

UNIVERSITY OF BIRMINGHAM

Applying Automation and Machine Learning to Scanning Transmission Electron Microscopy

by

Alexander James Pattison

A thesis submitted to the University of Birmingham for the degree of DOCTOR OF PHILOSOPHY

Nanoscale Physics Research Laboratory

School of Physics and Astronomy

College of Engineering and Physical Sciences

University of Birmingham

September 2020

UNIVERSITY OF
BIRMINGHAM

University of Birmingham Research Archive

e-theses repository

This unpublished thesis/dissertation is copyright of the author and/or third parties. The intellectual property rights of the author or third parties in respect of this work are as defined by The Copyright Designs and Patents Act 1988 or as modified by any successor legislation.

Any use made of information contained in this thesis/dissertation must be in accordance with that legislation and must be properly acknowledged. Further distribution or reproduction in any format is prohibited without the permission of the copyright holder.

Abstract

This work studies how the benefits of automation and machine learning can be applied to the creation, imaging and image analysis of scanning transmission electron microscopy (STEM) samples. Recrystallised tungsten tips are produced using a semi-automated multi-stage process for use as sample platforms in atomic electron tomography (AET). Two coating techniques are tested to see whether they may be viable methods of reducing sample oxidation. An automated microscope control software framework is presented and demonstrated in three different scenarios: the high-throughput acquisition of CdSe/CdS core-shell nanoparticles, the acquisition of CBED patterns of chiral tellurium nanoparticles and the search for candidate particles for alpha tomography. Finally, machine learning is used to classify the handedness of simulated chiral particles using stereopairs of simulated STEM projections. A 'weak labelling' approach is also demonstrated that takes advantage of the intrinsic nature of chirality to remove the need for manually labelling training datasets.

To my 10-year-old self,

whose grand plan led to all of this.

Acknowledgements

This work would not have been possible without the unwavering personal support of my parents, especially during the final write-up stage, and friends made through teaching, music and fencing. My sanity would have undoubtedly been called further into question without their humour and encouragement and I am grateful to have known them all.

I would like to thank the National Center for Electron Microscopy in Berkeley, California, both for granting me access to the TEAM 0.5 microscope and for the enthusiasm of my many collaborators there. A special mention must be made for Peter Ercius, whose energy and commitment to our collaboration have been invaluable.

I would also like to thank my numerous colleagues in the Nanoscale Physics department at the University of Birmingham, particularly Timothy Poon and Alessandra da Silva, with whom I shared quite a few long evenings operating the TEAM 0.5, and Megan Grose, my partner-in-crime since the first week of our degrees and an excellent friend. Finally, I must also thank all the staff of the department, including my secondary supervisor, Dr Ziyou Li, and, most importantly of all, my primary supervisor, Dr Wolfgang Theis, whose dedication to his work and his students is without equal and without whose insights and guidance I could never have hoped to reach this point.

This work was made possible by funding from the EPSRC.

Contents

List of Figures **x**

List of Tables **xxv**

List of Acronyms and Abbreviations **xxvi**

1 Introduction **1**

2 Scanning transmission electron microscopy (STEM) **4**

2.1 Fundamentals of Electron Microscopy 4

2.2 Types of Electron Microscopy 5

2.2.1 Scanning Electron Microscopy (SEM) 5

2.2.2 Transmission Electron Microscopy (TEM) 6

2.3 Scanning Transmission Electron Microscopy (STEM) 7

2.3.1 Annular Dark Field (ADF) imaging 8

2.4 Aberration correction 9

2.4.1 Chromatic (C_c) aberration 9

2.4.2 Spherical (C_s) aberration 10

2.4.3 Astigmatism 11

2.5 Radiation damage 11

2.6 Electron Tomography (ET) 12

2.6.1 Missing-Wedge Problem 13

3 TEAM 0.5 **15**

3.1 TEAM 0.5 capabilities . . . 15

3.2 Remote Operation Software (Wtip Server and Client) . . . 16

3.2.1 Defocus Slices . . . 18

3.2.2 Stigmation correction . . . 19

3.3 Alpha Tomography . . . 20

3.3.1 Choosing a suitable sample . . . 21

3.3.2 Setting up the experiment . . . 21

4 Artificial neural networks (ANN) **23**

4.1 Artificial Neural Network (ANN) structure . . . 23

4.1.1 Nodes . . . 23

4.1.2 Layers . . . 26

4.2 ANN training . . . 27

4.2.1 Optimisers . . . 29

4.2.2 Overfitting and Underfitting . . . 31

4.3 Convolutional Neural Networks . . . 31

4.3.1 Convolutional Layers . . . 32

4.3.2 Pooling Layers . . . 33

4.3.3 Typical CNN structure . . . 34

4.3.4 Notable image-recognition CNNs . . . 34

5 Creating tungsten tips for electron tomography (ET) **37**

5.1 Introduction . . . 37

5.2 Creating Tungsten Tips . . . 41

5.2.1 Recrystallisation . . . 41

5.2.2 Etching . . . 44

5.2.3 Annealing . . . 47

5.2.4 Sample creation . . . 47

5.2.5 Sample imaging . . . 49

5.2.6 Pucking, Back-etching and Packaging 49
5.3 Bismuth coating 51
5.4 Carbon coating 54
5.4.1 Future work 57
5.5 Summary and Outlook 58

6 Using automated control software to enable high-throughput STEM 60
6.1 Introduction 60
6.2 Software structure 63
6.3 Core functions 63
6.3.1 Registration 64
6.3.2 Focusing 64
6.3.3 Positioning-focusing-registration loop 65
6.4 Automation pipeline and interface 65
6.5 Cadmium selenide/cadmium sulphide core-shell nanoparticles (CdSe/CdS NPs) . . 68
6.5.1 Pipeline implementation 70
6.6 Tellurium nanoparticles (TeNPs) 72
6.6.1 Pipeline implementation 76
6.6.2 Convolutional Neural Network 77
6.7 Alpha tomography 81
6.8 Summary and Outlook 87

7 Using machine learning to determine the handedness of chiral nanoparticles 89
7.1 Introduction 89
7.1.1 Chiral Nanoparticles 92
7.2 Development of CNN for chiral geometry classification 95
7.2.1 Simulation, Augmentation, Pre-Processing and Training 100
7.2.2 Testing 109
7.3 Emulating 'realistic' experiments 118
7.3.1 Varying augmentation levels 122

7.3.2 Varying stereo angle . . . 123
7.4 Weak Labelling . . . 125
7.4.1 Varying weak labelling levels . . . 128
7.4.2 Different image conditions . . . 133
7.5 Interpreting and using model results . . . 137
7.6 Summary and Outlook . . . 138

8 Conclusion **141**

List of References **151**

Appendices **152**

A GUI Overview Panels 4-7 . . . 153
B Backpropagation . . . 154
C Hexagonal packing derivation . . . 154
D 'Realistic' Experiments . . . 155
D.1 Tables for dynamic augmentation [10,90] . . . 155
D.2 Tables for dynamic augmentation [10:10:90] . . . 164
D.3 Tables for single augmentation [10,90] . . . 165
D.4 Tables for single augmentation [10:10:90] . . . 170
E Varying stereo angles . . . 171
E.1 Tables for $n = 3$ . . . 171
E.2 Tables for $n = 4$ . . . 180
E.3 Tables for $n = 5$ . . . 189
F Weak Labelling . . . 198
F.1 Tables for 20% weak labelling, dynamic augmentation [10,90] . . . 198
F.2 Tables for 20% weak labelling, dynamic augmentation [10:10:90] . . . 201
F.3 Tables for 20% weak labelling, single augmentation [10,90] . . . 204
F.4 Tables for 20% weak labelling, single augmentation [10:10:90] . . . 207
F.5 Tables for 10% weak labelling, single augmentation [10,90] . . . 210

F.6 Tables for 5% weak labelling, single augmentation [10,90] 213
F.7 Tables for 0% weak labelling, single augmentation [10,90] 216
G Illustrated Acknowledgements 219

List of Figures

2.1 Comparison of SEM (left), TEM (centre) and STEM (right) lens configurations (CL: Condenser Lens, OL: Objective Lens, IL: Intermediate Lens, PL: Projector Lens, BSE: Backscattered Electron, SE: Secondary Electron). 6

2.2 Ray diagram of a) chromatic (C_c) aberration and b) spherical (C_s) aberration. . . 9

2.3 Comparison of stigmated (red) and non-stigmated (blue) electron beam shapes. . . 11

2.4 a) Path of the electron beam through a sample mounted on a slab-like support at normal incidence. b) Path of the electron beam through a sample mounted on a slab-like support at tilted incidence. 14

3.1 a) Schematic of TEAM 0.5 stage and puck schematic inset (images courtesy of Peter Ercius). Samples are mounted on the pucks (the puck shown is designed to accommodate an etched tungsten tip in the hole), which then slide between two 'chopsticks' courtesy of the grooves in the circumference of the puck. b) Translational and rotational axes of TEAM 0.5 stage (N.B. The direction of the γ axis rotates as the stage is rotated in α). 16

3.2 Connections between Wtip client and server and TEAM 0.5 microscope. 17

3.3 Main GUI window. 1) Server status bar. 2) Image window (with simulated image). 3) Slider range box. 4) Stage position bar. 5) 'Main control' panel with magnification, z stepper movement, defocus slices (see Section 3.2.1) and CBED acquisition controls. 6) Image save controls. 7) Monitor window (See Appendix A for magnified image of panels 4-7). 17

3.4 a) Simulated image of sharp edge. b) Line cut of sharp edge ($q = 54.2$). c) FFT of sharp edge line cut. d) Simulated image of Gaussian-filtered ($\sigma = 50$) sharp edge (with intensity normalised). e) Line cut of Gaussian-filtered sharp edge ($q = 3.0$). f) FFT of Gaussian-filtered edge line cut. 18
3.5 Astigmatism adjustment. a) Original image. b) Stigmation tableau corresponding to (a). c) Stigmation-corrected image. d) Stigmation tableau corresponding to (d). 20

4.1 Common ANN activation functions. 25
4.2 ReLU Function. 26
4.3 Convolution filter (size = 3×3, stride = 1) applied to an image to produce a convolved feature. 32
4.4 Max pooling filter (size = 2×2, stride = 2) applied to an image to produce a pooled feature. 33
4.5 Representation of generic CNN architecture. 34

5.1 Path of the electron beam through a sample mounted on a needle-like support. . . 39
5.2 a) Crystalline faceting of the side of a recrystallised W tip (ADF). b) Crystalline faceting of the side of a recrystallised W tip (BF). c) W tip with apex diameter of 5 nm (ADF). d) W tip with apex diameter of 5 nm (BF). 42
5.3 a) Recrystallisation flange (1) Copper Blocks. 2) Ceramic split-bushes. 3) Wire. 4) Springs.). b) Comparison of recrystallised and unrecrystallised wire. Recrystallised portions of the wire appear silver while the normal wire appears darker. 44
5.4 a) Etching Station. b) Loop Electrode. 45
5.5 Tungsten tips after (top) first-stage etching and (bottom) second-stage etching. . . 46
5.6 Annealing Flange. 1) Heating filament. 2) W tip. 3) Thermocouple wires. 48
5.7 a) Diagram of evaporator and annealing flange setup. b) Picture of evaporator and annealing flange setup. c) Evaporator. (1) Evaporator (two-pin electrical feedthrough). 2) Metal bead (on tungsten filament). 3) W tip (in stainless steel clamp). 4) Annealing flange (two-pin electrical feedthrough with thermocouple).) . 49
5.8 STEM tip sample plate for JEOL 2100-F. 50

5.9 a) Pucking setup (image courtesy of Dr Wolfgang Theis). b) CAD design of shipping container. c) 3D-printed shipping container with chopsticks holder and puck mounted in the clamp. d) Etched tungsten tip mounted on a puck held by chopsticks (image courtesy of Dr Wolfgang Theis). 51
5.10 Bismuth evaporation pocket (mounted between the two pins in the evaporation flange in Figure 5.7c). 52
5.11 Images for Bi removal test (720°C, 40 seconds). a) Rh-coated W tip (ADF). b) Rh-coated W tip (BF). c) Bi-coated Rh-coated W tip (ADF). d) Bi-coated Rh-coated W tip (BF). e) Tip after heating (ADF). f) Tip after heating (BF). 53
5.12 Images for Bi removal test (670°C, 40 seconds). a) Rh-coated W tip (ADF). b) Rh-coated W tip (BF). c) Bi-coated Rh-coated W tip (ADF). d) Bi-coated Rh-coated W tip (BF). e) Tip after heating (ADF). f) Tip after heating (BF). 54
5.13 Images for Bi removal test (530°C, 40 seconds). a) Rh-coated W tip (ADF). b) Rh-coated W tip (BF). c) Bi-coated Rh-coated W tip (ADF). d) Bi-coated Rh-coated W tip (BF). e) Tip after heating (ADF). f) Tip after heating (BF). 55
5.14 Diagram of vacuum chamber setup for carbon evaporation. 1) W tip on rotating flange. 2) Shutter on retractable bellows. 3) Carbon evaporator on retractable bellows. 56
5.15 Initial testing of C evaporation ($V = 1.5$ kV, $I = 150$ mA, d = 18 mm, t = 30 seconds (one side), 60 seconds (other side). a) Pt-coated W tip (ADF). b) Pt-coated W tip (BF). c) C-coated Pt-Coated W tip (ADF). d) C-coated Pt-Coated W tip (BF). e) C-coated Pt-Coated W tip (ADF). f) C-coated Pt-Coated W tip (BF). 57
5.16 C evaporation ($V = 1.5$ kV, $I = 140$ mA, d = 18 mm, t = 30 seconds. a) W tip (ADF). b) W tip (BF). c) C-coated W tip (ADF). d) C-coated W tip (BF). 58

6.1 Software diagram showing connections between automation software (top-left), Wtip client and server and TEAM 0.5 microscope systems. 64
6.2 Positioning-focusing-registration loop flowchart. 66
6.3 Graphical user interface. 67

6.4 Image sequence of CdSe/CdS NP (red box denotes area of subsequent step). The core-shell structure of the nanoparticles is not easily visible due to the low Z value of S and Se compared to Cd. a) 5k× magnification (Step 1: Tiling, 1024×1024), b) 20k× magnification (Step 2: ROI, registration, 1024×1024)), c) 80k× magnification (Step 2: ROI, 1024×1024), d) 160k× magnification (Step 3: ROI, registration, 1024×1024), e) 640k× magnification (Step 3: ROI, 512×512), f) 640k× magnification (Step 4: Record, 2048×2048) (red line denotes line cut used to calculate spacing between particles). 68

6.5 Region of interest selection using layer method. a) Original image (magnification = 40k×. b) Tiling positions (tiling magnification = 640k×, stride = 1.2). c) Tiling positions after passband filtering (lower threshold = 6000, upper threshold = 20000, pass fraction = 50%). 69

6.6 512×512 cutout of 2048×2048 image of CdSe/CdS NPs 71

6.7 Image sequence for TeNPs (red box denotes area of subsequent step). a) 5k× magnification (Step 1: Tiling, 512×512), b) 20k× magnification (Step 2: ROI), c) CBED pattern (camera length = 4.3 cm), d) CBED pattern (camera length = 6.8 cm), e) CBED pattern (camera length = 13.5 cm). 73

6.8 Laplacian-of-Gaussian-based particle detection method. a) Original image. b) Peaks identified by Laplacian-of-Gaussian blob detection. c) Flood-filled image. d) Identified individual particles. 74

6.9 Custom particle detection method. a) Original image. b) Feature map generated by ring-shaped maximum filter. c) Binarised image. d) Binarised feature map. e) Product of binarised image and feature map multiplied by original image. f) Detected particles in original image. 75

6.10 Comparison of LoG and custom particle detection methods. a) Original image. b) Peaks detected by LoG-based method. c) Peaks detected by custom method. . . . 75

6.11 Example images of TeNPs used to train the CNN. a) 'Suitable' TeNP. b) 'Unsuitable' TeNP. 77

6.12 Magnification series of a single (manually selected) NiPt nanoparticle. a) 5k× magnification, b) 10k× magnification, c) 40k× magnification, d) 160k× magnification, e) 1.3M× magnification, f) 1.8M× magnification. 81
6.13 Steps for the first stage of the alpha tomography particle-searching program (5k× magnification). a) Original image with left- and right-most points of each large agglomerate denoted by stars and circles, respectively. b) Positions for higher magnification images placed around each of the previously identified points (grid size = 2×3, overlap = 20%). c) Final positions for higher magnification images after intensity and distancing filters applied. 82
6.14 Particle-finding algorithm results for the second stage of the alpha tomography particle-searching program (40k× magnification). a) Original image. b) Particle-finding using custom method. c) Clearance-filtered peaks from b). d) Particle-finding using LoG method. e) Clearance-filtered peaks from d). (Arrows identify clusters of particles misidentified as single particles.). f)-h) Expanded regions of interest designated in a). 84
6.15 Clearance checking using the flood-filling algorithm. a) Original image with cutouts in b) and c) identified by boxes. b) Cutout of top-left particle in a). c) Binarised copy of a). d) Flood-filled copy of b), showing that the vertical clearance box is not clear and hence the particle is unsuitable for tomography. e) Cutout of centre particle in a). f) Binarised copy of e). g) Flood-filled copy of f), showing that the vertical clearance box is clear (aside from the particle) and hence the particle is suitable for tomography. 85

7.1 SEM images of tellurium nanoparticles. Sample created by Dr Assaf Ben-Moshe, Alivisatos group, University of California, Berkeley. Images courtesy of Alexander Mueller, National Center for Electron Microscopy. 94
7.2 Steps for simulating chiral geometries. a) Opposite-pointing pyramids offset by +15.0°. b) Chiral particle with chiral twist = +15.0°. c) Opposite-pointing pyramids offset by -15.0°. b) Chiral particle with chiral twist = -15.0°. 96

7.3 Simulated STEM projections of particles with a) 3-fold symmetry and b) 4-fold symmetry. 97
7.4 Simulated STEM projections of particles with chiral twists of a) +45° and b) -45°. 97
7.5 a) and b) Stereopair of chiral particles with chiral twist = +22.5° at stereo angles a) 0° and b) 10°. c) and d) Simulated STEM projections of a) and b), respectively. e) and f) Stereopair of chiral particles with chiral twist = -22.5° at stereo angles e) 0° and f) 10°. g) and h) Simulated STEM projections of e) and f), respectively. . . 98
7.6 a) and b) Simulated STEM stereopair of vertically aligned four-fold symmetry chiral particle (chiral twist = 22.5°, z-rotation = 0°) at stereo angles a) 0° and b) 90°. c) and d) Simulated STEM stereopair of diagonally aligned four-fold symmetry chiral particle (chiral twist = 22.5°, z-rotation = 45°) at stereo angles a) 0° and b) 90°. . 99
7.7 Comparison of stereopairs with stereo angle = $360°/2n$. a) and b) Stereopair of chiral particles (n = 4) with chiral twist = 22.5° and z-rotation = 0° (a) stereo rotation = 0°, b) stereo rotation = 45°). c) and d) Stereopair of chiral particles (n = 4) with chiral twist = -22.5° and z-rotation = 0° (c) stereo rotation = 180°, d) stereo rotation = 225°). e) and f) Stereopair of chiral particles (n = 4) with chiral twist = 22.5° and z-rotation = 45° (e) stereo rotation = 0°, f) stereo rotation = 45°). g) and h) Stereopair of chiral particles (n = 4) with chiral twist = -22.5° and z-rotation = 45° (g) stereo rotation = 180°, h) stereo rotation = 225°). 101
7.8 Rotation axes for the particle. These axes are stationary and do not rotate with the particle. 102
7.9 a) Simulation of chiral particle structure (height = 1.0, radius = 0.5, chiral twist angle = 22.5°, n-fold symmetry = 4, scale = 0.8, y-rotation = 10°, x-rotation = 31°, z-rotation = 10°). b) STEM image simulation with no shot noise or Gaussian blur. c) STEM image simulation with no shot noise and Gaussian blur = 0.5 pixels. d) STEM image simulation with shot noise (Poisson noise coefficients a = 10.0 and b = 0.1) and Gaussian blur = 0.5 pixels. Scale bars are an approximate representation of intended scale. 104

7.10 Effect of augmentations. Top row, left to right: original image, brightness +20%, contrast +20%, x-axis rescale +20%. Bottom row, left to right: x-shear +10%, up-down mirroring, left-right mirroring, random shift (x and y axis). 106
7.11 1D density projections of two enantiomers of a 2D chiral structure (black) at stereopair positions a, b and a, b + 180°. The colour of the block roughly corresponds to the projected density (blue = low density, green = medium density, yellow = high density). 106
7.12 Pre-Processing Steps. Left to right: original image, background subtraction and normalisation, centre of intensity alignment, mean pooling (2×2). 108
7.13 Graph of training accuracy vs number of epochs (Taken from Section 7.2.2.3, initial training, 111 network). 110
7.14 Flowchart of single and dynamic augmentation processes with size of dataset at each stage in brackets (for the case of 500 original simulated stereopairs). 119
7.15 Comparison of average test accuracies of each network configuration for a) dynamic augmentation [10,90], b) dynamic augmentation [10:10:90], c) single augmentation [10,90] and d) single augmentation [10:10:90]. 120
7.16 Comparison of average test accuracies of each augmentation type for a) 211 networks, b) 121 networks, c) 112 networks and d) 111 networks. 121
7.17 Comparison of number of successfully trained networks of each configuration using a) dynamic augmentation [10,90], b) dynamic augmentation [10:10:90], c) single augmentation [10,90] and d) single augmentation [10:10:90]. 122
7.18 Average test accuracy of trained networks vs augmentation levels. 123
7.19 Average test accuracy vs stereo angle for a) $n = 3$, chiral twist = 15.0°, b) $n = 3$, chiral twist = 22.5°, c) $n = 3$, chiral twist = 30.0°, d) $n = 4$, chiral twist = 15.0°, e) $n = 4$, chiral twist = 22.5°, f) $n = 4$, chiral twist = 30.0°, g) $n = 5$, chiral twist = 15.0°, h) $n = 5$, chiral twist = 22.5° and i) $n = 5$, chiral twist = 30.0°. 124

7.20 Comparison of STEM simulations of chiral particles with different symmetries and resolutions. a) 3-fold symmetry, 512×512 resolution. b) 4-fold symmetry, 512×512 resolution. c) 5-fold symmetry, 512×512 resolution. d) 3-fold symmetry, 64×64 resolution. e) 4-fold symmetry, 64×64 resolution. f) 5-fold symmetry, 64×64 resolution. 126

7.21 Number of successfully trained networks vs number of original stereopairs (20% weak labelling) for a) dynamic augmentation [10,90], b) dynamic augmentation [10:10:90], c) single augmentation [10,90] and d) single augmentation [10:10:90]. 129

7.22 Average final training (a-d) and test (e-h) accuracy of trained networks vs number of original stereopairs (20% weak labelling). a) dynamic augmentation [10,90] training accuracy. b) dynamic augmentation [10:10:90] training accuracy. c) single augmentation [10,90] training accuracy. d) single augmentation [10:10:90] training accuracy. e) dynamic augmentation [10,90] test accuracy. f) dynamic augmentation [10:10:90] test accuracy. g) single augmentation [10,90] test accuracy. h) single augmentation [10:10:90] test accuracy. 130

7.23 Average majority (a-d) and minority (e-h) test accuracy of trained networks vs number of original stereopairs (20% weak labelling). a) dynamic augmentation [10,90] majority test accuracy. b) dynamic augmentation [10:10:90] majority test accuracy. c) single augmentation [10,90] majority test accuracy. d) single augmentation [10:10:90] majority test accuracy. e) dynamic augmentation [10,90] minority test accuracy. f) dynamic augmentation [10:10:90] minority test accuracy. g) single augmentation [10,90] minority test accuracy. h) single augmentation [10:10:90] minority test accuracy. 131

7.24 Average test accuracy of trained networks vs number of original stereopairs for a) 20% weak labelling, b) 10% weak labelling, c) 5% weak labelling and d) 0% weak labelling. 132

