2 shows that all clustering metrics perform reasonably well, but *single* is clearly worse. Despite a good VNC F_1 score for *complete*, the remaining measures are bad, particularly the FDR for the minimum complete subsets. The *average* and *ward* methods performed strongly in both measures. Figure 5.2 demonstrates they both have similarly low areas under their FDR-Recall curves for both CB and VNC. For

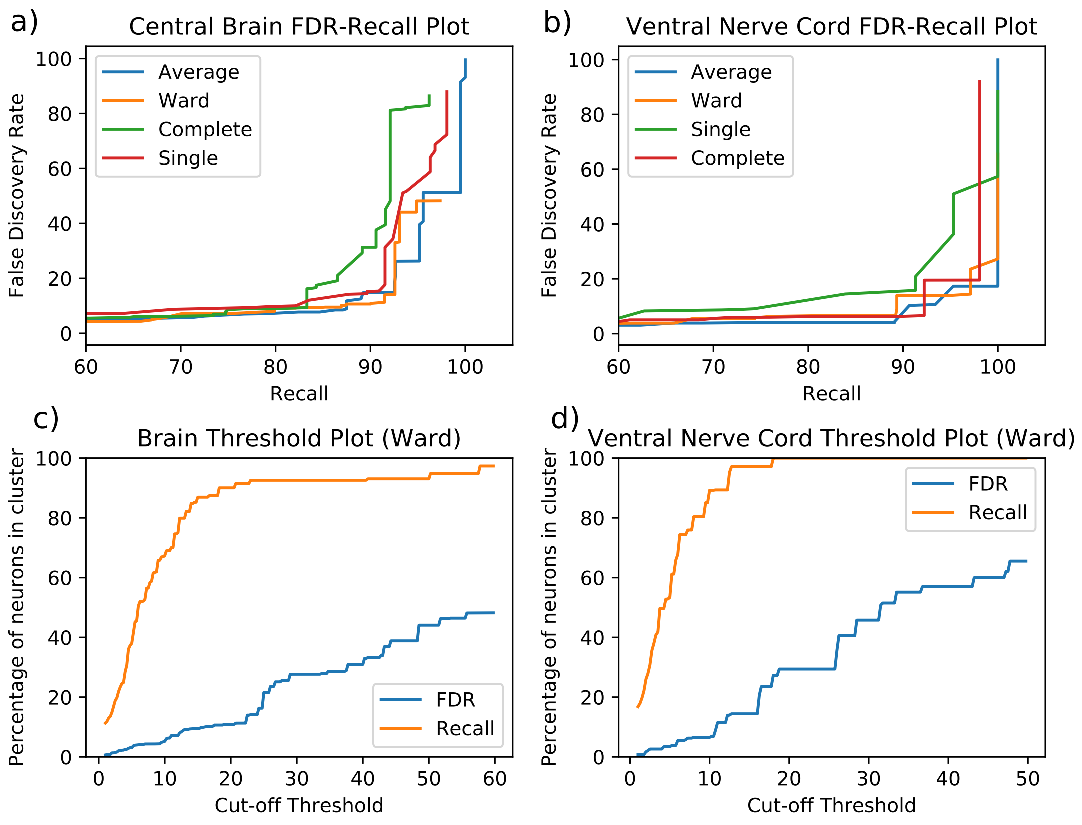


Figure 5.2: False Discovery Rate vs. Recall Graphs For Hierachical Clustering
As I want to maximise recall whilst minimising the False Discovery Rate (FDR), I wanted the graphs in a) and b) to stay low in FDR whilst recall increases prior to a rapid increase in FDR as close to 100% recall as possible. This is similar to the popular Receiver Operating Characteristic curve technique, except I have replaced the false positive rate with FDR as the number of true negatives would be very large and less meaningful. c) and d) give an example of FDR and Recall for different cut-offs used to generate lines in a) and b).

Table 5.2: Hierachical Clustering Results

Method	False Discovery Rate (FDR)			Optimum F_1 Score		
	CB	VNC	Average	CB	VNC	Average
Single	0.229	0.124	0.176	0.876	0.876	0.876
Complete	0.290	0.184	0.237	0.868	0.928	0.898
Average	0.168	0.111	0.139	0.893	0.924	0.908
Ward	0.205	0.111	0.158	0.901	0.913	0.907

The F_1 score is that for the threshold that gave the highest value.

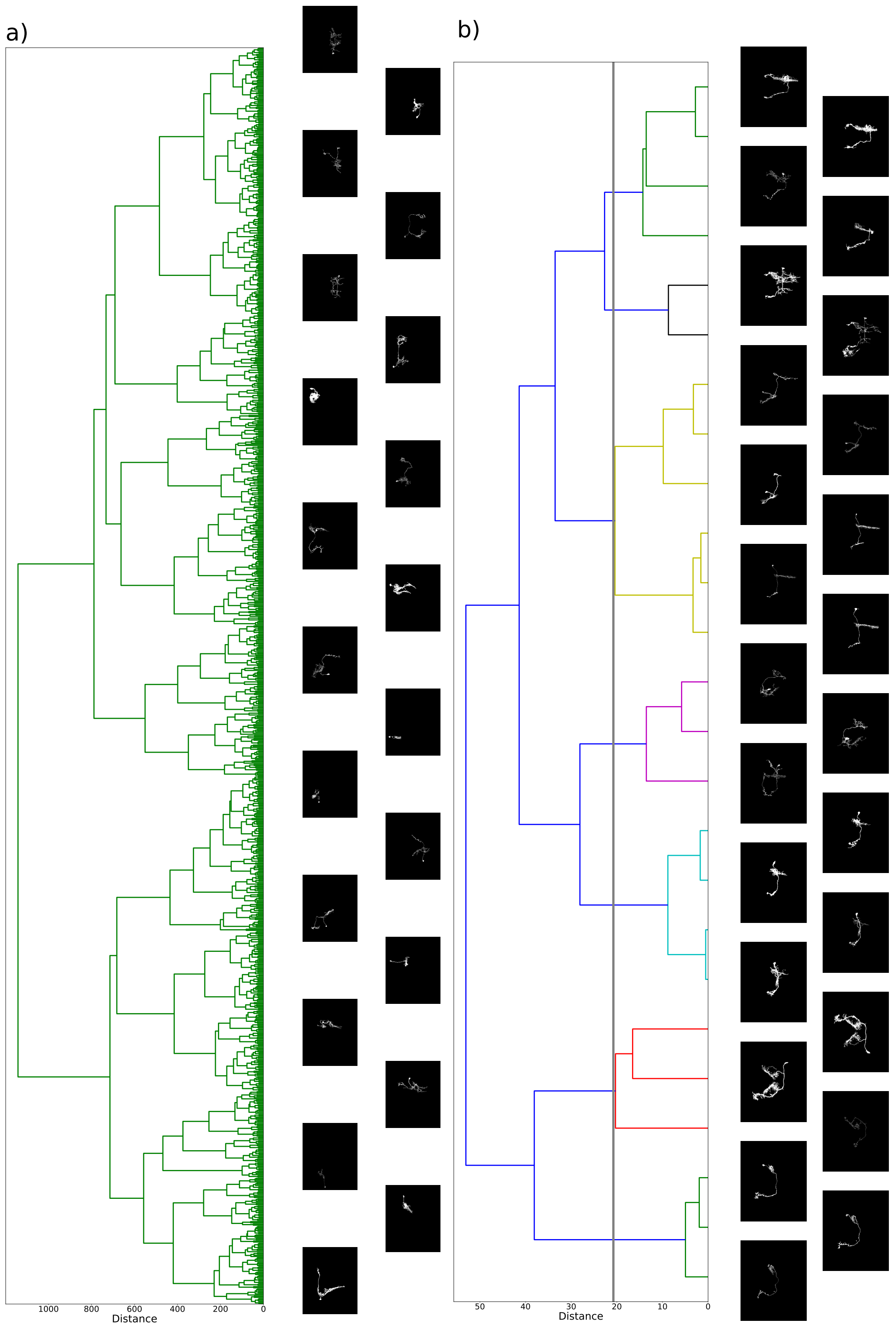


Figure 5.3: Hierachical Clustering of Brain Neurons
For ward's method: a) Whole single-cell data-set clustered with example maximum projections at approximate corresponding points. b) A close-up of a region with all leaf cell projections shown, colours correspond to given clusters at used threshold which is indicated by the thick vertical grey line.

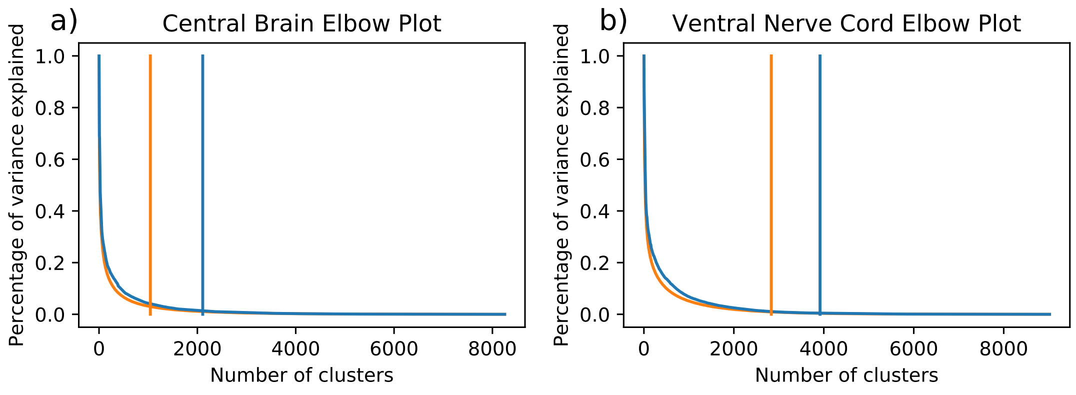


Figure 5.4: Elbow Plots for Hierachical Clustering
The elbow plots for a) CB and b) VNC with the thresholds that maximise F_1 indicated (blue: average, orange: ward) These are approximately at the points where the proportion of total data set variance explained by the clusters starts to increase rapidly. If clusters contain the same cells, the variance per cell within the clusters will not be very much, but will increase substantially if they contain different cells.

the *average* method, the best F_1 score gave 2 109 CB and 3 916 VNC cell-type clusters, whereas for *ward* the numbers were 1 044 and 2 832 respectively. I have plotted the clusters from the CB for the *ward* method in figure 5.3, which demonstrates a high degree of success of clustering similar cells together. An *elbow* plot of the sum of the total variance for each cluster as a proportion of the total variance of the whole dataset as the threshold - and total number of clusters - is varied can be used to validate these positions (see figure 5.4).

I also investigated clustering using affinity propagation Frey and Dueck (2007), as this was also employed by Costa et al. (2016). The F_1 scores were no better than with hierarchical clustering, achieving similar maximums of 0.880 and 0.862 for the CB and VNC respectively. I thus concluded that although it was clear I had of the order thousands of cells and that I could group them well with the M/L metric, I was not able to use it to obtain a clear estimate on the number of types of cells.

Having specified a distinct clusters using the above methods, it is also important to estimate the stability of these clusters, i.e. how robust they are to small perturbations. This is in order to determine my level of confidence in them and demonstrate that they are not simply artefactual, but do exist. I performed a bootstrap analysis of the clusters from the ward hierarchical method, as I will use this in section 5.4. As I clustered based on an nBlast metric distance, as opposed to one calculated from many Euclidean dimensions, I was not able to resample the distance dimensions. Instead, following Hennig (2007), applied a bootstrap procedure and resampled the cells with replacement to create an equally sized dataset (which contained repeated cells and omitted others). I clustered the resampled cells with the same hierarchical clustering method. Then for each cluster, C, in the original set of clusters I evaluated the degree of coherence with each and every cluster, D, in the resampled data, using the Jaccard Index, γ, Jaccard (1901):

$$\gamma\,(C,D) = \frac{|C \cap D|}{|C \cup D|} \tag{5.5}$$

Only cells that were present in both data sets were used to calculate the indices, and if this meant no cells in C were present in the resampled data set, then the index was not evaluated with this resampled set. The highest index returned for a specific cluster C over all D clusters in a resampled set is the most coherent resampled cluster for C and therefore the value that is given for the coherence of each original cluster C. An estimate of the robustness of each original cluster C can be given by taking the mean of the Jaccard index across many resampled cell clusterings (I performed 50). A score over 0.75 is widely regarded as "good" Hennig (2007).

For the CB I found that, for the 1 044 ward clusters I identified at the optimal F_1 score, the average Jaccard index was over 0.75 for 89.1% of the clusters, with an average mean of 0.91. In fact, 35.1% of these clusters had, in over 95% of the resamples, Jaccard indices of 1.0, i.e. were fully coherent with a 95% confidence. The figures for the 2 832 VNC clusters were 99.1% over 0.75 (mean 0.98) and 75.3% had a score of 1.0 within 5% confidence. Hence this evidence suggests that the clustering is justified and performs classification well for the vast majority of the data. I am now also able to identify clusters that are less clear and could prioritise investigation of these in the future.

5.3 ARBORISATION REGION REPRESENTATION

To investigate the validity of the annotated synaptic neuropil domains from sections 2.2 and A.1, I first extracted from the well registered neural volumes the regions of synaptic neuropil innervation, i.e. the arborisation regions as opposed to the tubular non-arborising neurites, many of which were in fascicles. This was to remove contaminating non-arborising tubular portions of cells passing through other regions of the neuropil and leave only the regions of arborisation. I utilised the tubeness score from section 4.4 to select pruned skeleton points about which to remove signal (those with a score over 0.009). Around these tubular skeleton points signal voxels were removed within surrounding tubes with radius 4µm and directions as determined for nBLAST (see section 4.5.2). Following this, 26-connected components of voxels of count less than 20 000 were also set to zero to remove any remaining small tubular amounts or contamination. Finally, I dilated the remaining signal to ensure that gaps in the dendritic arborisation patterns were filled in. I implemented this on the remaining binary non-zero signal of the neural volume. I applied a float-precision smoothing using a normalised Gaussian convolution filter with standard deviation 3.5μm. This smoothed image represented a probability-like distribution of where the arborisation regions of the cells were (see figure 5.5b).

To speed-up computation time in the following calculations and to fit within memory limitations on the compute servers I was using for this process, I took subsets of voxels from these full resolution images. Voxels were sampled from a sparse, regular grid at points within the neuropil mask. For both the PCA results and domain overlap calculations below, the sample grids were taken at $[5, 5, 2]$ in the $[x, y, z]$ dimensions. In the work that follows in this section I used only the single cell neural volumes and their mirrors (those I used for clustering in section 5.2 above).

Evaluating the degree that the neural arborisation regions support the annotated synaptic domains is not straight forward due to the cells often innervating several different regions of the CNS. Nevertheless, I performed some simple calculations to begin initial investigations. To reduce overlap between ROIs, I investigated a truncated part of the CB region that removed most of the VNC in the CB ROI. I did this by cutting the CB ROI at 550 voxels in the y direction. Within this region, I looked at cells with arborisation volumes over 100 $(\mu m)^3$ across all these domains. I first found, for each cell, the percentage of each domain that was occupied by the cell's high arborisation volume. These high arborisation volumes were all parts of the arborisation representations that were over 0.1. The results are shown in figure 5.5c over all combinations of cell and domain and indicate that arborisation patterns in general do not occupy entire domains, but partially fill them. This does not necessarily suggest the domains are not valid, however, as the individual arborisation regions could still be contained within neuropil domains rather than across several neighbouring domains. This did indeed appear to be the case during manual visual inspection.

To also investigate the question of how many domains an average cell innervates, I calculated, for each cell, how many domains are at least 10% occupied by arborisation (see figure 5.5d). I chose this threshold in order to exclude small amounts of bleeding into neighbouring domains. The results suggest that most neurons innervate a small number of domains, if any (mean 2.01). It is not particularly encouraging for the

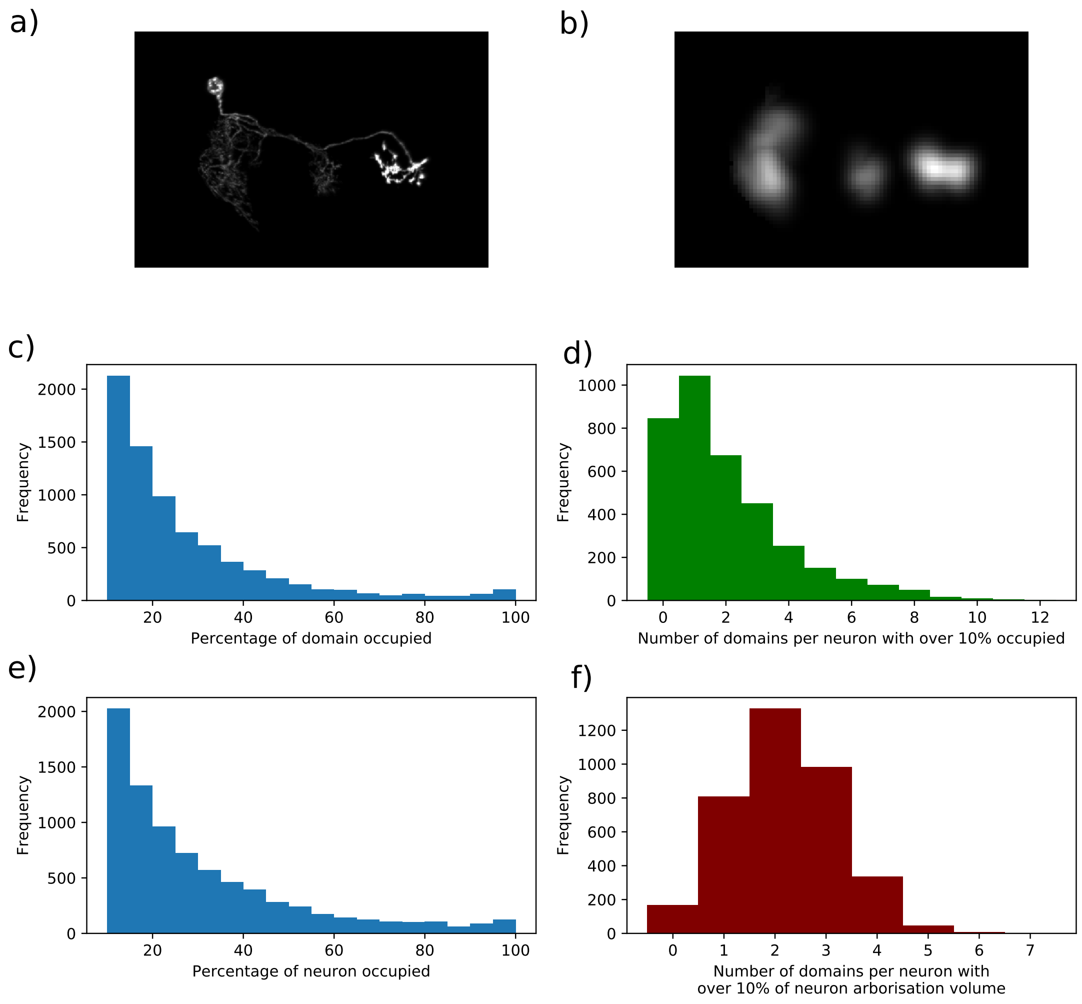


Figure 5.5: Arborisation Region Representation
Maximum z-projections of a) raw neuron signal b) arborisation region representation. Histograms for non-type 3 CB single cell neural volumes indicating c) the percentage of manually annotated domains occupied by the cell if over 10%, d) the number of manually annotated domains with which over 10% is occupied by the cell's arborisation signal, e) the percentage of cell neuron occupied by a manually annotated domain if over 10% and f) the number of manually annotated domains occupied by over 10% of the cell's arborisation signal. (The manually annotated domains were the lowest resolution except for the LT and split along the mid-line, except for the prFB).

parcellisation scheme, however, that there appear to be many neurons that do not occupy a single domain by this measure. Calculating the number of domains that contain over 10% of the arborisation volume of each neuron further demonstrates that most cells appear to innervate either 1 or 2 domains (mean 2.18, see figure 5.5f). That for these cells the arborisation regions are contained in a low number of domains does, however, suggest that there is some evidence for the parcellation scheme. The above results justify further investigating the suitability of the annotated synaptic neuropil domains and I will do this by attempting to find a natural parcellation scheme from the data itself.

5.4 VOXEL-BASED NEUROPIL DOMAIN CLUSTERING

5.4.1 *Introduction to Voxel-Based Clustering*

The synaptic neuropil domains described in section 2.2 were delineated based on years of experience by specialist anatomists who analysed many different gene and cell expression patterns in different specimens of individual larval central nervous systems and related them to the adult. More recently, however, computational approaches to neuroanatomy have been able to assist in adding evidence for these compartment-like subdivisions.

Much of this work has analysed gene expression patterns in the mouse brain, based on that of Bohland et al. (2010). Here it was shown that by treating the voxels as observations and their expression levels of each gene as a common feature across all voxels, it was possible to apply clustering algorithms to the voxels using the features to calculate a similarity metric between two voxels. These resultant clusters somewhat resembled, in standard brain space, the parcellation of the standardised and manually-devised Allen Mouse Brain Atlas Lein et al. (2007). More recently, a study automatically segmenting the adult *Drosophila* brain was able to postulate the existence of novel optic glomeruli domains Panser et al. (2016). Voxel clustering studies usually first employ a dimensionality reduction method in order to reduce the number of features so as to reduce redundancy and enable algorithms to run more efficiently, before employing a computationally intensive clustering technique. This is the technique I will follow here with the larval *Drosophila*. I will, however, consider only single cells rather than entire gene expression patterns. I believe that this will give more robust results since gene expression may not be related to domains whereas the neuropil volumes are delineated by the combinations of individual cell morphologies.

I will investigate the CB only in this study as the neuropil domains in this region has been described in the literature far more than the VNC and have more complex manually determined descriptions to supply a better comparison. First, I selected the cell clusters remaining above 4.0 in the ward cell clustering (2 949 clusters), and chose to represent each one with the cell with the lowest average distance to the other cells in the same cluster. This was to reduce the effect of over-represented cell types in the data set of single cells. I investigated voxel clustering using both the arborisation region representation and the raw signal values inside the neural volume, sampled at the same regular points as the arborisation region data. Again I used the reduced CB ROI (cut at $y = 550$). Any points outside the ROI were labelled as zero.

I detail below the two different methods that I tested for the dimensionality reduction that I required due to the large number of cells, namely PCA and t-distributed Stochastic Neighbour Embedding (tSNE). Following this I detail the clustering methods I applied, k-means and a Gaussian Mixture Model (GMM). Finally I will compare the results of all combinations of the above by relating them to the anatomically labelled domains.

5.4.2 *Principal Component Analysis*

PCA applies a linear mapping from the data axes to new orthogonal axes that can be ordered such that the variance of the data is maximised in each successive dimension following projections of the data onto the new axes Bishop (2006). This is usually achieved by either finding the eigenvectors of the covariance matrix or through finding the left singular vectors through SVD, and ordering the new dimensions by decreasing eigenvalue or singular value respectively. In the following I took the highest 500 singular vectors, which I found with the scikit-learn software package Pedregosa et al. (2011), thus reducing the number of dimensions by 88% (see figure 5.6). Figure 5.7 reminds us that the voxel representations in these PCA dimensions, or *modes*, are patterns in the voxels themselves - exhibiting spatial clusters - the weighted combinations of which build up the cellular arborisation patterns input into the procedure.

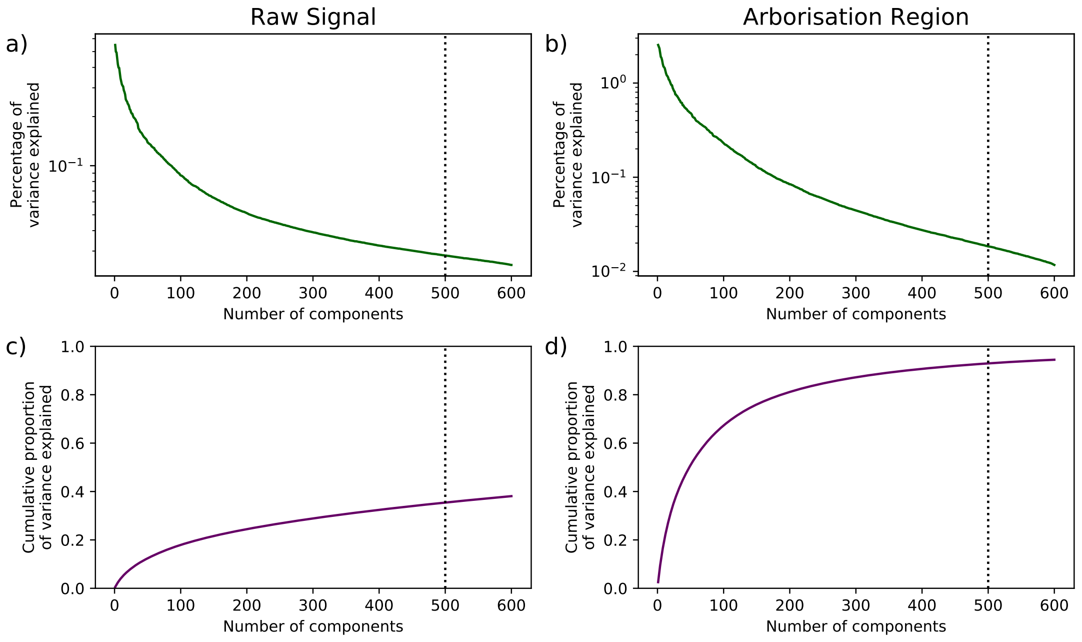


Figure 5.6: PCA Accounting for Variance in the Central Brain
a), b) Graph of the percentage of the total variance explained by each successive dimension in the PCA representation a) raw signal, b) arborisation regions. Dotted line at 500 components, where 35.4% of variance is explained for raw signal, 93.0% for arborisation regions. Note the logarithmic left Y-axis. c), d) Graphs as a) and b) respectively but for the cumulative variance explained.