7.25 Average test accuracy of trained networks vs number of original stereopairs for a) 0% weak labelling and b) 'normal' method (100% correctly labelled data). 132

7.26 Simulated particles (height = 1.0, radius = 0.5, $n = 4$, chiral twist = 22.5°, stereo angle = 10°, scale = 0.8, y-rotation = 0°, x-rotation = 0°, z-rotation = 0°) with sloped background (left to right). a) and b) Original stereopair. c) and d) Original stereopair mirrored along stereo axis with ordering reversed. 134

7.27 Simulated particles (height = 1.0, radius = 0.5, $n = 4$, chiral twist = 22.5°, stereo angle = 10°, scale = 0.8, y-rotation = 0°, x-rotation = 0°, z-rotation = 0°) with sloped background (top to bottom). a) and b) Original stereopair. c) and d) Original stereopair mirrored perpendicular to the stereo axis with ordering maintained. . . 137

7.28 Stereopair of TeNP taken at a) $\alpha = 0°$ and b) $\alpha = 10°$. N.B. The images have been rotated by 90° from their original positions so that the stereo axis is consistent with the other examples given in this chapter (i.e vertical). 140

8.1 Panels 4-7 in Figure 3.3. 4) Stage position bar. 5) 'Main control' panel with magnification, z stepper movement, defocus slices (see Section 3.2.1) and CBED acquisition controls. 6) Image save controls. 7) Monitor window. 153

8.2 Illustrated acknowledgements. Text by author, illustrations courtesy of Dr Michael Pattison. 219

List of Tables

7.1 Results of training with random starting y-rotation. 111

7.2 Test accuracies of networks trained with random starting y-rotation. 112

7.3 Results of training with random shapes and sizes. 112

7.4 Test accuracies of networks trained with random shapes and sizes. 113

7.5 Results of initial training with random y- and x-rotation. 114

7.6 Results of retraining with random z-rotation. 114

7.7 Test accuracies of networks trained with random z-rotation. 114

7.8 Results of initial training with pixel noise. 115

7.9 Results of retraining with pixel noise. 115

7.10 Test accuracies of networks trained with pixel noise. 115

7.11 Results of initial augmented training (500 original simulated stereopairs, 40× augmentation). 116

7.12 Results of augmented retraining (500 original simulated stereopairs, 40× augmentation). 116

7.13 Test accuracies of networks trained on augmented data (500 original simulated stereopairs, 40× augmentation). 116

7.14 Results of multiplicative initial training (500 original simulated stereopairs, 40× multiplication). 117

7.15 Results of multiplicative retraining (500 original simulated stereopairs, 40× multiplication) 117

7.16 Test accuracies of networks trained on multiplied data (500 original simulated stereopairs, 40× multiplication). 117
7.17 Validation data results for 111 networks trained on 500 original simulated stereopairs, 40× augmentation, then retrained on 20000 original simulated stereopairs, 1× augmentation. 118
7.18 Test accuracies of 20% weak labelling, single augmentation training (1000 original simulated stereopairs, 10× augmentation) [10,90] with left-to-right sloped backgrounds in the second images of the original stereopairs. Stereopair reordering and left-to-right mirroring used in formation of minority handedness dataset. 133
7.19 Test accuracies of 20% weak labelling, single augmentation training (1000 original simulated stereopairs, 10× augmentation) [10,90] with left-to-right sloped backgrounds in the second images of the original stereopairs. Stereopair reordering only used in formation of minority handedness dataset. 136

8.1 Results of initial dynamic augmentation training (125 original simulated stereopairs, 160× augmentation) 155
8.2 Results of dynamic augmentation retraining (125 original simulated stereopairs, 160× augmentation) 155
8.3 Test accuracies of dynamic augmentation training (125 original simulated stereopairs, 160× augmentation). 155
8.4 Results of initial dynamic augmentation training (250 original simulated stereopairs, 80× augmentation) 156
8.5 Results of dynamic augmentation retraining (250 original simulated stereopairs, 80× augmentation) 156
8.6 Test accuracies of dynamic augmentation training (250 original simulated stereopairs, 80× augmentation). 156
8.7 Results of initial dynamic augmentation training (500 original simulated stereopairs, 40× augmentation) 157

8.8 Results of dynamic augmentation retraining (500 original simulated stereopairs, 40× augmentation) . . . 157
8.9 Test accuracies of dynamic augmentation training (500 original simulated stereopairs, 40× augmentation). . . . 157
8.10 Results of initial dynamic augmentation training (1000 original simulated stereopairs, 20× augmentation) . . . 158
8.11 Results of dynamic augmentation retraining (1000 original simulated stereopairs, 20× augmentation) . . . 158
8.12 Test accuracies of dynamic augmentation training (1000 original simulated stereopairs, 20× augmentation). . . . 158
8.13 Results of initial dynamic augmentation training (20000 original simulated stereopairs, 1× augmentation) . . . 159
8.14 Results of dynamic augmentation retraining (20000 original simulated stereopairs, 10× augmentation) . . . 159
8.15 Test accuracies of dynamic augmentation training (20000 original simulated stereopairs, 1× augmentation). . . . 159
8.16 Results of initial dynamic augmentation training (250 original simulated stereopairs, 20× augmentation) . . . 160
8.17 Results of dynamic augmentation retraining (250 original simulated stereopairs, 20× augmentation) . . . 160
8.18 Test accuracies of dynamic augmentation training (250 original simulated stereopairs, 20× augmentation). . . . 160
8.19 Results of initial dynamic augmentation training (250 original simulated stereopairs, 40× augmentation) . . . 161
8.20 Results of dynamic augmentation retraining (250 original simulated stereopairs, 40× augmentation) . . . 161
8.21 Test accuracies of dynamic augmentation training (250 original simulated stereopairs, 40× augmentation). . . . 161

8.22 Results of initial dynamic augmentation training (250 original simulated stereopairs, 160× augmentation) 162
8.23 Results of dynamic augmentation retraining (250 original simulated stereopairs, 160× augmentation) 162
8.24 Test accuracies of dynamic augmentation training (250 original simulated stereopairs, 160× augmentation). 162
8.25 Results of initial dynamic augmentation training (250 original simulated stereopairs, 320× augmentation) 163
8.26 Results of dynamic augmentation retraining (250 original simulated stereopairs, 320× augmentation) 163
8.27 Test accuracies of dynamic augmentation training (250 original simulated stereopairs, 320× augmentation). 163
8.28 Test accuracies of dynamic augmentation training (125 original simulated stereopairs, 160× augmentation) [10:10:90] 164
8.29 Test accuracies of dynamic augmentation training (250 original simulated stereopairs, 80× augmentation) [10:10:90] 164
8.30 Test accuracies of dynamic augmentation training (500 original simulated stereopairs, 40× augmentation) [10:10:90] 164
8.31 Results of dynamic augmentation training (1000 original simulated stereopairs, 20× augmentation) [10:10:90] 165
8.32 Results of dynamic augmentation training (20000 original simulated stereopairs, 1× augmentation) [10:10:90] 165
8.33 Results of initial single augmentation training (125 original simulated stereopairs, 160× augmentation) 165
8.34 Results of single augmentation retraining (125 original simulated stereopairs, 160× augmentation) 166
8.35 Test accuracies of single augmentation training (125 original simulated stereopairs, 160× augmentation). 166

8.36 Results of initial single augmentation training (250 original simulated stereopairs, 80× augmentation) . . . 166
8.37 Results of single augmentation retraining (250 original simulated stereopairs, 80× augmentation) . . . 166
8.38 Test accuracies of single augmentation training (250 original simulated stereopairs, 80× augmentation). . . . 167
8.39 Results of initial single augmentation training (500 original simulated stereopairs, 40× augmentation) . . . 167
8.40 Results of single augmentation retraining (500 original simulated stereopairs, 40× augmentation) . . . 167
8.41 Test accuracies of single augmentation training (500 original simulated stereopairs, 40× augmentation). . . . 167
8.42 Results of initial single augmentation training (1000 original simulated stereopairs, 20× augmentation) . . . 168
8.43 Results of single augmentation retraining (1000 original simulated stereopairs, 20× augmentation) . . . 168
8.44 Test accuracies of single augmentation training (1000 original simulated stereopairs, 20× augmentation). . . . 168
8.45 Results of initial single augmentation training (20000 original simulated stereopairs, 1× augmentation) . . . 169
8.46 Results of single augmentation retraining (20000 original simulated stereopairs, 1× augmentation) . . . 169
8.47 Test accuracies of single augmentation training (20000 original simulated stereopairs, 1× augmentation). . . . 169
8.48 Test accuracies of single augmentation training (125 original simulated stereopairs, 160× augmentation) [10:10:90] . . . 170
8.49 Test accuracies of single augmentation training (250 original simulated stereopairs, 80× augmentation) [10:10:90] . . . 170

8.50 Test accuracies of single augmentation training (500 original simulated stereopairs, 40× augmentation) [10:10:90] 170
8.51 Test accuracies of single augmentation training (1000 original simulated stereopairs, 20× augmentation) [10:10:90] 171
8.52 Test accuracies of single augmentation training (20000 original simulated stereopairs, 1× augmentation) [10:10:90] 171
8.53 Test data results for $n = 3$, chiral twist = 15.0°, stereo angle = 5° (dynamic augmentation, [10,90], 250 original simulated stereopairs, 80× augmentation 171
8.54 Test data results for $n = 3$, chiral twist = 15.0°, stereo angle = 10° (dynamic augmentation, [10,90], 250 original simulated stereopairs, 80× augmentation 172
8.55 Test data results for $n = 3$, chiral twist = 15.0°, stereo angle = 15° (dynamic augmentation, [10,90], 250 original simulated stereopairs, 80× augmentation 172
8.56 Test data results for $n = 3$, chiral twist = 15.0°, stereo angle = 20° (dynamic augmentation, [10,90], 250 original simulated stereopairs, 80× augmentation 172
8.57 Test data results for $n = 3$, chiral twist = 15.0°, stereo angle = 25° (dynamic augmentation, [10,90], 250 original simulated stereopairs, 80× augmentation 173
8.58 Test data results for $n = 3$, chiral twist = 15.0°, stereo angle = 30° (dynamic augmentation, [10,90], 250 original simulated stereopairs, 80× augmentation 173
8.59 Test data results for $n = 3$, chiral twist = 15.0°, stereo angle = 35° (dynamic augmentation, [10,90], 250 original simulated stereopairs, 80× augmentation 173
8.60 Test data results for $n = 3$, chiral twist = 15.0°, stereo angle = 40° (dynamic augmentation, [10,90], 250 original simulated stereopairs, 80× augmentation 174
8.61 Test data results for $n = 3$, chiral twist = 15.0°, stereo angle = 45° (dynamic augmentation, [10,90], 250 original simulated stereopairs, 80× augmentation 174
8.62 Test data results for $n = 3$, chiral twist = 22.5°, stereo angle = 5° (dynamic augmentation, [10,90], 250 original simulated stereopairs, 80× augmentation 174
8.63 Test data results for $n = 3$, chiral twist = 22.5°, stereo angle = 10° (dynamic augmentation, [10,90], 250 original simulated stereopairs, 80× augmentation 175

8.64 Test data results for $n = 3$, chiral twist = 22.5°, stereo angle = 15° (dynamic augmentation, [10,90], 250 original simulated stereopairs, 80× augmentation 175
8.65 Test data results for $n = 3$, chiral twist = 22.5°, stereo angle = 20° (dynamic augmentation, [10,90], 250 original simulated stereopairs, 80× augmentation 175
8.66 Test data results for $n = 3$, chiral twist = 22.5°, stereo angle = 25° (dynamic augmentation, [10,90], 250 original simulated stereopairs, 80× augmentation 176
8.67 Test data results for $n = 3$, chiral twist = 22.5°, stereo angle = 30° (dynamic augmentation, [10,90], 250 original simulated stereopairs, 80× augmentation 176
8.68 Test data results for $n = 3$, chiral twist = 22.5°, stereo angle = 35° (dynamic augmentation, [10,90], 250 original simulated stereopairs, 80× augmentation 176
8.69 Test data results for $n = 3$, chiral twist = 22.5°, stereo angle = 40° (dynamic augmentation, [10,90], 250 original simulated stereopairs, 80× augmentation 177
8.70 Test data results for $n = 3$, chiral twist = 22.5°, stereo angle = 45° (dynamic augmentation, [10,90], 250 original simulated stereopairs, 80× augmentation 177
8.71 Test data results for $n = 3$, chiral twist = 30.0°, stereo angle = 5° (dynamic augmentation, [10,90], 250 original simulated stereopairs, 80× augmentation 177
8.72 Test data results for $n = 3$, chiral twist = 30.0°, stereo angle = 10° (dynamic augmentation, [10,90], 250 original simulated stereopairs, 80× augmentation 178
8.73 Test data results for $n = 3$, chiral twist = 30.0°, stereo angle = 15° (dynamic augmentation, [10,90], 250 original simulated stereopairs, 80× augmentation 178
8.74 Test data results for $n = 3$, chiral twist = 30.0°, stereo angle = 20° (dynamic augmentation, [10,90], 250 original simulated stereopairs, 80× augmentation 178
8.75 Test data results for $n = 3$, chiral twist = 30.0°, stereo angle = 25° (dynamic augmentation, [10,90], 250 original simulated stereopairs, 80× augmentation 179
8.76 Test data results for $n = 3$, chiral twist = 30.0°, stereo angle = 30° (dynamic augmentation, [10,90], 250 original simulated stereopairs, 80× augmentation 179
8.77 Test data results for $n = 3$, chiral twist = 30.0°, stereo angle = 35° (dynamic augmentation, [10,90], 250 original simulated stereopairs, 80× augmentation 179

8.78 Test data results for $n = 3$, chiral twist = 30.0°, stereo angle = 40° (dynamic augmentation, [10,90], 250 original simulated stereopairs, 80× augmentation 180
8.79 Test data results for $n = 3$, chiral twist = 30.0°, stereo angle = 45° (dynamic augmentation, [10,90], 250 original simulated stereopairs, 80× augmentation 180
8.80 Test data results for $n = 4$, chiral twist = 15.0°, stereo angle = 5° (dynamic augmentation, [10,90], 250 original simulated stereopairs, 80× augmentation 180
8.81 Test data results for $n = 4$, chiral twist = 15.0°, stereo angle = 10° (dynamic augmentation, [10,90], 250 original simulated stereopairs, 80× augmentation 181
8.82 Test data results for $n = 4$, chiral twist = 15.0°, stereo angle = 15° (dynamic augmentation, [10,90], 250 original simulated stereopairs, 80× augmentation 181
8.83 Test data results for $n = 4$, chiral twist = 15.0°, stereo angle = 20° (dynamic augmentation, [10,90], 250 original simulated stereopairs, 80× augmentation 181
8.84 Test data results for $n = 4$, chiral twist = 15.0°, stereo angle = 25° (dynamic augmentation, [10,90], 250 original simulated stereopairs, 80× augmentation 182
8.85 Test data results for $n = 4$, chiral twist = 15.0°, stereo angle = 30° (dynamic augmentation, [10,90], 250 original simulated stereopairs, 80× augmentation 182
8.86 Test data results for $n = 4$, chiral twist = 15.0°, stereo angle = 35° (dynamic augmentation, [10,90], 250 original simulated stereopairs, 80× augmentation 182
8.87 Test data results for $n = 4$, chiral twist = 15.0°, stereo angle = 40° (dynamic augmentation, [10,90], 250 original simulated stereopairs, 80× augmentation 183
8.88 Test data results for $n = 4$, chiral twist = 15.0°, stereo angle = 45° (dynamic augmentation, [10,90], 250 original simulated stereopairs, 80× augmentation 183
8.89 Test data results for $n = 4$, chiral twist = 22.5°, stereo angle = 5° (dynamic augmentation, [10,90], 250 original simulated stereopairs, 80× augmentation 183
8.90 Test data results for $n = 4$, chiral twist = 22.5°, stereo angle = 10° (dynamic augmentation, [10,90], 250 original simulated stereopairs, 80× augmentation 184
8.91 Test data results for $n = 4$, chiral twist = 22.5°, stereo angle = 15° (dynamic augmentation, [10,90], 250 original simulated stereopairs, 80× augmentation 184

8.92 Test data results for $n = 4$, chiral twist = 22.5°, stereo angle = 20° (dynamic augmentation, [10,90], 250 original simulated stereopairs, 80× augmentation 184
8.93 Test data results for $n = 4$, chiral twist = 22.5°, stereo angle = 25° (dynamic augmentation, [10,90], 250 original simulated stereopairs, 80× augmentation 185
8.94 Test data results for $n = 4$, chiral twist = 22.5°, stereo angle = 30° (dynamic augmentation, [10,90], 250 original simulated stereopairs, 80× augmentation 185
8.95 Test data results for $n = 4$, chiral twist = 22.5°, stereo angle = 35° (dynamic augmentation, [10,90], 250 original simulated stereopairs, 80× augmentation 185
8.96 Test data results for $n = 4$, chiral twist = 22.5°, stereo angle = 40° (dynamic augmentation, [10,90], 250 original simulated stereopairs, 80× augmentation 186
8.97 Test data results for $n = 4$, chiral twist = 22.5°, stereo angle = 45° (dynamic augmentation, [10,90], 250 original simulated stereopairs, 80× augmentation 186
8.98 Test data results for $n = 4$, chiral twist = 30.0°, stereo angle = 5° (dynamic augmentation, [10,90], 250 original simulated stereopairs, 80× augmentation 186
8.99 Test data results for $n = 4$, chiral twist = 30.0°, stereo angle = 10° (dynamic augmentation, [10,90], 250 original simulated stereopairs, 80× augmentation 187
8.100 Test data results for $n = 4$, chiral twist = 30.0°, stereo angle = 15° (dynamic augmentation, [10,90], 250 original simulated stereopairs, 80× augmentation 187
8.101 Test data results for $n = 4$, chiral twist = 30.0°, stereo angle = 20° (dynamic augmentation, [10,90], 250 original simulated stereopairs, 80× augmentation 187
8.102 Test data results for $n = 4$, chiral twist = 30.0°, stereo angle = 25° (dynamic augmentation, [10,90], 250 original simulated stereopairs, 80× augmentation 188
8.103 Test data results for $n = 4$, chiral twist = 30.0°, stereo angle = 30° (dynamic augmentation, [10,90], 250 original simulated stereopairs, 80× augmentation 188
8.104 Test data results for $n = 4$, chiral twist = 30.0°, stereo angle = 35° (dynamic augmentation, [10,90], 250 original simulated stereopairs, 80× augmentation 188
8.105 Test data results for $n = 4$, chiral twist = 30.0°, stereo angle = 40° (dynamic augmentation, [10,90], 250 original simulated stereopairs, 80× augmentation 189

8.106Test data results for $n = 4$, chiral twist = 30.0°, stereo angle = 45° (dynamic augmentation, [10,90], 250 original simulated stereopairs, 80× augmentation 189
8.107Test data results for $n = 5$, chiral twist = 15.0°, stereo angle = 5° (dynamic augmentation, [10,90], 250 original simulated stereopairs, 80× augmentation 189
8.108Test data results for $n = 5$, chiral twist = 15.0°, stereo angle = 10° (dynamic augmentation, [10,90], 250 original simulated stereopairs, 80× augmentation 190
8.109Test data results for $n = 5$, chiral twist = 15.0°, stereo angle = 15° (dynamic augmentation, [10,90], 250 original simulated stereopairs, 80× augmentation 190
8.110Test data results for $n = 5$, chiral twist = 15.0°, stereo angle = 20° (dynamic augmentation, [10,90], 250 original simulated stereopairs, 80× augmentation 190
8.111Test data results for $n = 5$, chiral twist = 15.0°, stereo angle = 25° (dynamic augmentation, [10,90], 250 original simulated stereopairs, 80× augmentation 191
8.112Test data results for $n = 5$, chiral twist = 15.0°, stereo angle = 30° (dynamic augmentation, [10,90], 250 original simulated stereopairs, 80× augmentation 191
8.113Test data results for $n = 5$, chiral twist = 15.0°, stereo angle = 35° (dynamic augmentation, [10,90], 250 original simulated stereopairs, 80× augmentation 191
8.114Test data results for $n = 5$, chiral twist = 15.0°, stereo angle = 40° (dynamic augmentation, [10,90], 250 original simulated stereopairs, 80× augmentation 192
8.115Test data results for $n = 5$, chiral twist = 15.0°, stereo angle = 45° (dynamic augmentation, [10,90], 250 original simulated stereopairs, 80× augmentation 192
8.116Test data results for $n = 5$, chiral twist = 22.5°, stereo angle = 5° (dynamic augmentation, [10,90], 250 original simulated stereopairs, 80× augmentation 192
8.117Test data results for $n = 5$, chiral twist = 22.5°, stereo angle = 10° (dynamic augmentation, [10,90], 250 original simulated stereopairs, 80× augmentation 193
8.118Test data results for $n = 5$, chiral twist = 22.5°, stereo angle = 15° (dynamic augmentation, [10,90], 250 original simulated stereopairs, 80× augmentation 193
8.119Test data results for $n = 5$, chiral twist = 22.5°, stereo angle = 20° (dynamic augmentation, [10,90], 250 original simulated stereopairs, 80× augmentation 193

8.120Test data results for $n = 5$, chiral twist = 22.5°, stereo angle = 25° (dynamic augmentation, [10,90], 250 original simulated stereopairs, 80× augmentation 194
8.121Test data results for $n = 5$, chiral twist = 22.5°, stereo angle = 30° (dynamic augmentation, [10,90], 250 original simulated stereopairs, 80× augmentation 194
8.122Test data results for $n = 5$, chiral twist = 22.5°, stereo angle = 35° (dynamic augmentation, [10,90], 250 original simulated stereopairs, 80× augmentation 194
8.123Test data results for $n = 5$, chiral twist = 22.5°, stereo angle = 40° (dynamic augmentation, [10,90], 250 original simulated stereopairs, 80× augmentation 195
8.124Test data results for $n = 5$, chiral twist = 22.5°, stereo angle = 45° (dynamic augmentation, [10,90], 250 original simulated stereopairs, 80× augmentation 195
8.125Test data results for $n = 5$, chiral twist = 30.0°, stereo angle = 5° (dynamic augmentation, [10,90], 250 original simulated stereopairs, 80× augmentation 195
8.126Test data results for $n = 5$, chiral twist = 30.0°, stereo angle = 10° (dynamic augmentation, [10,90], 250 original simulated stereopairs, 80× augmentation 196
8.127Test data results for $n = 5$, chiral twist = 30.0°, stereo angle = 15° (dynamic augmentation, [10,90], 250 original simulated stereopairs, 80× augmentation 196
8.128Test data results for $n = 5$, chiral twist = 30.0°, stereo angle = 20° (dynamic augmentation, [10,90], 250 original simulated stereopairs, 80× augmentation 196
8.129Test data results for $n = 5$, chiral twist = 30.0°, stereo angle = 25° (dynamic augmentation, [10,90], 250 original simulated stereopairs, 80× augmentation 197
8.130Test data results for $n = 5$, chiral twist = 30.0°, stereo angle = 30° (dynamic augmentation, [10,90], 250 original simulated stereopairs, 80× augmentation 197
8.131Test data results for $n = 5$, chiral twist = 30.0°, stereo angle = 35° (dynamic augmentation, [10,90], 250 original simulated stereopairs, 80× augmentation 197
8.132Test data results for $n = 5$, chiral twist = 30.0°, stereo angle = 40° (dynamic augmentation, [10,90], 250 original simulated stereopairs, 80× augmentation 198
8.133Test data results for $n = 5$, chiral twist = 30.0°, stereo angle = 45° (dynamic augmentation, [10,90], 250 original simulated stereopairs, 80× augmentation 198

8.134Test accuracies of 20% weak labelling, dynamic augmentation training (125 original simulated stereopairs, 80× augmentation) [10,90] 198
8.135Test accuracies of 20% , dynamic augmentation training (250 original simulated stereopairs, 40× augmentation) [10,90] 199
8.136Test accuracies of 20% weak labelling, dynamic augmentation training (500 original simulated stereopairs, 20× augmentation) [10,90] 199
8.137Test accuracies of 20% weak labelling, dynamic augmentation training (1000 original simulated stereopairs, 10× augmentation) [10,90] 199
8.138Test accuracies of 20% weak labelling, dynamic augmentation training (2000 original simulated stereopairs, 5× augmentation) [10,90] 200
8.139Test accuracies of 20% weak labelling, dynamic augmentation training (5000 original simulated stereopairs, 2× augmentation) [10,90] 200
8.140Test accuracies of 20% weak labelling, dynamic augmentation training (10000 original simulated stereopairs, 1× augmentation) [10,90] 200
8.141Test accuracies of 20% weak labelling, dynamic augmentation training (125 original simulated stereopairs, 80× augmentation) [10:10:90] 201
8.142Test accuracies of 20% weak labelling, dynamic augmentation training (250 original simulated stereopairs, 40× augmentation) [10:10:90] 201
8.143Test accuracies of 20% weak labelling, dynamic augmentation training (500 original simulated stereopairs, 20× augmentation) [10:10:90] 202
8.144Test accuracies of 20% weak labelling, dynamic augmentation training (1000 original simulated stereopairs, 10× augmentation) [10:10:90] 202
8.145Test accuracies of 20% weak labelling, dynamic augmentation training (2000 original simulated stereopairs, 5× augmentation) [10:10:90] 202
8.146Test accuracies of 20% weak labelling, dynamic augmentation training (5000 original simulated stereopairs, 2× augmentation) [10:10:90] 203
8.147Test accuracies of 20% weak labelling, dynamic augmentation training (10000 original simulated stereopairs, 1× augmentation) [10:10:90] 203

8.148Test accuracies of 20% weak labelling, single augmentation training (125 original simulated stereopairs, 80× augmentation) [10,90] 204
8.149Test accuracies of 20% weak labelling, single augmentation training (250 original simulated stereopairs, 40× augmentation) [10,90] 204
8.150Test accuracies of 20% weak labelling, single augmentation training (500 original simulated stereopairs, 20× augmentation) [10,90] 205
8.151Test accuracies of 20% weak labelling, single augmentation training (1000 original simulated stereopairs, 10× augmentation) [10,90] 205
8.152Test accuracies of 20% weak labelling, single augmentation training (2000 original simulated stereopairs, 5× augmentation) [10,90] 205
8.153Test accuracies of 20% weak labelling, single augmentation training (5000 original simulated stereopairs, 2× augmentation) [10,90] 206
8.154Test accuracies of 20% weak labelling, single augmentation training (10000 original simulated stereopairs, 1× augmentation) [10,90] 206
8.155Test accuracies of 20% weak labelling, single augmentation training (125 original simulated stereopairs, 80× augmentation) [10:10:90] 207
8.156Test accuracies of 20% weak labelling, single augmentation training (250 original simulated stereopairs, 40× augmentation) [10:10:90] 207
8.157Test accuracies of 20% weak labelling, single augmentation training (500 original simulated stereopairs, 20× augmentation) [10:10:90] 208
8.158Test accuracies of 20% weak labelling, single augmentation training (1000 original simulated stereopairs, 10× augmentation) [10:10:90] 208
8.159Test accuracies of 20% weak labelling, single augmentation training (2000 original simulated stereopairs, 5× augmentation) [10:10:90] 208
8.160Test accuracies of 20% weak labelling, single augmentation training (5000 original simulated stereopairs, 2× augmentation) [10:10:90] 209
8.161Test accuracies of 20% weak labelling, single augmentation training (10000 original simulated stereopairs, 1× augmentation) [10:10:90] 209

8.162Test accuracies of 10% weak labelling, single augmentation training (125 original simulated stereopairs, 80× augmentation) [10,90] 210
8.163Test accuracies of 10% weak labelling, single augmentation training (250 original simulated stereopairs, 40× augmentation) [10,90] 210
8.164Test accuracies of 10% weak labelling, single augmentation training (500 original simulated stereopairs, 20× augmentation) [10,90] 211
8.165Test accuracies of 10% weak labelling, single augmentation training (1000 original simulated stereopairs, 10× augmentation) [10,90] 211
8.166Test accuracies of 10% weak labelling, single augmentation training (2000 original simulated stereopairs, 5× augmentation) [10,90] 211
8.167Test accuracies of 10% weak labelling, single augmentation training (5000 original simulated stereopairs, 2× augmentation) [10,90] 212
8.168Test accuracies of 10% weak labelling, single augmentation training (10000 original simulated stereopairs, 1× augmentation) [10,90] 212
8.169Test accuracies of 5% weak labelling, single augmentation training (125 original simulated stereopairs, 80× augmentation) [10,90] 213
8.170Test accuracies of 5% weak labelling, single augmentation training (250 original simulated stereopairs, 40× augmentation) [10,90] 213
8.171Test accuracies of 5% weak labelling, single augmentation training (500 original simulated stereopairs, 20× augmentation) [10,90] 214
8.172Test accuracies of 5% weak labelling, single augmentation training (1000 original simulated stereopairs, 10× augmentation) [10,90] 214
8.173Test accuracies of 5% weak labelling, single augmentation training (2000 original simulated stereopairs, 5× augmentation) [10,90] 214
8.174Test accuracies of 5% weak labelling, single augmentation training (5000 original simulated stereopairs, 2× augmentation) [10,90] 215
8.175Test accuracies of 5% weak labelling, single augmentation training (10000 original simulated stereopairs, 1× augmentation) [10,90] 215

8.176Test accuracies of 0% weak labelling, single augmentation training (125 original simulated stereopairs, 80× augmentation) [10,90] 216
8.177Test accuracies of 0% weak labelling, single augmentation training (250 original simulated stereopairs, 40× augmentation) [10,90] 216
8.178Test accuracies of 0% weak labelling, single augmentation training (500 original simulated stereopairs, 20× augmentation) [10,90] 217
8.179Test accuracies of 0% weak labelling, single augmentation training (1000 original simulated stereopairs, 10× augmentation) [10,90] 217
8.180Test accuracies of 0% weak labelling, single augmentation training (2000 original simulated stereopairs, 5× augmentation) [10,90] 217
8.181Test accuracies of 0% weak labelling, single augmentation training (5000 original simulated stereopairs, 2× augmentation) [10,90] 218
8.182Test accuracies of 0% weak labelling, single augmentation training (10000 original simulated stereopairs, 1× augmentation) [10,90] 218

List of Acronyms and Abbreviations

Adam Adaptive Moment Estimation

ADF Annular Dark Field

AET Atomic Electron Tomography

AFM Atomic Force Microscopy/Microscope

ANN Artificial Neural Network

bcc Body-Centered Cubic

BF Bright Field

BID Beam-tilt-induced Image Displacement

CAT Computerised-Axial Tomography

CBED Coherent Beam Electron Diffraction

CD Circular Dichroism

CNN Convolutional Neural Network

CPU Central Processing Unit

CT Computerised Tomography

DF Dark Field

DNA Deoxyribonucleic Acid

DPC Differential Phase Contrast

EELS Energy Electron-Loss Spectroscopy

EDX/EDS Energy-Dispersive X-ray Spectroscopy

EM Electron Microscopy/Microscope

ET Electron Tomography

fcc Face-Centered Cubic

FEG Field-Emission Gun

FFT Fast Fourier Transform

FIB Focused Ion Beam

FIM Field Ion Microscopy

GD Gradient Descent

GPU Graphical Processing Unit

GUI Graphical User Interface

HAADF High-Angle Annular Dark Field

HPC High-Performance Computing

HRTEM High Resolution Transmission Electron Microscopy/Microscope

ILSVRC ImageNet Large Scale Visual Recognition Challenge

KAIST Korea Advanced Institute of Science and Technology

LBNL Lawrence Berkeley National Laboratory

LoG Laplacian of Gaussian

LM Low Magnification

MOSFET Metal Oxide Semiconductor Field Effect Transistor

ML Machine Learning

NCEM National Center for Electron Microscopy

NiPtNP Nickel-Platinum Nanoparticle

NP Nanoparticle

PET Positron Emission Tomography

ReLU Rectified Linear Unit

ROI Region of Interest

SBEM Sample Block-Face Scanning Electron Microscopy

SGD Stochastic Gradient Descent

SEM Scanning Electron Microscopy/Microscope

SPECT Single-Photon Emission Computer Tomography

SPM Scanning Probe Microscopy/Microscope

ss-TEM Serial-Section Transmission Electron Microscopy

STEM Scanning Transmission Electron Microscopy/Microscope

STM Scanning Tunnelling Microscopy/Microscope

SVM State Vector Machine

TEAM Transmission Electron Aberration-Corrected Microscope

TEM Transmission Electron Microscopy/Microscope

TeNP Tellurium Nanoparticle

TWS Trainable Weka Segmentation

Chapter 1

Introduction

The development of the first transmission electron microscope (TEM) by Ernst Ruska and Max Knoll in 1931 enabled scientists to harness high-energy electrons to probe the nature of materials at scales shorter than the wavelength of visible light [1]. Just four years later, Konrad Zuse started designing what would become the Z3, the world's first automatic, fully programmable computer, which he finished building in 1941 [2]. In the decades since, both technologies have flourished. Following the first successful direct observation of individual atoms by Müller and Bahadur in 1956 using Field Ion Microscopy (FIM) [3], the same feat was achieved in electron microscopy by Crewe et al. in 1970, who used a scanning transmission electron microscope (STEM) with an annular dark-field (ADF) detector to image uranium and thorium atoms on a thin carbon substrate [4]. Improvements in electron microscopy have been driven by the development of better electron filaments, electron lenses and, more recently, aberration-correction systems [5, 6]. Electron microscopes can now be found in scientific institutions across the world, where they are used to advance the frontiers of both life and material sciences [7]. Digital computers, on the other hand, are now ubiquitous in ordinary modern life thanks to the development and miniaturisation of the transistor, resulting in personal computers and smartphones with several orders of magnitude more processing power than the Z3. This has led to many low-skilled and/or repetitive human occupations being replaced by automated systems and the acquisition and processing of electron microscope data is starting to follow suit, as evidenced by the increasing use of automated microscope control

software in life sciences [8] and electron tomography [9]. Another emerging tool that could be of great benefit to microscopists is machine learning, where computers learn for themselves how to characterise and group data rather than having to be explicitly programmed to do so by humans. Such systems have already shown great promise in image recognition tasks [10–12] and are already starting to be applied to electron micrographs [13].

This thesis seeks to further expand the use of automation in electron microscopy, especially in the field of material science, which, to date, has yet to embrace the concept as readily as life sciences. Chapter 2 starts by discussing the fundamentals of electron microscopy with a particular focus on scanning transmission electron microscopy (STEM), the electron microscopy technique used throughout this thesis. It also discusses aberration correction, the effects of electron-beam irradiation on a sample and the basics of electron tomography, one of the fields of electron microscopy in which automation is already commonly applied. Chapter 3 discusses the TEAM 0.5 (Transmission Electron Aberration-Corrected Microscope), one of the world's most powerful transmission electron microscopes and the first to achieve 50 pm resolution in both TEM and STEM. The TEAM 0.5's custom-built ultra-stable five-axis stage is what makes a lot of the work presented in this thesis possible and its capabilities, along with the software used to operate it and the rest of the microscope remotely, are discussed here. This chapter also describes the experimental processes involved in alpha tomography, a form of single-axis electron tomography in which samples are rotated about the α axis of the TEAM stage. Chapter 4 outlines the theory behind and training of artificial neural networks (ANNs), one of the most promising and powerful forms of machine learning. It also examines their sub-variant, convolutional neural networks (CNNs), whose effectiveness in image-recognition tasks has been repeatedly demonstrated. Three of the most notable image-recognition CNNs are discussed in greater detail and their key features identified. Chapter 5 introduces the processes involved in creating and transporting samples mounted on tungsten tips for use in electron tomography, many of which are fully or semi-automated, and examines the viability of bismuth and carbon coating as means of inhibiting oxidation of the samples. Chapter 6 introduces a software framework used to automate the experimental operation of the TEAM 0.5 and three case studies of its application: the high-throughput acquisition of atomic-resolution images of cadmium selenide/cadmium sulphide core-shell nanoparticles, the acquisition

of coherent-beam electron diffraction (CBED) patterns of tellurium nanoparticles and the search for particles suitable for alpha tomography. It also demonstrates an example of how CNNs can be created and employed in automated control software. Chapter 7 details the development of CNNs capable of distinguishing the handedness of simulated particles with chiral geometry using stereopairs of simulated STEM projections. It also introduces a 'weakly labelled' approach to training such CNNs that takes advantage of the intrinsic nature of chirality to reduce the effort involved in training them.

Chapter 2

Scanning transmission electron microscopy (STEM)

2.1 Fundamentals of Electron Microscopy

Abbe's theory states that the resolution limit of any microscope is:

$$d = \frac{0.8\lambda}{\sin\alpha} \tag{2.1}$$

where λ is the wavelength of the imaging medium and α is the aperture angle of the objective lens [14]. If the illumination aperture is increased to equal the objective aperture α, this is improved to:

$$d = \frac{0.61\lambda}{\sin\alpha} \tag{2.2}$$

in accordance with Lord Rayleigh's resolution criterion [14–16]. The wavelength of a particle, as postulated by Louis de Broglie in 1923 [17] and verified by the Davisson-Germer experiment in 1928 [18], is given by:

$$\lambda = \frac{h}{p} \tag{2.3}$$

where λ is the particle's wavelength, h is Planck's constant and p is the particle's momentum. For a massive particle like an electron, this equation, with relativistic effects factored in, becomes:

$$\lambda = \frac{h}{\sqrt{2m_0eV(1 + \frac{eV}{2m_0c^2})}} \tag{2.4}$$

where m_0 is the particle's rest mass, e is the electron charge, V is the accelerating voltage and c is the speed of light. The accelerating voltages used in electron microscopy vary depending on application, but typically range from 20-300 kV [19, 20], corresponding to an electron wavelength range of 8.58-1.96 pm, which, in principle, is small enough to probe any atomic-scale phenomena. In practice, the resolution of electron microscopy is limited by aberrations induced by the lenses used to focus the electron beam, as will be discussed in Section 2.4, but atomic resolution images have been generated with voltages as low as 30 kV [21].

2.2 Types of Electron Microscopy

While all electron microscopes share a few key components, including an electron source, electromagnetic lenses for guiding and focusing these electrons and some form of electron detector, the configuration of these components varies between different forms of electron microscopy. Two of the broadest categories of electron microscopy are scanning electron microscopy (SEM) and transmission electron micoscopy (TEM). A diagram comparing the optics of these two configurations, along with that of scanning transmission electron microscopy (STEM), is show in Figure 2.1.

2.2.1 Scanning Electron Microscopy (SEM)

Scanning electron microscopy is a technique used to study the surface of samples. In it, electrons are focused by electromagnetic lenses into a tight spot, known as a probe, on the sample's surface, which is then rastered across the entire sample [19, 20]. These electrons interact with the sample (to a depth of roughly one micron) and generate backscattered and secondary electrons that are

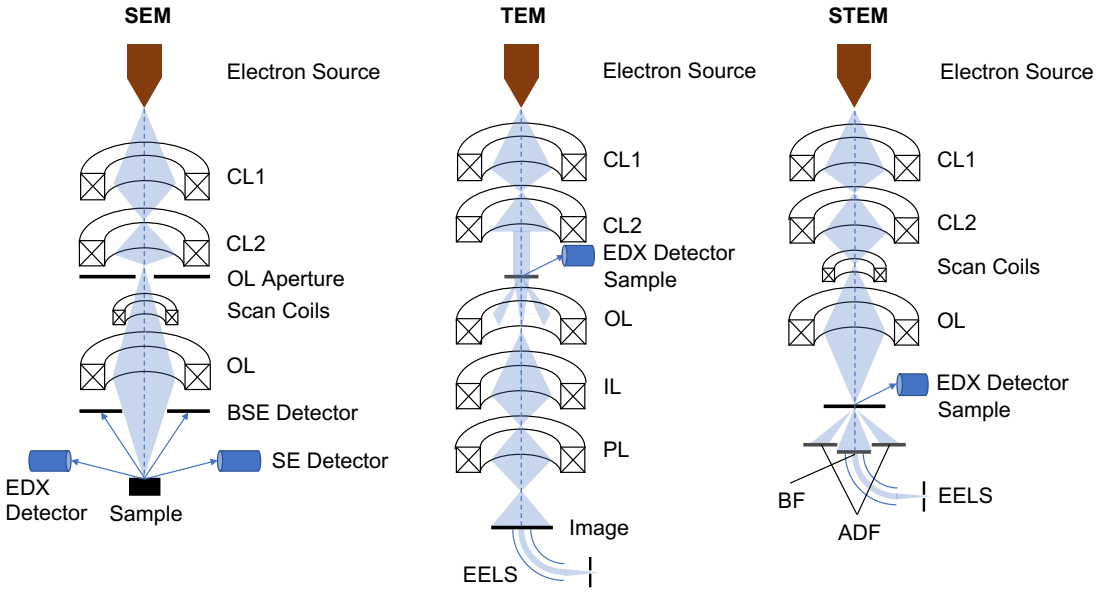


Figure 2.1: Comparison of SEM (left), TEM (centre) and STEM (right) lens configurations (CL: Condenser Lens, OL: Objective Lens, IL: Intermediate Lens, PL: Projector Lens, BSE: Backscattered Electron, SE: Secondary Electron).

then collected by detectors, which are mounted on the same side of the sample as the electron source. The intensity of each pixel in the resultant micrograph is proportional to the number of electrons counted at the corresponding detector position. The highest SEM resolution currently available is 0.4 nm [22], which is too low to examine atomic structure, but large enough to resolve surface microstructures. SEM is used in both life sciences [23] and material sciences [24] for various purposes, such as studying surface crystal structure [24] or monitoring surface defects like cracks in an alloy [25]. Sample composition can also be studied by examining the characteristic X-rays generated by beam-sample interactions, a technique known as energy-dispersive X-ray spectroscopy (EDS/EDX) [19, 20, 26].

2.2.2 Transmission Electron Microscopy (TEM)

Transmission electron microscopy refers to any electron microscopy technique that uses transmitted electrons to study a sample, usually to examine its internal structure. In a conventional TEM, a coherent, parallel beam of electrons with uniform amplitude and phase is fired at the sample,

which has to be thin enough (<100 nm [27]) to enable sufficient transmission of the electrons. Detectors with high pixel density (or films) are mounted on the opposite side of the sample to the electron source and capture both transmitted and forward-scattered electrons, with the intensity at each position in the micrograph corresponding to the number of electrons incident upon the corresponding portion of the detector [7]. The image that is formed depends on the configuration of the lenses and the scattering angle range of the electrons captured. In bright-field (BF) imaging, the objective aperture selects the unscattered beam and prevents scattered electrons from forming the image. In amorphous samples, the primary contrast mechanism is the mass-thickness relationship, with darker regions corresponding to heavier or thicker areas of the sample, which are more effective at impeding the transmission of electrons. In crystalline samples, contrast will be dominated by diffraction, with crystalline lattices appearing darker due to the coherent scattering of electrons [28]. In dark-field (DF) imaging, the beam and/or objective aperture is positioned so that the image is formed from scattered electrons outside the unscattered beam. In this mode, thin and/or amorphous samples appear dark in the resultant image due to the lack of scattering, whereas thick and/or crystalline samples appear bright due to increased scattering, which makes DF imaging particularly useful for studying crystal lattices [28]. In high-resolution TEM (HRTEM) imaging, a larger objective aperture is used to allow both scattered and unscattered electrons to interfere with one another, forming a high-resolution phase contrast image. With modern aberration correction, the resolution of this image is typically 0.05-0.1 nm [27]. TEMs can also be configured for electron holography as well as the recording of both parallel-beam and convergent-beam (CBED) diffraction patterns. Spectroscopic techniques, like electron energy-loss spectroscopy (EELS) and EDX, are also compatible with TEM [27].

2.3 Scanning Transmission Electron Microscopy (STEM)

Scanning transmission electron microscopy (STEM) is similar to conventional TEM in that it studies a sample using electrons passing through a sample. However, instead of focusing the electrons into a broad, coherent beam of electrons, they are instead focused into a 0.2-0.5 nm-sized probe [29] that rasters across the surface, like in SEM. Like in TEM, STEM detectors are positioned

on the opposite side of the sample to the electron source in order to capture transmitted electrons, but, unlike TEM, the detectors are typically monolithic (not pixelated)[1] and count all electrons scattered over a specific angular range. The intensity of each pixel in the resultant micrograph is proportional to the total number of electrons counted at each probe position. This enables spatial determination of the origin of each electron, transmitted or scattered, something that is not possible in conventional TEM and increases the amount of information that can be extracted [19].

2.3.1 Annular Dark Field (ADF) imaging

Annular dark-field (ADF) imaging is one of the most common imaging techniques in STEM, using annular (ring-like) detectors to collect these electrons from all lateral directions simultaneously [28]. At low angles (∼0-25 mrad for a 200 keV TEM [32]), these electrons are a combination of coherently (diffracted) and incoherently scattered electrons. Incoherent scattering is the scattering of electrons from individual nuclei rather than crystal lattices, the cross-section of which is proportional the square of the atomic number, as shown by:

$$\frac{d\sigma}{d\Omega} \sim Z^2 \tag{2.5}$$

where σ is the scattering cross-section, Ω is the solid angle and Z is the atomic number [26]. At higher angles (∼50-200 mrad for a 200 keV TEM [33]), the contribution from coherently scattered electrons drops away, leaving incoherently scattered electrons as the dominant signal, which enables Z-contrast imaging (image intensity is a function of Z). This is the guiding principle of high-angle annular dark-field (HAADF) imaging, which is often used in material science for imaging heavy elements [7, 19]. It should be noted that, in practice, the relationship between intensity and atomic number is:

$$I \propto Z^{\gamma} \tag{2.6}$$

[1] This is only true for '2D-STEM'. '4D-STEM', which includes techniques like differential phase contrast (DPC) imaging [30, 31] and electron ptychography [31], utilises segmented and/or pixelated detectors to record additional information. Even here, however, a basic '2D' STEM micrograph can be generated by summing the number of electrons counted by all segments and/or pixels within a specified angular range of scattering.

where γ is a value (typically in the range of 1.4-1.9[33, 34]) that depends on factors of microscope geometry like collection angle [35] and, to a lesser extent, sample thickness [36].

2.4 Aberration correction

While the wavelengths of the high-energy electrons used in electron microscopy are of the order of picometres, in practice, the resolution of an electron microscope is limited by aberrations introduced by the electron lenses. This limitation was predicted by Otto Scherzer as far back as 1936, in which he demonstrated that chromatic (C_c) and spherical (C_s) aberrations could not be entirely eliminated from any electron lens that was static, rotationally symmetric and free of space charges [37]. Figure 2.2 shows ray diagrams of these two aberrations.

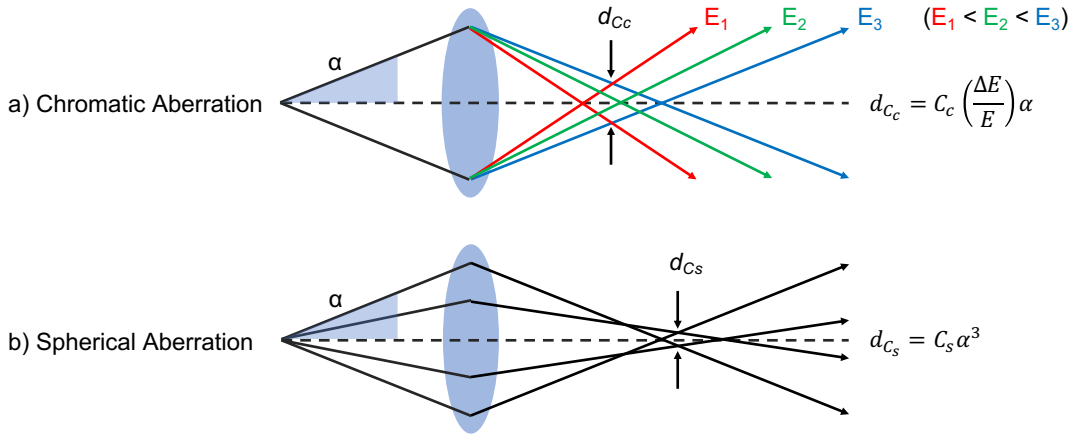


Figure 2.2: Ray diagram of a) chromatic (C_c) aberration and b) spherical (C_s) aberration.