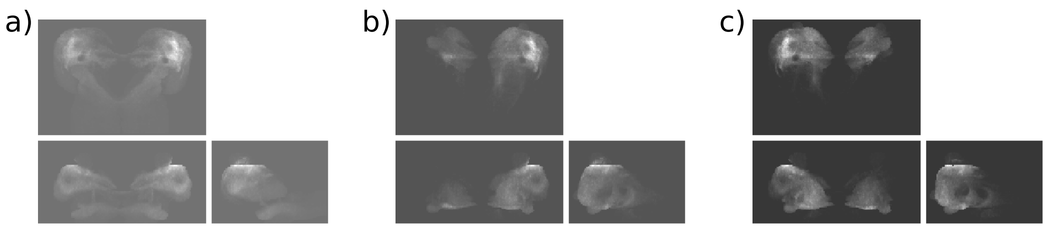


Figure 5.7: Central Brain PCA Mode Plots
Maximum projections of the a) first, b) second and c) third modes of PCA for the raw signal. Top-left: z-projection, bottom-left: y-projection, bottom-right: x-projection.

5.4.3 *t-distributed Stochastic Neighbour Embedding*

tSNE Maaten and Hinton (2008) is a reasonably recently developed technique used for visualising clusters in data sets in low (usually 2 or 3) dimensions, now frequently employed in data science. Ji (2013) and Mahfouz et al. (2015) have shown that this technique can be applied as the dimensionality reduction step in order to produce parcellation results more in line with the Allen Brain Atlases for human and mouse brains than those of PCA. tSNE first defines a probability distribution for each point, i, to all other points, that is as follows:

$$p(i,j) = \frac{p(j|i) + p(i|j)}{2N} \tag{5.6}$$

where N is the total number of points. $p(j|i)$ is the probability that a point i would pick other point j as its only neighbour, with this probability being proportional to the value at point j of a Gaussian in the original space centred on point i with a uniform variance, σ_i^2, in all dimensions. $p(i,j)$ is then mapped to the low dimensional space so that the probability in this space, $q_{i,j}$, is proportional to the *Cauchy distribution* (heavy-tailed Student's t-distribution) acting on the distance squared between the two points:

$$q(i,j) = \frac{\left(1+\|y_i - y_j\|^2\right)^{-1}}{\sum_{k\neq i}\left(1+\|y_i - y_k\|^2\right)^{-1}} \tag{5.7}$$

This is achieved by minimising the Kullback Leibler (KL) divergence between the probability distributions for each point, so that they match as closely as possible:

$$KL(P\|Q) = \sum_{i\neq j} q(i,j)\log\left(\frac{p(i,j)}{q(i,j)}\right) \tag{5.8}$$

The gradient of the KL divergence can be calculated analytically, enabling an iterating algorithm to update the positions of the points in the low dimensionality space using a given learning rate.

Converting the points between the two spaces with different probability distributions using tSNE, when applied correctly, will produce meaningful clusters that are clearly separated, although the distances between clusters (and variances within them) are usually not meaningful. It goes some way to addressing the "crowding problem" that occurs in low dimensional representations where the distances between close points are expanded to be comparable to those of moderately separated points and clear clusters do not form. The heavy tail of the Cauchy distribution adds a re-

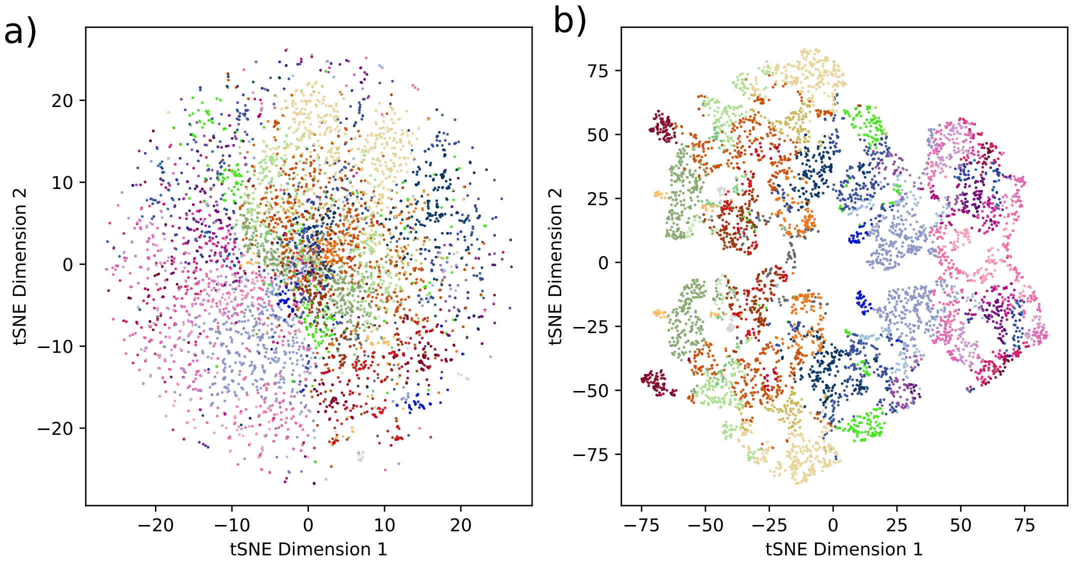


Figure 5.8: 2D Representation of Neuropil Domains Using t-SNE
The 2D t-SNE representations of points from the Central Brain neuropil domains colour coded as they were labelled from anatomical experience in section 2.2. These were the aesthetically best clustered I could find in 2D by varying perplexity a) is for the raw signal with perplexity 200, b) is for the arborisation regions with perplexity 30. Notably, high learning rates were required to avoid local minimums.

pulsive force to mitigate this.

In using the Ulyanov (2016) parallel python library implementation of tSNE, I was able to make use of the Barnes-Hut tree approximation for calculating the gradient contributions for distantly separated points Barnes and Hut (1986). I tuned the parameters (most notably the learning rate and *perplexity* that controls σ_i^2) so that distinct clusters formed across most of the data. It was reassuring to observe that, for the arborisation region data, these apparent clusters grouped together very well voxels belonging to the same annotated domains (those based on the literature). See figure 5.8. The 2D map recreates the gross CB structure well as a flattened brain, including regions such as the AL and VL. It is important to note that as the mapping is learnt directly on the dataset, it is not possible to find the 2D representation of other extra points from the model. As the algorithm required a large amount time and data, I ran the tSNE on a grid of $[10, 10, 6]$. I interpolated this back to a $[5, 5, 3]$ grid by finding each point's nearest neighbour *in the original feature space* when assigning voxels to clusters.

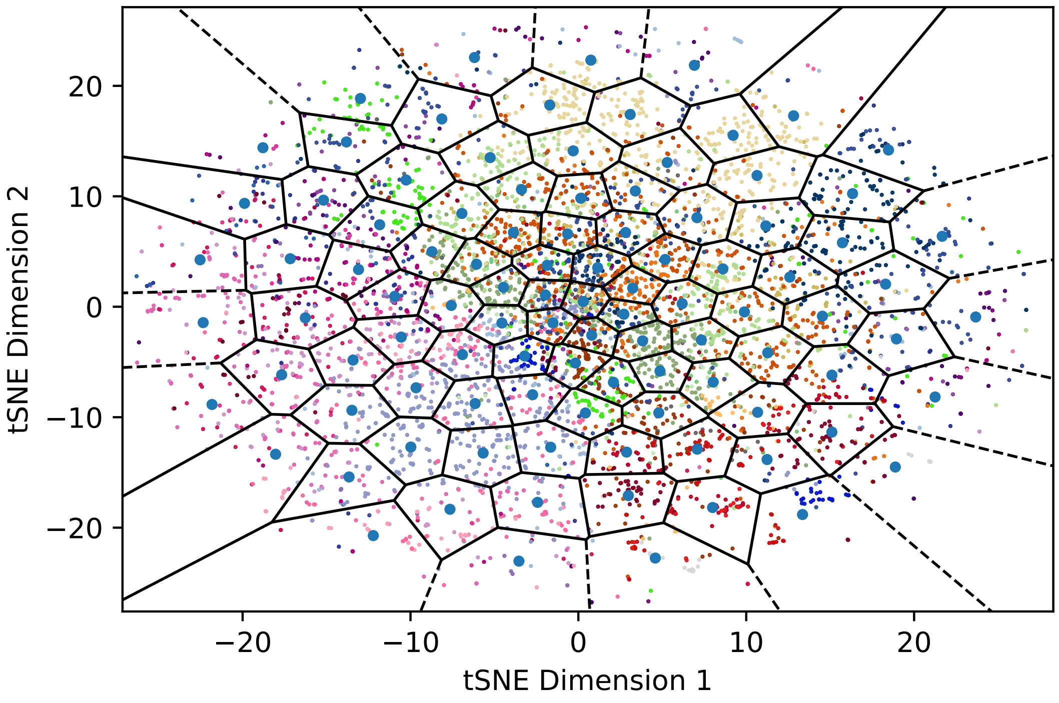


Figure 5.9: K-means Clustering
K-means clustering as applied to the 2D map from tSNE in section 5.4.3. The region partitions are shown via a *Voronoi diagram* using the means of the clusters as centre points (blue dots). The black lines demonstrate the boundaries that are equidistant from two or more cluster centres.

5.4.4 *k-means Clustering*

K-means is an iterative clustering algorithm that attempts to minimise the intra-cluster variances of the points in the feature space (as noted previously, this also minimises the total intra-cluster point separations in Euclidean space). The constraint of the model is that each point observation is assigned to the cluster whose mean position is closest (see figure 5.9). Following an initial assignment of values as "means" for a pre-determined number of clusters, each observation is assigned by the constraint, and the new means are calculated from the cluster's constituent observations. The process iterates until none of the observations' cluster assignments change (or a set limit is reached). It can be shown that the k-means algorithm is a special case of this Expectation-Maximisation (E-M) procedure explained in the following sub-section and in Dempster et al. (1977). I utilised the implementation of the scikit-learn python toolbox Pedregosa et al. (2011), using the k-means++ algorithm for mean initialisations Arthur and Vassilvitskii (2007).

5.4.5 *Gaussian Mixture Model*

K-means tends to prefer clusters to be of similar spatial extents, and - in the PCA space at least - it is not clear that some domains will be separated by similar amounts. I attempted to mitigate this problem by also experimenting with using a GMM. This model treats the class (or cluster) to which each voxel observation belongs as a latent variable $\mathbf{Z}$, and models the probability of any set of observations, $\mathbf{X}$, as the joint distribution:

$$p\left(\mathbf{X}, \mathbf{Z} | \boldsymbol{\mu}, \boldsymbol{\Sigma}, \boldsymbol{\pi}\right) = \prod_{n=1}^{N} \prod_{k=1}^{K} \pi_k^{z_{nk}} \mathcal{N}\left(\mathbf{x_n} | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k\right)^{z_{nk}} \tag{5.9}$$

where there are K clusters and N observations. z_{nk} is the kth component of the $\mathbf{z_n}$ cluster determinant vector which is one for the observation's assigned cluster and zero for all others. $\mathcal{N}$ is the normalised multi-dimensional Gaussian distribution with as many dimensions as the feature space, mean $\boldsymbol{\mu}_k$, variance $\boldsymbol{\Sigma}_k$ and *mixing coefficients* $\boldsymbol{\pi}$. One desires finding the parameters, θ, i.e. $\boldsymbol{\mu}, \boldsymbol{\Sigma}$ and $\boldsymbol{\pi}$, of the model that give the highest likelihood, $p\left(X = x|\theta\right)$, for the observed data, x. This cannot be analytically maximised, but the log likelihood (and hence likelihood) can be iteratively increased by changing the parameters via the two-step E-M process Dempster et al. (1977):

After first setting initial parameters (randomly or as the results of k-means) we can calculate the *expectation*, $p\left(\mathbf{Z}|\mathbf{X}, \theta^{\text{old}}\right)$ given the current parameters, θ^{old}, using the model in equation 5.9. Next, the remaining appearance of the parameters of the clusters in the following quantity can be maximised analytically (see Bishop (2006)):

$$\sum_{\mathbf{Z}} p\left(\mathbf{Z}|\mathbf{X}\theta^{\text{old}}\right) \ln p\left(\mathbf{X}, \mathbf{Z}|\theta\right) = \underbrace{\sum_{\mathbf{Z}} p\left(\mathbf{Z}|\mathbf{X}\theta^{\text{old}}\right) \ln p\left(\mathbf{Z}|\mathbf{X}, \theta\right)}_{\text{KL Part}} + \underbrace{\sum_{\mathbf{Z}} p\left(\mathbf{Z}|\mathbf{X}\theta^{\text{old}}\right) \ln p\left(\mathbf{X}|\theta\right)}_{\text{Likelihood Part}} \quad (5.10)$$

This works towards a local maximum for the likelihood because any deviation from θ^{old} in the first "KL part" of equation 5.10 will decrease the quantity due to KL divergence, leaving the other, "Likelihood part" to be guaranteed to increase, and hence increasing $\ln p\left(\mathbf{X}|\theta\right)$. Again I utilised the scikit-learn python toolbox for implementation, which allowed me to include regularisation in the form of adding a small value to the diagonals of the covariance matrices and thus help preventing Gaussians collapsing onto a single point.

5.4.6 *Choosing the Best Model*

The human-expert segmented brain from section 2.2 provides some domain boundaries that are very clear in the background staining and appear to be respected on visual inspection of the cells. This is particularly true for the Mushroom Bodies and the Antennal Lobe, and therefore I expect some similarity between the automated and manual clustering systems. This also appears to be the case from figure 5.8. I will investigate similarity between these using a measure of overlap. I will use the *S-Index*, defined by Bohland et al. (2009), which does not penalise the case of several sub-clusters contained within another domain. This is useful since I found that I needed to increase the number of domains beyond the expected number in order to delineate some of the smaller domains and would be able to combine some clusters to better match the manually segmented domains. This also allows me to potentially discover new sub-domains.

The S-Index first calculates a domain i to cluster j score, X_{ij} - that is the proportion of the smaller domain in the pair that overlaps the other:

$$X_{ij} = \frac{|r_i \cap r_j|}{\min\left(|r_i|, |r_j|\right)} \quad (5.11)$$

This ranges from 0 to 1, where r is the set of voxels in each domain/cluster. The S-Index is then given by:

$$S = 1 - 4 \sum_{ij} W_{ij} X_{ij} \left(1 - X_{ij}\right) \tag{5.12}$$

Where W_{ij} is a weight, for each ij contribution, that is proportional to the size of the smallest domain in the overlap calculated as follows:

$$U_{ij} = \begin{cases} \min\left(|r_i|, |r_j|\right) & \text{if } X_{ij} > 0. \\ 0 & \text{otherwise.} \end{cases} \tag{5.13}$$

$$W_{ij} = \frac{U_{ij}}{\sum_{ij} U_{ij}} \tag{5.14}$$

Since $X_{ij}(1 - X_{ij})$ has a maximum value of 0.25, S ranges from 0 for bad correspondence to a maximum of 1 for complete matching. It penalises partial overlapping most.

I calculated the S-Index for all lowest level domains except the Lower Toe (part of the Medial Lobe) and split the CB domains into both left and right along the medial line (except prFB). I also calculated an S-Index exclusively for the MB region as this is a well defined volume that should be segmented clearly by the clustering. I decided to evaluate the S-Index when allowing the clustering algorithms to use 80 clusters. This corresponds to less than the number of domains in the same volume (95), but as many were in the VNC region where there was little signal, I believed this was more appropriate and would enable me to investigate more of the gross structure.

Table 5.3 demonstrates that taking as input the raw signal rather than the arborisation region gave results closer to expectations from the original template annotations from section 2.2. In addition, the arborisation region method domains were visually more appealing since they suffered a lot less from disparate, separated signal classed as the same domain, most likely due to the removal of the tubes. As the results of the tSNE were noisier than those of the PCA, I smoothed a binary representation of each domain in the 3D space using a Gaussian filter and assigned the voxel to its new highest valued domain at its point. This reduced the performance somewhat (see table 5.3), especially for the raw signal. It was still generally better, however, than the PCA results in terms of S-Index and greatly improved the visual appearance

Table 5.3: Voxel Clustering Results (S-Index)

Input Type		Arborisation Region		Raw Signal	
Method Type		K-Means++	GMM	K-Means++	GMM
PCA-500	S-Index	0.618	0.638	0.646	0.729
	S-Index MB Only	0.580	0.575	0.582	0.446
TSNE-3D	S-Index	0.649	0.658	0.898	0.729
	S-Index MB Only	0.619	0.669	0.839	0.686
TSNE-3D Smoothed	S-Index	0.622	0.627	0.721	0.650
	S-Index MB Only	0.605	0.649	0.742	0.687

Clustering results averaged over 10 runs each due to random initialisations, PCA GMM initialised using K-Means++ to converge in a reasonable time, tSNE randomly. The tSNE had perplexity 35 for raw signal and 40 for arborisation regions, optimised for results here. The MB-only S-Index is calculated for the singular, MB compartment only. Note that I used 3 dimensions for the tSNE as this significantly improved the results over 2D (not shown). tSNE smoothing Gaussians had standard deviation 2.1 for raw signal and 1 for arborisation regions.

of the domains. The PCA stacks suffered from smaller domains being grouped into one large domain that included many near-cortical regions and applying the GMM method tended to produce overly large, merged domains for random initialisations.

As a result of the high S-Index scores and the above considerations, I have based my following comparison analysis on the smoothed tSNE k-means clusters for raw signal, although I also studied the other stacks in order to assist in identifying key trends. I manually assigned each automatically segmented domain to what appeared to be the most suitable template domain to produce the maps in figure 5.10. Note that some of the resultant domains might be made up of more than one subdomains from the above clustering.

5.4.7 *Evaluation of Automated Domains*

The reference point in figure 5.10 is shown in square brackets in the following.

On both sides, the MB is well segmented from the rest of the stack, with both the VL [b1] and LT [b2] clearly separate subdomains. The MAML is also well segmented into its own domain [a1].

On the right side both the SMPp [d2] and SMPa [b3] domains are very well segmented apart from some bleeding of the SMPp into the template's IPp [d1]. On the left

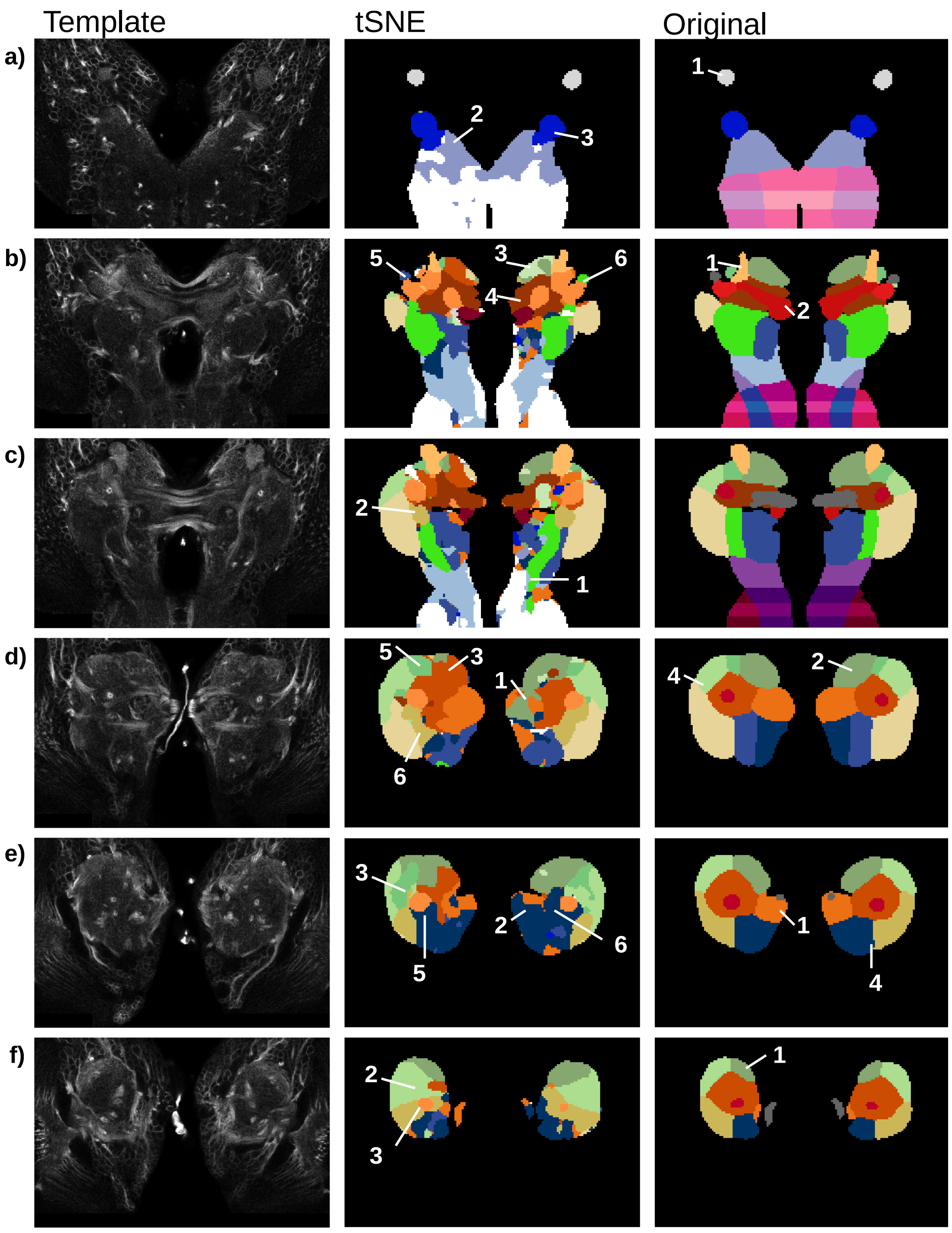


Figure 5.10: Comparison of Automated to Manual Domains
Individual slices at various points in the Central Brain ROI to illustrate differences between the original manual neuropil domains and those automatically produced using 3D tSNE with K-Means++ and manually matched up. Note that there are some subdivisions that are not visible in the automated domains as those domains have been matched to the same original domain. The annotations are referred to in the text.

side, the posterior of the SMPp is similarly segmented [f1], but more anteriorly both the SMPp and SMPa become merged with the Clamp [d3]. The IPp on the other hand matches well on the left side but on the right is replaced in part (as mentioned) with the SMPp [d1] and more posteriorly with the VMCp [e2]. The segmentation of the IPp and SMPp were more within the bounds of the original annotations in PCA k-means, suggesting that perhaps a higher resolution k-means tSNE of these zones may be able to separate them.

The SLP is similarly shaped to as it is defined in the template, except the thin border with the VLP is more ventral in the tSNE plot [d4] and posteriorly it replaces the Clamp to abut the Peduncle [f2]. There is happily evidence for the existence of a separate SIP domain [d5], though it extends much further posterior, also projecting into the Clamp [e3].

In the ventral CB, the LAL boundaries match well within the CB except that it extends some way into the Tritocerebrum dorsal (TRd) on both sides [c1]. The VLP is clearly defined and matches the original very well except for the aforementioned thin border with the SLP and some dorsal replacement with the PLP more anterior than expected [d6]. The template PLP domain is found in the tSNE stack but it appears this is linked to a domain that descends through the dorso-medial part of the VLP to meet the LAL [c2,d6]. The VMCa and VMCp are broadly occupying the same regions as in the original, except that there is additional posterior bleeding over into the IPp (especially right side [e2]) and the Clamp [e5,e6]. A notable mention for strong similarity in the ventral region is that the *virtual* sagittal planes separating the domains are shown to be broadly accurate [e4].

The Crepine surrounds the medial lobe as expected, which is pleasing given its complex shape [b4]. The prFB are included in the Crepine in the automated stack, however. The Clamp is perhaps the most problematic domain, although its anterior is slightly better despite the bleeding into the SMPp [d3]. Much of the Clamp has been replaced by either extensions of the surrounding domains inwards (the VMCp [e5,e6] and more posteriorly, PLP [f3] and SLP [f2]) or antero-postero projections of them (notably the already mentioned PLP [d6,c2] and SIP [e3]). This suggests the Clamp is a heavily integrative zone of the neuropil.

Besides demonstrating possession of a clear ventral domain matching expectations, the Tritocerebrum was difficult to interpret with conflicting information from each side. Claim by the Tritocerebrum ventral (TRv) was made, however, for the former

BCv compartment extension into the CB [not shown].

Another feature to note was that although some tSNE domains (notably the MB) were present on both sides, most were of one side only. Where this dual listing occurs, this suggests the presence of many cells that innervate the same domains on both sides. The prAOTU was also separated clearly from surrounding domains and took on the identity of the more distant VMCa (left, [b5]) or LAL (right, [b6]).

Some parts of the Clamp may be too thin to be realised by the segmentation (though this would be surprising since the Crepine was successfully segmented). It could also be the case that the Clamp is very varied in terms of its signal profile, with many projections from other regions giving many varied voxel profiles. In this regard, I will mention two specific automated domains that were difficult to match to the original domains as they overlapped large portions of two domains (a *one-many* relationship). First was one that encircled the right Clamp and also met the cortex across where the VMCp was expected (assigned to the VMCp) [e6]. The other was an tSNE domain that covered both the Clamp and the left SMPa (assigned to the Clamp) [d3].

After the above analysis it is clear that there is somewhat support for the parcellisation scheme, particularly in the MB and outer neuropil and ventral regions. There was not time during my project, unfortunately, to investigate this in further detail or for the VNC.