2.4.1 Chromatic (C_c) aberration

Chromatic aberration is caused by electrons (or photons, in the optical case) being refracted by different amounts in the lens depending on their wavelength, which is inversely proportional to their energy. The geometrical blur for chromatic aberration is given by:

$$d_{C_c} = C_c\left(\frac{\Delta E}{E}\right)\alpha \tag{2.7}$$

where α is the convergent semi-angle, E is the accelerating energy and ΔE is the electron energy spread [19]. This aberration has been largely mitigated by advances in electron sources, with a general shift from thermionic emission sources, where a filament (e.g. W or LaB_6) is heated to give its conduction band electrons sufficient energy to overcome the material's work function and become free electrons, to field emission guns (FEGs), where an electric potential is applied to a sharp tip, typically electrochemically etched from a piece of wire (e.g. W or ZrO/W), and the enhanced electric field at the apex of the tip reduces the height of the potential barrier sufficiently for electrons to tunnel through. FEGs have a smaller energy spread (ΔE) than thermionic sources, which is one of the reasons why they are favoured in modern electron microscopes [7, 20].

2.4.2 Spherical (C_s) aberration

Spherical aberration is caused by lenses failing to focus electrons to the same point. The geometrical blur for spherical aberration is given by:

$$d_{C_s} = C_s \alpha^3 \tag{2.8}$$

where α is the convergence semi-angle [19]. Setting $\alpha = 0$ would eliminate the blur, but, from Equation 2.2, this would also make the minimum resolvable distance = ∞. Achieving optimal performance in an electron microscope is therefore a balancing act between the two effects. In STEM, the minimum probe size achievable is given by:

$$d_{min} = 0.43 \lambda^{\frac{3}{4}} C_s^{\frac{1}{4}} \tag{2.9}$$

and the optimum convergence semi-angle for resolution, α_0, is given by:

$$\alpha_0 = \left(\frac{4\lambda}{C_s} \right)^{\frac{1}{4}} \tag{2.10}$$

where λ is the electron wavelength [16].

In 1949, Scherzer suggested that spherical (and chromatic) aberrations could be corrected by breaking any of the three lens conditions outlined in his 1936 paper: static, rotational symmetry,

space charge free. It is the second of these that is broken in modern multipole-based aberration correction systems, which were first used to correct spherical aberrations in 1998 and achieve a point resolution 0.14 nm in a CM 200 TEM [5]. In 2009, a resolution of 50 pm was achieved in STEM mode on the TEAM 0.5 microscope at NCEM, LBNL [38]. Aberration correction is now an essential and widespread component of high-resolution electron microscopy.

2.4.3 Astigmatism

An aberration closely related to spherical aberration is astigmatism, where the electron beam is focused into a shape other than a perfectly round spot as a result of exposure to a non-uniform magnetic field [39] (such as those created by multipole-based aberration correction systems). The lowest order of this effect is 2-fold astigmatism, which causes the beam to be focused into an ellipse rather than a round spot, as shown in Figure 2.3, and results in point-like objects appearing as ellipsoids in the resultant image [40]. This, along with defocus, must be routinely corrected by a microscopist, which is usually done using octupole correctors that produce compensating fields to cancel out the astigmatism [41].

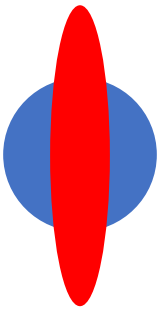


Figure 2.3: Comparison of stigmated (red) and non-stigmated (blue) electron beam shapes.

2.5 Radiation damage

Although high-energy electrons are a useful tool for examining the structure and topology of nanomaterials, these electrons can also damage the samples that they are imaging. Various damage mechanisms exist that can be categorised based on the type of electron scattering. Elastic scat-

tering, which occurs when the electrons are deflected by the nucleus and gives rise to most of the scattering seen in TEM and STEM, as well as back-scattering seen in SEM, can cause atomic displacement within the lattices, also known as knock-on damage, and the sputtering of surface atoms [42–44]. Inelastic scattering, which occurs when the electrons in the beam are deflected by atomic electrons and is responsible for secondary electron emission in SEM, as well as the generation of X-rays for EDX, can cause ionization damage (radiolysis) of the sample, altering its structure or removing material [26, 42, 44]. If hydrocarbons are present, it is also the mechanism by which hydrocarbon contamination, which distorts both TEM and SEM images, occurs [42].

The impact of each damage mechanism is extremely sample-dependent. Good conductors, for example, are extremely resistant to radiolytic damage, since any hole that is created by the excitation of an electron above the Fermi level of the sample will be quickly filled due to the high density of conduction band electrons, which is not the case for insulating or semiconducting materials [44]. Different materials also have different threshold energies for knock-on damage effects, with higher-mass elements (e.g. W, Pt, Au) generally having higher thresholds than lower-mass elements (e.g. Li, C, Al) [43]. Various ways of mitigating these problems exist, however, such as cooling the sample, which both counteracts the heating effect of the electron beam and reduces the radiation sensitivity of samples (cryo-EM is commonly used to examine biological samples [45]), changing the electron energy, either by decreasing it to limit knock-on damage or, in some cases, increasing it to reduce radiolysis, or lowering the overall electron dose, which can be achieved by reducing the beam intensity, beam diameter or simply the number of images taken [44]. These are all key considerations when setting up any electron microscopy experiment, especially when using HAADF-STEM, which requires a higher electron beam intensity than conventional TEM due to the relatively low probability of high-angle scattering events [7].

2.6 Electron Tomography (ET)

Electron tomography is a technique used to obtain three-dimensional structural data from two-dimensional micrographs taken via TEM or STEM. This is performed by rotating a sample inside a microscope and capturing images at different angles, creating a ‘tilt series’ from which the 3D model

can then be generated (also referred to as 'reconstruction'). Most samples are only rotated around a single axis, due to the relative simplicity of data acquisition and reconstruction [7], although dual-axis tomography has been used to overcome the 'missing-wedge problem' (see Section 2.6.1) [9, 46, 47].

A fundamental requirement of tomography is that the measured quantity (i.e. intensity of each pixel) is a monotonic function of a specific property of the sample (e.g. mass or magnetic field strength) in every projection, with the voxel intensities in the final reconstruction correlating to this property; this is known as the 'projection requirement' [7, 48]. Another implicit but no less important requirement is that the sample remain stable for the duration of the imaging, or else the images in the tilt series will be projections of effectively different samples. Together, these two requirements play an important part in determining the technique used. Conventional TEM fulfils the projection requirement when the sample being imaged has no crystalline structure and the intensity therefore correlates primarily to the mass/thickness of the sample. This is often the case for certain biological samples like cells and protein complexes and conventional TEM has used to generate tomographic reconstructions of IgG1 antibodies [49, 50], high-density lipoproteins [51] and DNA origami Bennett linkages [52]. This requirement is violated, however, if the sample is crystalline, as so many materials science samples are, because the contrast of TEM images of such samples is dominated by electron diffraction rather than any quantifiable property of the sample. The Z-contrast mechanism of HAADF-STEM imaging, on the other hand, fulfils the projection requirement for all samples, since the pixel intensity is solely proportional the sample mass, but the higher beam intensity of STEM makes it unsuitable for materials with low knock-on damage thresholds [7].

2.6.1 Missing-Wedge Problem

Most TEM and STEM samples are mounted on slab-like supports for imaging, most commonly a TEM grid: a 3mm-wide, 0.1mm-thin circular mesh grid (usually copper) with a thin, electron-transparent carbon film covering the mesh holes [53]. In standard microscopy, the default orientation for these grids is perpendicular to the electron beam, meaning that the beam passes through the thinnest possible cross-section of carbon film. In tomography, however, these grids have to be

rotated and, as this happens, the effective thickness t_{eff} of the carbon film increases becomes:

$$t_{eff} = \frac{t}{\cos\theta} \tag{2.11}$$

where t is the thickness of the carbon film (effective thickness at normal incidence) and θ is the tilt angle, as demonstrated in Figure 2.4. This added thickness that the electron beam must pass through increases the background noise and decreases the signal-to-noise ratio, which, in turns, limits the effective spatial resolution of tilted images. At high tilt angles ($\sim \pm 80°$), the grid bars shadow the sample and prevent the transmission of electrons, making it impossible to obtain useful images [7, 9, 46, 47]. On a related note, the space between focusing lenses in most microscopes is often too small to enable extreme tilting of the sample holder, meaning that the holder and/or the microscope must be specifically configured for electron tomography. In practice, $\sim \pm 70°$ is the angular limit for most tilt series, meaning that projections cannot be acquired in the region of $\pm$(70-90)°. This results in a 'missing wedge' of data from these high tilt angles and the absence of these projections causes elongation of features in the final reconstruction, especially along the original projection direction (the beam direction at normal incidence to the TEM grid) [7, 54].

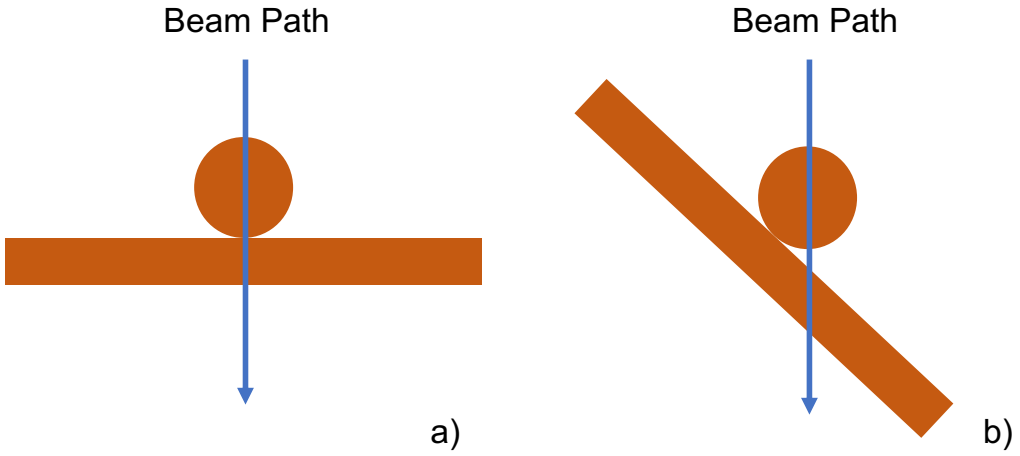


Figure 2.4: a) Path of the electron beam through a sample mounted on a slab-like support at normal incidence. b) Path of the electron beam through a sample mounted on a slab-like support at tilted incidence.

Chapter 3

TEAM 0.5

3.1 TEAM 0.5 capabilities

The Transmission Electron Aberration-Corrected Microscope 0.5 (TEAM 0.5) is a double-aberration-corrected (scanning) transmission electron microscope situated at the National Center for Electron Microscopy at the Lawrence Berkeley National Laboratory in Berkeley, California. It was the first microscope to achieve 50 pm resolution in both TEM and STEM modes, thanks in large part to its double-aberration-correction system, which consists of a probe corrector and an image corrector. The probe corrector corrects coherent axial aberrations up to 4th order, 5th-order spherical aberrations and six-fold astigmatism, while the image corrector corrects coherent axial aberrations up to 3rd order and partially corrects 4th- and 5th-order aberrations. Its source monochromator can also achieve an energy resolution of 0.15 eV [55, 56].

The stage for the TEAM 0.5 is a custom-built, ultra-stable, piezo-electric stage with a drift rate of 2 pm/sec and five-axis movement: x, y, z, α and γ, the axes of which are shown in Figure 3.1. α and γ have a full $\pm 180°$ range of motion ($\pm 180°$) [55, 57]. Samples, like TEM grids or W tips, are mounted on pucks with grooves along their rims, which, in turn, are held between two Ti wires known as 'chopsticks'. There are two modules in the stage: the alpha module, which is connected to the microscope column and controls y-z-α motion, and the beta module, which is connected to the alpha module and controls x-γ motion. Moving the chopsticks in and out of

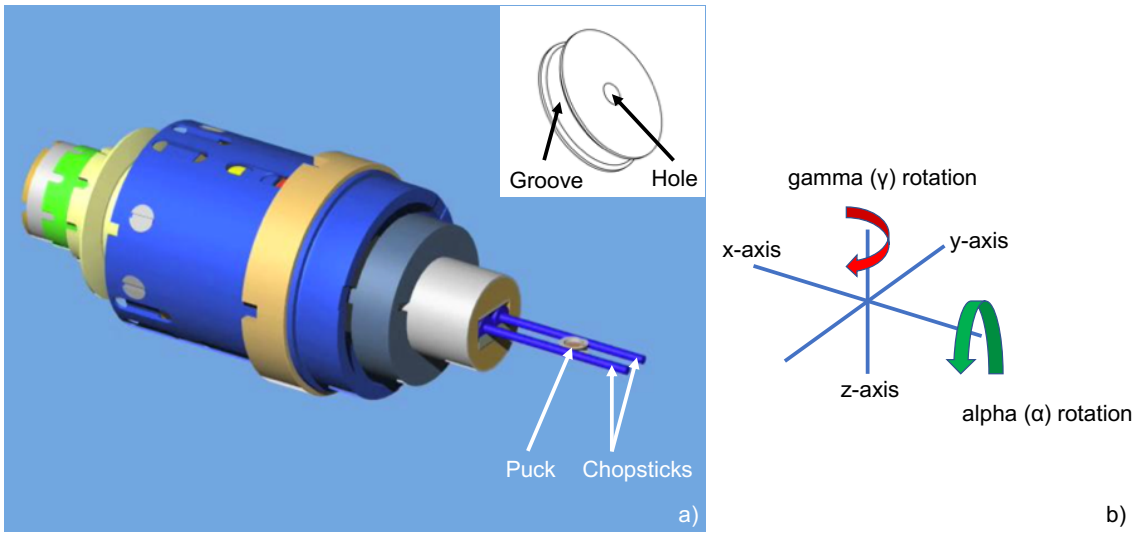


Figure 3.1: a) Schematic of TEAM 0.5 stage and puck schematic inset (images courtesy of Peter Ercius). Samples are mounted on the pucks (the puck shown is designed to accommodate an etched tungsten tip in the hole), which then slide between two 'chopsticks' courtesy of the grooves in the circumference of the puck. b) Translational and rotational axes of TEAM 0.5 stage (N.B. The direction of the γ axis rotates as the stage is rotated in α).

the stage achieves x motion, moving the chopsticks in opposite directions achieves γ rotation and angling the chopsticks away from the normal achieves α rotation [57].

3.2 Remote Operation Software (Wtip Server and Client)

The remote operation software for the TEAM 0.5, which consists of the Wtip client and the Wtip server, was primarily developed by Dr Wolfgang Theis to control the TEAM 0.5 in STEM mode both from within NCEM and from other locations over the internet. Since it was originally developed for use in tungsten tip (hence 'Wtip') electron tomography experiments, where cumulative electron beam dose is a major concern, many of its functions have been designed to minimise the electron beam dose that the sample receives, with the beam being blanked automatically between images to prevent unnecessary exposure. More information regarding tungsten tip tomography can be found in Chapter 5. As shown in Figure 3.2, which shows how the Wtip client and server connect to the TEAM 0.5 microscope software, the software allows multiple remote users to interface with the microscope at any one time, enabling multiple parties to observe and participate in

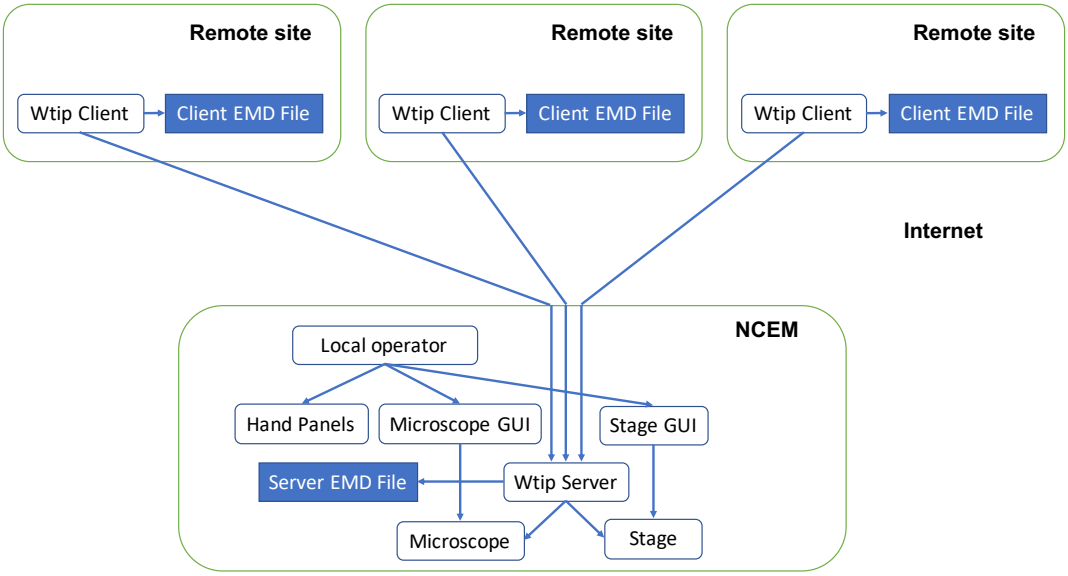


Figure 3.2: Connections between Wtip client and server and TEAM 0.5 microscope.

any given experiment.

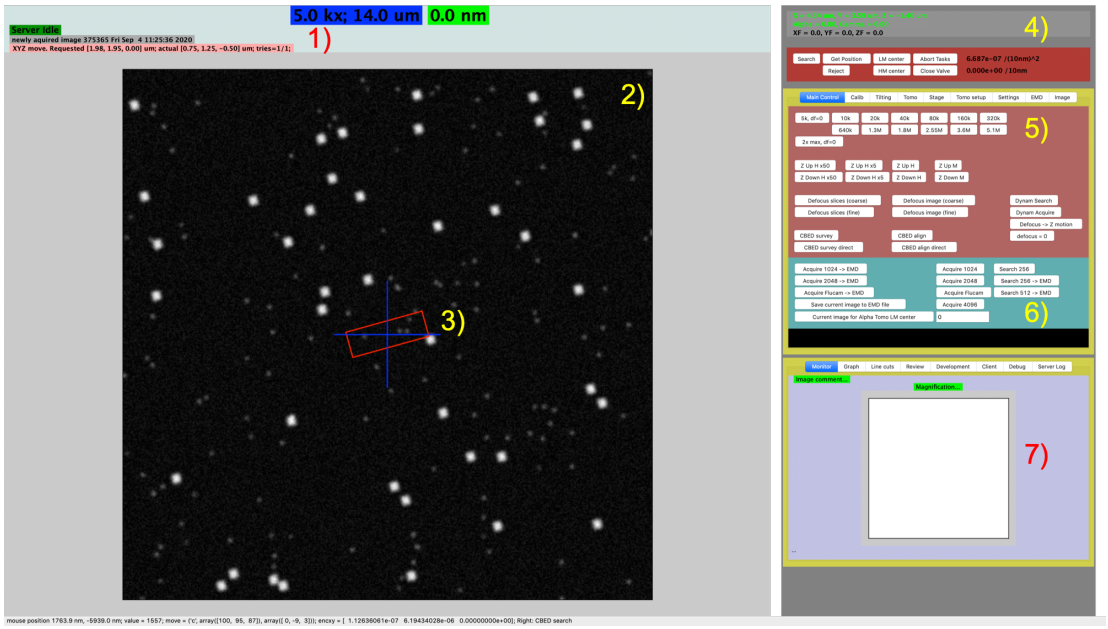


Figure 3.3: Main GUI window. 1) Server status bar. 2) Image window (with simulated image). 3) Slider range box. 4) Stage position bar. 5) 'Main control' panel with magnification, z stepper movement, defocus slices (see Section 3.2.1) and CBED acquisition controls. 6) Image save controls. 7) Monitor window (See Appendix A for magnified image of panels 4-7).

Figure 3.3 shows the main control display for the Wtip client. All images are saved in an emd file (this is an hdf5 file with a custom extension so as to be compatible with a custom viewer) that also records experimental parameters such as microscope conditions, calibration constants and logs of commands used. The large window on the left displays either the last image taken in the microscope or last image opened from the emd file. Movement of the stage in the x and y directions is achieved by left-clicking on any point in the image, which moves the stage to designated point.

3.2.1 Defocus Slices

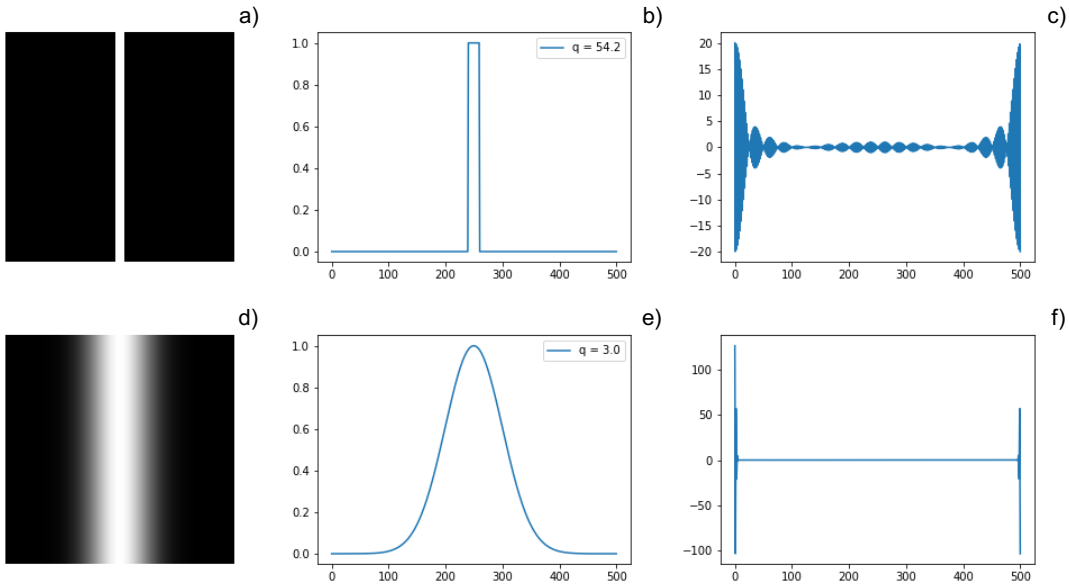


Figure 3.4: a) Simulated image of sharp edge. b) Line cut of sharp edge ($q = 54.2$). c) FFT of sharp edge line cut. d) Simulated image of Gaussian-filtered ($\sigma = 50$) sharp edge (with intensity normalised). e) Line cut of Gaussian-filtered sharp edge ($q = 3.0$). f) FFT of Gaussian-filtered edge line cut.

Defocus slices are the primary method of adjusting focus using the Wtip client. They are, in essence, a focal series of line cuts taken through a designated pixel (usually the centre or highest-intensity pixel of an image). A Fast Fourier Transform (FFT) is then applied to these line cuts and the absolute values of each frequency above the zero frequency in the FFT summed and divided by the value of the zero frequency to produce a metric of focus. A perfectly sharp edge is equivalent

to the sum of all possible frequencies with a uniform amplitude, which, in an FFT, would look like a flat line of that amplitude. A non-sharp edge can be thought of as a sharp edge with a Gaussian filter applied to it, the effect of which is to suppress the higher frequencies and thereby lower the sum of all possible frequencies, as demonstrated in Figure 3.4. Therefore, this sum of frequencies, q, can be used as a metric of sharpness and hence focus, with high q values corresponding to an image being in focus.