5.5 INTEGRATION WITH THE VIRTUAL FLY BRAIN WEBSITE

The versatile VFB website has been designed to serve as an atlas of a range of templates with linked annotations. Therefore it was easily expanded to integrate my work by adding my template to the template repository and setting up the web address `larva.virtualflybrain.org` to load it. The domains and lineage tracts / fascicles were linked to the template and their FlyBase references. This data was stored along with centre of mass in a table and binary image stacks and binary volume 3D representations (see below) were also added to the repository. Finally, a table of neural volumes was added with their meta data and individual signal image stacks and cell 3D representations also calculated and deposited.

The binary representations of the domains were easily exported from TrackEM using the built in area list exporter. There was not, unfortunately, a readily available

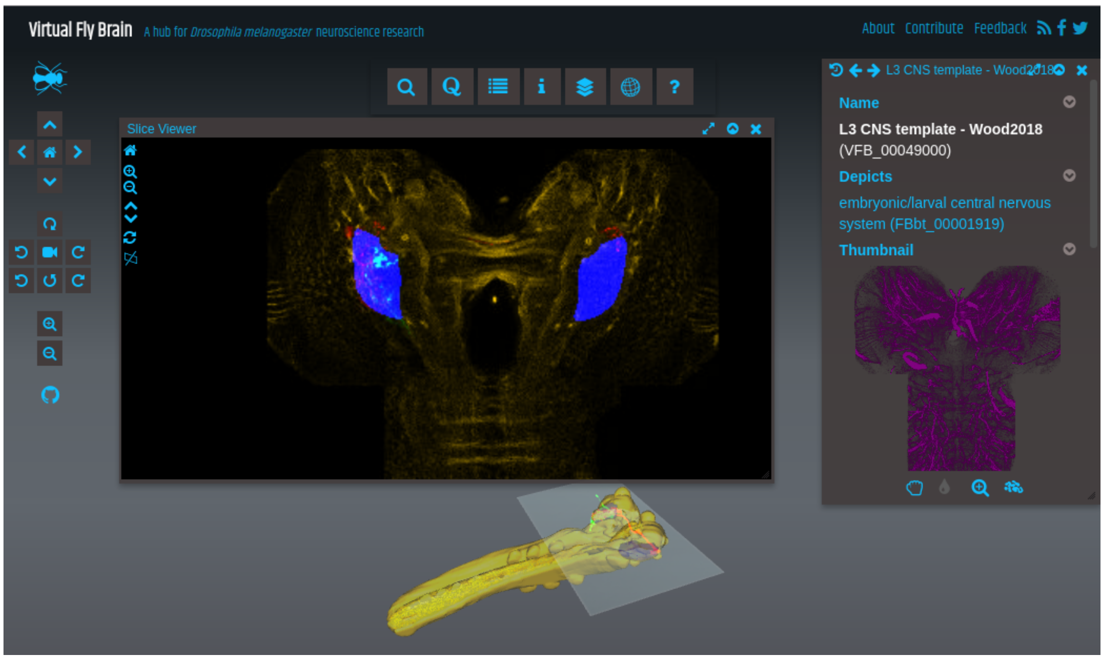

Virtual Fly Brain
A hub for Drosophila melanogaster neuroscience research
About
Contribute
Feedback
Slice Viewer
L3 CNS template - Wood2018
Name
L3 CNS template - Wood2018
(VFB_00049000)
Depicts
embryonic/larval central nervous system (FBbt_00001919)
Thumbnail

Figure 5.11: Example of Slice Viewer in Larval Virtual Fly Brain.
A slice (in any 3 co-ordinate planes) can be viewed and navigated through using the controls on the left of the Slice Viewer window pane (top left). The position of the slice in the 3D model is displayed below in the main window. The right Information Panel displays meta data concerning the presently selected domain, cell or - in this instance - the template. The cell meta data includes the GMR lines from which it has been expressed. There are also tools in the Information Panel allowing a user to change an item's display colour and visibility. See figure 1.5 for a further explanation of the VFB interface.

method to extract pipes from TrackEM for the lineage tracts and fascicles. Therefore I wrote a python script to extract these from the eXtensible Markup Language (XML) TrackEM file in which they were recorded as *Bezier curves* and redrew them in the same resolution as the template and domains using a self-written fast and specific C program. I converted these binary representations to 3D surface meshes using the Lewiner marching cube algorithm Lewiner et al. (2012) which is implemented in the scipy library. These were simplified using the Clustering Decimation and Laplacian Smoothing procedures of MeshLab Cignoni et al. (2008) in order to reduce memory requirements.

I only took the good quality, non-type 3, neural volumes for inclusion as part of the Larval Virtual Fly Brain. Some further processing was required for 3D views: One method of 3D representation of cells on the virtual fly brain website, devised by RC, is that of point clouds (see figure 5.12). All voxels above a threshold intensity are included with the threshold adjusted to give a large number of points (of the order 15 000 - 50 000) (RC, personal communication). In addition to performing this

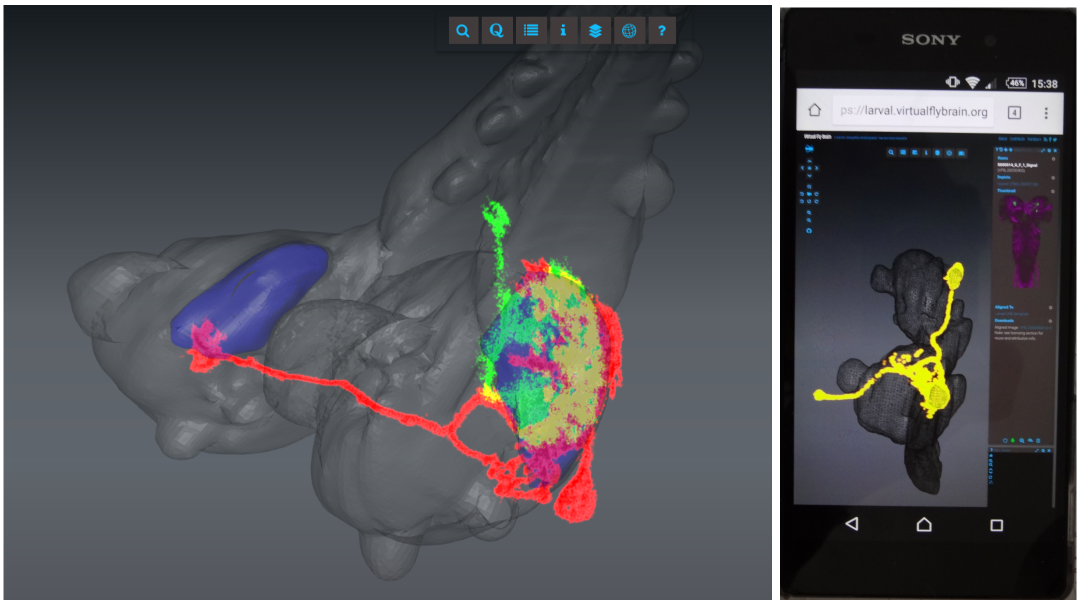


Figure 5.12: Example of 3D View in Larval Virtual Fly Brain.
Left: Interactive 3D view of the cells shown in figure 5.11. The two cells are in red and green with the innervated VLP compartment displayed in blue. Right: A similar situation but with different cells as viewed through the browser on a mid-range smartphone.

processing, I also exported the skeletons as uniform-radius pipes for an alternate 3D view that may be included in a future version of the website.

Neurons are named based on the GAL4 line for which they are subsets of and a unique identifier. It is therefore now possible for a researcher to find neurons in the larval brain either by directly searching for the cell's name or GMR line or instead by first finding a domain and running a query for all cells that innervate this domain. I implemented this by simply calculating the overlap of the signal in the same method as in 5.3 and storing this as a relationship in the global *Drosophila* ontology. If the domain name is not known, a point in the neuropile of the stack viewer can be selected and the domain at that point identified and overlaid (see figure 5.11). Once a neuron has been found, queries can be run from this, including interfacing to the BLAST query API to find similar cells and those that overlap (indicating possible synaptic contact). The genetic line for each neuron is displayed upon selection, allowing researchers to compile lists of lines of interest for particular cell types.

5.6 CHAPTER SUMMARY

I manually checked and curated a high quality dataset of a over 8 000 single cells in the larval *Drosophila* CNS and integrated them with the world-leading resources for

Drosophila neuroanatomy and gene expression, VFB and FlyBase. I further validated the M/L comparison metric by clustering neurons of similar types together and I created a robust tree of neurons based on this similarity metric. It was, however, not possible to obtain a clear estimate of the number of cell types that were present in the data set. I clustered the CB neuropil based on the individual cell signal expression patterns and found strong evidence for most of the manually annotated domains and noted some domains to investigate further, namely the Clamp and those of the centro- and dorsal Tritocerebrum. This was the first data driven analysis of neuropil domains in the *Drosophila* larva.

5.7 CHAPTER DISCUSSION

The manual cell inspection made it clear that there were high failure rates for the segmentation (41.2% for CB and 69.0% for VNC). Although this sometimes occurred for very faintly labelled cells, it occurred much more often when stacks were very "busy" (i.e. not very sparse) which was more often the case in the VNC. It was disappointing to throw away so much data, but most were from clumps of many cells which my project was not designed to target since it was not of great interest to the VFB website where single cells were desired. Investing the time in reviewing all cells was informative for familiarising myself with the general morphologies of the cells. I also now have a large training set if I wanted to develop an automated procedure to review the segmentation quality. The imaged ROI, however, often did not coincide with the bounding boxes of the registered stacks, an artefact of the original image production pipeline (and out of my control). This suggests that detecting type 2 deficiency (image ROI bordering) cells would be difficult to automate. A final point to note here is that cells that do pass out of the imaged/template ROIs could have their cell number labels compromised by the fact that some discretely interwoven additional cell might have its cell body outside the ROI. One cannot simply count cell bodies.

The large differences in number of cells for similarly performing cut-off measures illustrates the difficulty in determining the number of cell types - not only with these automated methods but also in general. The large variation in scales and many partial similarities as well as biological variation make it difficult for my and other comparison metrics to have a uniform cut-off point for number of cell types. This task can also be difficult for human experts; for a few of the clustering test sub-sets there were a small number of cases it was not clear if the cell was indeed the same or not, and I conservatively did not mark them as the same. This has also been stated as a problem by JT (personal communication). When combining sparsely expressed signal from different specimens, it is not possible to know what other cells are present in the candidate cell's stack that have not been expressed but might be a better match to a query. To tackle this, one could inspect the EM data where all neurons are traced to identify the number of cell types of this morphology that actually exist in a single specimen. As multiple EM volumes are traced this should allow cell types to be identified across specimens, where most types consist of just one cell in each side of each specimen's CNS.

It was interesting to observe that although plots of the tSNE 2D (and 3D) representations of the originally annotated domains appeared much better for the arborisation region method, the raw signal gave higher S-Indices. I also found the process of manually matching up the manually annotated domains to the automatically segmented ones easier for the raw signal (since there were less one-many cases). The choice of method based on S-Index appeared justified, and the chosen method was a clear good performer, especially for the MB and confirmed the applicability of tSNE to this problem. The S-Indices here compare favourably with those in Ji (2013), with my results at the top of the range reported in that study.

The similarity of both sides of the brain following automated clustering should be expected from inclusion of the mirrored cells. It is less clear why the raw signal performed better than the arborisation regions. One consideration could be that the arborisation regions were not precise enough due to the smoothing.

In the adult brain, the VLP and PLP border the Peduncle and this appears to in fact be implied by the automated tSNE stack. There was also little evidence of a lateral-medial split in the Clamp for either the tSNE or PCA stacks but instead a suggestion of a dorsal-ventral split, again as in the adult. These observations and the problems mentioned in the text regarding the Clamp suggest that this is a region of neuropil that requires attention. It was notable that the manually specified Clamp inhabited a region in the tSNE plots that was highly compact but with much noise from other domains, consistent with the idea that it is a complex mixture of many cells (which one might expect with it being in the centre of the CB).

Another region of interest is the Tritocerebrum, a region that it has been difficult to define synaptic neuropil domains in (VH, personal communication). It is likely that this under-studied region will receive updated domains in the future by anatomists, and the results here might be helpful in defining them, as they will for reviewing all domain definitions. Of particular interest was the spreading out of some of the proto- and deutocerebrum domains into the Tritocerebrum, suggesting these regions are fused more than perhaps previously thought. Though it should also be recalled that this is the region where registration performed worst and automated domains varied somewhat from left to right, so results should be treated with caution.

Adding more clusters might resolve some of the one-many relationships between the automated and manually segmented domains when matching them. Adding more clusters, however, will eventually enable matching to any prior parcellation scheme (by creating predominantly many-one relationships). I would only increase the number of clusters having devised a model to determine the probability of the automated segmentation indeed being from the manual prior rather than one that is random. I briefly investigated hierarchical clustering with the dimensionality-reduced data but the results were not satisfactory. Alternative clustering methods could explicitly include the spatial relationships of the voxels so as to preserve domains as connected components, which might help alleviate some of the disjointed domains in the tSNE stack. This could be added as a constraint in the hierarchical clustering when merging the clusters or as part of an alternative image segmentation algorithm such as *Fuzzy C-Means* that was applied by Liu (2014) to human brain images. Given more time I would explore these methods.

It should be noted that here I have attempted to improve and refine the neuropil domains by analysing a clustering scheme that produced compatible (high S-Index) clusters to those already manually annotated. This limits the discovery of an alternate domain regime that might also be valid. It remains to be seen how important the domains are in the function of the larval brain, and indeed there may be several valid parcellation schemes depending on the function. The domains might also be fuzzy, in which case a probability of membership of each domain could be given for each voxel.

6 GENERAL DISCUSSION

I successfully produced a searchable database of single-cell neurons for the larval *Drosophila* CNS, integrating it with an online and interactive viewer in which they can be displayed in both 2D and 3D and explored in relation to each other and the surrounding anatomy. Researchers are actively using this tool in order to find GAL4 lines to use in their experiments to genetically target specific neurons. In the process of developing the above, I devised novel methods to evaluate the registration accuracy of images and employed recently developed Machine Learning methods to enable much faster accurate identification of similar cells. I also demonstrated that the manually annotated neuroanatomy has a mostly sound grounding in the data and identified regions for further investigation. Although the EM reconstructions will provide better resolution for determining potential cell connectivity, my database provides another route for this and one that can demonstrate what biological variance exits as its samples are taken from thousands of individuals rather than a single CNS.

6.0.1 *Deficiencies and Potential Improvements*

In addition to those mentioned in the individual chapter discussions, there are a few areas where I would allocate more effort were I attempting to improve the system. Unfortunately, as with many of those potential improvements I have suggested previously, due to the linear nature of the pipeline, they would impact heavily on the following results and as such the feasibility of employing them for this project as a whole is limited by the restrictive timeframe. I have created and well documented, however, the baseline for a searchable database of single-cell Drosophila larva cells with clear performance metrics that alternative methods can be tested against.

In particular, just over half the neural volumes were judged not usable due to poor registration. Although the registration pipeline appears to perform comparably to other recent studies, other methods should still be investigated. The improved speed of the elastix toolkit used in Muenzing et al. (2018) might enable more accurate registration by utilising longer time periods. In addition, a statistical template should be compared to the one I have created.

The other major part of the process causing neural volumes to not be added to the final database was their failure to meet the standards of segmentation for inclusion. This was mostly down to poor segmentation from the background - i.e. missing sections in the cell. Whilst many of the neural volumes that failed are from stacks containing a large amount of signal, and would thus likely not be single cells, enough are not to warrant further investigation of this, particularly for the VNC. Approaches that have been trialed elsewhere to segment the neural signal from the background include utilising a limited watershed segmentation Russ (2011) and a M/L algorithm with CNNs Li et al. (2017a). I would investigate both of these if I had the time to not only evaluate them but re-evaluate the rest of the pipeline.

One of the reasons for the poor segmentation method being utilised was that I did not initially evaluate it empirically or as thoroughly as in section 5.1 (i.e. manually). Evaluating all methods as I did in 5.1 would be time consuming. It might therefore be preferable to create a well segmented test set to evaluate against. At the very least a set of image stacks with their optimal threshold (found manually) could be used to evaluate threshold finding algorithms and perhaps increase the amount of signal remaining at the threshold for VNC stacks. Indeed this is the first thing I would adjust in the pipeline. Other methods might require comparisons of the shapes and sizes of resultant neural volume voxel components, with a test set containing manually or semi-automatically produced volumes for comparison.

Rather than improving segmentation directly, however, it might be preferable to utilise an advanced tracing algorithm that works with the raw signal of the image stack, such as several of those detailed in section 3.9. The segmentation could then be inferred back from a good tracing by including surrounding neural volume. Since the segmentation had a direct impact on the performance of the skeletonisation method, it makes sense to address these problems together. This approach might also be able to split some multiple neural volumes using techniques to separate cell traces.

An improved tracing algorithm might also better the results of the non-fixed points matching algorithms. I am doubtful they will improve much, however, since the pruning approach I applied utilised the main neurite branches in the traces. Nevertheless, this should be tested. Although I considered the matching performance my ultimate goal and could evaluate tracing accuracy against this, it would be more convenient to evaluate the tracing accuracy directly. For this a test set is required of manually traced cells. Accurate traces could be used for displaying the neurons in 3D on the VFB website. The limited resources, particularly with respect to time and expert biological knowledge required to perform the manual tracing, limited its use in this project, but having identified the problem I believe it would have been worth acquiring. This tracing test set would also allow evaluation and benchmarking of the pruning algorithm approach to test my theory that it is fairly accurate.

Possessing the ability to assign cells to lineages accurately would enable a data-driven study of the cell morphologies of each lineage that would, when coupled with knowledge of the genetics of the neuroblasts, be highly useful in determining how these morphological features are encoded in the DNA and what proteins are involved Hartenstein et al. (2015). The differences as mentioned in section 1.3 between PATs that the cells follow and the SATs that I have annotated may explain some of the problems, although when I attempted to weight the tracts by proximity to neuropil entry point (and hence more conserved regions between PATs and SATs) this did not improve the situation (results not shown). Therefore this might require an alternative matching algorithm, such as *curve fitting* which, although it would take longer per lineage, would be satisfactory as there are so few lineages.

6.0.2 *Scope and Limitations*

For the CB I believe I have a database of at least 1 000 unique morphological single-cell types (including the mirrored cells as separate types), out of the 3 000 cells that are present in the CB. This is based on the results of section 5.2 and noting that some of the cell types in the CB region will originate in the VNC and some cell types have multiple examples that are very difficult to distinguish from one another in the LM (they are usually of the same lineage). I would estimate, from having viewed all cells included in my database and JT's original FileMaker database, the proportion of cell types included is much higher, closer to 2 000 (the higher estimate from section 5.2). The VNC is much less well covered, since I have at least 2000 cell types out of approximately 7000. Since the cell types are repeated, in many instances I do not have an example of each type in a specific segment, but will have the type for other segments. Hence it is possible to infer between segments (note that the GAL4 lines often include the same cell in each segment).

One method to evaluate the coverage more concretely would be to match each cell in the database to its entry in JT's database. This way the names of that database could also be applied to the cells. Unfortunately this requires the input of JT's time to be confirmed, which draws on a very limited resource. Nevertheless, the clustering from section 5.2 should greatly assist in identifying similar sets of cells that can be tentatively matched to a small number of listed cell types in JT's database. If this is completed, the performance of the pipeline could then also be checked for any cell type dependence. It would also be of great interest to see how many traced EM cells are not matched to a cell type in my database, although for the CB, it appears very few (MW, personal communication). Identified missing cells will help in the targeted generation of further GAL4 lines.

As the pipeline for image integration is mostly fully automated, it will be straightforward to add more cells to the database and improve coverage. It is important to note that as all imaged brains here followed the same preparation guidelines, it is not given that my system will work as is if others are adopted, for instance at different laboratories (although there are no clear impediments I can foresee). Regarding the registration, in practice this may be achieved by registering the images from a new dataset to a template for that dataset and then creating a single, high-quality *bridging registration* between the two templates Manton et al. (2014). In this way the recently registered gene expression patterns of lower resolution whole-CNS images Muenzing et al. (2018) might be quickly integrated.

Considering that the VNC is segmentally repeated and I have examples of cell types in some but not others, it might be of interest to create a typical segment space into which all the segments can be registered. This could be achieved through landmark registration between template segments, for example, and the registered stacks further registered to the new space.

A further area of limitation is the difficulty in viewing the cells that have parts that innervate both the CB and VNC. Unfortunately, records were not kept of where these cells occur across stacks, however they will be from the same GAL4 lines and, due to overlap in the images it is feasible to manually identify matching stacks. When finding a CB and VNC stack from the same specimen to create the whole CNS template candidates, I found that this could be achieved by observing the original stacks and matching deviations in the reference channel tract and orientations of remaining external tubular appendages. This could be aided by application of a tiling algorithm that found the best candidate matches e.g. Chalfoun et al. (2017).

6.0.3 *Theoretical Implications*

The difficulty in determining the number of cell types illustrates a fundamental problem with what is meant by *cell type*. The definition in the context of this work is not precise, relying on what an individual can determine from the LM image stacks as morphologically distinguishable cells. Although all cells are unique at a level of precise morphology and will exhibit some variation due to environmental stochasticity, I have classed cells as the same type if they have the same gross morphologies. A more precise definition of two cells of the same type (from different or the same larval brain specimens) are those that are near identical genetically and born at the same point in the lineage phylogeny. Each larva has the same number of cells of each type, with maybe 1 or 2 variations per individual (JT, personal communication). Cells from the same type as defined this way will have the same gross morphology and function.

Through years of experience observing many thousands of cells one can note the consistencies that appear in the data and determine that cells of a certain morphological consistency are a certain type, but this classification may include more than one cell from each individual (and perhaps more than one genetic type as defined above and hence more than one functional type). JT has built the most comprehensive dataset of cells classed in types by morphological consistency and provided it as a great resource for *Drosophila* neurobiologists. It is this database and usefulness that I have attempted to emulate in my work. In most cases, the cell types defined by JT correspond to individual cells with functional properties that can be determined and linked to individual cells in the EM stack, such as in Ohyama et al. (2015).

I decided, however, not to invest significant time into manually matching the single cell observations to the database of JT (although I did create an ImageJ Plugin to enable JT to do this). My reasoning was based on the use of the EM stack to identify matching LM cells and my expectation that cell names will, as the literature progresses, take those that are used in the EM tracing. This is due to those who trace them naming them in that database and then using these names (which are often more memorable) in publications regarding the cell's function e.g. goro neurons (Japenese - *rolling*) Ohyama et al. (2015).

The EM image stack will eventually have traced all neurons and therefore it should be possible, as each LM cell is matched to one or more EM cells, to determine how many actual (EM) cells each type possesses and define a type in the LM stack as including all cells with a particular EM cell as their best (or above threshold) EM match. I have not spent further time investigating the distinction of cell types as the ability to search the database for similar cells allows the biologist to make their own choice on what is the same cell. In addition, it would be best to wait until the full

EM stack is available so that the above criteria can be applied. Further EM specimens should also be analysed to investigate levels of variation within cell types and verify or otherwise the numbers per type. One could imagine, eventually, a very robust hierarchy of cell class definitions, similar to those of the lineages, but also relating to function. Average cells could be produced from all those within, perhaps, a given hierarchical cluster.

One should similarly consider the neuropil domains hierarchically, which I do here through the ontology model. It was clear from the voxel clustering that there were some subdomains and that there could be multiple ways of defining these as they may overlap. These may relate to different functional units and hierarchical processing levels, for instance. It is therefore important that in future ontologies are kept flexible so as to allow for multiple schemes of parcellation of smaller, overlapping domains. That said, I would strongly encourage researchers to refer to the anatomical definitions included in this thesis for gross neuroanatomy. While these definitions may well - and should - change based on evidence, such as from computational studies similar to section 5.4, there can now be a clear record on how they have changed.

6.0.4 *Future Directions*

Perhaps the most obvious immediate extension to this project would be to investigate the automatically segmented synaptic domains for the VNC since there has not been a great deal of research into defining these. I would also consider investigating performing hierarchical clustering of cells by calculating cluster membership instead by similarity to its average common *morphology*, produced in a method similar to that in section 4.8 (but much faster). This would give clear morphological definitions to each cell cluster.

It is worth noting that the VFB online atlas model and the techniques employed and developed here could also be applied to other organisms used for developmental studies with extensive genetic tools, such as zebrafish.

More extensively, an area in *Drosophila* larva neuroscience in which computational resources could be greatly applied, given the morphological cell data I have and even more so the high resolution EM, is the study of what controls how neurons grow and arborise. The eventual outcome, i.e. the actual neuroanatomy that I have, is a validation tool for mathematical models that consider the geometries of cell proliferation and gradients of signalling molecules influencing the neural *growth cones* Zubler and Douglas (2009) Van Ooyen (2011).

There is an information explosion, much like the population explosion, how on earth are you going to scan all that information? Of course you can get computers to help you...
...but by Parkinson's Law the sooner you become more efficient in doing this, the more this will stimulate cataloguing so that you will have to have more efficient computers still to assimilate all the information. ***You may get ahead, but only for a short time.***

- Alan Watts

A APPENDIX

A.1 LARVAL DROSOPHILA NEUROPIL DOMAIN DEFINITIONS

The following definitions are described in the BP104 synaptic density staining of an *in vitro* Drosophila larval central nervous system. They are based on experience and analysis of neural stainings reported in the literature. Note that the neuraxis has been flattened to the y-axis in the template so that the CB anterior direction matches the ventral VNC direction and the CB posterior direction is that of the dorsal of the VNC. In the following text, boundaries with previously described neuropil domains can generally be referred back to by the reader when these neighbouring domains are mentioned in italics without repetition of description. Boundaries with cortex are generally not mentioned. Lineages are shown in brackets for the parts of fascicles that they are a part of when these fascicles act as demarcation. Volumes are averaged over both sides, except for the Fan-shaped Body primordium (prFB). References to compartments in italics refers to those defined in older work by VH and collaborators as described in Nassif et al. (2003), Pereanu et al. (2010) and elsewhere. Square bracket references refer to the figures where the described observations can be seen.