Defocus slice results are displayed in the monitor window in the bottom-right of the main control panel (see Figure 3.1) and clicking on a point in that window adjusts the defocus to the corresponding value. If the optimal focus is judged to be between two values, clicking between the two points will adjust the defocus to the corresponding intermediary value. There are two types of defocus slices — coarse and fine — with coarse slices having larger defocus intervals and fine slices have smaller defocus intervals. The client provides useful default values for these defocus intervals, with the 'Client' tab enabling a user to manually input values for these if desired.

3.2.2 Stigmation correction

Astigmatism can be corrected using the Wtip client via the stigmation tableau: a set of images taken at the same position — usually one with a clearly visible atomic lattice structure — with varying x and y stigmator values, from which the user can manually select the best image. If the optimal adjustment is judged to be between two (or three) images, clicking between these images will adjust the astigmatism to the corresponding intermediary value. An example of this correction is shown in Figure 3.5, with the image in a) showing clear streakiness. The stigmation tableau in b) shows that the best image is generated by the stigmation settings used in the bottom-right image. c) shows the stigmation-corrected image with the streakiness removed, while d) shows that the current settings, represented by the image in the centre, produce the best image. Since the quality of the images degrades evenly in all directions spreading out from the centre, the current settings are optimal.

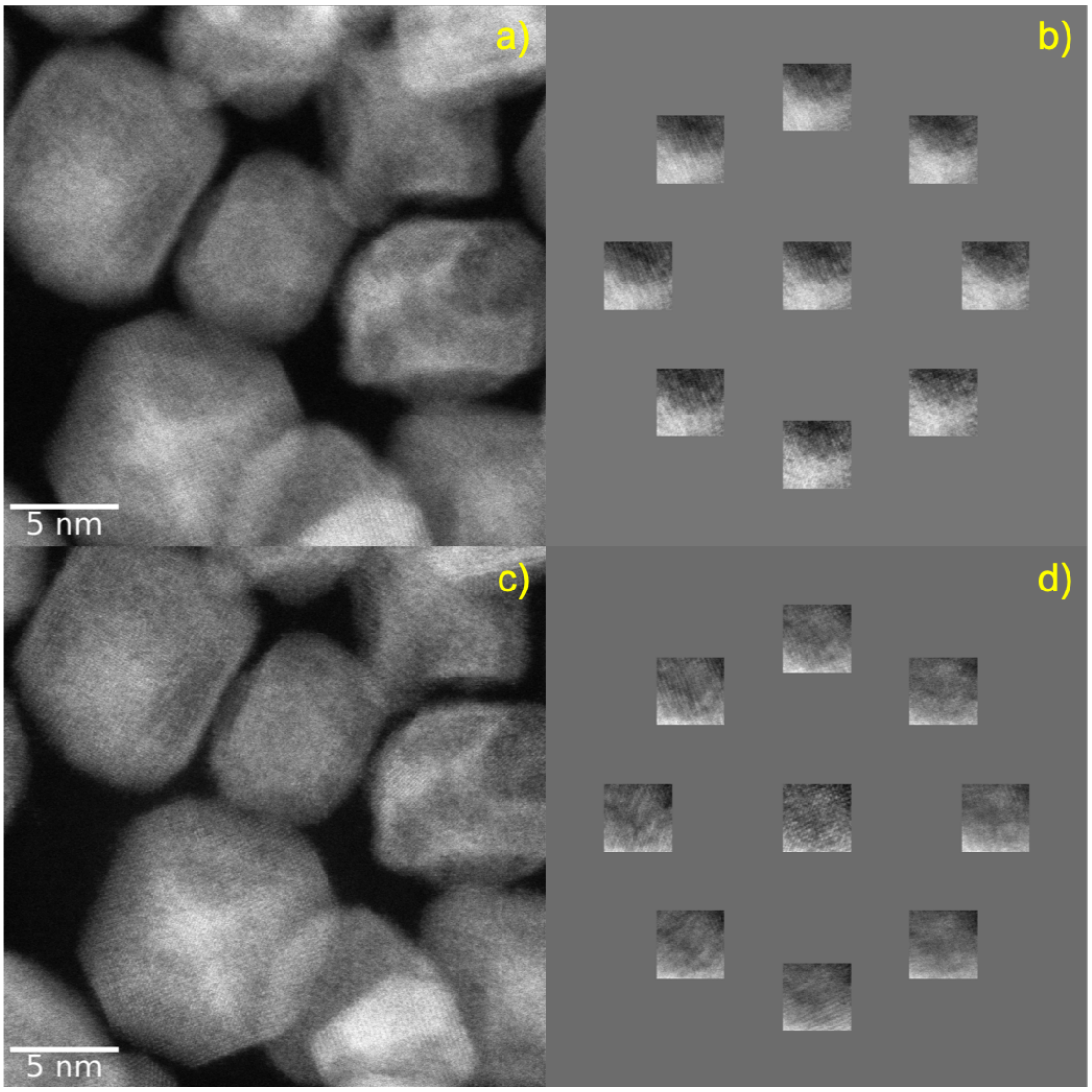


Figure 3.5: Astigmatism adjustment. a) Original image. b) Stigmation tableau corresponding to (a). c) Stigmation-corrected image. d) Stigmation tableau corresponding to (d).

3.3 Alpha Tomography

Alpha Tomography is a form of single-axis electron tomography in which the sample is rotated in the α-direction of the stage (as shown in Figure 3.1b). This scheme has been used by several groups to generate atomic-resolution tomographic reconstructions of a Au nanoparticle [58], Pt nanoparticle [59], FePt nanoparticle [60] and amorphous Pt nanoparticles and Ta film [61]. Since

acquiring a tomography tilt series is a long, repetitive and complicated process in which all measures must be taken to minimise the electron beam dose, the capability to automate this process has been built into the software.

3.3.1 Choosing a suitable sample

In order to obtain a high-quality tilt series that will likely result in an atomic-resolution reconstruction of a nanoparticle, any sample nanoparticle must be sufficiently distant from neighbours in the direction orthogonal to the axis of tilt that the neighbours do not shadow the sample particle as the sample plate is tilted to high angles. In alpha tomography, the scan rotation is typically set so that the tilt axis is parallel to the horizontal axis of the images, so the clearance is required in the vertical direction. Since most single-axis tomography experiments on slab-like supports do not tilt beyond $\pm 70°$, the clearance h required at normal incidence ($\alpha = 0°$) for a particle of radius r can be calculated by:

$$h = \frac{r}{\cos(70)} \tag{3.1}$$

This is the distance that two particles of the same size must be separated by. If one of these particles is replaced by a large agglomerate consisting of several layers of particles stacked on top of each other, this distance must be increased (the minimum vertical clearance accepted in experiments is usually $\sim 5\times$ this value). However, as the stage rotates in α, it also inevitably moves in x, y and z, so it is necessary to select a particle that is in the vicinity of a feature that is prominent even at strong defocus and low magnification, like an agglomeration, that can act as a 'landmark' to help the operator relocate the sample particle after tilting.

3.3.2 Setting up the experiment

Once the stage movement has been calibrated, the microscope has been aligned (the stigmation can be adjusted mid-experiment so long as there is a suitable lattice structure on which to do so) and a particle of interest has been identified, the first step is set up the LM (low magnification) centering by taking a series of images of the sample particle at increasing magnifications from 5k×

to 1.3M× (the final image is taken at 1.8M×). After a step in α, LM centering cross-correlates images taken at the new α angle with those from the reference set (at one magnification step lower) for registration, determining the offset in x and y resulting from the step in α and correcting it (and using defocus slices to do the same for z). The magnification series is updated every 10° in α and uses 512×512 resolution images to speed up the process.

The second step is to generate a small database of images of the sample particle at different tilt angles (α = start value + 0.5°, 1°, 2°, 4°), the position of which can then be used to fit curves of changes in x, y and z with α. Since the LM centering has already been set up, this can be used to automatically locate the particles at the new α angles. Once this has been completed, the α range, number of projections and starting α value are all manually set by the operator in the 'Tomo setup' tab, from which the tilt series can be calculated. At this point, the experiment is ready to begin. For each step in the series, the software will use LM centering to locate the particle, acquire an image and repeats (taking repeats enables drift to be compensated for, as well as providing several backup images if one is of lower quality) and a reference magnification series for the next α, then move to the next α angle in the sequence and move in x, y and z as predicted by database. The series will terminate automatically either at the end of the specified sequence or if there is an error, such as being unable to locate the particle of interest, in which case the column valve of the microscope will close automatically. However, it is often necessary to manually pause the sequence as the tilt angle increases to adjust the stigmation value, since the change in orientation of the chopsticks slightly affects the magnetic field and perturbs the beam. In these instances, the sequence can be easily resumed from the point where it was paused.

Chapter 4

Artificial neural networks (ANN)

4.1 Artificial Neural Network (ANN) structure

Artificial neural networks consist of layers of interconnected nodes across which tasks are distributed. This is inspired by the inner workings of the human brain, which distributes operations across multiple neurons, each carrying out a simple task. Instead of passing all the information pertaining to the problem from neuron to neuron, the total information is instead encoded in the interconnection of the neural network itself with each neuron only receiving a small piece of the overall picture, thus reducing the required 'bandwidth' and increasing the processing speed for complex tasks like image recognition [62, 63].

4.1.1 Nodes

The ANN analogue of a neuron is called a 'node', the behaviour of which is again heavily inspired by their biological counterparts. Neighbouring neurons are connected by synapses: small gaps across which signals are mediated by a neurotransmitter [64]. A neuron only fires once the level of neurotransmitter surpasses a threshold value and, in the same way, a node only activates and passes the signal through an activation, or transfer, function once the weighted sum of the inputs surpasses the node's threshold value. Mathematically, this can be represented as:

$$y = f\left(\sum_{i=0}^{n} w_i x_i - T\right) \tag{4.1}$$

where y is the node output, f is the activation function, w_i is the weight of of input x_i and T is the threshold value [62]. Within the machine learning community, this equation is typically reworked to give:

$$y = f\left(\sum_{i=1}^{n} w_i x_i + w_0\right) \tag{4.2}$$

where w_0 is the 'bias', replacing the threshold value. Node biases are one of the trainable parameters of a neural network.

Activation functions translate the input signal to an output signal and can take many forms, with non-linear functions typically being more useful since few problems are linearly separable [62, 63]. Common mathematical functions that fulfil this criterion include step functions:

$$H(z) = \begin{cases} 0 & (z < 0) \\ 1 & (z > 0) \end{cases} \tag{4.3}$$

sigmoid functions:

$$\sigma(z) = \frac{1}{1 + e^{-z}} \tag{4.4}$$

and hyperbolic tangent functions [62, 63]:

$$\tanh(z) = \frac{e^{z} - e^{-z}}{e^{z} + e^{-z}} \tag{4.5}$$

These activation functions are graphically represented in Figure 4.1.

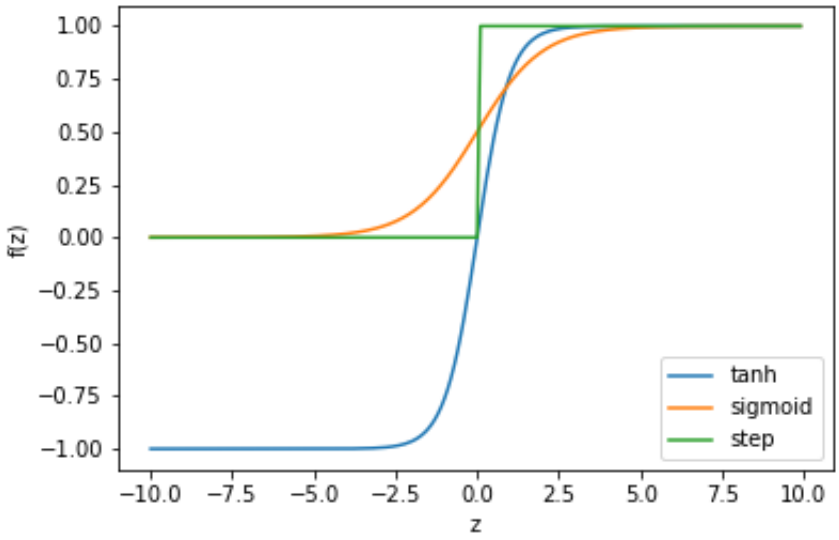


Figure 4.1: Common ANN activation functions.

In recent years, the Rectified Linear Unit (ReLU) has become the most commonly used activation function for hidden layers [65] as it is computationally inexpensive, both because it does not require the computation of an exponential function [66] and because only 50% of the neurons fire at any one time [67]. ReLU is defined by:

$$g(z) = max\{0, z\} \tag{4.6}$$

as shown in Figure 4.2. Most ANNs intended for classification employ the 'softmax' activation function in their final layer, defined by:

$$s(z_k) = \frac{e^{z_k}}{\sum_{l=1}^{K} e^{z_l}} \tag{4.7}$$

which, by design, transforms the node activations from the final hidden layer into probabilities for each label [63, 68].

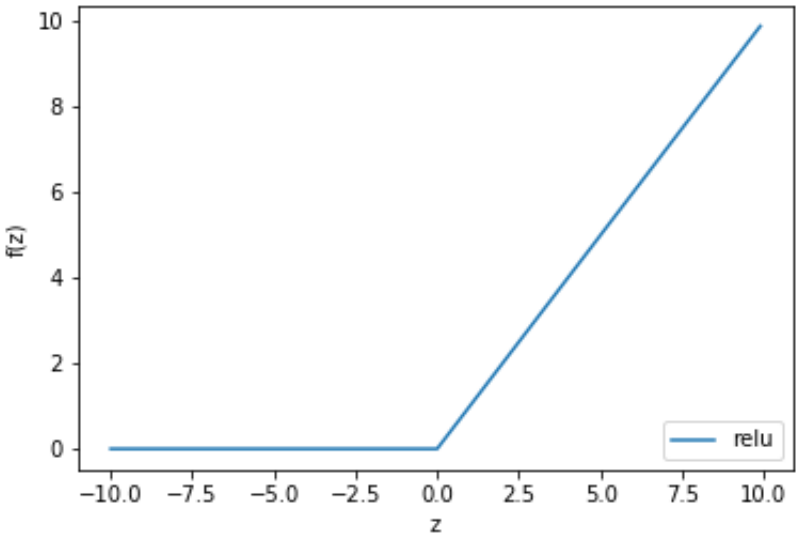


Figure 4.2: ReLU Function.

4.1.2 Layers

Nodes in ANNs are often organised into linear arrays known as layers. All ANNs have multiple layers, with most possessing an input layer, where the number of nodes equals the number of input variables, and an output layer, where the number of nodes equals the number of possible outputs (for a classification task, this would be the number of groups into which the data is to be categorised). Typically sandwiched between these layers are 'hidden layers', of which there can be any number, all with different sizes (numbers of nodes) and characteristics (type, activation functions, levels of connectivity with other layers). These parameters must be tailored to the specific task at hand, since different network architectures will be better suited to some problems than others. This is usually achieved by using the structures of pre-existing ANNs created for a similar purpose as a starting point, then optimising the architecture through experimentation [62].

Nodes in different layers are connected together to form the overall network. These connections have weights, as represented by w_i in Equation (4.2), which are trainable parameters of the network. A layer in which each node connects to every node in the previous and subsequent layer is known as a 'dense' or 'fully connected' layer [68].

Neural networks can be divided into two categories based on the ordering of connections between nodes. In a feed-forward network, the output of every layer is only ever passed forward to a subsequent layer and never back to a previous layer. In a feed-back network, some outputs are passed back to previous layers, resulting in a feedback loop within the network that changes the weights and biases over many cycles until the network reaches an equilibrium point [62]. Examples of feed-back networks include the Hopfield net — a single-layer network in which each node is connected to every other node [69] — and the Kohonen (self-organising) map, which consists of an input layer that is fully connected to a 2D array of nodes whose weights update iteratively to better adjust to the input (this is typically used for clustering) [70]. All networks discussed in this work are feed-forward networks unless explicitly stated otherwise.

4.2 ANN training

Upon initialisation, each weight and bias in an ANN is determined randomly. To train the network, data is passed through it repeatedly and the weights and biases are adjusted to try to optimise the output for the intended task. In the case of classification tasks in a feed-forward network, this typically equates to minimising a loss function, which represents the difference between the desired and actual outputs of the network. The amount of data points required to train a network effectively depends on the complexity of the problem and can range from hundreds [71] to millions [10–12], with thousands being a typical range [71]. This can limit the viability of neural networks (and machine learning in general [72]) in situations where obtaining large quantities of data is unfeasible, but this can be overcome by data 'augmentation', in which real data is duplicated and modified to create new, distinct data that is still sufficiently similar to be representative of the original sample, such as by mirroring or translating an image [10]. An alternative is to train the network on simulated data that is indistinguishable (or nearly indistinguishable) from real data.

Various feedback mechanisms exist for training neural networks. Classification problems like the work being presented here employ 'supervised learning', in which all the training data is labelled and the network is adjusted so that it correctly labels as many data points as possible. Clustering problems typically employ 'unsupervised learning', in which none of the training data

is labelled and the network is left to create its own classification criteria for grouping the data. Other examples include 'semi-supervised learning', in which some of the data is labelled and the network is left to try to extrapolate these labels and apply them to neighbouring points within the parameter space, and 'reinforcement learning', in which a software agent is trained to take actions that maximise a reward function (or minimise a cost function). [72–74]. All the examples discussed here will employ supervised learning unless explicitly stated otherwise.

In a 'forward pass' of the data through a network, the network calculates the probabilities of each data point having a specific label based on the network's current weights and biases. In supervised learning of classification problems, an error can be calculated by subtracting the probability assigned to the correct label from 1, thus turning the training process into a large-scale optimisation problem, where the error — mathematically represented by some kind of loss function — for the entire training dataset is minimised by altering the weights and biases. This occurs during the 'backward pass'. During training, data is typically passed through the network in batches, with one forward and backward pass of a single batch being known as an 'iteration'. A full run of the entire training dataset through the network is known as an 'epoch' [68].

The training of an ANN is typically evaluated using validation datasets and/or test datasets. Validation datasets usually contain data generated in the same manner as, but separate from, the training dataset, against which the network is checked at the end of each epoch, giving snapshots of the state of the network during training. This can enable the early detection of problems like overfitting (see Section 4.2.2) [75]. Test datasets are often representative of the real data that the model is training to categorise, which the training data might not be (e.g. if the training data is simulated), and are checked against once the network has finished training.

Validation datasets can also be used to tune hyperparameters: parameters that affect the speed and quality of model training [76]. One example of a hyperparameter is batch size: the number of data points per batch. Increasing batch size enables faster computation of every epoch, but slows down network convergence due to the reduced number of updates per computation, thus requiring more epochs. An intermediate value, often around 32, is usually chosen to balance these two competing effects [76]. Other examples include learning rate (see Section 4.2.1) and dropout rate (see Section 4.2.2) [76].

4.2.1 Optimisers

A number of different optimisation functions exist for neural network training, most of which are based on gradient descent (GD) methods. In GD, with every iteration (i.e. for every batch), the average of the derivatives (or gradients) of the loss with respect to the weights and biases of the network is calculated via a process known as backpropagation (see Appendix A). These gradients are multiplied by a hyperparameter known as the 'learning rate' or 'step size', a typical value for which is 0.001, giving:

$$\theta_t = \theta_{t-1} + \alpha \frac{\partial L(z_t, \theta_t)}{\partial \theta_t} \tag{4.8}$$

where θ_t is the value of a parameter (a weight or bias) at iteration t, z is the state of the network at iteration t, L is the loss of the network and α is the learning rate. The weights and biases are then altered by this value in an effort to minimise the loss of the entire network for the training dataset. The learning rate is another hyperparameter whose appropriate selection is crucial. Increasing it too much, although initially speeding up the rate at which the system moves towards its optimal state, will increase the average loss and increase the risk of overshooting the optimum state. [76].

Stochastic Gradient Descent (SGD), in which the gradients are analysed in random batches as opposed to the entire training set at once, is the most commonly encountered variant of GD. Note that the term SGD can be used to refer to both online gradient descent (batch size set to 1) and mini-batch gradient descent (batch size set to value above 1 but much lower than the size of the training dataset) [76].

4.2.1.1 Adam

Various techniques exist for dynamically adjusting the learning rate during training, one example of which is Adaptive Moment Estimation (Adam). 'Moment', in this context, refers to the expected value of a random variable, which a gradient of loss against weight or bias can be treated as since it is evaluated from a random subset of the data. An n-th order moment is the expected value of a random variable to the power of n, which can be represented as:

$$m_n = E[X^n] \tag{4.9}$$

where m is the moment, E is the expected value, X is the random variable and n is the order. The first and second moments of any random variable are its mean and uncentered variance, respectively. Adam computes exponentially moving averages of these values with every mini-batch of data:

$$m_t = \beta_1 m_{t-1} + (1 - \beta_1) g_t \tag{4.10}$$

$$v_t = \beta_2 v_{t-1} + (1 - \beta_2) g_t^2 \tag{4.11}$$

where m_t and v_t are the moving averages of the mean and variance at iteration t, respectively, g_t is the net gradient of the mini-batch at iteration t and β_1 and β_2 are hyperparameters, the default values of which are 0.9 and 0.999 respectively. When initialised with zeros, these rolling averages have a bias towards zero, which is corrected by:

$$\hat{m}_t = \frac{m_t}{1 - \beta_1^t} \tag{4.12}$$

$$\hat{v}_t = \frac{v_t}{1 - \beta_2^t} \tag{4.13}$$

The model weights are then updated by:

$$w_t = w_{t-1} - \alpha \frac{\hat{m}_t}{\sqrt{\hat{v}_t} + \epsilon} \tag{4.14}$$

where w_t is the network weights at iteration t and α, the learning rate, and ϵ, a small scalar value used to prevent division by zero if v_t vanishes to zero (which typically happens after a few epochs), are two more hyperparameters, the default values of which are 0.001 and 10^{-8}, respectively. The magnitude of the effective step-size of this equation decreases with the average of means, which will occur as the gradients decrease (i.e. as the network approaches its optimal state) [77].

4.2.2 Overfitting and Underfitting

The aim of the training process is to create a network that is capable of both classifying the training dataset and generalising to other, similar data. If the network learns too little, it will be unable to classify even the training dataset, let alone anything else; this is known as 'underfitting'. If the network trains too much, it will effectively 'learn' the training dataset and fail to generalise to other, similar data; this is known as 'overfitting'.

Underfitting can usually be fixed by increasing the complexity of the ANN, typically by increasing the number or size of the hidden layers to enable it to fit more functions to map input to output. Since this is easily solved, overfitting is the more common problem.

Overfitting can be detected using validation or test datasets. If the classification accuracy for these datasets is significantly lower than the training accuracy, overfitting has likely occurred [75]. One way of countering this is to increase the size of the training dataset, either by obtaining more data or augmenting pre-existing data. This increases the difficulty of 'learning' the dataset, thereby increasing the probability that the network will instead seek a general solution.

Alternatively, overfitting can be countered by reducing the complexity of the model, impeding its ability to learn a specific training dataset. This can be done by reducing the size and/or number of layers, or by introducing 'dropout' layers. Dropout layers randomly silence a certain proportion of nodes with every forward pass of the training data, which limits 'co-adaptation': a phenomena where individual nodes become strongly dependent on particular nodes in other layers. The dropout mechanism limits this and ensures that each node develops robust decision mechanisms based on the ensemble of nodes, rather than any specific subset [10]. The dropout rate — the proportion of nodes silenced — is a hyperparameter of the network. Increasing the dropout rate decreases the probability of overfitting at the expense of increased network convergence time and increased risk of underfitting [10].

4.3 Convolutional Neural Networks

Convolutional Neural Networks (CNN), or 'convnets', are a sub-category of ANNs that possess one or more convolutional layers. As their name implies, these layers perform convolutional operations

to find spatial and/or temporal patterns within data and, as such, are useful in situations where neighbouring values of data are correlated, like image or speech recognition [63, 73].

4.3.1 Convolutional Layers

Rather than consisting of nodes, convolutional layers are made up of trainable convolutional filters, also known as kernels, the number, dimensions (receptive field) and stride (spacing between placement of filters) of which can be varied in each layer. These filters are convolved with the input image to generate maps of features like edges, curves and gradients [63]. Subsequent convolutional operations performed on these feature maps enable the detection of higher-level features like shapes and, with enough layers, faces and abstract objects [11, 73].

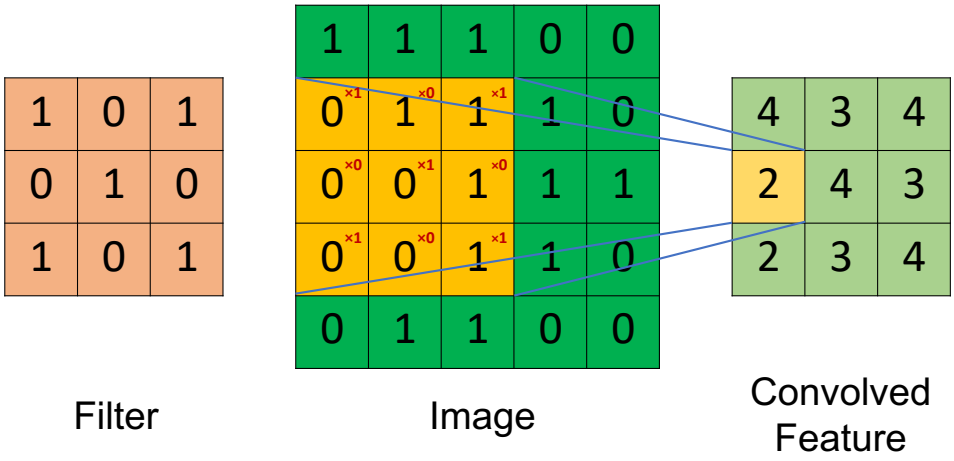


Figure 4.3: Convolution filter (size = 3×3, stride = 1) applied to an image to produce a convolved feature.

It should be noted that, due to the nature of the convolution process, the dimensions of the output feature map are always smaller than the input image, as shown in Figure 4.3. This is important when considering the behaviour of CNNs with successive convolution layers, since it means that the effective receptive field of a given filter over the original image expands with each successive layer [11]. This can be seen by considering the application of a second convolution filter with size 3×3 and stride 1 to the output (the convolved feature) in Figure 4.3, which would be drawing information from every pixel in the original 5×5 image. Padding the edges of the original

image with values can mitigate this reduction if required without affecting the mathematics of the convolution.

4.3.2 Pooling Layers

Pooling layers reduce the dimensions of an image and the sizes of features within them, decreasing the parameters (and thus computational resources) involved in the network. It can also have the added benefit of extracting dominant features, thereby simplifying the training process [78], as well as negating the impact of small spatial shifts in the input [63].

Two types of pooling are typically used in CNNs: max pooling and average pooling. As the names imply, max pooling selects the maximum value of the receptive field (as shown in Figure 4.4), while average pooling returns the mean of all values within the field. Max pooling has the added benefit of suppressing noise since it discards noisy activations altogether, which is why it is generally favoured over average pooling [78].

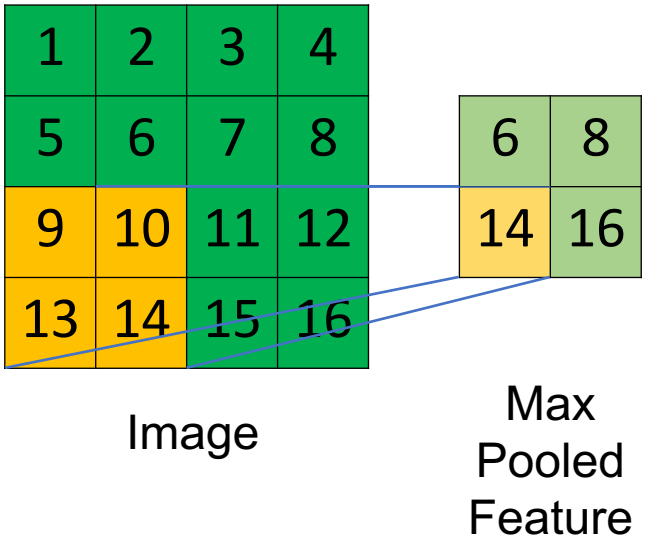


Figure 4.4: Max pooling filter (size = 2×2, stride = 2) applied to an image to produce a pooled feature.

4.3.3 Typical CNN structure

Most simple CNNs use a combination of convolutional and pooling layers, typically alternating between them, followed by one or more dense layers that use the features identified by the convolutional layers to try to classify the input, then ending with a dense layer with softmax activation, as shown in Figure 4.5 [10–12, 63].

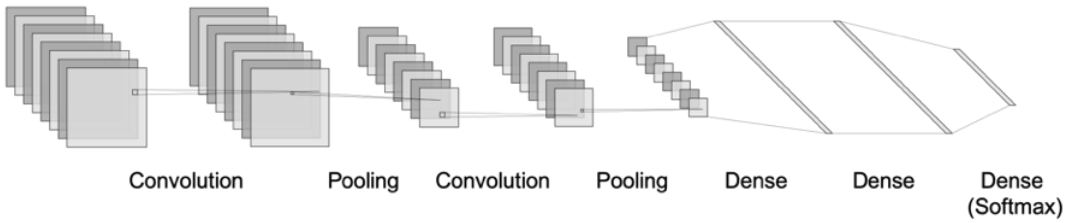


Figure 4.5: Representation of generic CNN architecture.

4.3.4 Notable image-recognition CNNs

4.3.4.1 AlexNet

In recent years, there has been a surge in the number of CNNs used for image recognition, many of which can trace their roots back to AlexNet [12, 79], a groundbreaking CNN that pioneered the use of ReLU (Rectified Linear Unit) as an activation function, demonstrating it to be quicker than the previously used tanh function, and the use of dropout layers to prevent overfitting. It consists of five convolutional layers, the first two of which were followed by a max pooling layer and a dropout layer, followed by three fully-connected dense layers. The first convolutional layer applies 96 11×11 filters to the 224×224-pixel image with a stride of 4, the second applies 256 5×5 filters and the third, fourth and fifth layers apply 384, 384 and 256 3×3 filters. The following two dense layers have 4096 neurons each, followed by the final layer with 1000 (the number of image labels). AlexNet gained wide recognition after winning the 2012 ImageNet Large-Scale Visual Recognition Challenge (ILSVRC) with a top-5 test error rate (percentage of images not identified within the top 5 guesses) of 15.3% (compared to the 26.2% error rate of the next competitor) [80], having trained on 1.2 million images [10].

4.3.4.2 ZF Net

The ZF Net, which achieved an error rate of 13.5% in the ILSVRC-2013 classification task [81], was a refinement of AlexNet whose accompanying paper outlined a new tool for visualising the behaviour of its different layers called a 'deconvolutional network' [11]. Previous work had mostly been limited to analysis of the first layer, where direct projections onto pixel space are possible. The deconvolutional network, however, can reverse the convolutional operations applied to reach any point in the network, enabling the behaviour of nodes in layers beyond the first to be studied. This analysis demonstrated that the convolutional filters in the first layer are typically low-level feature detectors, like edges and curves, with filters in subsequent layers activating for progressively more complex and abstract features, from simple shapes right up to, in the case presented in the paper, dog faces and flowers. Using this tool, key features of the ZF Net were optimised. The 11×11 filter with a stride of 4 used in the first layer of the AlexNet was found to be ignoring a lot of relevant pixel information from the initial image, so this was replaced by a 7×7 filter with a stride of 2, with the successive four convolutional layers all employing 3×3 filters, the first with a stride of 2 and the following three with a stride of 1. It is important to remember that, due to the downsizing effect of both the convolutional and pooling layers, successive filters, despite being the same size, draw on information from progressively larger regions of the original image, which tallies with the notion of deeper layers recognising more complex structures [11].

4.3.4.3 VGG Net

The VGG Net developed in 2014 took a slightly different approach, favouring depth over individual layer complexity. It consists solely of convolutional layers with 3×3 filters and a pad and stride of 1 and 2×2 max pooling layers with a stride of 2. Various network architectures were designed and tested, with sequences of multiple convolutional layers without any spatial pooling layers between them, the reasoning for which is that placing several filters sequentially increases their effective receptive field (2 3×3 filters with a stride of 1 have a receptive field of 5×5, 3 have a receptive field of 7×7, etc.). When comparing against the use of single large filters in AlexNet and ZF Net, this method has several advantages, including a reduction in the number of computational parameters and a more discriminatory filter, since multiple rectification layers are used instead of just one. The

VGG Net also introduces the practice of doubling the number of filters after each pooling layer, thereby doubling the number of features extracted in every layer and greatly increasing the final number of high-level features used for classification. In training, the VGG Net converged three times faster than the AlexNet, despite its larger number of overall parameters and greater depth, which its creators conjectured was, at least in part, due to the improvements in discrimination resulting from the stacking of smaller filters [12].

Chapter 5

Creating tungsten tips for electron tomography (ET)

5.1 Introduction

Tomography, derived from the Greek word 'tomos', meaning slice or section [82], is the acquisition of three-dimensional information about a sample's internal structure using two-dimensional images (slices) captured using a penetrative imaging technique [83]. Medical imaging is one of the most notable applications of tomography, where it comes in various forms, such as computerised tomography (CT) and computerised-axial tomography (CAT), which image the body using X-rays [84], positron emission tomography (PET), which detects the gamma rays produced by the annihilation of electrons and positrons emitted by radioactive isotopes injected into the body [85], and single-photon emission computed tomography (SPECT), which detects the gamma rays produced by gamma-producing isotopes [86]. Transmission electron microscopy (TEM), as previously mentioned, examines the internal structure of samples by measuring the transmission of electrons through it and is therefore compatible with the concept of tomography, a partnership known as electron tomography (ET).

The application of tomography in TEM can be traced back at least as far as the late 1960s,

with studies of gold nanoparticles [87] and the tail of bacteriophage T4 [88] using multiple images taken at different angles both being published in 1968. Subsequent improvements in instrumentation, advances in computational power and the development of better reconstruction algorithms have all contributed to the development of electron tomography [7], with numerous tomographic studies of both 'soft' biological samples [47, 49, 89–91] and 'hard' material samples [46, 59, 92, 93] being published. More recently, atomic-resolution electron tomography (AET) has been made feasible by the development of aberration correction and its application to scanning transmission electron microscopy (STEM). A handful of atomic-resolution tomographic reconstructions have been published in recent years, including a 784-atom Ag nanoparticle [94], the top nine atomic layers (3,769 atoms) of an electrochemically etched W tip [95] and a 23,196-atom FePt particle in which the element of each atom was also identified [60].

As discussed in Chapter 2, the 'missing-wedge problem', where samples mounted on slab-like supports like TEM grids become shadowed at high angles by the supports themselves, making imaging at these angles practically impossible, causes features in the tomographic reconstruction to be stretched along the original projection direction [7, 54]. One solution to this problem is dual-axis tomography, in which two tilt series are taken around different axes and the information from the two used to create a single reconstruction. This technique has been used to create reconstructions of gold nanoparticles [9], CdTe tetrapods [46] and carbon nanotubes [47], but the acquisition and processing of data using this method is significantly more complicated than for single-axis tomography. Obtaining two tilt series around different axes requires either a dual-tilt holder or the manual rotation of the sample after extraction from the microscope and will increase the electron dose that the sample is exposed to due to the greater number of images, increasing the risk of knock-on damage to the sample. Additionally, the alignment and reconstruction of these tilt series is inherently more complicated than for single-axis tilt series, which is itself a complicated task [7, 9, 46, 47].

Another way of solving this problem is to use needle-shaped samples and supports. In this setup, the needle is inserted perpendicular to the direction of the beam and a sample mounted on the apex or the apex itself is imaged as the needle is rotated about its own axis, as shown in Figure 5.1. Since the sample thickness is (roughly) the same for all viewing angles, all projections

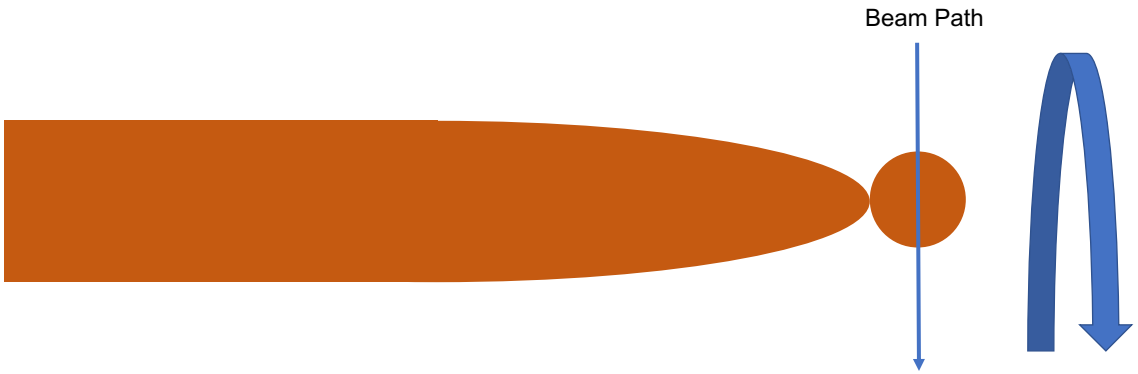


Figure 5.1: Path of the electron beam through a sample mounted on a needle-like support.

will be of equal quality, regardless of tilt angle, and a larger lens gap is not required since the thickness of the sample holder does not change with angle either. This makes the acquisition of single-axis tilt series of up to ±90° feasible, which solves the missing-wedge problem and simplifies the subsequent processing and reconstruction of the data. This setup also has the added bonus of having no background contributions from the sample support as are present when using a TEM grid, which improves the signal-to-noise ratio [54]. Przybilla et al. used this method to acquire a tilt series of a 2µm zeolite particle transferred onto a tip using a FIB/SEM pickup system [96]. The tilt series for the aforementioned reconstruction of the top nine layers of a tungsten tip performed by Xu et al. was also acquired by this method [95], as was the tilt series for the atomic-resolution reconstruction of a 5nm gold nanoparticle mounted on the apex of a crystalline tungsten tip performed by Poon [54]. Tungsten is an ideal candidate for this task due its exceptionally high knock-on damage threshold (over 500 keV [42, 43]), which means that it a stable platform for tomography. Additionally, the science of electrochemically etching tungsten wires to create sharp tips is well-understood, primarily due to its usage in the manufacture of probes for scanning probe microscopy (SPM) and electrodes for field-emission guns [7, 20, 97].

The ultimate goal of this work is to be able to use tungsten tips as a platform for atomic-resolution tomography of any arbitrary sample, as was done in the case of the gold nanoparticle on the tungsten tip. Creating such samples is practically difficult, however, as a number of stringent conditions must be fulfilled:

- The diameter of the reconstructed volume (i.e. the sample) must be small (5-10 nm). This is because, in order to obtain the same resolution, the amount of data that needs to be taken increases as the size of the reconstructed volume increases. The amount of data corresponds to the number of projections at different angles, but, as the angular increments between projections decreases, so too does the amount of new information that they provide, until eventually, adding new projections stops adding information. This limits the size of the volume that can be reconstructed.

- The tip must remain straight. If the tip bends off-axis, the sample may become shadowed by the puck at high alpha angles ($\alpha = 70°$), resulting in a missing wedge. Tips should not bend more than $\sim \pm 15°$.

- The sample must be mounted on the apex. If the sample is not mounted on the apex, it will become shadowed at certain angles, resulting in a missing wedge.

Ideally, particles could be transferred to the apex of a tip using a pickup system like the FIB/SEM used by Przybilla et al. [96]. That particular system, however, has insufficient resolution to handle samples small enough for atomic-resolution reconstruction. One proposed solution is to use some sort of scanning probe microscope in which the tungsten tips pick up particles deposited on a flat surface, but this system is still in development and has not yet been shown to be feasible [54]. Another simpler method is to dip the tips in a colloidal solution, but, due to the relatively small surface area of the apex, the particles would be more likely to stick to the sides of the tip rather than the apex. At present, samples are created by evaporating metal onto the apex of an annealed tip, then annealing the tips again so that the evaporated metal forms nanoparticles. This is an extremely delicate process, however. If the etching cut-off is not perfect, the tip may etch for slightly longer than desired and become blunt. If this happens, the size of the apex's surface may become too large to guarantee that a particle sits on the 'true' apex. If the apex of the tip is knocked against any solid object after etching, it will bend and no longer be straight. If the tip is overheated during either annealing stage, the atoms in the apex will become mobile and form the shape with the lowest energy — a sphere (or hemisphere) with the largest possible radius — which, put simply, will make the apex blunt. As such, tens or hundreds of tips may need to be created

before a usable sample emerges, which can take several months, so relying on being able to create such samples during short visits to NCEM — the facility in which the TEAM 0.5 microscope used for tomography is based — is impractical. Instead, samples are created in our laboratories and shipped to NCEM, but the time required for such transportation means that there is a significant risk of sample oxidation. One solution is to use small vacuum storage units, but these tend to be bulky (relative to the size of the samples that they hold), expensive and must be carefully handled to avoid breaching the vacuum during transport. Another solution is to coat the tips in a thin layer of another material to block oxygen atoms from the sample. This material should either be easily removable or have a low atomic number so that they have minimal Z-contrast during imaging.

This chapter demonstrates the processes involved in creating tungsten tips that are suitable sample platforms for electron tomography, many of which have been partially or wholly automated. The process of creating a suitable tomography sample by evaporating metal onto the apex of a tip is also described, as are the steps and equipment needed to mount the tips into the pucks used to manipulate samples in the TEAM 0.5 and ship them in suitable containers. Finally, the results of two experiments in which the tips are coated in two materials — bismuth and carbon — in an effort to reduce sample oxidation are outlined.

5.2 Creating Tungsten Tips

5.2.1 Recrystallisation

Commercially available tungsten wire is typically cold-drawn and polycrystalline, with grain sizes on the order of micrometres. During electrochemical etching, the least stable atoms, like those in the vicinity of defects or on certain crystal faces, are preferentially etched, which creates rough surfaces [97]. Thus, the apexes of tips etched from polycrystalline wire tend to have diameters of the order of $\sim$100 nm and multiple protrusions on the surface [97], neither of which is ideal for the analysis of the relatively small samples used in AET, which tend to be 5-10 nm in size [54, 60, 95].

An ideal tip for tomography is one that is smooth, so that the sample more readily binds to the apex, and has roughly the same apex diameter as the sample. This can be achieved by etching monocrystalline wire, which produces smooth, sharp tips with apex diameters as small as

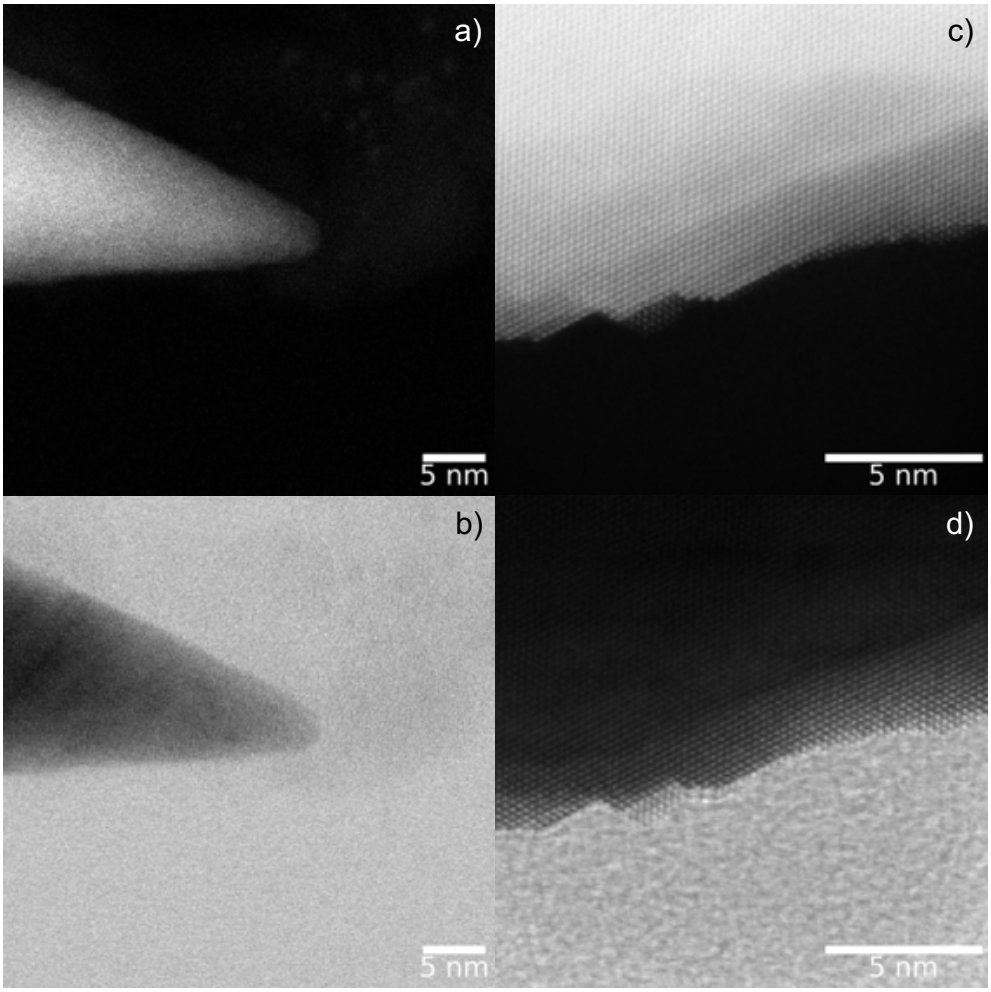


Figure 5.2: a) Crystalline faceting of the side of a recrystallised W tip (ADF). b) Crystalline faceting of the side of a recrystallised W tip (BF). c) W tip with apex diameter of 5 nm (ADF). d) W tip with apex diameter of 5 nm (BF).

5 nm, as shown in Figure 5.2 a and b. Such tips have numerous parallel step edges and apexes that can look polyhedral, as shown in Figure 5.2 c and d. While single-crystal tungsten wire is 10,000 times more expensive than ordinary tungsten wire, annealing (heating to a temperature where substantial solid-state diffusion takes place) cold-drawn wire can increase the grain size to the order of millimetres. The first stage of this process is called 'recovery', in which the densities of the point defects and dislocations introduced during the deformation (cold-drawing process) are reduced. 'Recrystallisation' follows on from this, with new, defect-free grains nucleating and

growing to consume and replace the recovered microstructure, a process driven by the stored energy resulting from deformation. The temperature required for this to take place can be decreased by increasing the strain applied to the wire as this increases the stored energy within the wire. Previous studies have found that 0.25 mm tungsten wire recrystallises at 1700°C when no tension is applied [97] and that 0.18 mm tungsten wire recrystallises at 1100-1200°C when high tension is applied (strain = 7.7) [98]. The temperature required can also be decreased by increasing the duration of annealing as the mechanisms that govern recrystallisation are thermally activated (the temperature dependence of the recrystallisation rate is given by the Arrhenius equation). Once recrystallisation is complete, the size of grains may continue to increase via 'grain growth', in which the driving force is the reduction of energy stored in grain boundaries [99]. For bcc materials like tungsten, the cold-drawing process aligns ⟨110⟩ direction of the fibre texture to the axis of the wire and this texture is maintained after recrystallisation [100]. This predictable crystal structure means that the tip itself can be used to determine the tilt angle at any given position without directly imaging the region of interest, thereby decreasing the electron dose applied to said region [54]. In practice, the faceting of the apex also creates an atomically flat surface to which the sample particle can more readily bind.