NB: The boundaries, particularly those along tracts, should be viewed as guiding definitions rather than precise, absolute definitions due to their often indistinct nature.

Total CNS volume is 1 390 010 $(\mu m)^3$ with CB domain volume 525 850$(\mu m)^3$ and combined proto-deuterocerebrum volume is 439 510$(\mu m)^3$.

A.1.1 *Mushroom Bodies*

Each side of the symmetrical Central Brain of the *Drosophila* larva is orientated around the Mushroom Body (MB) which is easily identifiable in BP104 staining with high intensity tracts and surrounding neuropil. They are split into domains as follows: Three, approximately perpendicular, axes can be drawn, with the origin at the Spur, which is the domain where the following three other domains meet [A.2a]: Along the ventro-dorsal axis, the Vertical Lobe (VL) projects dorsally of

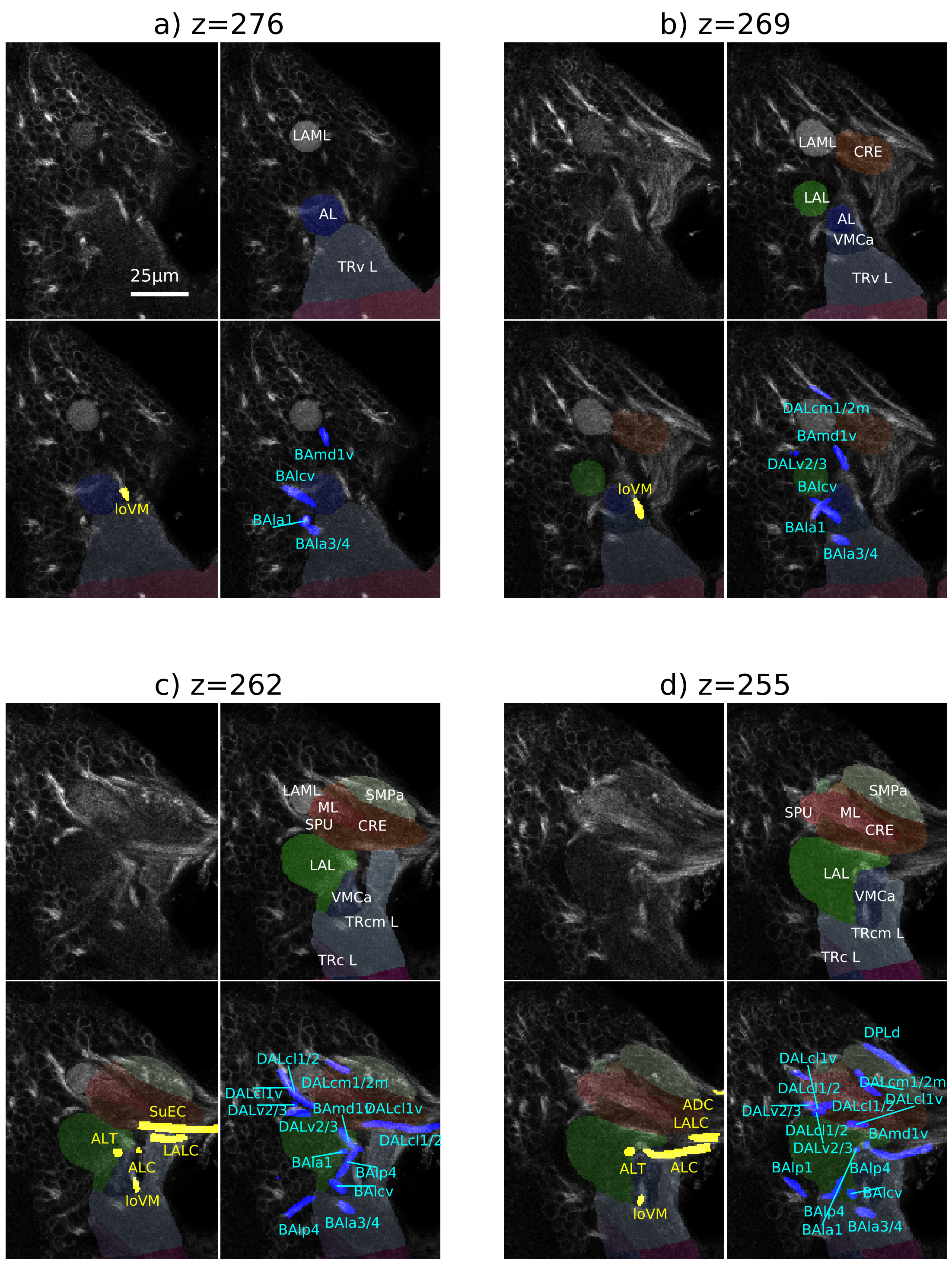


Figure A.1: Central Brain Domain Definitions A
Slices at various z-positions for a template CB hemisphere. Top left: raw signal, top right: shaded neuropil domains overlayed with labels. Bottom left: Yellow overlays of labelled fascicles and bottom right: Blue and cyan overlays of lineage tracts used in demarcation of the neuropil domains.

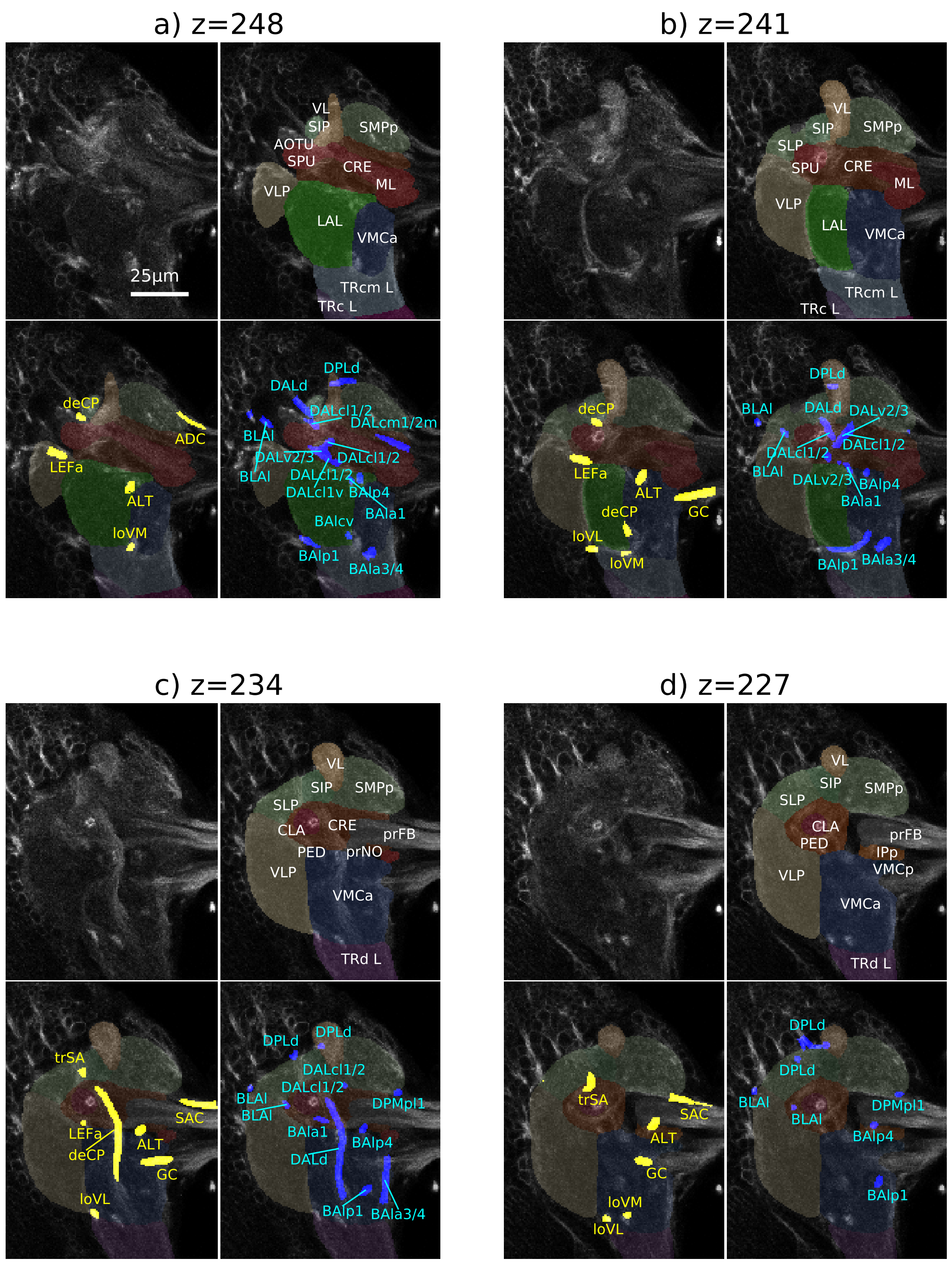


Figure A.2: Central Brain Domain Definitions B
Slices at various z-positions for a template CB hemisphere. Top left: raw signal, top right: shaded neuropil domains overlayed with labels. Bottom left: Yellow overlays of labelled fascicles and bottom right: Blue and cyan overlays of lineage tracts used in demarcation of the neuropil domains.

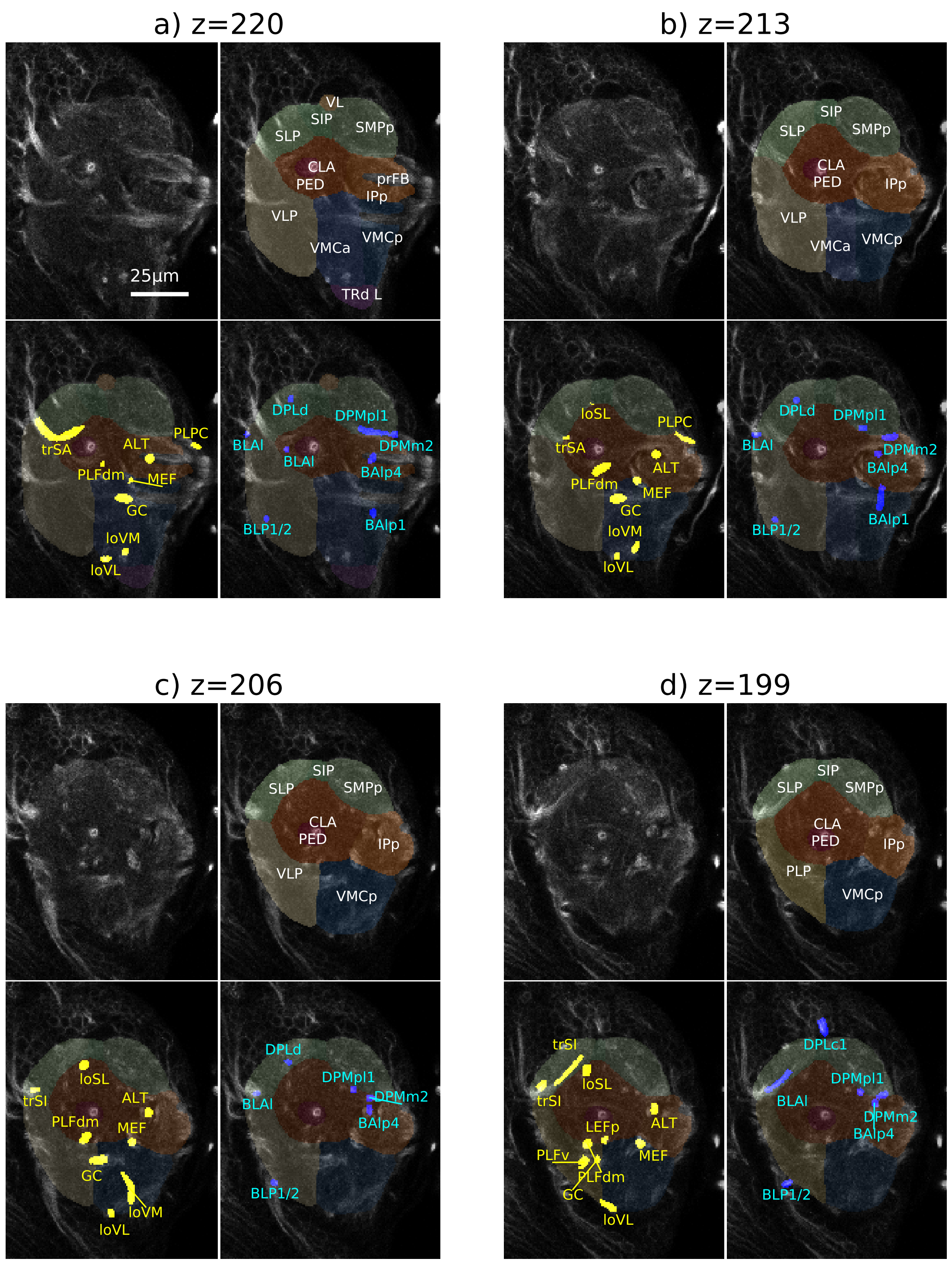


Figure A.3: Central Brain Domain Definitions C
Slices at various z-positions for a template CB hemisphere. Top left: raw signal, top right: shaded neuropil domains overlayed with labels. Bottom left: Yellow overlays of labelled fascicles and bottom right: Blue and cyan overlays of lineage tracts used in demarcation of the neuropil domains.

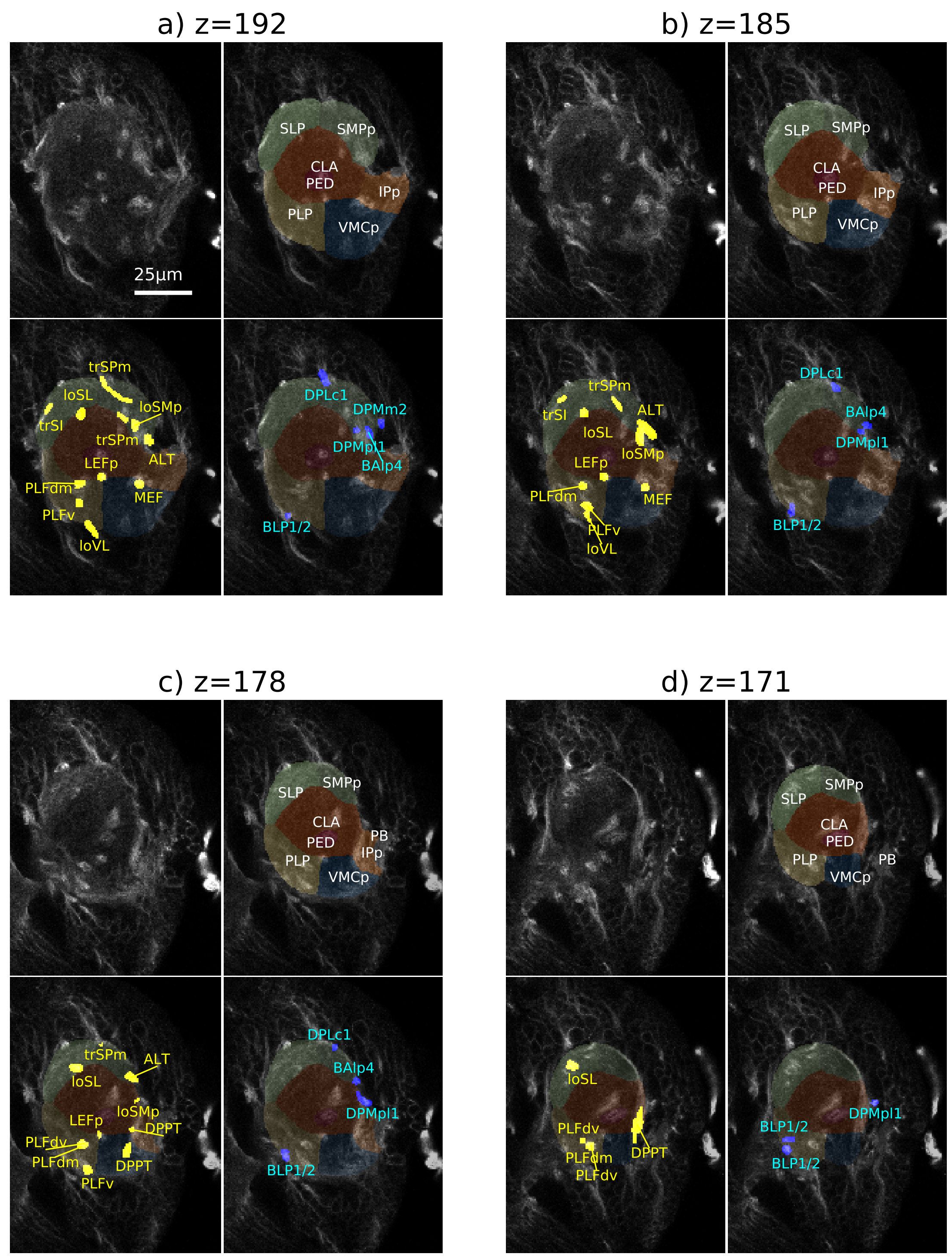


Figure A.4: Central Brain Domain Definitions D
Slices at various z-positions for a template CB hemisphere. Top left: raw signal, top right: shaded neuropil domains overlayed with labels. Bottom left: Yellow overlays of labelled fascicles and bottom right: Blue and cyan overlays of lineage tracts used in demarcation of the neuropil domains.

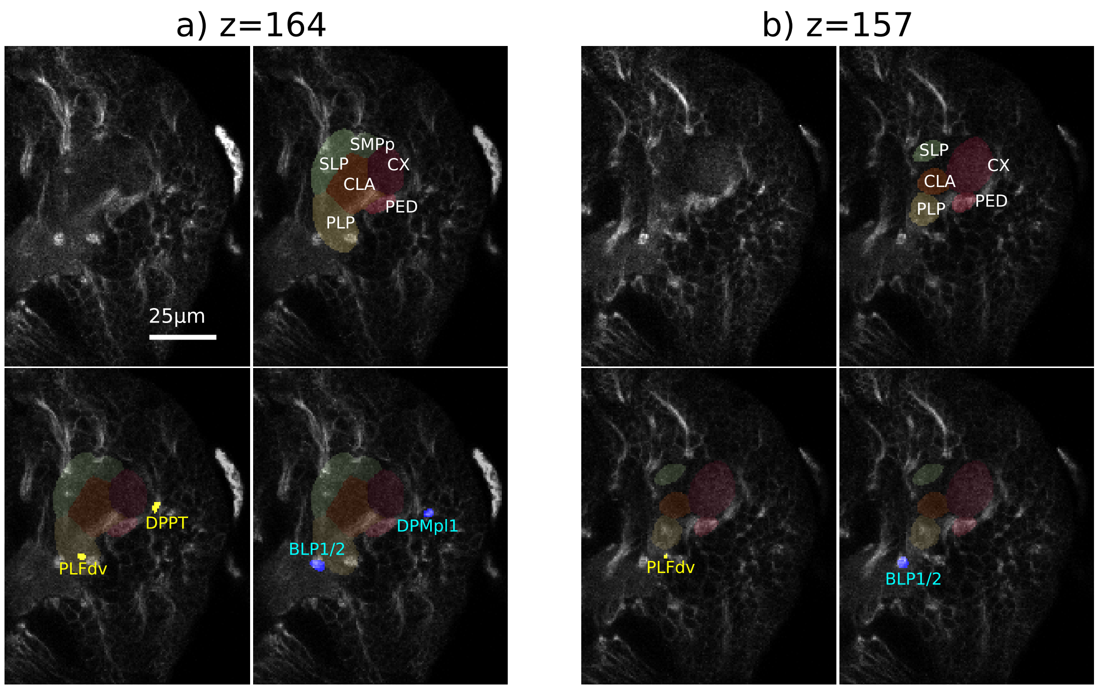


Figure A.5: Central Brain Domain Definitions E
Slices at various z-positions for a template CB hemisphere. Top left: raw signal, top right: shaded neuropil domains overlayed with labels. Bottom left: Yellow overlays of labelled fascicles and bottom right: Blue and cyan overlays of lineage tracts used in demarcation of the neuropil domains.

the Spur [A.2b], the Medial Lobe (ML) projects medially [A.2a] and the Peduncle posteriorly [A.2c-A.5b]. For most of the coronal CB slices, the Peduncle is the inner-most neuropil surrounding a very bright tract (which is also classed as part of the Peduncle domain). The Peduncle itself is surrounded by the Clamp domain (see below). At the posterior end of the peduncle, the spherical Calyx (CX) protrudes from the rest of the neuropil, surrounded on all but part of its anterior side by cortex [A.5b].

A small, spherical sub-domain of the MB ML is also annotated - as the Lower Toe (LT) (formerly Medial Appendix of the Medial Lobe (MAML)) [A.2b]. It is at the ventral and medial end of the ML, posterior of the LALC (CM1) commissure and anterior of a dull low intensity gap between it and the posterior Inferior Protocerebrum (IPp). Intensity differences separate it from the Crepine laterally and the anterior Ventro Medial Cerebrum (VMCa) ventrally. Further anatomical subdivision of the MB lobes has recently been published (see Saumweber et al. (2018)), although as these are not immediately visible in the template reference here they have not been labelled as yet.

Finally, a small, spherical domain is appended to the above described MB (of which it is also part). The Lateral Appendix of the Medial Lobe (LAML) borders the anterior of the Spur, proud from the rest of the neuropil and surrounded by cortex otherwise [A.1a-A.1c].

Average volumes over both lobes are:

Domain	CX	PED	SPU	ML	VL	MAML	MB	LT
Volume / $(\mu m)^3$	8 180	3 950	1 520	5 240	3 150	2 120	24 160	1 190

A.1.2 *Inferior Protocerebral (Centro Central Brain) Domains*

A.1.2.1 *CLA: Larval Clamp (Inferior Protocerebrum)*

Corresponds to the CPI and CPL compartments combined, although sometimes split into these domains as the medial Clamp (CLAm), also called the IPm, and lateral Clamp (CLAl), also called the IPl, along a sagittal plane through the Peduncle Pereanu et al. (2010). 28 060$(\mu m)^3$.

A cylindrically shaped, central domain surrounding the Peduncle, this reaches as far anterior as the coronal level of the trSA (DPLal1-3) [A.2c] and stretches right back to the posterior cortex where it borders the Calyx [A.5b]. The boundary between the CLAm and the dorsally adjacent posterior Superior Medial Protocerebrum (SMPp) is demarcated by the trSPm fascicle (DPLc lineages) [A.4b] and the loSMp fascicle (DPMpl1/2,DPMpm2) [A.4a]. Medially, the CLAm is bounded from the IPp by a sagittal plane that intersects with the antero-posterior fascicles MEF(CM1,3,4, ventrally, Hartenstein et al. (2015)) and the loSMp (DPMpl1/2, DPMpm2, dorsally) [A.4a, A.4b]. A virtual axial plane intersecting the MEF (CM1,3,4, medially) and LEFp (CP1v, laterally) constitutes the ventral boundary of the CLAm with the posterior Ventro Medial Cerebrum (VMCp) and VMCa Hartenstein et al. (2015) [A.3d-A.4b]. The boundary with the Lateral Accessory Lobe (LAL) continues along this plane further anteriorly.

Considering the CLAl, ventro-laterally this domain is separated from the ventrally adjacent Ventro Lateral Protocerebrum (VLP) by the curved virtual plane defined by the LEFp (CP1v) and PLFdm (CP2/3v) to LEFa (DALv1) [A.2b-A.4c]. The boundary with the Posterior Lateral Protocerebrum (PLP) is a continuation posteriorly of this CLAl-VLP border Hartenstein et al. (2015). Dorso-laterally of the Clamp, the Superior Lateral Protocerebrum (SLP) has a clearly visible higher level intensity in BP104. The following fascicles are inside the SLP along the boundary with the CLAl: trSA(DPLal1-3 Lovick et al. (2013)) [A.3a], trSI (BLAd/BLAl/BLD1-4) [A.3d] and loSL (DPLl2/3 Hartenstein et al. (2015)) [A.3b-A.4c]. Finally, the dorso-laterally located Superior Intermediate Protocerebrum (SIP) can also be distinguished from the CLAl by its higher level of synaptic density (BP104 intensity) [A.2d- A.3d]. Anteriorly, at the coronal level of the posterior surface of the ML and the deCP (DALd,DALcm) fascicle [A.2c], the CLAm borders the Crepine domain, that surrounds the medial lobe anteriorly. The CLAl extends more anterior laterally, reaching the coronal level of the trSA (DPLal1-3) [A.2c], before ending at the MB Spur [A.2b].

A.1.2.2 *CRE: Larval Crepine (anterior Inferior Protocerebrum)*

Corresponds to the CA compartment. 12 380$(\mu m)^3$. This domain surrounds the ML, reaching as posterior as the prFB [A.2c], from which it is clearly separated by its lower level of intensity level in BP104. Posterior of this coronal level the IPp begins. The dorsally located anterior Superior Medial Protocerebrum (SMPa) and SMPp also display a higher BP104 intensity than the Crepine and are separated along the tracts of the DALcm1m [A.1c-A.2a] and DPMpl1 [A.3a] respectively. Ventrally, the boundary between Crepine and LAL/VMCa extends along the DALv2/3 Hartenstein et al. (2015) [A.1c-A.2b], DALcl1v [A.1c] and BAmd1v [A.1c, A.1d] lineage tracts and the commissural fascicle SuEC (DALcl1v Hartenstein et al. (2015)) [A.1c]. The LALC (CM1) and ALC (BAmd1/2) commissure Hartenstein et al. (2015), however, separate the Crepine from the tritocerebrum [A.1c, A.1d], which is also ventral. Moving anteriorly, the Crepine expands laterally to almost completely surround the Spur of the MB [A.2b]. It thus continues the borders with the VLP and SIP given for the CLAl, before these domains come into contact with the Spur themselves at the coronal level of the Medial Lobe [A.2a]. The most anterior part of the Crepine flanks the anterior of the ML to the cortex [A.1b]. *The Clamp, Spur and LAML are all lateral and the LT medial.*

A.1.2.3 *IPp: posterior Inferior Protocerebrum*

Corresponds to the CPM compartment. 11 740$(\mu m)^3$. This domain fills the medial gap of neuropil between the Clamp and the space between the two CB spheres, from the coronal level of the posterior prFB [A.2d] to the entry of the Protocerebral Bridge (prPB) [A.4c] (clearly demarcated by its brighter density intensity in BP104). Dorsally, the IPp is separated from the SMPp by the commissural axon bundles formed by the SAC [A.2d], PLPC (DPLp1) [A.3b] and commissural part of DPMm2 [A.3b]. The longitudinal fascicle loSMp (DPMpl2) [A.4c] marks the junction of the CLAm, IPp and SMPp. Ventrally, the VMCp is split from the IPp by the DPPT (DPMl1) [A.4c] fascicle and a virtual axial plane drawn from the MEF medially to the medial neuropil edge [A.3a-A.4b]. *The Crepine is anterior, Clamp lateral.*

A.1.3 *Superior Protocerebral (Dorsal Central Brain) Domains*

A.1.3.1 *SLP: Superior Lateral Protocerebrum*

Corresponds to most of the CPLd compartment. 16 470$(\mu m)^3$. This domain extends over a large range antero-posteriorly, in the dorso-lateral part of the CB. Posteriorly and medially, the border between the SLP and SMPp is defined by the trSPm entry point (DPLc) [A.4c]. At a more anterior level, the small SIP domain is wedged between the SLP and SMPp. The SIP is notable by its higher BP104 intensity and the SLP-SIP boundary can be followed posterior from the SLP-Anterior Optic TUbercle primordium (prAOTU) boundary (at the DALd lineage entry point) [A.2a] posteriorly, past the distal segment (internal, lateral) curve of the trSA (DPLal1-3) [A.2c, A.2d] and continuing further posteriorly until the entry point of DPLc [A.3d, A.4a].

At a more posterior level, the SLP forms a slim ventral border with the PLP, which is defined by the entry point of the trSI (BLD1-4) fascicle [A.3c, A.3d]; in addition, a higher intensity of BP104 label demarcates the posterior SLP from the PLP. Further anterior, the ventral border of the SLP is with the VLP and can be defined by the lineage tract of BLAl Hartenstein et al. (2015) [A.3b-A.3d] and the entry point of the trSA (DPLal1-3 Hartenstein et al. (2015)) [A.3a]. Anteriorly, the SLP ends at the MB Spur and the prAOTU (located anterior of the SLP-SIP boundary) [A.2a], both of which stand out by possessing a higher BP104 intensity. *Towards ventro-medially, the SLP borders the CLAl.*

A.1.3.2 *SMPa: anterior Superior Medial Protocerebrum*

Corresponds to the DA compartment. 2 940$(\mu m)^3$. This is the part of the dorso-medial part of the CB that is anterior-medial of the VL and split from the SMPp at the coronal level of the ADC (DAMd1) commissure [A.2a] and a glial septum Pereanu et al. (2010). *Ventrally, the SMPa borders the Crepine.*

A.1.3.3 *SMPp: posterior Superior Medial Protocerebrum*

Corresponds to the DP compartment. 21 830$(\mu m)^3$. This part of the dorso-medial portion of the CB extends posterior of the SMPa as far as the Calyx, postero-medially of the VL, SLP and SIP, which has a higher BP104 intensity than the SMPp. *Ventrally the*

SMPp is bordered (in antero-posterior order) by the Crepine, a small gap in BP104 intensity to the prFB and finally the IPp. The CLAm is ventro-medial.

A.1.3.4 *SIP: Superior Intermediate Protocerebrum*

Corresponds to the dorso-lateral part of the CPLd compartment. 3 460$(\mu m)^3$. The small SIP has the shape of a hemi-cylinder that flanks the VL laterally and posteriorly, where it wedges in between SLP and SMPp. *The SIP borders the CLAl and (briefly) Crepine domains and (very anteriorly and ventrally) the Spur.*

A.1.4 *Ventral Central Brain Domains*

A.1.4.1 *AL: Larval Antennal Lobe*

Corresponds to the BA compartment, deuterocerebrum Hartenstein et al. (2018). 4 060$(\mu m)^3$. On the antero-ventral side of the CB, this spherical domain sits anterior of the VMCa and LAL, the borders demarcated by the root of the Antennal Lobe Tract (ALT), formed by the BAla1 and BAlp4 lineages just as they enter the neuropil Hartenstein et al. (2015) [A.1c]. The loVM fascicle (BAmv1/2) passes along the medial boundary Hartenstein et al. (2015) [A.1a, A.1b] and the BAlcv lineage tract [A.1a] and a glial septum demarks the AL from the ventral Tritocerebrum.

A.1.4.2 *LAL: Lateral Accessory Lobe*

Corresponds to the BC compartment, protocerebrum. 13 080$(\mu m)^3$. This domain is located in the ventro-lateral anterior CB, posterior of the Antennal Lobe (AL) and mostly anterior of the VLP and VMCa. The posterior side of the LAL ends between the VLP and VMCa at the coronal level of the deCP (DALd) fascicle [A.2b]. The LAL is separated on its lateral side from the VLP by a ridge of cortex Hartenstein et al. (2015) as well as the loVL (BAlp2 Pereanu (2006)) [A.2b].

Medially, a virtual sagittal plane intersecting the deCP (DALd, posteriorly) and loVM (Bamv1/2, anteriorly) separates the LAL from the VMCa Pereanu (2006) [A.2b]. Ventrally, the boundary with the centro-medial tritocerebrum is formed along the BAlp1/4 lineage tracts Hartenstein et al. (2018) [A.1c, A.2b] and the approximately axial virtual plane between the ventral sections of the loVM (BAmv1/2, medial) and loVL (BAlp2, lateral) fascicles [A.2b]. A medial, finger-like protrusion from the main mass of the LAL occurs at the coronal level of the ALC and splits the VMCa (which contains the ALC) from the Crepine [A.1d]. *Dorsally, the LAL borders the Clamp and Crepine and the AL borders antero-medially.*

A.1.4.3 *PLP: Posterior Lateral Protocerebrum*

Corresponds to the posterior part of the BPL compartment, protocerebrum. 10 940$(\mu m)^3$. This is the ventro-postero-lateral domain of the CB. Anteriorly, the GC (BLAv1, BLD5) defines the boundary, in the coronal plane, with the VLP domain [A.3c]. Dorsally, there is a slim border with the SLP. Medially, a virtual sagittal plane intersecting the LEFp (CP1v, posteriorly) and the loVL (Balp2/3, anteriorly) separates the PLP from the VMCp [A.3d-A.4b]. Laterally, a glial septum and BLP1/2 separate the PLP from the lobula [A.4d, A.5a]; *the Clamp is dorso-medial and laterally, the SLP dorsal.*

A.1.4.4 *VLP: Ventro Lateral Protocerebrum*

Corresponds to the anterior part of the BPL compartment, protocerebrum. 26 340$(\mu m)^3$. This is the ventro-antero-lateral domain of the CB. *The anterior part of the VLP has a medial boundary with the LAL*. More posteriorly, the medial boundaries - with the VMCa and VMCp - are defined as is the PLP-VMCp boundary, i.e. by the virtual sagittal plane intersecting the LEFp (CP1v, posteriorly) and the loVL (Balp2/3, anteriorly). *The PLP is located posteriorly of the VLP, the SLP is dorsal and the Clamp is dorso-medial.*

A.1.4.5 *VMCa: anterior Ventro Medial Cerebrum*

Corresponds to the anterior part of the BPM compartment, deuterocerebrum Hartenstein et al. (2018). 21 830$(\mu m)^3$. This is the ventro-antero-medial domain of the CB. It abuts the VMCp at the coronal slice level of the Great Commissure, GC (BLAv1, BLD5 Pereanu et al. (2010)) [A.3a]. Ventrally, the boundary with the tritocerebrum (ventral, centro-medial and dorsal domains) is defined by a somewhat curved, more or less axial plane defined by the loVL (Balp2/3 Hartenstein et al. (2018)), loVM (Bamv1/2 Hartenstein et al. (2018)) [A.2b-A.3b] and, more anterior, by the BAla3 tract Kumar et al. (2009) [A.1d-A.2b]. Medially, an extension of the centro-medial tritocerebrum is separated from the VMCa by the loVM (BAmv1/2) fascicle and a glial septum Pereanu (2006) [A.1c]; more posteriorly, this boundary continues as a discontinuity in BP104 staining intensity. *Laterally, the VMCa borders the LAL; at a more posterior level, the VLP flanks the VMCa laterally. The Clamp and Crepine border dorso-laterally and dorsally respectively, with the LT of the ML dorso-medial. A dull BP104 intensity gap separates the VMCa from the prFB.*

A.1.4.6 *VMCp: posterior Ventro Medial Cerebrum*

Corresponds to the posterior part of the BPM compartment, deuterocerebrum Hartenstein et al. (2018). 17 610$(\mu m)^3$. This is the ventro-postero-medial domain of the CB. *It* *borders the IPp dorso-medially, the Clamp dorso-laterally and the PLP laterally. It is posterior of the VMCa.*

A.1.5 *Central Brain Protocerebral Primordia*

Primordia are domains formed by secondary lineage tracts and filopodia; devoid of synapses, denoted by the "pr" prefix.

A.1.5.1 *prAOTU: Anterior Optic TUbercle*

430$(\mu m)^3$.Encircled by the DALcl1/2 lineages [A.2a] at their entry points into the lateral neuropil, this small primordium is dorsolateral of the Spur of the MB, between the SLP and SIP domains. It is marked by a different intensity in BP104 from the SLP and separated from the SIP by the DALd lineage [A.2a, A.2b]. The prAOTU has its anterior boundary where the LAML begins [A.1d] and posterior boundary at the coronal level of the DPLd input [A.2b].

A.1.5.2 *prFB: Fan-shaped Body*

Previously included in the CPM compartment. Total volume 6 740$(\mu m)^3$. Formed from the DPMm, DPMpm and CM4 lineages Pereanu et al. (2010), this commissural primordium is at the centre of the CB, mostly clearly separable by intensity differences of BP104 [A.2c-A.3a]. It is posterior of the Crepine and anterior of the IPp, directly posterior of the ML. The Noduli are to the ventro-lateral side of the prFB.

A.1.5.3 *prNO: NOduli*

Previously included in the CPM compartment. 190$(\mu m)^3$. Also formed from the DPMm1, DPMpm1/2 and CM4 lineages Pereanu et al. (2010), this small primordium appears as a ventrally directed process of the lateral prFB [A.2c]. It is demarcated from the surrounding CLAm/Crepine by different intensity patterns in BP104.

A.1.5.4 *prPB: Protocerebral Bridge*

Previously included in the CPM compartment. 880$(\mu m)^3$. Also formed from the DPMm1 and DPMpm1/2 and CM4 lineages Pereanu et al. (2010), this small tubular-like primordium is shaped like a dorsally convex crescent, attached to the posterior surface of the IPp domain [A.3b-A.4d]. Turning anterior-medially, it connects to its contralateral counterpart right posteriorly of the PLPC (DPLp1) commissure [A.3b] and prFB.

A.1.6 *Ventral Nerve Cord (VNC) and Tritocerebrum*

The definitions follow those set out in Hartenstein et al. (2018). See figures A.6 and A.7.

A.1.6.1 *Antero-Posterior Segmentation*

The VNC is split into segments, or synaptic domain neuromeres, along the antero-posterior axis by a repeated, approximately coronal plane defined by features of each segment: Ventrally, this plane passes just posterior to the posterior-lateral and posterior-medial vertical bundles, formed by the entry portals of lineages 19/23 and 3/5/6/12, respectively; in the ventral midline, the plane intersects with the entry point of lineage zero, at the centre of the ventral side of the VNC. More dorsally it reaches just posterior of the posterior intermediate commissure. Unless noted below, this splits the VNC into (from anterior to posterior):

1. Tritocerebrum (part of the CB, not VNC)
2. Mandibula
3. Maxilla
4. Labium
5. Thoracic segments 1-3
6. Abdominal segments 1-9

The boundary between the Tritocerebrum and Mandibula is not well defined, since counterparts of the critical lineages (0, 3, or 12) have not been identified. Hence one uses a virtual coronal plane defined by a cleft in the neuropil surface (particularly noticeable in the dorsal VNC) as the boundary. The tritocerebral commissure extends along the posterior boundary of the Tritocerebrum.

The boundary between the Mandibula and Maxilla is defined by lineage 3 which forms a sole vertical bundle and this, approximately coronal, plane can be drawn up to just posterior of the single Mandibula intermediate commissure.

A.1.6.2 *Intrasegmental Subdivisions of the Neuropil*

Excluding the most posterior segment of the VNC (abdominal segment 9, A9), the ventral nerve cord can be subdivided along the dorso-ventral axis into three tiers, called dorsal, central, and ventral. A system of longitudinal and transverse fascicles (connectives and commissures, respectively) define the boundaries of these tiers, and

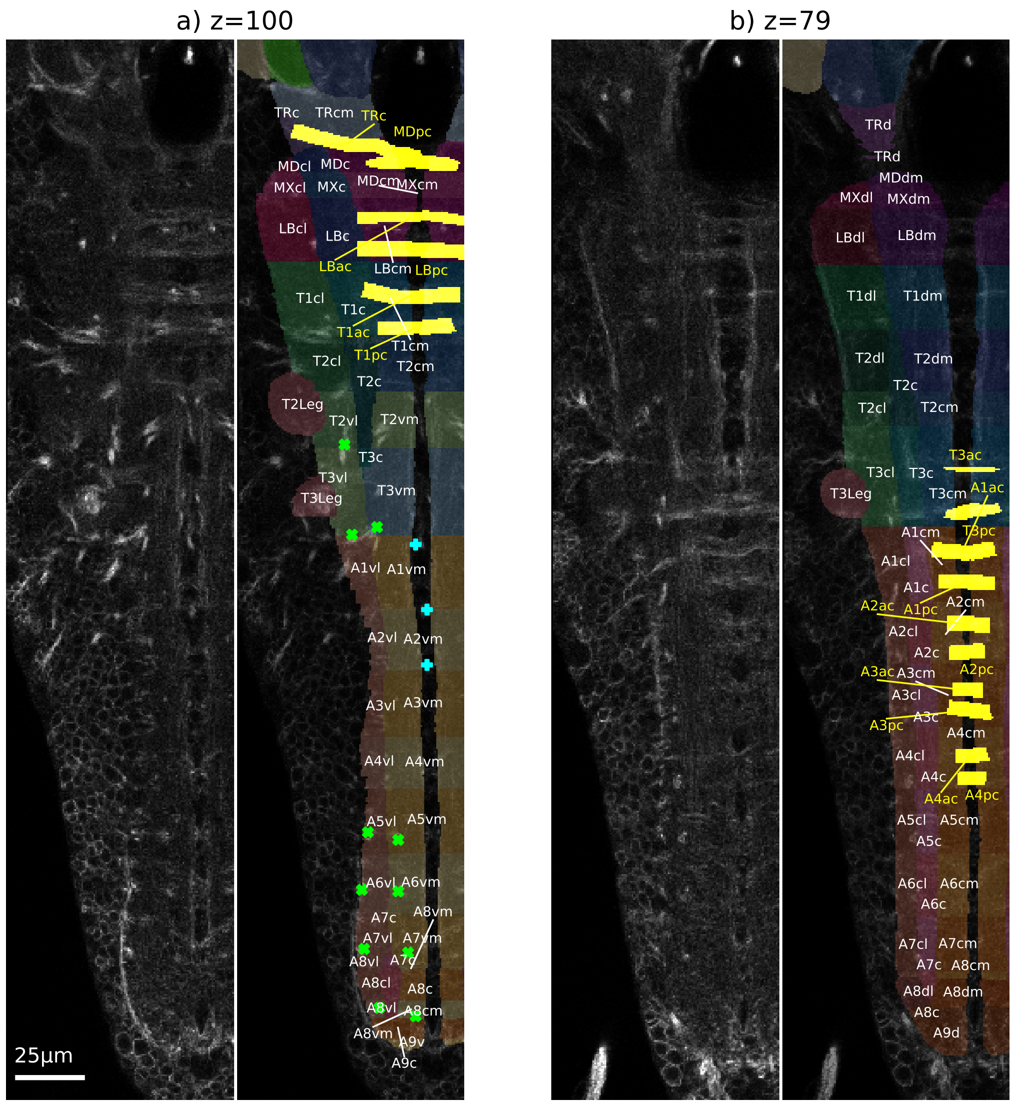


Figure A.6: Horizontal sections of Ventral Nerve Cord
For each subplot: Left: slice through background BP104 signal of template. Right: Overlays of neuropile domains and (in yellow) main fascicles. Cyan "+" markers are lineage 0 entry points and lime green "x" markers are the lateral 19/23 and medial 3/5/6/12 lineage tracts. Note that "TRc" notation refers to the Tritocerebrum commisural fascicle in yellow (the Tritocerebrum central (TRc) domain is labelled white).

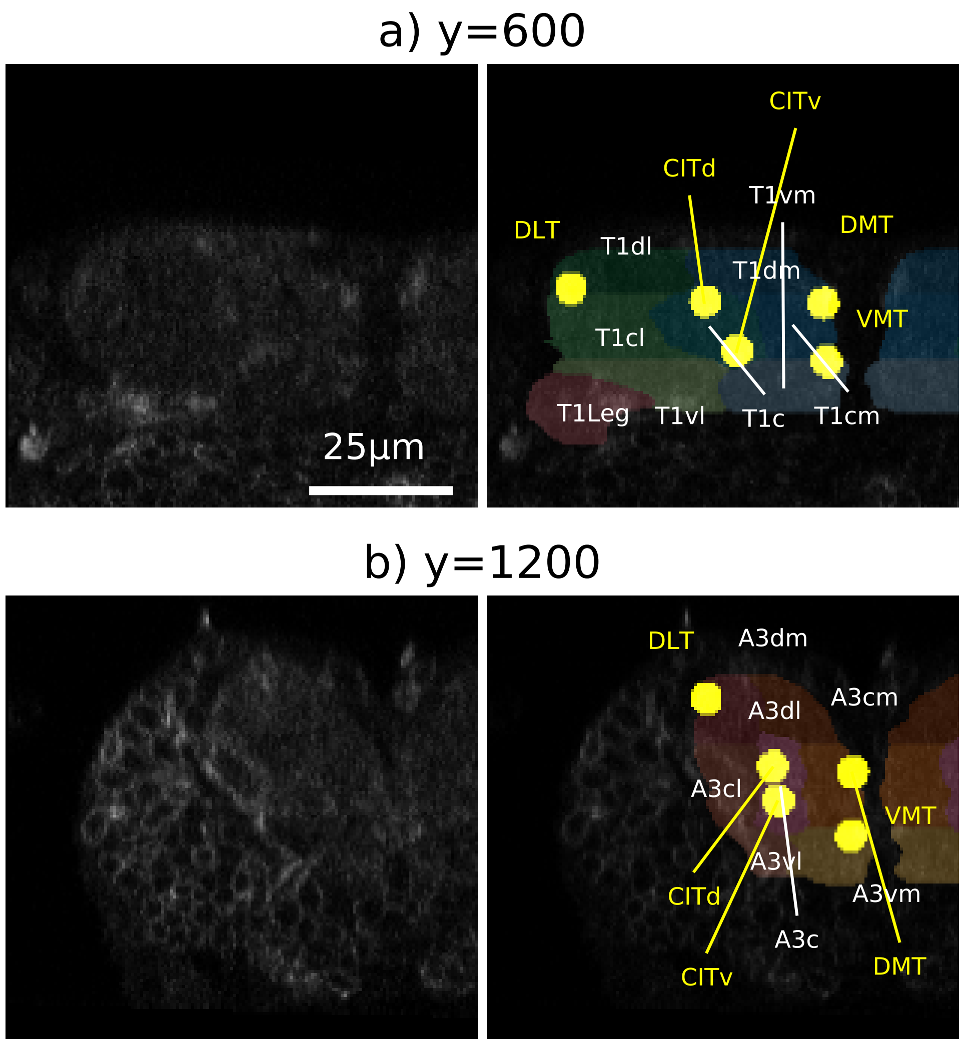


Figure A.7: Coronal cross sections of Ventral Nerve Cord
Left: slice through background BP104 signal of template. Right: Overlays of neuropile domains and (in yellow) main fascicles.

further delineate subdivisions along the medio-lateral axis:

The CITd and CITv fascicles surround - and pass through - the center of the central tier, referred to simply as the central domain. Medially of this domain, containing the intermediate commissures (aI, pI), is the centro-medial (cm) domain, laterally the centro-lateral (cl) domain. A virtual axial plane intersecting the CITv and the VMT separate the central from the ventral domains; an axial plane placed just dorsally of the intermediate commissures and CITd separates the central from the dorsal domains. Due to the nature of the Central (c) neuropil, it protrudes slightly from the central domain into the dorsal and ventral domains in the segments in which it occurs.

The ventral beginnings of the vertical bundles (antero-medial and posterior medial vertical bundle) divide the ventral neuropil into ventro-medial (vm) and ventro-lateral (vl) domains. In the dorsal neuropil, the dorso-lateral (dl) and dorso-medial (dm) domains are separated by a sagittal plane intersecting the CITd fascicle.

The A9 Segment is at present only split into ventral, central and dorsal domains as defined above. *Volume sizes averaged over both sides are 1 980*$(\mu m)^3$, *1 590*$(\mu m)^3$ *and 4 720*$(\mu m)^3$ *for A9v, A9c and A9d respectively.*

The Tritocerebrum does not contain the vertical bundles required to split the ventral tier medio-laterally and the CITd/v fascicles gradually turn laterally in the Tritocerebrum, heading towards the rest of the CB domains. They reach the lateral surface of the neuropil denying existence to either a dl or cl domain in the Tritocerebrum. The central domain is pyramidal shaped and does not meet the CB, following the lateral curve of the CITd/v fascicles. The remaining four domains - dm, central, cm and vm - fill the neuropil between the medulla and CB. dm is renamed simply Tritocerebrum dorsal (TRd) and vm renamed Tritocerebrum ventral (TRv). As the intermediate commissures are ill defined, so is the centro-dorsal boundary. The centro-ventral border continues that of the VNC described above. The cm Tritocerebrum extends alongside the medial part of the anterior CB. *This protruding part of the cm corresponds to the former BCv compartment. Volume sizes averaged over both sides are 25 970*$(\mu m)^3$, *2 100*$(\mu m)^3$, *9 820*$(\mu m)^3$ *and 5 280*$(\mu m)^3$ *for TRv, TRc, Tritocerebrum centromedial (TRcm) and TRd respectively, (total 43 170*$(\mu m)^3$*).*

VNC Synaptic Neuropil Domain Sizes (averaged over both left and right, $(\mu m)^3$)

Segment	vl	vm	cl	c	cm	dl	dm	Total
MD	6 560	7 190	920	1 970	3 580	290	2 270	2 279
MX	7 910	6 160	2 400	2 190	2 640	1 380	2 240	2 491
LB	8 300	5 870	6 980	5 190	5 080	4 500	4 900	4 083
T1	7 440	7 790	6 910	4 910	4 830	4 680	4 880	4 144
T2	5 340	6 170	7 750	5 710	6 080	5 740	6 350	4 314
T3	3 310	4 150	5 770	4 850	5 530	5 920	6 620	3 615
A1	2 050	2 940	4 520	3 900	4 210	6 550	6 250	3 042
A2	1 600	2 670	3 690	3 700	3 690	6 300	6 340	2 799
A3	1 800	3 270	3 950	3 670	4 020	4 530	4 900	2 615
A4	2 090	3 400	2 960	3 400	3 110	4 470	4 770	2 422
A5	3 250	4 560	2 570	3 140	2 820	4 550	4 580	2 546
A6	2 740	4 240	2 730	3 130	3 190	5 560	5 280	2 687
A7	2 050	2 630	2 420	2 430	2 930	4 560	4 290	2 130
A8	2 610	2 390	1 220	1 300	1 290	2 990	3 340	1 514
Total	57 040	63 420	54 800	49 510	53 000	62 030	67 020	406 820

Total VNC Volume, including both sides, leg domains and the A9 segment is 864 200$(\mu m)^3$.

A.1.6.3 *Thoracic Leg Domains*

Each thoracic segment contains a protruding, spherical domain on its dorso-lateral side. These leg neuropil are embedded into the dorso-lateral domains of T1-3 and also the centro-lateral domains of T2-3. *Volume sizes averaged over both sides are 6 060*$(\mu m)^3$*, 5 660*$(\mu m)^3$ *and 5 240*$(\mu m)^3$ *for T1,T2 and T3 respectively.*

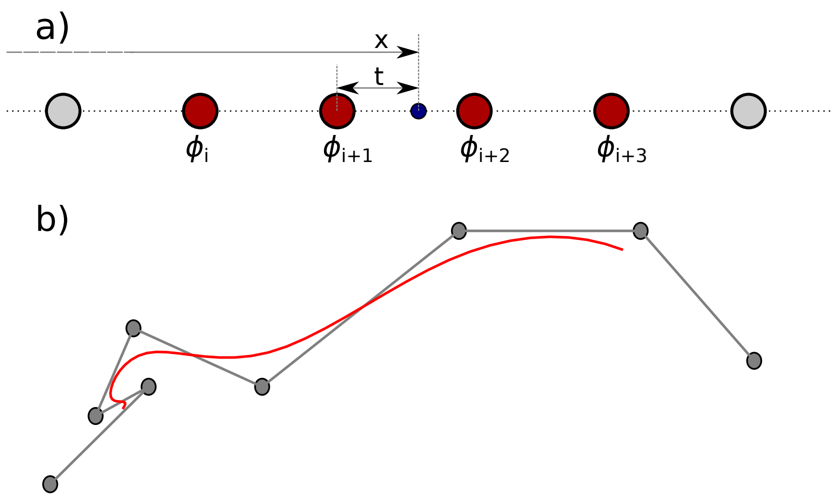


Figure A.8: B-Spline Equations Illustrations

a) B-Spline knots used (red) and not used (grey) in third order B-Spline calculation at a point (blue dot) requiring interpolation between two knots. This also illustrates the relationship of x, the parametric value for the entire curve to t, the *proportional* distance between the two knots contributing ϕ_{i+1} and $\phi + i + 2$. (This diagram assumes a unit separation between knots) b) A plot of the third order B-Spline (red line) drawn using the positions of the knots (grey points) ordered along the grey line as the vectors to interpolate between.