To recrystallise the wire, 5 cm sections of 0.01" (0.25 mm) diameter tungsten wire (99.95% purity, Alfa Aesar) were fixed under tension between two copper clamps mounted on a two-pin electrical feedthrough. These clamps were held apart by springs and electrically isolated from different pins by ceramic split bushes. The wire was then resistively heated under vacuum ($\sim 10^{-5}$ mbar), with a starting current of 3.9 A and a ramp rate of 0.2 A every 30 seconds. As per the Hall-Petch relationship, the increase in grain size resulting from the annealing process decreased the yield stress of the wire [101], meaning that the force of the springs ultimately caused the recrystallised wire to break in a ductile manner [102] and separate into two pieces, both with recrystallised ends. This entire process was controlled by a custom-written program on an Arduino Mego 2560 microcontroller and the final breaking current was typically around 8.0 A. The exact temperature that this corresponded to was not recorded in this work.

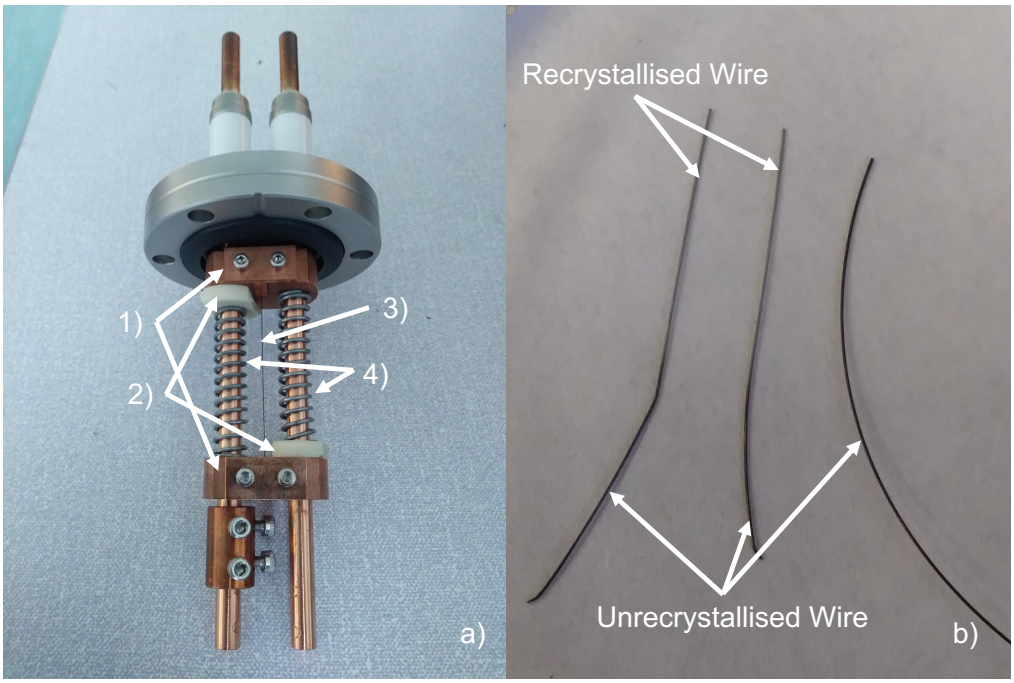


Figure 5.3: a) Recrystallisation flange (1) Copper Blocks. 2) Ceramic split-bushes. 3) Wire. 4) Springs.). b) Comparison of recrystallised and unrecrystallised wire. Recrystallised portions of the wire appear silver while the normal wire appears darker.

5.2.2 Etching

The process of electrochemically etching wires to produce sharp tips is well-documented and widely used in SPM [103]. The half-equations for the reaction are given by equations 5.1 (anode) and 5.2 (cathode) [103].

$$W(s) + 8OH^- \rightarrow WO_4^{2-} + 4H_2O + 6e^- \tag{5.1}$$

$$6H_2O + 6e^- \rightarrow 3H_2(g) + 6OH^- \tag{5.2}$$

The overall equation is:

$$W(s) + 2OH^- + 2H_2O \rightarrow WO_4^{2-} + 3H_2(g) \tag{5.3}$$

It is the meniscus that forms when the tungsten wire is inserted into the hydroxide solution (typically NaOH or KOH) that results in the formation of a needle shape. The concentration of hydroxide ions decreases up the meniscus, meaning that the rate of etching is highest at the bottom of the meniscus and lowest at the top. Additionally, throughout the etching process, tungsten anions etched off the top of the wire flow down into the solution, effectively shielding the immersed wire from the reaction. Therefore, etching occurs fastest at the level of the solution and it is here where the tip is formed [103].

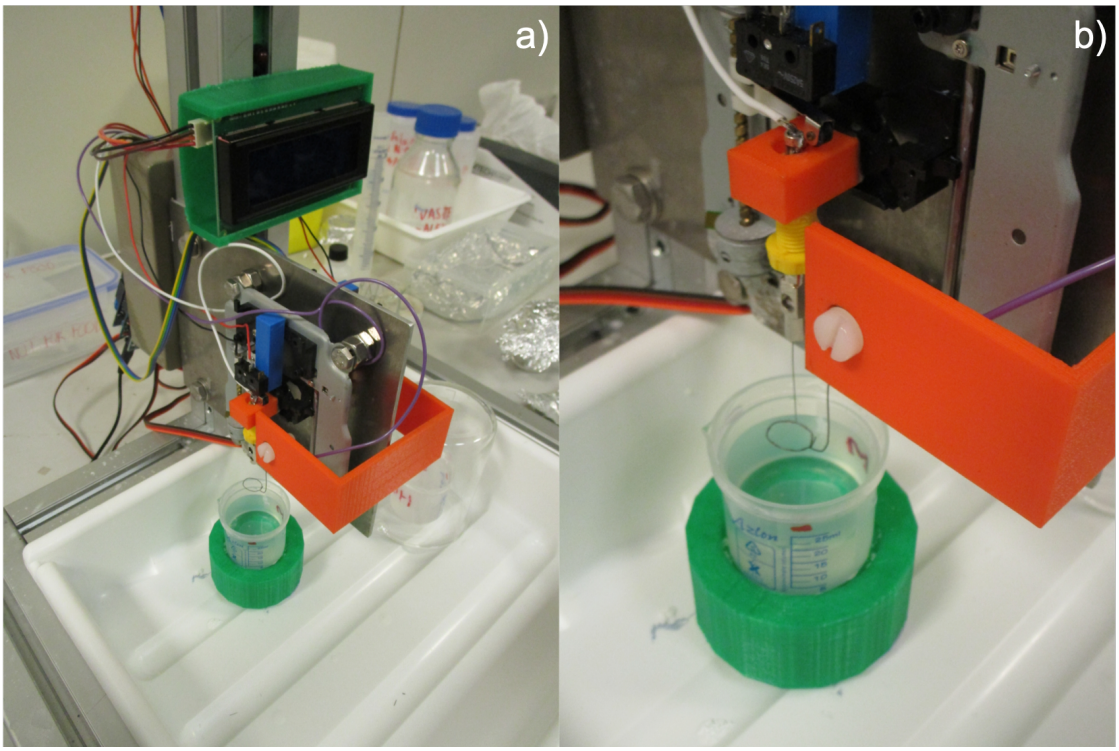


Figure 5.4: a) Etching Station. b) Loop Electrode.

As well as having a sharp apex, a small neck needed to be created in order the tip to fit into the pucks that hold samples in the TEAM stage. Additionally, a notch was etched 10 mm above the final apex so that the tips could be broken off the wire, ensuring that the tip length was consistent and reproducible. To achieve this, a custom-built etching station, as shown in Figure 5.4a, and two-stage etching process were employed. Unlike standard etching stations, in which the counter-electrode is a piece of wire, the counter-electrode in this setup was a loop, as shown in Figure 5.4b,

enabling the notch-etching at the back of the tip. The movements of the loop electrode and tip in each stage were all controlled by a specially programmed Arduino Mego 2560 microcontroller, only requiring human intervention to wash the tip between stages and change the NaOH solution used.

The first stage used 5M NaOH to speed up the overall etching process. 2 mm of the recrystallised wire were also etched off at the start to ensure that the shape of the break during the recrystallisation process did not affect the outcome. Once this was done, the loop electrode was pulled out of the solution, creating a bubble-like film inside the loop that was used to etch the notch that denotes the back of the tip. The loop was then dipped back into the solution and etching recommenced in the standard fashion. The front 2.375 mm of the tip was etched until the wire diameter was reduced to 230 µm, which was the diameter required to fit into the puck, after which the front 2.125 mm was etched to a diameter of 100 µm to reduce the duration of the second stage. Once this stage was complete, the tip was washed under a tap to remove any NaOH crystal residue that may have formed on the wire.

The second stage used 1M NaOH to produce a sharp tip. The tip was dipped 2 mm into the solution (where the tip had already been thinned) and was etched until the section beneath the meniscus broke off, cutting the circuit and ending the etching immediately. Like standard scanning tunnelling microscopy (STM) tip etching stations, the etching electronics used MOSFETs to break the connection, since the Arduino controller cut-off speed was insufficient to stop further etching, which would reduce the sharpness of the tip. Figure 5.5 shows how the tips appeared after the two stages.



Figure 5.5: Tungsten tips after (top) first-stage etching and (bottom) second-stage etching.

5.2.3 Annealing

Once etching was complete, the tip was annealed by direct-current heating under vacuum ($<5\times10^{-6}$ mbar) to remove the oxide layer that commonly builds up during etching [104]. The tip was placed in a steel clamp mounted on one pin of a two-pin electrical feedthrough and electrically connected to the other pin via a 0.4mm tungsten filament that touched the tip higher up. A K-type thermocouple was used to measure the tip temperature during heating, with the two thermocouple wires connected to the tip (but not directly to each other) above the filament (see Figure 5.6). Care had to be taken not to heat the tip too high, however, since this would enable diffusion of the atoms in the tip and blunt the apex. Testing showed that heating to 850°C for 5 seconds was generally sufficient to clean the tip without causing significant restructuring of the apex.

5.2.4 Sample creation

To create films or particles on the tip, metal atoms were thermally evaporated onto the apex under vacuum ($<5\times10^{-6}$ mbar) and the tip was then annealed again. Annealing was done using the same flange as before, while evaporation was carried out by a two-pin electrical feedthrough with a 0.01" (0.025mm) tungsten wire filament running between the two pins, as shown in Figure 5.7c. This filament had a bend in the centre into which a small bead of metal had been melted (a small piece of wire was pressed into the filament with pliers, then the filament resistively heated to melt it). Upon resistive heating, this bead melted and atoms evaporated. A separate filament was created for each type of metal used. Originally, the tip was placed in a clamp mounted to one pin of the feedthrough and the apex of the tip placed 5 mm directly beneath the metal on the filament. This required the tip to be taken out of vacuum between tip annealing and evaporation, as well as between the evaporation and the subsequent annealing stage, which, on top of slowing down the overall preparation process due to the requirement for two extra pump-down cycles in the vacuum chamber, increased the risk of oxidation and contamination of the tip at every stage. To eliminate this risk, the vacuum chamber was reconfigured to accommodate both the annealing and evaporation feedthroughs at right angles to one another, with the metal on the evaporation filament being placed 5 mm above where the apex of the tip sits in the annealing flange, as shown

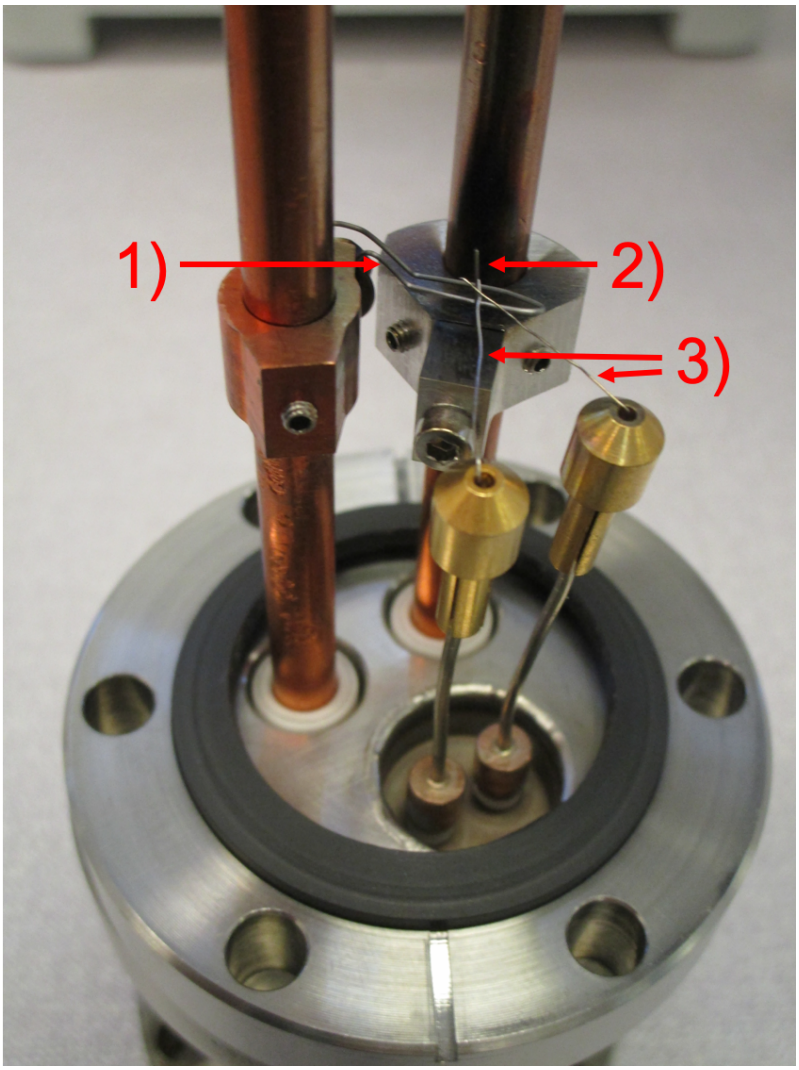


Figure 5.6: Annealing Flange. 1) Heating filament. 2) W tip. 3) Thermocouple wires.

in Figure 5.7a and b. This allowed the evaporation to be done immediately following annealing (after a suitable cool-down period) and post-evaporation annealing to be performed immediately after that.

The exact evaporation conditions used depended on the metal being evaporated, due to the differing melting points of different metals and the length of the individual filaments, which differed slightly between successive filaments, since longer filaments reached higher temperatures with less current. Additionally, certain metals, like platinum, alloyed with tungsten and shortened the

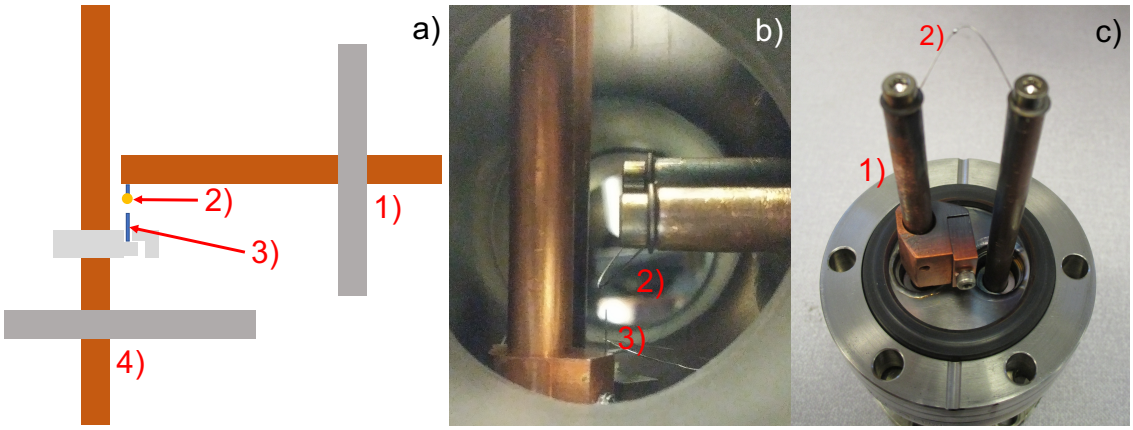


Figure 5.7: a) Diagram of evaporator and annealing flange setup. b) Picture of evaporator and annealing flange setup. c) Evaporator. (1) Evaporator (two-pin electrical feedthrough). 2) Metal bead (on tungsten filament). 3) W tip (in stainless steel clamp). 4) Annealing flange (two-pin electrical feedthrough with thermocouple).)

lifespan of the filament, so, to mitigate this, filaments with rounder bends were used to avoid creating a specific weakness within them. The minimum current required for evaporation was determined experimentally for each individual by increasing the current 0.1 A at a time, which was then used for all subsequent runs with that filament. The final thickness of deposited metal was varied by adjusting the evaporation time.

5.2.5 Sample imaging

Sample tips were imaged using a JEOL 2100-F field-emission microscope in STEM mode at 200 keV. A specialised sample plate was designed and built to accommodate these tips, with a linear groove into which the tips could lie and be clamped without slipping from side to side, as shown in Figure 5.8. The plate was made out of titanium and the spring was made out of molybdenum.

5.2.6 Pucking, Back-etching and Packaging

Once the samples had been created, the tips were inserted into the pucks that the TEAM 0.5 uses to manipulate samples using the specially designed tool shown in Figure 5.9a. Pucks were mounted in a pair of chopsticks in the central gap of the tool and tips were pushed through the holes. This had to be performed under an optical microscope, since the scales involved are of the

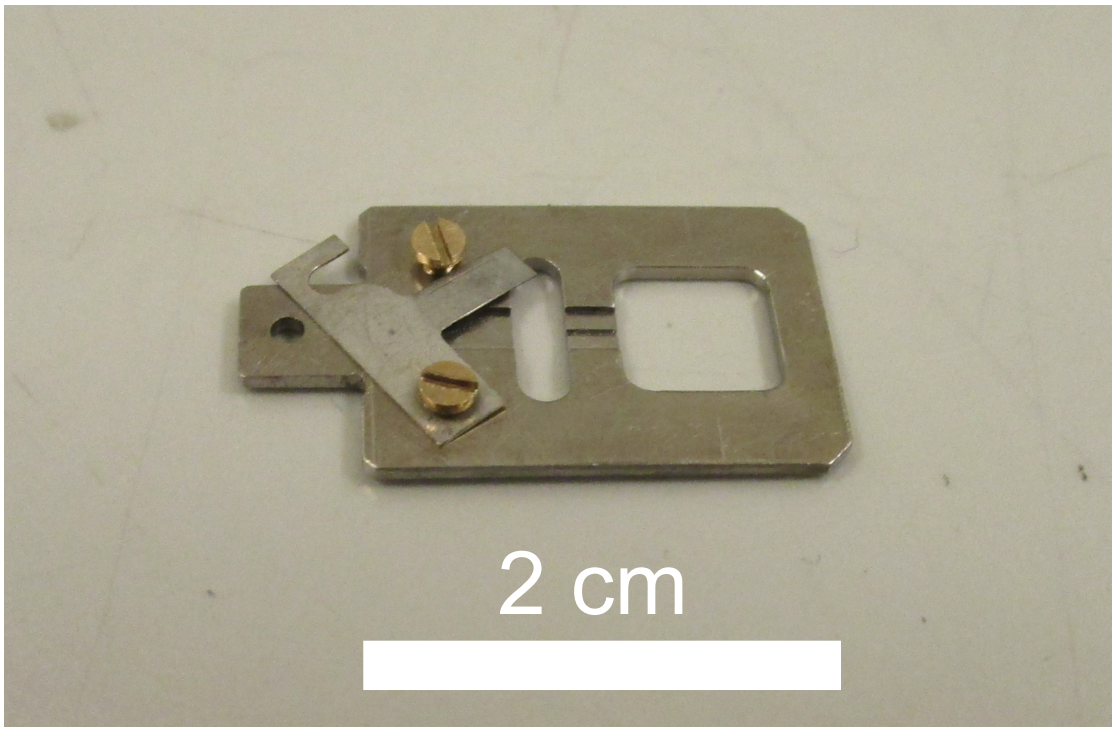


Figure 5.8: STEM tip sample plate for JEOL 2100-F.

order of millimetres.

To ensure that the tip-puck assembly was small enough to fit within the 2.5 mm pole piece gap of the TEAM 0.5, the backs of the tips had to be removed. They could not simply be cut with wire cutters as this would generate a shockwave that would propagate through the wire and resonate at a certain diameter of the neck, damaging the apex. Laser cutting would generate similar shocks due to heat, so the backs were simply etched off instead. This was done by taking the pucks out of the chopstick holders, attaching a hemispherical plastic cap to the backs of the tips and then etching them in 5M NaOH until the wire broke off.

Once back-etched, the pucks were mounted back into chopstick holders that were then themselves placed into the 3D-printed plastic shipping containers shown in Figures 5.9b and c. The chopstick holders were held in a clamp that was tested to ensure that the holders could not accidentally slip out, as shown in Figure 5.9c. The caps screwed onto the bases courtesy of the screw threads in both and the sidebars, the height of which was identical to the height of the chopsticks

when the holder was placed in the clamp, ensured that the tip was not accidentally touched while the cap was being screwed on, whilst still leaving enough space for a pair of tweezers to easily fit in to extract the holder. The depth of the hole inside the cap was designed so that, when the chopstick holder with chopsticks was placed into the cap and the cap screwed on, there would be a clearance of 0.5 mm between the top of the chopsticks and the 'roof' of the cap. This ensured that the cap did not apply any pressure to the chopsticks that may cause them to part and the puck to slip out, but also ensured that the puck, which was 2 mm in diameter, could not slide down the chopsticks and slip out of its own accord. The same 0.5 mm gap existed between the tops of the sidebars and the roof of the cap, which ensured that the two did not rub together and cause small pieces of plastic to break off that could damage the tip. Prior to use, the containers were sonicated to ensure that any loose material left behind by the printing process was removed.

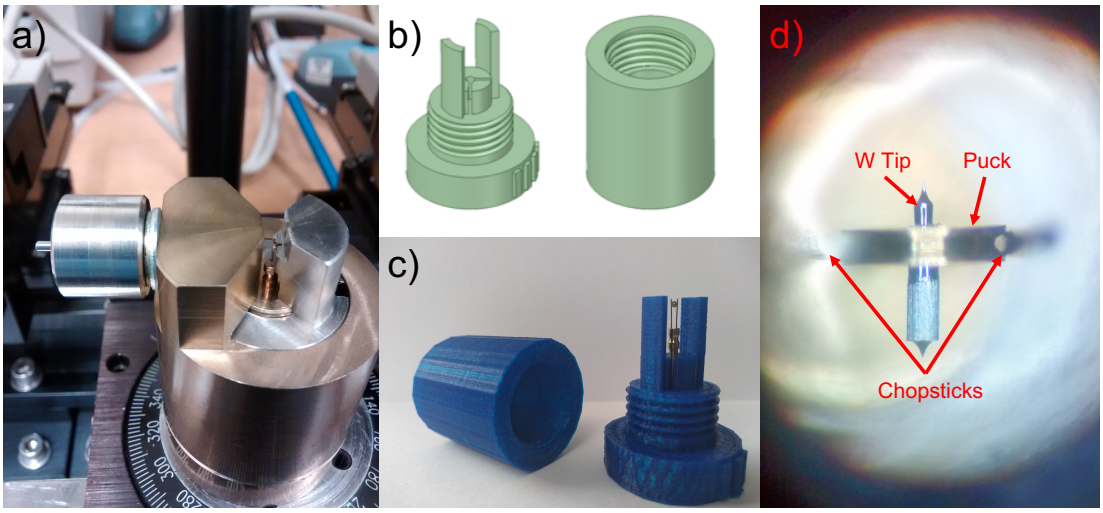


Figure 5.9: a) Pucking setup (image courtesy of Dr Wolfgang Theis). b) CAD design of shipping container. c) 3D-printed shipping container with chopsticks holder and puck mounted in the clamp. d) Etched tungsten tip mounted on a puck held by chopsticks (image courtesy of Dr Wolfgang Theis).