A.2 B-SPLINES

B-Spline (or *Basis* Splines) are a method of interpolation of a vector at a point in an ordered list of vector data points. This is achieved by a weighted sum of the vectors (here denoted by $\boldsymbol{\phi}$) at its surrounding, known data points. These known data points are referred to as *knots* and the weights depend on the proportion between the two surrounding knots the data point is (designated here as t, see figure A.8a). Specifically, the weights for a single nth-order B-Spline is a polynomial in t of order n, with $n + 1$ surrounding points contributing to a point within the spline. If the vector is the position of points, this allows a line to be drawn using the knot points as guides (see figure A.8b). Applied to free form deformation, the knots are the grid points in 2D or 3D space ordered by their position in each dimension and the vectors are the calculated deformation shifts *not their position*. Here the interpolated shift, $\mathbf{T}$ at a point (x, y, z) is calculated by summing weighted contributions from the surrounding grid / cube of grid points (knots). The weightings (B-Splines, B) in each dimension are simply multiplied together as in equation 3.1, reiterated below:

$$\mathbf{T}(x, y, z) = \sum_{l,m,n=0}^{3} B_l(u) B_m(v) B_n(w) \boldsymbol{\phi}_{i+l,j+m,k+n} \tag{A.1}$$

Here the u,v and w represent t in the three different directed dimensions that are multiplied together. $\boldsymbol{\phi}_{i,j,k}$ is the vector at the lowest indexed contributing control point involved in the calculation when the point of interpolation is in the cuboid grid spacing of which the line between knots $(i+1, j+1, k+1)$ and $(i+2, j+2, k+2)$ is a diagonal.

For third-order B-Splines in one dimension the equation for the shift is as follows:

$$\begin{aligned} T(t) &= B_{0,3}\phi_0 + B_{1,3}\phi_1 + B_{2,3}\phi_2 + B_{3,3}\phi_3 = \\ &(a_0 t^3 + b_0 t^2 + c_0 t + d_0)\phi_0 + (a_1 t^3 + b_1 t^2 + c_1 t + d_1)\phi_1 \\ &\quad + (a_2 t^3 + b_2 t^2 + c_2 t + d_2)\phi_2 + (a_3 t^3 + b_3 t^2 + c_3 t + d_3)\phi_3 \end{aligned} \tag{A.2}$$

where $B_{i,n}$ refers to the nth order B-Spline component for knot i (again, see figure A.8a).

B-Splines enforce a few sensible constraints that can be used to determine the coefficients of the polynomials:

- The contributions from all knots must be smooth throughout their contribution, enforced by the requirement for continuity in the weights and their first $n-1$ derivatives as one changes which knots one places t between (and which knots contribute).
- For all values of t the total weighting of the contributions from all knots is 1.
- The boundary condition that the contribution weight and weight derivatives at a knot must be zero when the limit of influence on the curve of that knot is met (i.e. when $t = 0$ at knot contribution ϕ_3 and $t = 1$ at ϕ_0).
- The contribution weights must be symmetrical about the contributing knots, so that replacing t with $(1-t)$ in the contributing equation on one side (e.g. ϕ_0) gives the equation of the other side (ϕ_3 in this example).

The equations for B-Spline weighting components for a knot contributing ϕ_i are defined recursively on a parameter x that is ramped from 0 to $p - n$ where p is the number of knots in the spline:

$$B_{i,0}(x) := \begin{cases} 1 & \text{if} \quad t_i \leqslant x < t_{i+1} \\ 0 & \text{otherwise} \end{cases} \tag{A.3}$$

$$B_{i,n}(x) := \frac{x - t_i}{t_{i+n} - t_i} B_{i,n-1}(x) + \frac{t_{i+n+1} - x}{t_{i+n+1} - t_{i+1}} B_{i+1,n-1}(x). \tag{A.4}$$

Thus all the various weightings, $B_{i,n}$, from this specific point for any point between any two knots can be calculated from the above equations. It will be instructive to follow through the derivation of the weightings from all the surrounding knots for a position t between two *specific* knots. For third order B-Splines:

$$\begin{aligned} B_{i,3}(x) &= \frac{x - t_i}{t_{i+3} - t_i} B_{i,2}(x) + \frac{t_{i+3+1} - x}{t_{i+3+1} - t_{i+1}} B_{i+1,2}(x) && \text{(A.5)} \\ &= \frac{x - t_i}{t_{i+3} - t_i} \left[\frac{x - t_i}{t_{i+2} - t_i} B_{i,1}(x) + \frac{t_{i+2+1} - x}{t_{i+2+1} - t_{i+1}} B_{i+1,1}(x) \right] \\ &+ \frac{t_{i+4} - x}{t_{i+4} - t_{i+1}} \left[\frac{x - t_{i+1}}{t_{i+1+2} - t_{i+1}} B_{i+1,1}(x) + \frac{t_{i+1+2+1} - x}{t_{i+1+2+1} - t_{i+1+1}} B_{i+2,1}(x) \right] && \text{(A.6)} \end{aligned}$$

This can be further written as:

$$\begin{aligned} B_{i,3}(x) = \frac{x - t_i}{t_{i+3} - t_i} &\left[\frac{x - t_i}{t_{i+2} - t_i} \left(\frac{x - t_i}{t_{i+1} - t_i} B_{i,0}(x) + \frac{t_{i+1+1} - x}{t_{i+1+1} - t_{i+1}} B_{i+1,0}(x) \right) \right. \\ &\left. + \frac{t_{i+3} - x}{t_{i+3} - t_{i+1}} \left(\frac{x - t_{i+1}}{t_{i+1+1} - t_{i+1}} B_{i+1,0}(x) + \frac{t_{i+1+1+1} - x}{t_{i+1+1+1} - t_{i+1+1}} B_{i+2,0}(x) \right) \right] \\ + \frac{t_{i+4} - x}{t_{i+4} - t_{i+1}} &\left[\frac{x - t_{i+1}}{t_{i+3} - t_{i+1}} \left(\frac{x - t_{i+1}}{t_{i+1+1} - t_{i+1}} B_{i+1,0}(x) + \frac{t_{i+1+1+1} - x}{t_{i+1+1+1} - t_{i+1+1}} B_{i+2,0}(x) \right) \right. \\ &\left. + \frac{t_{i+4} - x}{t_{i+4} - t_{i+2}} \left(\frac{x - t_{i+2}}{t_{i+2+1} - t_{i+2}} B_{i+2,0}(x) + \frac{t_{i+2+1+1} - x}{t_{i+2+1+1} - t_{i+2+1}} B_{i+3,0}(x) \right) \right] \end{aligned} \tag{A.7}$$

This formulation allows one to note a few properties of the equations that give B-Splines the properties required. Working from the inner-most components, $B_{-,0}$, one notes that these binary values, when equal to 1, are multiplied by a ramping of x (at a constant rate) as the knot interval is passed. As x increases, the left term increases from zero to 1 and the right term decreases from 1 to zero. As one moves outwards, through the brackets, it is clear that the inner components are multiplied by similar weightings, determined by the proportional distance x currently is between the ends of a *series* of knots of ever increasing number.

Note that continuity in the contributions is preserved as x passes a knot, as in each of the inner most brackets, the $B_{-,0}$ terms are multiplied by a zero weighting at the x values on their bounds of contribution, i.e. for the first central bracket term when $x = t_i$ and $x = t_{i+2}$. The $B_{-,0}$ terms are also both multiplied by the same weighting at the middle knot, $x = t_{i+1}$, as one approaches this knot from above or below. As these two terms are mutually exclusive, equal to 1 at the knot and follow consecutively, this means that each bracket term is continuous. Since these terms are multiplied by clearly continuous functions, hence the whole equation is continuous.

The derivative of equation A.4 with respect to x is as follows:

$$\frac{dB_{i,n}(x)}{dx} = \frac{x - t_i}{t_{i+n} - t_i}\frac{dB_{i,n-1}(x)}{dx} + \frac{t_{i+n+1} - x}{t_{i+n+1} - t_{i+1}}\frac{dB_{i+1,n-1}(x)}{dx} + B_{i,n-1}(x) - B_{i+1,n-1}(x). \tag{A.8}$$

From noting that the components are continuous it can be shown that all derivatives are continuous. Firstly, this is because the B terms are continuous as detailed above. Secondly, if one were to follow through this equation to lower n value derivatives one would be left with inner bracket $\frac{dB_{-,0}(x)}{dx}$ terms such as:

$$\frac{dB_{i,1}(x)}{dx} = \frac{x - t_i}{t_{i+1} - t_i}\frac{dB_{i,0}(x)}{dx} + \frac{t_{i+2} - x}{t_{i+2} - t_{i+1}}\frac{dB_{i+1,0}(x)}{dx} \tag{A.9}$$

At all points between knots $\frac{dB_{-,0}(x)}{dx} = 0$ as $B_{-,0}(x)$ is a constant here. Even at the central knot ($i + 1$ above) the derivatives cancel as one value of $B_{-,0}(x)$ decreases from 1 to 0 and the other increases by the same amount (and the fraction multipliers are both equal to 1). At the end knots (i and $i + 2$) the $\frac{dB_{-,0}(x)}{dx}$ terms are multiplied by zero so their value is not relevant.

The $B_{-,0}$ terms of equation A.7 can be collected as follows:

$$\begin{aligned}
B_{i,3}(x) = & \frac{x-t_i}{t_{i+3}-t_i}\frac{x-t_i}{t_{i+2}-t_i}\frac{x-t_i}{t_{i+1}-t_i}B_{i,0}(x) \\
& + \left[\frac{x-t_i}{t_{i+3}-t_i}\left(\frac{x-t_i}{t_{i+2}-t_i}\frac{t_{i+2}-x}{t_{i+2}-t_{i+1}} + \frac{t_{i+3}-x}{t_{i+3}-t_{i+1}}\frac{x-t_{i+1}}{t_{i+2}-t_{i+1}}\right)\right. \\
& \left. + \frac{t_{i+4}-x}{t_{i+4}-t_{i+1}}\frac{x-t_{i+1}}{t_{i+3}-t_{i+1}}\frac{x-t_{i+1}}{t_{i+2}-t_{i+1}}\right] B_{i+1,0}(x) \\
& + \left[\frac{x-t_i}{t_{i+3}-t_i}\frac{t_{i+3}-x}{t_{i+3}-t_{i+1}}\frac{t_{i+3}-x}{t_{i+3}-t_{i+2}}\right. \\
& \left. + \frac{t_{i+4}-x}{t_{i+4}-t_{i+1}}\left(\frac{x-t_{i+1}}{t_{i+3}-t_{i+1}}\frac{t_{i+3}-x}{t_{i+3}-t_{i+2}} + \frac{t_{i+4}-x}{t_{i+4}-t_{i+2}}\frac{x-t_{i+2}}{t_{i+3}-t_{i+2}}\right)\right] B_{i+2,0}(x) \\
& + \frac{t_{i+4}-x}{t_{i+4}-t_{i+1}}\frac{t_{i+4}-x}{t_{i+4}-t_{i+2}}\frac{t_{i+4}-x}{t_{i+4}-t_{i+3}}B_{i+3,0}(x) \qquad \text{(A.10)}
\end{aligned}$$

To reiterate, the above, in x, represents the contributions of a single point across all knot intervals, whereas it is desired to have an expression for all contributions at a point t between two specific, consecutive knots, located at $x = t_z$ and $x = t_{z+1}$. The knot contributions for a point in the curve, x, between $x = t_z$ and $x = t_{z+1}$, come from the knots at $x = t_z$ and before, here $x = t_{z-1}$,$x = t_{z-2}$ and $x = t_{z-3}$. If it is set that $x = t - t_z$, then the contributions during this interval from these previous knots can be found. For point $x = t_{z-3}$ for example, in equation A.7 one thus only needs to look at $B_{0,z} = B_{0,i+3}$ terms. Note that, as is implied, in the following equation t_i is used to represent what was t_{z-3} in the proceeding explanation. The separation between the grid points is a constant, τ, so it is possible to also express the above equations in a kinder form, noting that $x = t\tau + t_j$ for $B_{j,-}(x)$.

The above also explains how the normalisation is built in, for all contributions for a particular value of x. Looking at it with Equation A.4 for $B_{j,n}(x)$ (where it is set $i = j$), each first term, $\frac{x-t_j}{t_{j+n}-t_j}B_{j,n-1}(x)$ will have a counterpart from $i = j-1$ of $\frac{t_{j-1+n+1}-x}{t_{j-1+n+1}-t_{j-1+1}}B_{j-1+1,n-1}(x) = \frac{t_{j+n}-x}{t_{j+n}-t_j}B_{j,n-1}(x)$. The sum of the two weights for the $B_{j,n-1}(x)$ component is equal to 1, and as the B values for lower n are calculated, each summing to 1 the total will always be 1 as the weightings multiply out (similar to random event probabilities). This also demonstrates the symmetrical nature of the components in t and $1-t$ below.

Hence the polynomial terms for $B(t)$ can be written:

$$
\begin{aligned}
B(t) = & \frac{t\tau}{3\tau}\frac{t\tau}{2\tau}\frac{t\tau}{\tau}\phi_0 \\
& + \left[\frac{t\tau+\tau}{3\tau}\left(\frac{t\tau+\tau}{2\tau}\frac{\tau-t\tau}{\tau}+\frac{2\tau-t\tau}{2\tau}\frac{t\tau}{\tau}\right)+\frac{3\tau-t\tau}{3\tau}\frac{t\tau}{2\tau}\frac{t\tau}{\tau}\right]\phi_1 \\
& + \left[\frac{t\tau+2\tau}{3\tau}\frac{\tau-t\tau}{2\tau}\frac{\tau-t\tau}{\tau}+\frac{2\tau-t\tau}{3\tau}\left(\frac{t\tau+\tau}{2\tau}\frac{\tau-t\tau}{\tau}+\frac{2\tau-t\tau}{2\tau}\frac{t\tau}{\tau}\right)\right]\phi_2 \\
& + \frac{\tau-t\tau}{3\tau}\frac{\tau-t\tau}{2\tau}\frac{\tau-t\tau}{\tau}\phi_3 \qquad (A.11)
\end{aligned}
$$

Which simplifies to:

$$
\begin{aligned}
B(t) = & \frac{t^3}{6}\phi_0 \qquad (A.12) \\
& + \frac{(t+1)\left[(t+1)(1-t)+(2-t)t\right]+(3-t)t^2}{6}\phi_1 \\
& + \frac{(t+2)(1-t)^2+(2-t)\left[(t+1)(1-t)+(2-t)t\right]}{6}\phi_2 \\
& + \frac{(1-t)^3}{6}\phi_3 \\
= & \frac{1}{6}t^3\phi_0+\frac{1}{6}\left[-3t^3+3t^2+3t+1\right]\phi_1 \qquad (A.13) \\
& + \frac{1}{6}\left[3t^3-6t^2+4\right]\phi_2+\frac{1}{6}(1-t)^3\phi_3
\end{aligned}
$$

A.3 INITAL REGISTRATION SCORING GUIDE

The following is a guide for scoring the test stack registrations. Particular attention was paid to the boundary of the neuropil, the Mushroom Bodies and the fascicles within the neuropil.

A: Fascicles within the neuropil nearly always overlap at least partially and the general neuropil domain structure has no noticeable distortions. Outside the neuropil cell bodies do not need to align but the general structure should be mostly within 10μm of where expected.

B: General neuropil domain structure has no noticeable distortions, fascicles may not overlap at all in one or two small subparts of the neuropil but are very close (almost all within 5μm). Mushroom bodies are expected to be well aligned, but some extra-neuropil distortion over 10μm at the stack edges is permitted.

C: One or two small noticeable distortions (with displacements of around 5μm-10μm from corresponding positions, less than 5% of volume) occur in the neuropil domain structure, often involving mushroom bodies or volumes near the neuropil boundary. Distortions such as in B nearly always also occur.

D: Up to 10% of the image stack is more than 10μm away from its corresponding position, or more than 5% of the image is more than 5μm away from corresponding positions.

E: Significant distortions with many regions shifted more than 10μm from corresponding location. Some of the image stack (between 25-90%) does match within 10μm, however, allowing the distortions to be placed in context.

F: Degree of distortions not measurable as features are not near to each other.

A.4 CMTK PARAMETERS

For the parameter and template test set, the signal of each stack was thresholded using the same thresholding algorithm as for the post-registered stacks with varying proportions of signal to retain, chosen to return volumes that best exclusively represent the neurons of interest.

Align principal axes and centre of mass.

```
cmtk make_initial_affine -principal-axes TEMPLATE FLOAT INITIAL_OUTPUT
```

Affine registration using correlation ratio metric

```
cmtk registration -initial INITIAL_OUTPUT -cr OPTIONS -o AFFINE_OUTPUT
TEMPLATE FLOAT
```

Option Name	Option Command	Value
Number of resolution levels	auto-multi-levels	4
Degrees of freedom	dofs	6,9
Search space exploration (initial step size)	exploration	8

Non rigid registration using correlation ratio metric

```
cmtk warp -o WARP_OUTPUT -cr -fast OPTIONS -t WARP_INFO -initial
AFFINE_OUTPUT TEMPLATE FLOAT
```

Option name	Option Command	Value
CPG		
Control point grid spacing (initial)	grid-spacing	80
Number of refinements of CPG	refine	1
Optimisation		
Search space exploration (initial step size)	exploration	30
Search step reduction factor	stepfactor	0.5
Search accuracy (final step size)	accuracy	0.2
Upper limit for image sampling in multiresolution hierarchy	coarsest	4
Regularisation		
Grid bending energy constraint weight	energy-weight	0.1
Local rigidity constraint weight	rigidity-weight	0 (off)
Jacobian-based local volume preservation constraint weight	jacobian-weight	0 (off)
Weight relaxation factor	relax	-1 (off)

Dimensions in voxel widths in xy dimensions (i.e. 0.293µm). Step sizes refer to decreasing shifts in parameters during optimization.

A.5 CELL TYPE TRAINING AND TEST SETS SUMMARY

Set 1

1. Used as Registration Accuracy Evaluation (134 stacks, 18 cell types)
2. Subset (75 well registered stacks, 15 cell types) used for BLAST method training (generating histograms to estimate probabilities). Examples from this also used for M/L Model evaluation (test set).

Set 2 Used to test the matching methods (140 stacks, 13 cell types). Includes 3 sub-cell type categories, namely "single cell neural volume of this type" (76 stacks), "multiple cell neural volumes containing cell of this type" (51 stacks), "unsure is neural volume contains cell of same type" (13 stacks).

Set 3 Used to train the Machine Learning Local Neuron Representations (184 stacks, 52 cell type groups).

A.6 NEURAL MATCHING CONVOLUTIONAL NEURAL NETWORK PARAMETERS

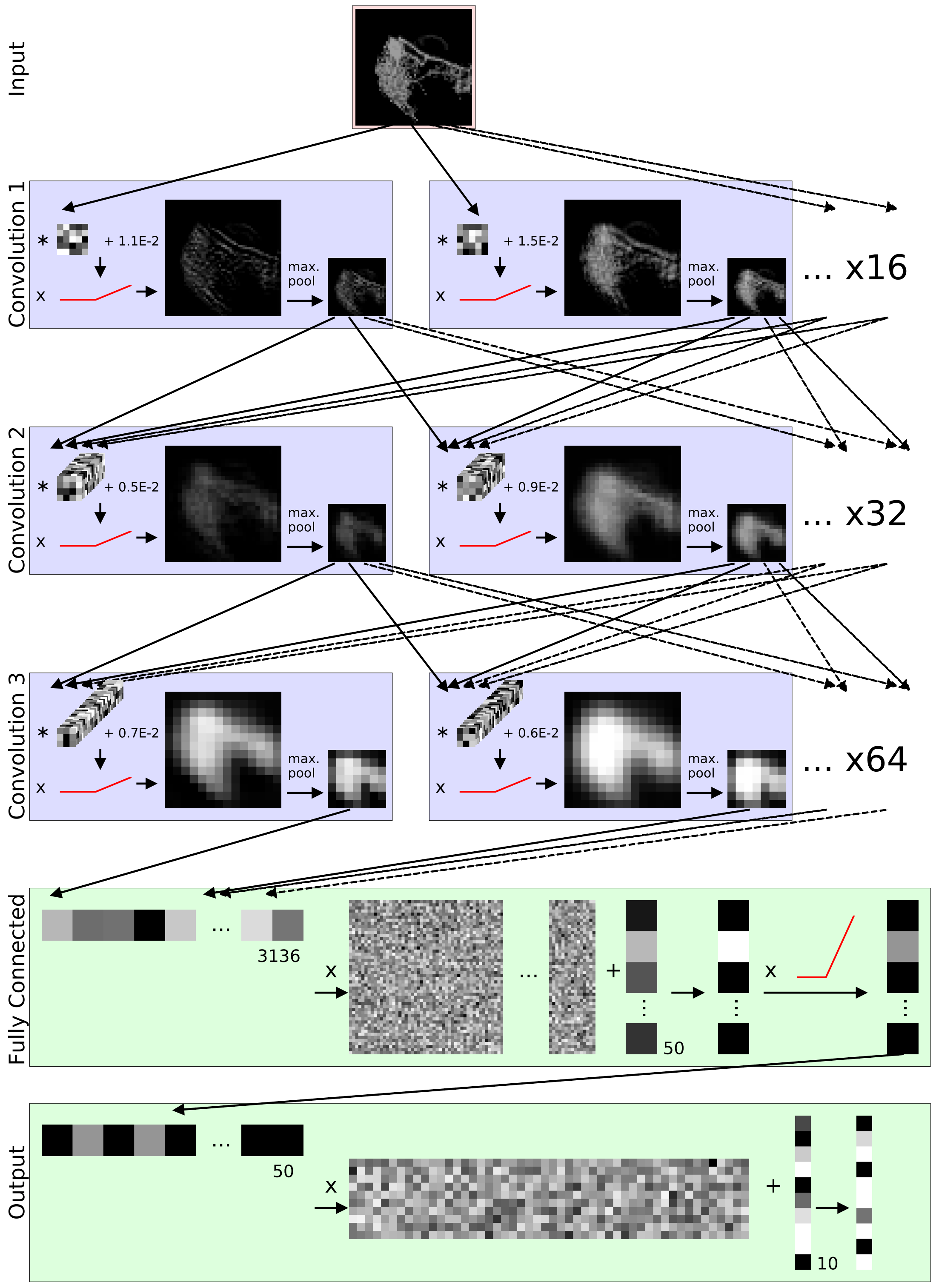


Figure A.9: Convolutional Neural Network For Matching Neurons.
Diagram of the neural network for condensing the maximum-projection windows to 10 feature vectors. The neuron weights and outputs are represented by the different shades of grey and intermediate images shown following the convolutions (scale

varies among images). The red line graph represents the rectifier linear neurons used and the arrows the flow of information between components. There were 16 convolutional dimensions in the first layer, 32 in the second and 64 in the third, with each layer taking the max. pool of a 2x2 grid, giving 3136 inputs to the fully connected layer that reduced to 50 output neurons and, through the final linear layer, a 10 dimensional feature vector. for a soft max.

A.7 AVERAGE NEURON ALGORITHM

1. Preprocessing steps:
   The directions are calculated for all points using the SVD method with points within 3 voxel traversals as in section 4.5.2 and the nBLAST calculation is performed for each point to its nearest neighbour point in each of the other skeletons in order to give an average nBLAST score for each point. Next, for each point in each skeleton, all possible linear paths of up to 100 voxels along the skeleton (using 26-neighbour connectivity) starting at this point, without repeating voxels, were constructed. These paths were given an nBLAST score that was the average of all average point scores of voxels in the path.

2. Find initial matching paths:
   To begin matching a well matched region, the path with the best (highest) nBLAST path score for as yet untried paths of length over 10 voxels from all skeletons was found. Then, for each other skeleton, a path-path nBLAST score was calculated from the best path to all paths over length 10 starting at voxels within 20μm of the position of the best path starting voxel in the registered template space. Rather than finding the nearest neighbour voxel for each point in the best path, however, each point was paired with that of the corresponding point in sequence as far as the lengths allowed. The path-path matching score was the average nBLAST score calculated for these pairs. The best matching path from each skeleton gave me a group of paths, one from each skeleton. If all had path-path matching scores with the initial best nBLAST path of greater than -1 then this gave an initial average point, otherwise the process was repeated with the next highest nBLAST scored, untried path in all skeletons until a satisfactory group was found (or none at all and the process skipped to step 7).

3. Building Tubes:
   The starting points of this group of best matching paths represent an average point, the position of which is the average position of these original skeleton points. The tube is then extended in each skeleton by first using the approximate string matching algorithm Skiena (2008) to align points in each path to all others. This method efficiently finds the best path-path alignment for the common minimum length of points in the two paths. It allows insertions and deletions and points matching to multiple adjacent other skeleton points (i.e. bunching up of the paths). For a particular alignment of points-points for two

paths the alignment score is the sum of the dot products of each point SVD direction for each point pair plus the penalty for each insertion / deletion (0.25). For each path-path combination, the first point-point pair in the alignment that does not include either of the points in the first point-point pair are designated the next point in this path-path combination. To estimate the position of the next point in each skeleton's tube, I take the rounded average increment along the path for all combinations involving that skeleton's path.

All points after the original up to and including this new point are marked as matched and added to the local skeleton's tube. The next point in the average tube contains the new point from each skeleton and its position is the average of those of its group of points. All paths that still have an unmatched second voxel and that begin at the one voxel in each skeleton representing the new point from the rest of a skeleton form the new candidate set of paths. The best matching group within this set is found as above in step 2 for all paths but without constraints on the minimum length of the paths and a lower path-path nBLAST score threshold of -2.5. Providing a satisfactory group of paths can be found for this best path, the tube is incremented further, otherwise it ends.

4. Iterating To Add More Tubes:
   The matched voxels form a connected component in each skeleton, helping to preserve some morphology between the skeletons. All paths from all voxels in the currently matching connected components that still had an unmatched second voxel from the rest of a skeleton form the new candidate set of paths. The best matching group within this set is found again as above in step 2 for all paths but again without constraints on the minimum length of the paths. A new tube is then started and extended and this process iterated until no more satisfactory groups of well matched paths were found starting at voxels in the ever-growing connected component.

5. Adding More Connected Components:
   It may be the case that there are multiple well matching connected components that are not linked by a clearly matchable region, and as such steps 2-4 are repeated only considering paths with still unmatched second-voxels.

6. Average skeleton:
   The average tubes may include portions that are in regions of high arborisation and as such I found the average tubeness score for each average point from the tubeness scores for its constituent original skeleton points. Any points above 0.004 are designated as high tubeness points. The final designation of

"good" tubeness points comes after a smoothing by a weighted Gaussian with standard deviation 10µm. Any points with over 50% tube weighting in a length of at least 5 are good tubeness points. For each original skeleton, its points at and between these good tubeness points are then used as key centre points for drawing 3D tubes of radius 3µm that interpolate linearly between the key points. Any signal within these tubes is then removed.

7. Average cell:
   The signal for each neural volume is that above the threshold in the original image stack used to segment the neural volumes. Following removal of good tubeness regions, the remaining arborisation signal is expanded to fill small gaps using a Gaussian convolution of standard deviation 3µm and all remaining voxels above 0.025 kept as signal. All voxels that are present in all dilated arborisation signal and part of a 26-connected component larger than 2000 voxels are kept as part of the arborisation regions in the average cell. To this the average tubes are added (with radius 1.5µm) to create the final, average cell representation.

LIST OF FIGURES

Figure 1.1 *Drosophila* Melanogater Life Cycle and Anatomy 2
Figure 1.2 Overview of Larval Central Nervous System 3
Figure 1.3 Overview of Neuropil of the CNS and Development 5
Figure 1.4 Explanation of Genetic Tools 9
Figure 1.5 The Virtual Fly Brain (Adult) 12
Figure 2.1 Previous FileMaker Database 16
Figure 2.2 TrackEM Viewer 18
Figure 2.3 Brain Axes Orientation 19
Figure 2.4 3D Neuropil Domains 20
Figure 2.5 3D Neuropil Domain Cross Sections 21
Figure 3.1 Overview of the image registration pipeline 27
Figure 3.2 Overview of registration process 28
Figure 3.3 Simple 2D summary of CMTK image registration steps 31
Figure 3.4 B-Splines Explanation 32
Figure 3.5 Examples of overlays of registered reference channels to a template 36
Figure 3.6 Illustration of the overlap score calculation method 38
Figure 3.7 Overlap scores for example cell type 39
Figure 3.8 Illustration of some unsuitable features in a template 41
Figure 3.9 The template 44
Figure 3.10 Average of registered image stacks 45
Figure 3.11 Illustration of contrast normalisation including image intensity histograms 47
Figure 3.12 Gaussian smoothing of image pre-registration 48
Figure 3.13 VNC dorso-ventral orientation determination 50
Figure 3.14 Voxel connectivity 52
Figure 3.15 Successful segmentation and neural volume examples 53
Figure 3.16 Examples of removed small signal volumes 54
Figure 3.17 Limitations of segmentation process 54
Figure 3.18 Histograms of number of voxels and accuracy scores per neural volume 55
Figure 3.19 Correlations between overlap and unsatisfactory accuracy scores 57

Figure 3.20 Explanation of implemented registration accuracy scoring method . . . 59
Figure 3.21 Random, known warp applied to template . . . 62
Figure 3.22 Correlations between overlap and implemented accuracy score 64
Figure 3.23 Examples of correlations in the subgroups . . . 65
Figure 3.24 Example accuracy score maps within a neural volume . . . 66
Figure 3.25 Outliers of overlap and accuracy scores . . . 67
Figure 3.26 Density map of successfully registered neural volumes . . . 68
Figure 3.27 Example of mirrored stack . . . 70
Figure 3.28 Skeletonisation example . . . 73
Figure 3.29 Skeletonisation and biological cell variation . . . 74
Figure 3.30 Time taken to run CMTK registration for the test data set . . . 75
Figure 3.31 Time taken to run scoring and skeletonisation for all stacks . . 75
Figure 4.1 Examples of the Cell Body Detection ROIs . . . 81
Figure 4.2 Cell Body Results for Gaussian Filters . . . 83
Figure 4.3 Cell Body Results for Linear Model Filter . . . 84
Figure 4.4 Layer Diagram of Convolutional Neural Network For Cell Body Detection . . . 87
Figure 4.5 Convolutional Neural Network Results . . . 88
Figure 4.6 Errors Of Convolutional Neural Network . . . 88
Figure 4.7 Validation of Cell Body Detection . . . 90
Figure 4.8 Skeleton Pruning Examples . . . 93
Figure 4.9 Tubeness Example . . . 94
Figure 4.10 Fixed Point Boolean Comparison Example . . . 96
Figure 4.11 Fixed Point Boolean Comparison Matching Results . . . 98
Figure 4.12 BLAST Score Examples . . . 99
Figure 4.13 pBLAST Functional Mappings . . . 101
Figure 4.14 Percentage of Matching Cells Found Above Threshold for BLAST Methods . . . 102
Figure 4.15 nBLAST Functional Mappings . . . 103
Figure 4.16 nBLAST Example Ranking . . . 106
Figure 4.17 Percentage of Matching Cells Found Above Threshold for Machine Learning Methods . . . 109
Figure 4.18 Training Convolutional Neural Network For Matching Neurons 110
Figure 4.19 Convolutional Neural Network Clusters For Neural Projection Windows . . . 111
Figure 4.20 Convolutional Neural Network Projection Window Representation . . . 112

Figure 4.21 Landmarks and EM to LM Test Set Results 115
Figure 4.22 Examples of EM to LM Mapping 117
Figure 4.23 Example of average neuron skeleton matching 119
Figure 4.24 Average neuron 3D Representations 120
Figure 4.25 Explanation For Limited Search Result Perfomance 123
Figure 4.26 Explanation For More Than 150 Points Not Being Beneficial . . 123
Figure 5.1 Example of Labelling Web App 128
Figure 5.2 False Discovery Rate vs. Recall Graphs For Hierachical Clustering 132
Figure 5.3 Hierachical Clustering of Brain Neurons 133
Figure 5.4 Elbow Plots for Hierachical Clustering 134
Figure 5.5 Arborisation Region Representation 137
Figure 5.6 PCA Accounting for Variance in the Central Brain 140
Figure 5.7 Central Brain PCA Mode Plots 140
Figure 5.8 2D Representation of Neuropil Domains Using t-SNE 142
Figure 5.9 K-means Clustering 143
Figure 5.10 Comparison of Automated to Manual Domains 148
Figure 5.11 Example of Slice Viewer in Larval Virtual Fly Brain. 151
Figure 5.12 Example of 3D View in Larval Virtual Fly Brain. 152
Figure A.1 Central Brain Domain Definitions A 164
Figure A.2 Central Brain Domain Definitions B 165
Figure A.3 Central Brain Domain Definitions C 166
Figure A.4 Central Brain Domain Definitions D 167
Figure A.5 Central Brain Domain Definitions E 168
Figure A.6 Horizontal sections of Ventral Nerve Cord 176
Figure A.7 Coronal cross sections of Ventral Nerve Cord 177
Figure A.8 B-Spline Equations Illustrations 180
Figure A.9 Convolutional Neural Network For Matching Neurons. 190

BIBLIOGRAPHY

Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G. S., Davis, A., Dean, J., Devin, M., Ghemawat, S., Goodfellow, I., Harp, A., Irving, G., Isard, M., Jia, Y., Jozefowicz, R., Kaiser, L., Kudlur, M., Levenberg, J., Mané, D., Monga, R., Moore, S., Murray, D., Olah, C., Schuster, M., Shlens, J., Steiner, B., Sutskever, I., Talwar, K., Tucker, P., Vanhoucke, V., Vasudevan, V., Viégas, F., Vinyals, O., Warden, P., Wattenberg, M., Wicke, M., Yu, Y., and Zheng, X. (2015). {TensorFlow}: Large-Scale Machine Learning on Heterogeneous Systems. (Cited on pages 85 and 111.)

Acciai, L., Soda, P., and Iannello, G. (2016). Automated Neuron Tracing Methods: An Updated Account. (Cited on pages 69 and 91.)

Aldus Developers Desk (1992). *TIFF (Tagged Image File Format), Revision 6.0*. Aldus Corporation. (Cited on page 16.)

Alpert, N. M., Bradshaw, J. F., Kennedy, D., and Correia, J. A. (1990). The principal axes transformation–a method for image registration. *Journal of Nuclear Medicine*. (Cited on page 31.)

Altschul, S., Gish, W., Miller, W., Myers, E., and Lipman, D. (1990). Basic Local Alignment Search Tool. *J. Mol. Biol.* (Cited on page 98.)

Apostolopoulou, A. A., Rist, A., and Thum, A. S. (2015). Taste processing in Drosophila larvae. *Frontiers in Integrative Neuroscience*. (Cited on page 2.)

Arganda-Carreras, I., Fernández-González, R., Muñoz-Barrutia, A., and Ortiz-De-Solorzano, C. (2010). 3D reconstruction of histological sections: Application to mammary gland tissue. *Microscopy Research and Technique*. (Cited on page 91.)

Arganda-Carreras, I., Manoliu, T., Mazuras, N., Schulze, F., Iglesias, J. E., Bühler, K., Jenett, A., Rouyer, F., and Andrey, P. (2018). A Statistically Representative Atlas for Mapping Neuronal Circuits in the Drosophila Adult Brain. *Frontiers in Neuroinformatics*, 12:13. (Cited on page 42.)

Armstrong, J. D. and van Hemert, J. I. (2009). Towards a virtual fly brain. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*. (Cited on page 29.)

Arthur, D. and Vassilvitskii, S. (2007). k-means++: the advantages of careful seeding. In *Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms*. (Cited on page 144.)

Avants, B. B., Epstein, C. L., Grossman, M., and Gee, J. C. (2008). Symmetric diffeomorphic image registration with cross-correlation: Evaluating automated labeling of elderly and neurodegenerative brain. *Medical Image Analysis*. (Cited on page 29.)

Bargmann, C. I. (2012). Beyond the connectome: How neuromodulators shape neural circuits. *BioEssays*. (Cited on page 1.)

Barnes, J. and Hut, P. (1986). A hierarchical O(N log N) force-calculation algorithm. *Nature*. (Cited on page 142.)

Basu, S., Condron, B., and Acton, S. T. (2011). Path2Path: Hierarchical path-based analysis for neuron matching. In *Proceedings - International Symposium on Biomedical Imaging*. (Cited on page 95.)

Bate, M. (1990). The embryonic development of larval muscles in Drosophila. *Development*. (Cited on page 3.)

Beck, K., Beedle, M., Bennekum, A., Cockburn, A., Cunningham, W., Fowler, M., Grenning, J., Highsmith, J., Hunt, A., Jeffries, R., Kern, J., Marick, B., Martin, R. C., Mellor, S., Schwaber, K., Sutherland, J., and Thomas, D. (2001). Manifesto for Agile Software Development. (Cited on page 26.)

Bello, B. C., Izergina, N., Caussinus, E., and Reichert, H. (2008). Amplification of neural stem cell proliferation by intermediate progenitor cells in Drosophila brain development. *Neural Development*. (Cited on page 6.)

Berrigan, D. and Pepin, D. J. (1995). How maggots move: Allometry and kinematics of crawling in larval Diptera. *Journal of Insect Physiology*. (Cited on page 3.)

Bishop, C. M. (2006). *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer-Verlag New York, Inc., Secaucus, NJ, USA. (Cited on pages 139 and 145.)

Bohland, J. W., Bokil, H., Allen, C. B., and Mitra, P. P. (2009). The brain atlas concordance problem: Quantitative comparison of anatomical parcellations. *PLoS ONE*. (Cited on page 145.)

Bohland, J. W., Bokil, H., Pathak, S. D., Lee, C. K., Ng, L., Lau, C., Kuan, C., Hawrylycz, M., and Mitra, P. P. (2010). Clustering of spatial gene expression patterns in the mouse brain and comparison with classical neuroanatomy. *Methods*. (Cited on page 138.)

Bossing, T., Udolph, G., Doe, C. Q., and Technau, G. M. (1996). The embryonic central nervous system lineages of Drosophila melanogaster. I. Neuroblast lineages derived from the ventral half of the neuroectoderm. *Developmental Biology*. (Cited on page 6.)

Bradski, G. (2000). The OpenCV Library. *Dr. Dobb's Journal of Software Tools*. (Cited on page 107.)

Brand, A. H. and Perrimon, N. (1993). Targeted gene expression as a means of altering cell fates and generating dominant phenotypes. *Development (Cambridge, England)*. (Cited on page 7.)

Campos-Ortega, J. A. and Hartenstein, V. (1997). *The Embryonic Development of Drosophila melanogaster*. (Cited on pages 2, 3, and 4.)

Cardona, A., Saalfeld, S., Arganda, I., Pereanu, W., Schindelin, J., and Hartenstein, V. (2010). Identifying Neuronal Lineages of Drosophila by Sequence Analysis of Axon Tracts. *Journal of Neuroscience*. (Cited on page 120.)

Cardona, A., Saalfeld, S., Schindelin, J., Arganda-Carreras, I., Preibisch, S., Longair, M., Tomancak, P., Hartenstein, V., and Douglas, R. J. (2012). TrakEM2 software for neural circuit reconstruction. *PLoS ONE*. (Cited on page 18.)

Cardona, A., Saalfeld, S., Tomancak, P., and Hartenstein, V. (2009). Drosophila brain development: Closing the gap between a macroarchitectural and microarchitectural approach. In *Cold Spring Harbor Symposia on Quantitative Biology*. (Cited on page 6.)

Chalfoun, J., Majurski, M., Blattner, T., Bhadriraju, K., Keyrouz, W., Bajcsy, P., and Brady, M. (2017). MIST: Accurate and Scalable Microscopy Image Stitching Tool with Stage Modeling and Error Minimization. *Scientific Reports*. (Cited on page 159.)

Chiang, A. S., Lin, C. Y., Chuang, C. C., Chang, H. M., Hsieh, C. H., Yeh, C. W., Shih, C. T., Wu, J. J., Wang, G. T., Chen, Y. C., Wu, C. C., Chen, G. Y., Ching, Y. T., Lee, P. C., Lin, C. Y., Lin, H. H., Wu, C. C., Hsu, H. W., Huang, Y. A., Chen, J. Y., Chiang, H. J., Lu, C. F., Ni, R. F., Yeh, C. Y., and Hwang, J. K. (2011). Three-dimensional reconstruction of brain-wide wiring networks in drosophila at single-cell resolution. *Current Biology*. (Cited on page 12.)

Chothani, P., Mehta, V., and Stepanyants, A. (2011). Automated tracing of neurites from light microscopy stacks of images. *Neuroinformatics*. (Cited on pages 71 and 78.)

Church, R. B. and Robertson, F. W. (1966). A biochemical study of the growth of Drosophila melanogaster. *Journal of Experimental Zoology*. (Cited on page 2.)

Cignoni, P., Callieri, M., Corsini, M., Dellepiane, M., Ganovelli, F., and Ranzuglia, G. (2008). MeshLab: an Open-Source Mesh Processing Tool. In Scarano, V., Chiara, R. D., and Erra, U., editors, *Eurographics Italian Chapter Conference*. The Eurographics Association. (Cited on page 151.)

Costa, M., Manton, J. D., Ostrovsky, A. D., Prohaska, S., and Jefferis, G. S. (2016). NBLAST: Rapid, Sensitive Comparison of Neuronal Structure and Construction of Neuron Family Databases. *Neuron*. (Cited on pages 95, 98, 103, 129, and 134.)

Costa, M., Reeve, S., Grumbling, G., and Osumi-Sutherland, D. (2013). The Drosophila anatomy ontology. *Journal of Biomedical Semantics*. (Cited on page 10.)

Court, R. (2017). *Dissecting the ventral nervous system in Drosophila melanogaster*. Phd, Edinburgh. (Cited on pages 30 and 46.)

Davis, R. L. (2011). Traces of Drosophila Memory. (Cited on page 8.)

Dempster, A. P., Laird, N. M., and Rubin, D. B. (1977). Maximum likelihood from incomplete data via the em EM algorithm. *Journal of the Royal Statistical Society, Series B*. (Cited on page 144.)

Eichler, K., Li, F., Litwin-Kumar, A., Park, Y., Andrade, I., Schneider-Mizell, C. M., Saumweber, T., Huser, A., Eschbach, C., Gerber, B., Fetter, R. D., Truman, J. W., Priebe, C. E., Abbott, L. F., Thum, A. S., Zlatic, M., and Cardona, A. (2017). The complete connectome of a learning and memory centre in an insect brain. *Nature*. (Cited on page 69.)

Encyclopedia of Mathematics (2017). Diffeomorphism. (Cited on page 28.)

Foley, D., O'Brien, D. J., León-Vintró, L., McClean, B., and McBride, P. (2016). Phase correlation applied to the 3D registration of CT and CBCT image volumes. *Physica Medica*. (Cited on page 46.)

Frey, B. J. and Dueck, D. (2007). Clustering by passing messages between data points. *Science*. (Cited on page 134.)

Gala, R., Chapeton, J., Jitesh, J., Bhavsar, C., and Stepanyants, A. (2014). Active learning of neuron morphology for accurate automated tracing of neurites. *Frontiers in Neuroanatomy*. (Cited on page 78.)

Gerber, B. and Stocker, R. F. (2007). The drosophila larva as a model for studying chemosensation and chemosensory learning: A review. (Cited on pages 3 and 8.)

Gillette, T. A., Hosseini, P., and Ascoli, G. A. (2015). Topological characterization of neuronal arbor morphology via sequence representation: II - global alignment. *BMC Bioinformatics*. (Cited on pages 95, 118, and 126.)

Golic, K. G. and Lindquist, S. (1989). The FLP recombinase of yeast catalyzes site-specific recombination in the drosophila genome. *Cell*. (Cited on page 9.)

Golik, P., Doetsch, P., and Ney, H. (2013). Cross-entropy vs. Squared error training: A theoretical and experimental comparison. In *Proceedings of the Annual Conference of the International Speech Communication Association, INTERSPEECH*. (Cited on page 86.)

Goshtasby, A. A. (2005). *2-D and 3-D Image Registration: for Medical, Remote Sensing, and Industrial Applications*. (Cited on page 29.)

Goshtasby, A. A. (2012). Image Registration: Principles, Tools and Methods. *Advances in Computer Vision and Pattern Recognition*. (Cited on pages 28 and 29.)

Hales, K. G., Korey, C. A., Larracuente, A. M., and Roberts, D. M. (2015). Genetics on the fly: A primer on the drosophila model system. *Genetics*. (Cited on pages 2 and 7.)

Hartenstein, V. (2008). Drosophila Brain Lineage Atlas. (Cited on page 5.)

Hartenstein, V. (2011). Morphological diversity and development of glia in Drosophila. *GLIA*. (Cited on page 4.)

Hartenstein, V., Omoto, J. J., Ngo, K. T., Wong, D., Kuert, P. A., Reichert, H., Lovick, J. K., and Younossi-Hartenstein, A. (2018). Structure and development of the subesophageal zone of the Drosophila brain. I. Segmental architecture, compartmentalization, and lineage anatomy. *Journal of Comparative Neurology*. (Cited on pages 17, 172, 173, and 175.)

Hartenstein, V., Rudloff, E., and Campos -Ortega, J. A. (1987). The pattern of proliferation of the neuroblasts in the wild-type embryo of Drosophila melanogaster. *Roux's Archives of Developmental Biology*. (Cited on page 6.)

Hartenstein, V., Spindler, S., Pereanu, W., and Fung, S. (2008). The Development of the Drosophila Larval Brain. In *Brain Development in Drosophila melanogaster*. (Cited on page 4.)

Hartenstein, V., Younossi-Hartenstein, A., Lovick, J. K., Kong, A., Omoto, J. J., Ngo, K. T., and Viktorin, G. (2015). Lineage-associated tracts defining the anatomy of the Drosophila first instar larval brain. *Developmental Biology*. (Cited on pages 6, 17, 23, 158, 169, 170, 171, and 172.)

Heckscher, E. S., Lockery, S. R., and Doe, C. Q. (2012). Characterization of Drosophila Larval Crawling at the Level of Organism, Segment, and Somatic Body Wall Musculature. *Journal of Neuroscience*. (Cited on page 3.)

Hennig, C. (2007). Cluster-wise assessment of cluster stability. *Computational Statistics and Data Analysis*. (Cited on pages 134 and 135.)

Heumann, H. and Wittum, G. (2009). The tree-edit-distance, a measure for quantifying neuronal morphology. *Neuroinformatics*. (Cited on page 95.)

Hoffer, E. and Ailon, N. (2015). Deep metric learning using triplet network. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. (Cited on page 110.)

Honjo, K., Hwang, R. Y., and Tracey, W. D. (2012). Optogenetic manipulation of neural circuits and behavior in Drosophila larvae. *Nature Protocols*. (Cited on page 7.)

Horton, R. E. (1945). Erosional deviation of streams and their drainage basins; hydrophysical approach to quantitative morphology. *Geological Society of America Bulletin*. (Cited on page 91.)

Hwang, R. Y., Zhong, L., Xu, Y., Johnson, T., Zhang, F., Deisseroth, K., and Tracey, W. D. (2007). Nociceptive Neurons Protect Drosophila Larvae from Parasitoid Wasps. *Current Biology*. (Cited on pages 3 and 5.)

Ito, K., Shinomiya, K., Ito, M., Armstrong, J. D., Boyan, G., Hartenstein, V., Harzsch, S., Heisenberg, M., Homberg, U., Jenett, A., Keshishian, H., Restifo, L. L., Rössler, W., Simpson, J. H., Strausfeld, N. J., Strauss, R., and Vosshall, L. B. (2014). A systematic nomenclature for the insect brain. *Neuron*. (Cited on pages 13 and 19.)

Iyengar, B. G. (2008). Balapagos Blog. (Cited on page 2.)

Jaccard, P. (1901). Étude comparative de la distribution florale dans une portion des Alpes et des Jura. *Bull. Soc. Vaudoise Sci. Nat.* (Cited on page 135.)

Jähne, B. (2004). *Practical Handbook on Image Processing for Scientific and Technical Applications*. (Cited on page 46.)

Jenett, A., Rubin, G. M., Ngo, T. T., Shepherd, D., Murphy, C., Dionne, H., Pfeiffer, B. D., Cavallaro, A., Hall, D., Jeter, J., Iyer, N., Fetter, D., Hausenfluck, J. H., Peng, H., Trautman, E. T., Svirskas, R. R., Myers, E. W., Iwinski, Z. R., Aso, Y., DePasquale, G. M., Enos, A., Hulamm, P., Lam, S. C. B., Li, H. H., Laverty, T. R., Long, F., Qu, L., Murphy, S. D., Rokicki, K., Safford, T., Shaw, K., Simpson, J. H., Sowell, A., Tae, S., Yu, Y., and Zugates, C. T. (2012). A GAL4-Driver Line Resource for Drosophila Neurobiology. *Cell Reports*. (Cited on page 69.)

Ji, S. (2013). Computational genetic neuroanatomy of the developing mouse brain: Dimensionality reduction, visualization, and clustering. *BMC Bioinformatics*. (Cited on pages 141 and 154.)

Jones, E., Oliphant, T., Peterson, P., and Others (2001). SciPy.org. (Cited on pages 72 and 129.)

Joshi, S., Davis, B., Jomier, M., and Gerig, G. (2004). Unbiased diffeomorphic atlas construction for computational anatomy. In *NeuroImage*. (Cited on page 42.)

Kanai, R., Feilden, T., Firth, C., and Rees, G. (2011). Political orientations are correlated with brain structure in young adults. *Current Biology*. (Cited on page 29.)

Kingma, D. P. and Ba, J. L. (2014). Adam: a Method for Stochastic Optimization. *International Conference on Learning Representations*. (Cited on page 110.)

Klein, A., Andersson, J., Ardekani, B. A., Ashburner, J., Avants, B., Chiang, M. C., Christensen, G. E., Collins, D. L., Gee, J., Hellier, P., Song, J. H., Jenkinson, M., Lepage, C., Rueckert, D., Thompson, P., Vercauteren, T., Woods, R. P., Mann, J. J., and Parsey, R. V. (2009). Evaluation of 14 nonlinear deformation algorithms applied to human brain MRI registration. *NeuroImage*. (Cited on page 37.)

Klein, S., Staring, M., Murphy, K., Viergever, M. A., and Pluim, J. P. W. (2010). Elastix: A toolbox for intensity-based medical image registration. *IEEE Transactions on Med-*

ical Imaging. (Cited on pages 29, 30, and 77.)

Knowles-Barley, S. F. (2012). *Proteins, anatomy and networks of the fruit fly brain*. Phd, Edinburgh. (Cited on page 30.)

Kohsaka, H., Okusawa, S., Itakura, Y., Fushiki, A., and Nose, A. (2012). Development of larval motor circuits in Drosophila. (Cited on page 5.)

Kohsaka, H., Takasu, E., Morimoto, T., and Nose, A. (2014). A group of segmental premotor interneurons regulates the speed of axial locomotion in drosophila larvae. *Current Biology*. (Cited on page 8.)

Kumar, A., Fung, S., Lichtneckert, R., Reichert, H., and Hartenstein, V. (2009). Arborization pattern of Engrailed-positive neural lineages reveal neuromere boundaries in the Drosophila brain neuropil. *Journal of Comparative Neurology*. (Cited on pages 17 and 173.)

Kwon, J. Y., Dahanukar, A., Weiss, L. A., and Carlson, J. R. (2011). Molecular and Cellular Organization of the Taste System in the Drosophila Larva. *Journal of Neuroscience*. (Cited on page 5.)

Lancaster, J. L., Tordesillas-Gutiérrez, D., Martinez, M., Salinas, F., Evans, A., Zilles, K., Mazziotta, J. C., and Fox, P. T. (2007). Bias between MNI and talairach coordinates analyzed using the ICBM-152 brain template. *Human Brain Mapping*. (Cited on page 42.)

Lance, G. N. and Williams, W. T. (1967). A general theory of classificatory sorting strategies: 1. Hierarchical systems. *The computer journal*, 9(4):373–380. (Cited on page 130.)

Lazzaro, D. and Montefusco, L. B. (2002). Radial basis functions for the multivariate interpolation of large scattered data sets. *Journal of Computational and Applied Mathematics*. (Cited on page 115.)

LeCun, Y. and Bengio, Y. (1995). Convolutional networks for images, speech, and time series. *The handbook of brain theory and neural networks*. (Cited on page 110.)

LeCun, Y., Bottou, L., Bengio, Y., and Haffner, P. (1998). Gradient based learning applied to document recognition. *Proceedings of the IEEE*. (Cited on page 86.)

Lee, P. C., Chuang, C. C., Chiang, A. S., and Ching, Y. T. (2012). High-throughput Computer Method for 3D Neuronal Structure Reconstruction from the Image Stack of the Drosophila Brain and Its Applications. *PLoS Computational Biology*. (Cited on pages 71 and 91.)

Lee, S., Wolberg, G., and Shin, S. Y. (1997). Scattered data interpolation with multilevel b-splines. *IEEE Transactions on Visualization and Computer Graphics*. (Cited on page 31.)

Lee, T., Kashyap, R., and Chu, C. (1994). Building Skeleton Models via 3-D Medial Surface Axis Thinning Algorithms. *CVGIP: Graphical Models and Image Processing*. (Cited on pages 71 and 72.)

Lein, E. S., Hawrylycz, M. J., and al., E. (2007). Genome-wide atlas of gene expression in the adult mouse brain. *Nature*. (Cited on page 138.)

Lewiner, T., Lopes, H., Vieira, A. W., and Tavares, G. (2012). Efficient Implementation of Marching Cubes' Cases with Topological Guarantees. *Journal of Graphics Tools*. (Cited on page 151.)

Li, H. H., Kroll, J. R., Lennox, S. M., Ogundeyi, O., Jeter, J., Depasquale, G., and Truman, J. W. (2014). A GAL4 driver resource for developmental and behavioral studies on the larval CNS of Drosophila. *Cell Reports*. (Cited on pages 4, 11, 69, and 77.)

Li, R., Zeng, T., Peng, H., and Ji, S. (2017a). Deep Learning Segmentation of Optical Microscopy Images Improves 3-D Neuron Reconstruction. *IEEE Transactions on Medical Imaging*. (Cited on page 157.)

Li, Y., Wang, D., Ascoli, G. A., Mitra, P., and Wang, Y. (2017b). Metrics for comparing neuronal tree shapes based on persistent homology. *PLoS ONE*. (Cited on page 81.)

Linford, N. J., Bilgir, C., Ro, J., and Pletcher, S. D. (2013). Measurement of lifespan in Drosophila melanogaster. *Journal of visualized experiments : JoVE*. (Cited on page 2.)

Liu, Z. (2014). An Adapted Spatial Information Kernel-based Fuzzy C-means Clustering Method. *International Congress on Image and Signal Processing*. (Cited on page 155.)

Longair, M. (2009). *Computational Neuroanatomy of the Central Complex of Drosophila melanogaster*. PhD thesis, University of Edinburgh. (Cited on page 93.)

Lovick, J. K., Kong, A., Omoto, J. J., Ngo, K. T., Younossi-Hartenstein, A., and Hartenstein, V. (2016). Patterns of growth and tract formation during the early development of secondary lineages in the Drosophila larval brain. *Developmental Neurobiology*. (Cited on page 6.)

Lovick, J. K., Ngo, K. T., Omoto, J. J., Wong, D. C., Nguyen, J. D., and Hartenstein, V. (2013). Postembryonic lineages of the Drosophila brain: I. Development of the lineage-associated fiber tracts. *Developmental Biology*. (Cited on pages 17, 23, and 170.)

Luan, H., Peabody, N. C., Vinson, C., and White, B. H. (2006). Refined Spatial Manipulation of Neuronal Function by Combinatorial Restriction of Transgene Expression. *Neuron*. (Cited on page 13.)

Luce, J., Gray, J., Hoggarth, M. a., Lin, J., Loo, E., Campana, M. I., and Roeske, J. C. (2014). Medical Image Registration Using the Fourier Transform. *International Journal of Medical Physics, Clinical Engineering and Radiation Oncology*. (Cited on page 28.)

Maaten, L. V. D. and Hinton, G. (2008). Visualizing Data using t-SNE. *Journal of Machine Learning Research 1*. (Cited on page 141.)

Maes, F., Collignon, A., Vandermeulen, D., Marchal, G., and Suetens, P. (1997). Multimodality image registration by maximization of mutual information. *IEEE Transactions on Medical Imaging*. (Cited on page 48.)

Mahfouz, A., van de Giessen, M., van der Maaten, L., Huisman, S., Reinders, M., Hawrylycz, M. J., and Lelieveldt, B. P. (2015). Visualizing the spatial gene expression organization in the brain through non-linear similarity embeddings. *Methods*. (Cited on page 141.)

Manton, J. D., Ostrovsky, A. D., Goetz, L., Costa, M., and Rohlfing, T. (2014). Combining genome-scale Drosophila 3D neuroanatomical data by bridging template brains. *bioRxiv*. (Cited on page 159.)

Markow, T. A., Beall, S., and Matzkin, L. M. (2009). Egg size, embryonic development time and ovoviviparity in Drosophila species. *J Evol Biol*. (Cited on page 2.)

Masse, N. Y., Cachero, S., Ostrovsky, A. D., and Jefferis, G. S. X. E. (2012). A Mutual Information Approach to Automate Identification of Neuronal Clusters in Drosophila Brain Images. *Frontiers in Neuroinformatics*. (Cited on pages 93 and 103.)

Mesbah, S., Conjeti, S., Kumaraswamy, A., Rautenberg, P., Navab, N., and Katouzian, A. (2015). Hashing forests for morphological search and retrieval in neuroscientific image databases. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. (Cited on pages 95 and 96.)

Milyaev, N., Osumi-sutherland, D., Reeve, S., Burton, N., Baldock, R. A., and Armstrong, J. D. (2012). The virtual fly brain browser and query interface. *Bioinformatics*. (Cited on page 10.)

Mottini, A., Descombes, X., and Besse, F. (2015). From Curves to Trees: A Tree-like Shapes Distance Using the Elastic Shape Analysis Framework. *Neuroinformatics*. (Cited on pages 95, 118, and 126.)

Muenzing, S. E., Strauch, M., Truman, J. W., Bühler, K., Thum, A. S., and Merhof, D. (2018). larvalign: Aligning Gene Expression Patterns from the Larval Brain of Drosophila melanogaster. *Neuroinformatics*. (Cited on pages 77, 78, 156, and 159.)

Murakami, R., Shigenaga, A., Matsumoto, A., Yamaoka, I., and Tanimura, T. (1994). Novel tissue units of regional differentiation in the gut epithelium of Drosopbila, as revealed by P-element-mediated detection of enhancer. *Roux's Archives of Developmental Biology*. (Cited on page 3.)

Narasimha, S., Kolly, S., Sokolowski, M. B., Kawecki, T. J., and Vijendravarma, R. K. (2015). Prepupal building behavior in Drosophila melanogaster and its evolution under resource and time constraints. *PLoS ONE*. (Cited on page 3.)

Nassif, C., Noveen, A., and Hartenstein, V. (2003). Early development of the Drosophila brain: III. The pattern of neuropile founder tracts during the larval period. *Journal of Comparative Neurology*. (Cited on pages 19 and 163.)

Nern, A., Pfeiffer, B. D., and Rubin, G. M. (2015). Optimized tools for multicolor stochastic labeling reveal diverse stereotyped cell arrangements in the fly visual system. *Proceedings of the National Academy of Sciences*. (Cited on pages 9 and 10.)

Nielsen, M. A. (2015). Neural Networks and Deep Learning. In *Neural Networks and Deep Learning*. (Cited on page 85.)

Nusslein-Volhard, C. and Wieschaus, E. (1980). Mutations affecting number and polarity in Drosophila. *Nature*. (Cited on page 7.)

Ohyama, T., Schneider-Mizell, C. M., Fetter, R. D., Aleman, J. V., Franconville, R., Rivera-Alba, M., Mensh, B. D., Branson, K. M., Simpson, J. H., Truman, J. W., Cardona, A., and Zlatic, M. (2015). A multilevel multimodal circuit enhances action selection in Drosophila. *Nature*. (Cited on pages 8, 13, and 160.)

Oliphant, T. E. (2006). *A guide to NumPy*. Trelgol Publishing. (Cited on page 72.)

Pandey, U. B. and Nichols, C. D. (2011). Human Disease Models in Drosophila melanogaster and the Role of the Fly in Therapeutic Drug Discovery. *Drug Delivery*. (Cited on page 8.)

Panser, K., Tirian, L., Schulze, F., Villalba, S., Jefferis, G. S., Bühler, K., and Straw, A. D. (2016). Automatic Segmentation of Drosophila Neural Compartments Using GAL4 Expression Data Reveals Novel Visual Pathways. *Current Biology*. (Cited on page 138.)

Pauls, D., Selcho, M., Gendre, N., Stocker, R. F., and Thum, A. S. (2010). Drosophila Larvae Establish Appetitive Olfactory Memories via Mushroom Body Neurons of Embryonic Origin. *Journal of Neuroscience*. (Cited on page 8.)

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., and Duchesnay, E. (2011). Scikit-learn: Machine Learning in {P}ython. *Journal of Machine Learning Research*, 12:2825–2830. (Cited on pages 139 and 144.)

Peng, H., Bria, A., Zhou, Z., Iannello, G., and Long, F. (2014). Extensible visualization and analysis for multidimensional images using Vaa3D. *Nature Protocols*. (Cited on page 16.)

Peng, H., Chung, P., Long, F., Qu, L., Jenett, A., Seeds, A. M., Myers, E. W., and Simpson, J. H. (2011). BrainAligner: 3D registration atlases of Drosophila brains. *Nature Methods*. (Cited on page 30.)

Peng, H., Ruan, Z., Long, F., Simpson, J. H., and Myers, E. W. (2010). V3D enables real-time 3D visualization and quantitative analysis of large-scale biological image data sets. *Nature Biotechnology*. (Cited on page 69.)

Pereanu, W. (2006). Neural Lineages of the Drosophila Brain: A Three-Dimensional Digital Atlas of the Pattern of Lineage Location and Projection at the Late Larval Stage. *Journal of Neuroscience*. (Cited on pages 17, 23, 172, and 173.)

Pereanu, W., Kumar, A., Jennett, A., Reichert, H., and Hartenstein, V. (2010). Development-based compartmentalization of the drosophila central brain. *Journal of Comparative Neurology*. (Cited on pages 17, 163, 169, 171, 173, and 174.)

Pereanu, W., Spindler, S., Cruz, L., and Hartenstein, V. (2007). Tracheal development in the Drosophila brain is constrained by glial cells. *Developmental Biology*. (Cited on page 4.)

Python, F. and Stocker, R. F. (2002). Adult-like complexity of the larval antennal lobe of D. melanogaster despite markedly low numbers of odorant receptor neurons. *Journal of Comparative Neurology*. (Cited on page 4.)

Richardson, L., Venkataraman, S., Stevenson, P., Yang, Y., Moss, J., Graham, L., Burton, N., Hill, B., Rao, J., Baldock, R. A., and Armit, C. (2014). EMAGE mouse embryo spatial gene expression database: 2014 update. *Nucleic Acids Research*. (Cited on page 29.)

Rickert, C., Kunz, T., Harris, K.-L., Whitington, P. M., and Technau, G. M. (2011). Morphological Characterization of the Entire Interneuron Population Reveals Principles of Neuromere Organization in the Ventral Nerve Cord of Drosophila. *Journal of Neuroscience*. (Cited on page 6.)

Roche, A., Malandain, G., Pennec, X., and Ayache, N. (1998). The Correlation Ratio as a New Similarity Measure for Multimodal Image Registration. *Proceedings of the International Conference on Medical Image Computing and Computer-Assisted Intervention (MICCAI)*. (Cited on page 49.)

Rohlfing, T. (2011). User Guide to The Computational Morphometry Toolkit. (Cited on pages 31, 33, and 34.)

Rohlfing, T., Brandt, R., Menzel, R., and Maurer, C. R. (2004). Evaluation of atlas selection strategies for atlas-based image segmentation with application to confocal microscopy images of bee brains. *NeuroImage*. (Cited on page 42.)

Rohlfing, T. and Maurer, C. R. (2001). Intensity-based non-rigid registration using adaptive multilevel free-form deformation with an incompressibility constraint. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. (Cited on page 33.)

Rohlfing, T. and Maurer, C. R. (2003). Nonrigid image registration in shared-memory multiprocessor environments with application to brains, breasts, and bees. *IEEE Transactions on Information Technology in Biomedicine*. (Cited on pages 30, 33, and 34.)

Rubin, G. M. and Spradling, A. C. (1982). Genetic transformation of Drosophila with transposable element vectors. *Science*. (Cited on page 7.)

Rueckert, D., Sonoda, L. I., Hayes, C., Hill, D. L. G., Leach, M. O., and Hawkes, D. J. (1999). Nonrigid registration using free-form deformations: application to breast MR images. *Medical Imaging, IEEE Transactions on*. (Cited on pages 31 and 33.)

Russ, J. C. (2011). The image processing handbook. In *The image processing handbook*. (Cited on page 157.)

Sanborn, A. (2015). Detection of fluorescent neuron cell bodies using convolutional neural networks. Technical report, Stanford University. (Cited on page 85.)

Sarkar, A., Santiago, R. J., Smith, R., and Kassaee, A. (2005). Comparison of manual vs. automated multimodality (CT-MRI) image registration for brain tumors. *Medical Dosimetry*. (Cited on page 29.)

Sato, Y., Nakajima, S., Shiraga, N., Atsumi, H., Yoshida, S., Koller, T., Gerig, G., and Kikinis, R. (1998). Three-dimensional multi-scale line filter for segmentation and visualization of curvilinear structures in medical images. *Medical Image Analysis*. (Cited on page 93.)

Saumweber, T., Rohwedder, A., Schleyer, M., Eichler, K., Chen, Y.-c., Aso, Y., Cardona, A., Eschbach, C., Kobler, O., Voigt, A., Durairaja, A., Mancini, N., Zlatic, M., Truman, J. W., Thum, A. S., and Gerber, B. (2018). Functional architecture of reward learning in mushroom body extrinsic neurons of larval Drosophila. *Nature Communications*. (Cited on pages 17 and 168.)

Schaefer, S., McPhail, T., and Warren, J. (2006). Image deformation using moving least squares. In *ACM SIGGRAPH 2006 Papers on - SIGGRAPH '06*. (Cited on page 115.)

Schindelin, J., Arganda-Carreras, I., Frise, E., Kaynig, V., Longair, M., Pietzsch, T., Preibisch, S., Rueden, C., Saalfeld, S., Schmid, B., Tinevez, J. Y., White, D. J., Hartenstein, V., Eliceiri, K., Tomancak, P., and Cardona, A. (2012). Fiji: An open-source platform for biological-image analysis. (Cited on pages 43 and 81.)

Schleyer, M., Saumweber, T., Nahrendorf, W., Fischer, B., von Alpen, D., Pauls, D., Thum, A., and Gerber, B. (2011). A behavior-based circuit model of how outcome expectations organize learned behavior in larval Drosophila. *Learning & Memory*. (Cited on page 8.)

Schneider-Mizell, C. M., Gerhard, S., Longair, M., Kazimiers, T., Li, F., Zwart, M. F., Champion, A., Midgley, F. M., Fetter, R. D., Saalfeld, S., and Cardona, A. (2016). Quantitative neuroanatomy for connectomics in Drosophila. *eLife*. (Cited on pages 8 and 69.)

Schoofs, A., Niederegger, S., van Ooyen, A., Heinzel, H. G., and Spieß, R. (2010). The brain can eat: Establishing the existence of a central pattern generator for feeding in third instar larvae of Drosophila virilis and Drosophila melanogaster. *Journal of Insect Physiology*. (Cited on page 5.)

Schrader, Š. and Merritt, D. J. (2007). Dorsal longitudinal stretch receptor of Drosophila melanogaster larva - Fine structure and maturation. *Arthropod Structure and Development*. (Cited on page 5.)

Scorcioni, R., Polavaram, S., and Ascoli, G. A. (2008). L-Measure: A web-accessible tool for the analysis, comparison and search of digital reconstructions of neuronal morphologies. *Nature Protocols*. (Cited on page 95.)

Sederberg, T. W. and Parry, S. R. (1986). Free-form deformation of solid geometric models. *ACM SIGGRAPH Computer Graphics*. (Cited on page 33.)

Shulman, J. M. and Feany, M. B. (2003). Genetic Modifiers of Tauopathy in Drosophila. *Genetics*. (Cited on page 8.)

Shuvaev, S. A., Lazutkin, A. A., Kedrov, A. V., Anokhin, K. V., Enikolopov, G. N., and Koulakov, A. A. (2017). DALMATIAN: An Algorithm for Automatic Cell Detection and Counting in 3D. *Frontiers in neuroanatomy*, 11:117. (Cited on page 82.)

Silvestri, L., Sacconi, L., and Pavone, F. S. (2013). The connectomics challenge. *Functional Neurology*. (Cited on page 1.)

Skiena, S. S. (2008). *The algorithm design manual: Second edition*. (Cited on pages 120 and 192.)

Solís Montero, A. and Lang, J. (2012). Skeleton pruning by contour approximation and the integer medial axis transform. In *Computers and Graphics (Pergamon)*. (Cited on page 91.)

Sprecher, S. G., Cardona, A., and Hartenstein, V. (2011). The Drosophila larval visual system: high-resolution analysis of a simple visual neuropil. *Dev Biol*. (Cited on page 4.)

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R. (2014). Dropout: A Simple Way to Prevent Neural Networks from Overfitting. *Journal of Machine Learning Research*. (Cited on pages 86 and 111.)

Stocker, R. F. (2008). Design of the larval chemosensory system. (Cited on page 3.)

Studholme, C., Hawkes, D. J., and Hill, D. L. (1998). Normalized entropy measure for multimodality image alignment. In *Proceedings of SPIE*. (Cited on page 49.)

Studholme, C., Hill, D. L. G., and Hawkes, D. J. (1997). Automated three-dimensional registration of magnetic resonance and positron emission tomography brain images by multiresolution optimization of voxel similarity measures. *Medical Physics*. (Cited on page 31.)

Studholme, C., Hill, D. L. G., and Hawkes, D. J. (1999). An overlap invariant entropy measure of 3D medical image alignment. *Pattern Recognition*. (Cited on page 49.)

Sui, D., Wang, K., Chae, J., Zhang, Y., and Zhang, H. (2014). A pipeline for neuron reconstruction based on spatial sliding volume filter seeding. *Computational and Mathematical Methods in Medicine*. (Cited on page 71.)

Thirion, J. P. (1998). Image matching as a diffusion process: An analogy with Maxwell's demons. *Medical Image Analysis*. (Cited on page 30.)

Truman, J. W. (2004). Developmental architecture of adult-specific lineages in the ventral CNS of Drosophila. *Development*. (Cited on page 6.)

Turetken, E., Benmansour, F., and Fua, P. (2012). Automated reconstruction of tree structures using path classifiers and Mixed Integer Programming. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. (Cited on page 71.)

Tweedie, S., Ashburner, M., Falls, K., Leyland, P., McQuilton, P., Marygold, S., Millburn, G., Osumi-Sutherland, D., Schroeder, A., Seal, R., Zhang, H., and FlyBase Consortium (2008). FlyBase: enhancing Drosophila Gene Ontology annotations. *Nucleic Acids Research*. (Cited on page 10.)

Ulyanov, D. (2016). Multicore-TSNE. \url{https://github.com/DmitryUlyanov/Multicore-TSNE}. (Cited on page 142.)

University of Muenster (2002). FlyMove. (Cited on page 2.)

van der Bliek, A. M. and Meyerowrtz, E. M. (1991). Dynamin-like protein encoded by the Drosophila shibire gene associated with vesicular traffic. *Nature*. (Cited on page 7.)

Van Ooyen, A. (2011). Using theoretical models to analyse neural development. (Cited on page 162.)

Vercauteren, T., Pennec, X., Perchant, A., and Ayache, N. (2007). Non-parametric Diffeomorphic Image Registration with the Demons Algorithm. *Medical Image Computing and Computer-Assisted Intervention–MICCAI*. (Cited on page 29.)

Vijendravarma, R. K., Narasimha, S., and Kawecki, T. J. (2013). Predatory cannibalism in Drosophila melanogaster larvae. *Nature Communications*. (Cited on page 3.)

Viswanathan, S., Williams, M. E., Bloss, E. B., Stasevich, T. J., Speer, C. M., Nern, A., Pfeiffer, B. D., Hooks, B. M., Li, W. P., English, B. P., Tian, T., Henry, G. L., Macklin, J. J., Patel, R., Gerfen, C. R., Zhuang, X., Wang, Y., Rubin, G. M., and Looger, L. L. (2015). High-performance probes for light and electron microscopy. *Nature Methods*. (Cited on page 10.)

W3C OWL Working Group (2012). OWL 2 Web Ontology Language Document Overview. (Cited on page 10.)

Wahba, G. (1990). *Spline Models for Observational Data*. (Cited on page 33.)

Wan, Y., Long, F., Qu, L., Xiao, H., Hawrylycz, M., Myers, E. W., and Peng, H. (2015). BlastNeuron for Automated Comparison, Retrieval and Clustering of 3D Neuron Morphologies. *Neuroinformatics*. (Cited on pages 95 and 129.)

Wang, C., Chang, K. C., Somers, G., Virshup, D., Ang, B. T., Tang, C., Yu, F., and Wang, H. (2009). Protein phosphatase 2A regulates self-renewal of Drosophila neural stem cells. *Development*. (Cited on page 3.)

Webb, B. (2006). Validating biorobotic models. (Cited on page 1.)

Wood, D. (2013). *The effects of online and offline Transcranial Magnetic Stimulation on Binocular Rivalry*. Masters, Edinburgh. (Cited on page 29.)

Xiang, Y., Yuan, Q., Vogt, N., Looger, L. L., Jan, L. Y., and Jan, Y. N. (2010). Light-avoidance-mediating photoreceptors tile the Drosophila larval body wall. *Nature*. (Cited on page 4.)

Xiao, H. and Peng, H. (2013). APP2: Automatic tracing of 3D neuron morphology based on hierarchical pruning of a gray-weighted image distance-tree. *Bioinformatics*. (Cited on pages 71 and 91.)

Younossi-Hartenstein, A., Nguyen, B., Shy, D., and Hartenstein, V. (2006). Embryonic origin of the Drosophila brain neuropile. *Journal of Comparative Neurology*. (Cited on page 17.)

Younossi-Hartenstein, A., Salvaterra, P. M., and Hartenstein, V. (2003). Early development of the Drosophila brain: IV. Larval neuropile compartments defined by glial septa. *Journal of Comparative Neurology*. (Cited on pages 4 and 17.)

Yuan, X., Trachtenberg, J. T., Potter, S. M., and Roysam, B. (2009). MDL constrained 3-d grayscale skeletonization algorithm for automated extraction of dendrites and spines from fluorescence confocal images. *Neuroinformatics*. (Cited on page 71.)

Zeng, H. and Sanes, J. R. (2017). Neuronal cell-type classification: Challenges, opportunities and the path forward. (Cited on page 1.)

Zhang, F., Aravanis, A. M., Adamantidis, A., De Lecea, L., and Deisseroth, K. (2007). Circuit-breakers: Optical technologies for probing neural signals and systems. (Cited on page 1.)

Zhang, W., Yan, Z., Li, B., Jan, L. Y., and Jan, Y. N. (2014). Identification of motor neurons and a mechanosensitive sensory neuron in the defecation circuitry of Drosophila larvae. *eLife*. (Cited on page 5.)

Zhao, T., Xie, J., Amat, F., Clack, N., Ahammad, P., Peng, H., Long, F., and Myers, E. (2011). Automated reconstruction of neuronal morphology based on local geometrical and global structural models. *Neuroinformatics*. (Cited on page 71.)

Zheng, Q., Sharf, A., Tagliasacchi, A., Chen, B., Zhang, H., Sheffer, A., and Cohen-Or, D. (2010). Consensus skeleton for non-rigid space-time registration. *Computer Graphics Forum*. (Cited on page 118.)

Zubler, F. and Douglas, R. (2009). A framework for modeling the growth and development of neurons and networks. *Frontiers in computational neuroscience*. (Cited on page 162.)