5.3 Bismuth coating

Bismuth is a metallic element ($Z = 83$) that, in its pure form, is a hard, brittle and lustrous crystal with red-tinged grey-white colour. It has a low melting point of 271.3°C [105], which was

the characteristic of interest for this study. The idea for this test was to evaporate bismuth under vacuum ($<5\times10^{-6}$ mbar) onto a sample tip, transport it, then gently heat the tip to evaporate the bismuth. Since the melting point of bismuth is considerably lower than the temperature at which the tungsten tips had been found to deform (∼900-1000°C), the belief was that the bismuth could be evaporated without damaging the sample, therefore making bismuth coating a simple and effective method of reducing oxidation.

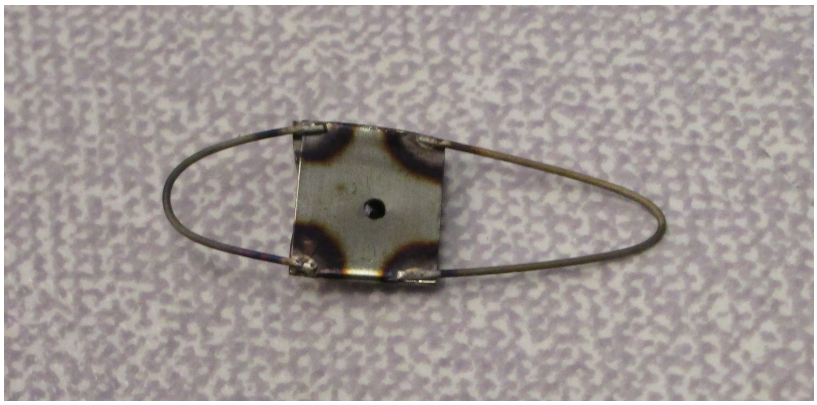


Figure 5.10: Bismuth evaporation pocket (mounted between the two pins in the evaporation flange in Figure 5.7c).

To evaporate bismuth onto the tips, a shard of bismuth was contained inside a molybdenum pocket into which a 0.5 mm aperture had been drilled. This pocket was suspended by 0.01" tungsten wire between two pins of an electrical feedthrough, as shown in Figure 5.10, and the tip mounted directly beneath the aperture. The evaporator was then placed in a vacuum chamber and heated until a layer of bismuth was seen on the tip, with testing revealing that a minimum current of 15.8 A was required to evaporate bismuth with this setup and 5 seconds being an appropriate deposition time. After imaging to confirm the presence of bismuth, the tips were then heated to remove the bismuth and imaged again. In these tests, the tip was initially coated with a small layer of rhenium before the bismuth was deposited. The bismuth removal was tested at three different temperatures — 530°C, 670°C and 720°C — for 40 seconds, the results of which are shown in Figures 5.11, 5.12 and 5.13.

As shown in parts c and d of Figures 5.11, 5.12 and 5.13, the bismuth was successfully deposited onto the tip. Contrary to expectations, however, the bismuth failed to cleanly evaporate from the

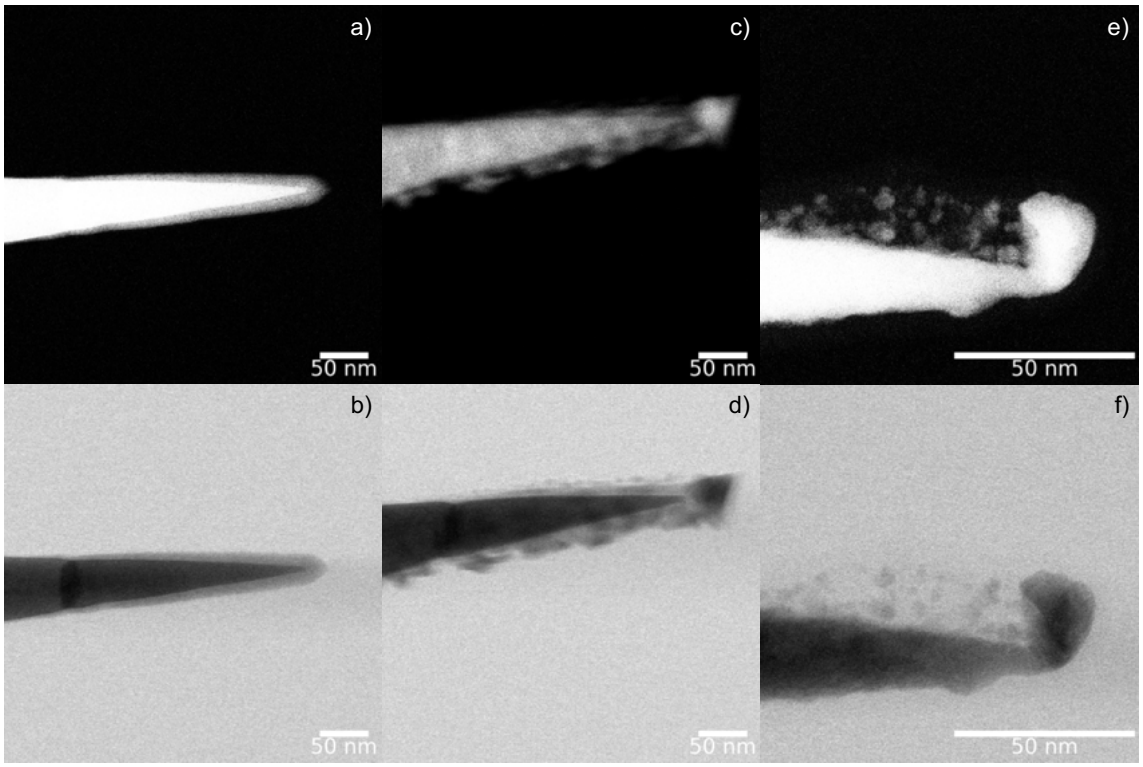


Figure 5.11: Images for Bi removal test (720°C, 40 seconds). a) Rh-coated W tip (ADF). b) Rh-coated W tip (BF). c) Bi-coated Rh-coated W tip (ADF). d) Bi-coated Rh-coated W tip (BF). e) Tip after heating (ADF). f) Tip after heating (BF).

tips in any of the three cases. As shown in Figures 5.11 and 5.12, even at 720°C and 670°C, below the temperature at which bare tips had been seen to deform, the Bi-coated tips underwent significant structural changes as a result of the heating, resulting in the deformation seen in Figure 5.11 and the apparent removal of the Rh film and/or possible alloying of the Bi and/or Rh with the W tip seen in Figure 5.12. At the lower temperature of 530°C, the bismuth partially evaporated without deforming the tip, but small crystals of it were still present, as shown by the bright spots in Figure 5.13e. This failure to fully evaporate makes this method unsuitable for use in tomography samples, since these crystals would shadow the sample at all angles and lower the signal-to-noise ratio of the images, thereby lowering the resolution of the final reconstruction.

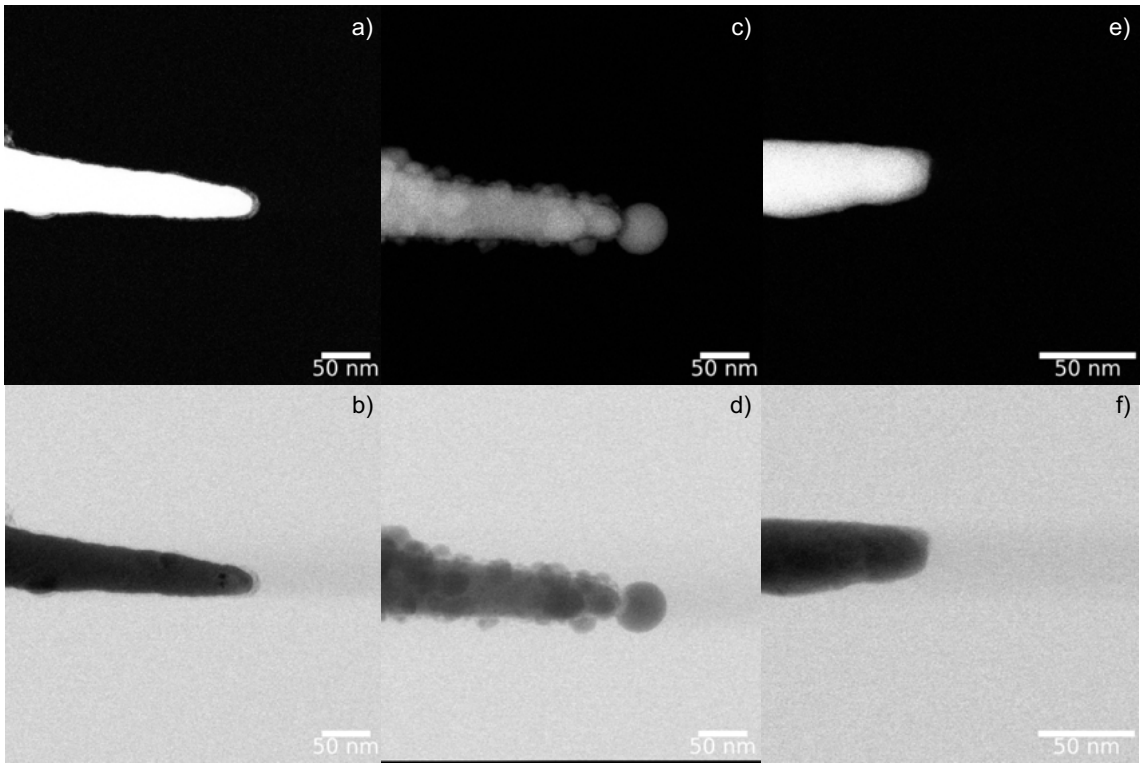


Figure 5.12: Images for Bi removal test (670°C, 40 seconds). a) Rh-coated W tip (ADF). b) Rh-coated W tip (BF). c) Bi-coated Rh-coated W tip (ADF). d) Bi-coated Rh-coated W tip (BF). e) Tip after heating (ADF). f) Tip after heating (BF).

5.4 Carbon coating

In HAADF-STEM, the intensity of carbon is much lower than metals like tungsten due to its low Z, which makes it an interesting candidate for coating the tips. Unlike bismuth, a carbon layer, if sufficiently thin, may not need to be removed prior to imaging due to its low intensity relative to the sample metal. If this is the case, this method offers another advantage over both the vacuum container transport method and the bismuth coating method, which is that the tips are never exposed to air, even when being transferred from the container to the microscope, since the carbon film is remains present at all times. Carbon coating is already employed at NCEM to mitigate the effect of radiation damage in samples [106] as it suppresses the activity of catalytic samples (e.g. AuNP, PtNP), which may be catalysing the radiolysis of surface-mobile hydrocarbons caused by electron-beam irradiation.

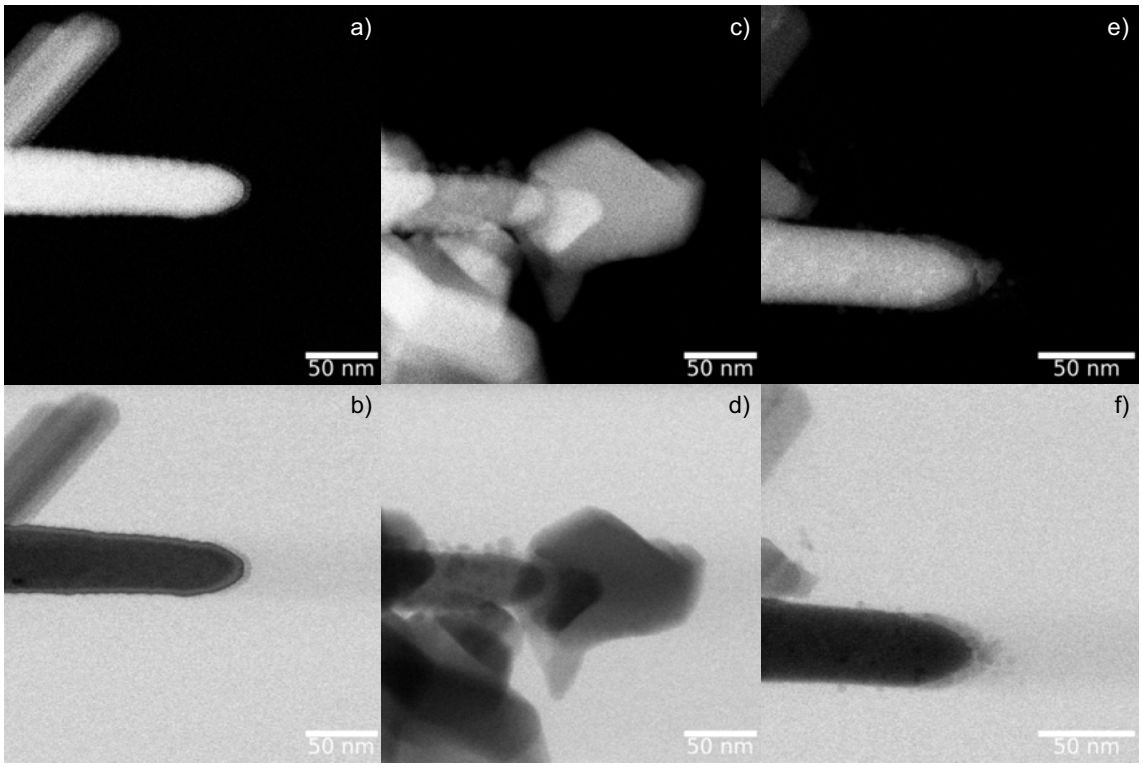


Figure 5.13: Images for Bi removal test (530°C, 40 seconds). a) Rh-coated W tip (ADF). b) Rh-coated W tip (BF). c) Bi-coated Rh-coated W tip (ADF). d) Bi-coated Rh-coated W tip (BF). e) Tip after heating (ADF). f) Tip after heating (BF).

To create a carbon film on the apex of the tip, an electron-beam evaporator was used to evaporate carbon atoms onto the apex of the tip under vacuum ($<5\times10^{-6}$ mbar). The tip was placed on a rotating flange so that the apex was above the evaporator aperture. With a shutter in place, the evaporator's voltage and current were ramped up to the target voltage and current before the shutter was then withdrawn to fully expose the aperture and deposit carbon atoms for a set amount of time, after which the shutter was put back in place and the system ramped down. Since the tip was mounted at a 45° angle to carbon evaporator, the tip was rotated smoothly on a rotary feedthrough during the evaporation period to ensure a uniform deposition of carbon across the apex of the tip. The diagram for the vacuum chamber setup is shown in Figure 5.14.

In an early test, carbon was deposited onto one side of a Pt-coated tip for 30 seconds and the other side for 60 seconds using an evaporator voltage of 1.5 kV, an evaporator filament emission

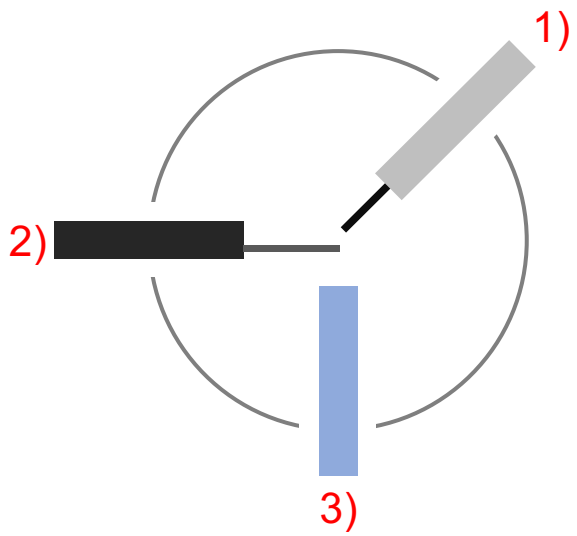


Figure 5.14: Diagram of vacuum chamber setup for carbon evaporation. 1) W tip on rotating flange. 2) Shutter on retractable bellows. 3) Carbon evaporator on retractable bellows.

current of 150 mA and a tip-evaporator distance of 18 mm. As shown by the STEM images in Figure 5.15, the resultant carbon layer was considerably thicker on one side than the other, as expected. The orientation of the tip when viewed in STEM in Figures 5.15c-f was not purposely correlated with the orientations in which the carbon was deposited, so the apparent thickness (∼20 nm) does not necessarily represent the true thickness of the carbon layer at its thickest point, but the deposition rate under these conditions can be estimated to be around 20 nm/minute under these conditions.

Since the average tomography sample is ∼5-10 nm in diameter, an initial target of 2 nm was set for the carbon layer thickness based on previous experience with carbon layers on tomography samples. Further testing showed that the threshold current for evaporation of carbon from the evaporator was just below 140 mA ($V = 1.5$ kV), so carbon was evaporated onto a bare W tip at $V = 1.5$ kV, $I = 140$ mA for 30 seconds while smoothly rotating the tip in the holder to ensure even coverage. As shown in Figure 5.16, this resulted in the desired carbon layer thickness of 2 nm, demonstrating that the layer thickness can be controlled by this method as desired.

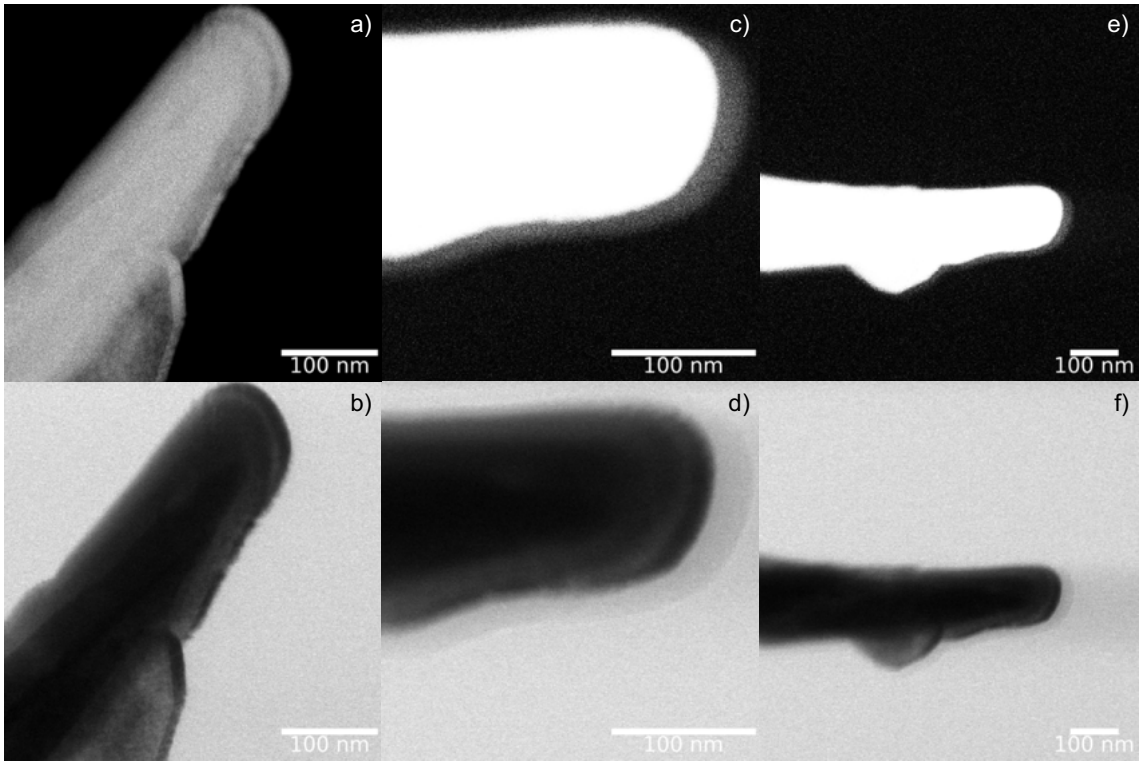


Figure 5.15: Initial testing of C evaporation (V = 1.5 kV, I = 150 mA, d = 18 mm, t = 30 seconds (one side), 60 seconds (other side). a) Pt-coated W tip (ADF). b) Pt-coated W tip (BF). c) C-coated Pt-Coated W tip (ADF). d) C-coated Pt-Coated W tip (BF). e) C-coated Pt-Coated W tip (ADF). f) C-coated Pt-Coated W tip (BF).

5.4.1 Future work

To demonstrate that a carbon layer of a given thickness protects against oxidation, a C-coated W tip would need to be placed in a controlled environment — either a climate-controlled box or a known oxidising solution — for a few weeks and then placed into a microscope to see if an oxide layer had formed. If the carbon layer fails to guard against oxidation, the process could be repeated with thicker carbon layers. This would be less favourable from a microscopy standpoint, but this could be mitigated by plasma treatment to remove some of the carbon.

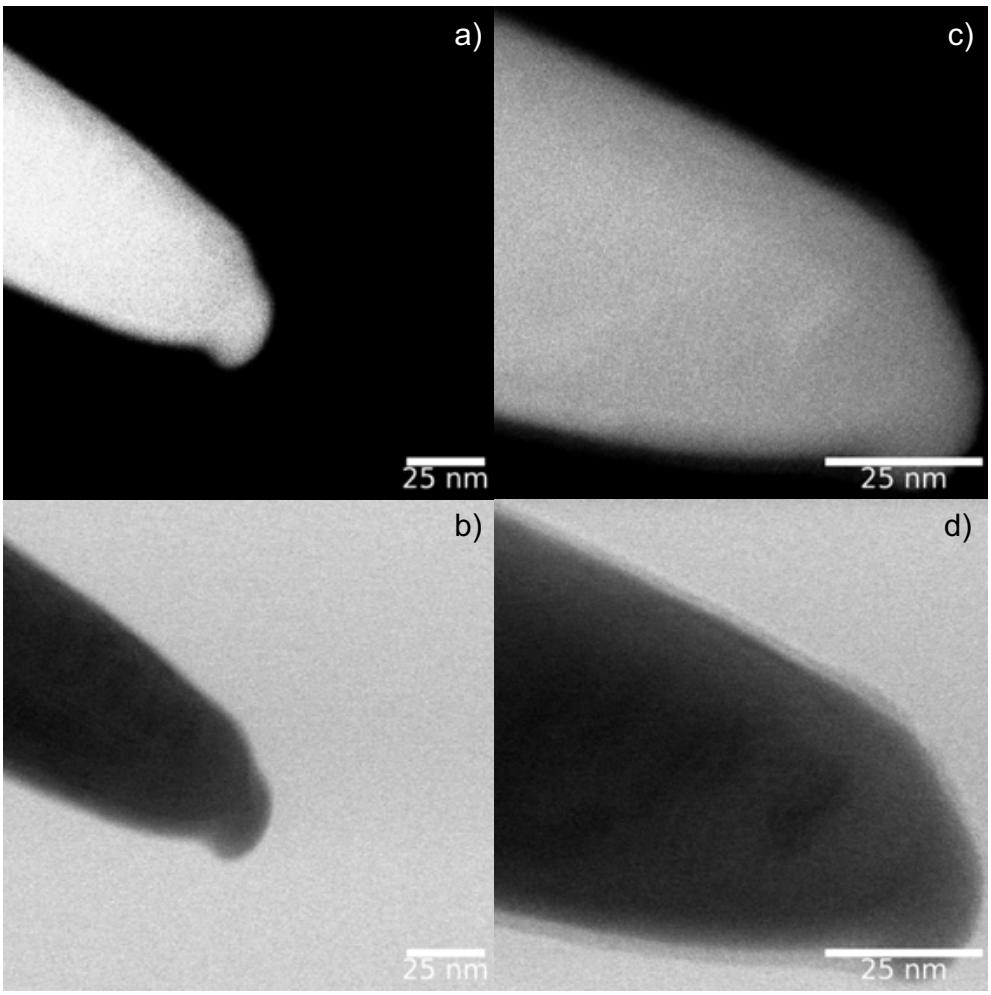


Figure 5.16: C evaporation (V = 1.5 kV, I = 140 mA, d = 18 mm, t = 30 seconds. a) W tip (ADF). b) W tip (BF). c) C-coated W tip (ADF). d) C-coated W tip (BF).

5.5 Summary and Outlook

This chapter has introduced the processes by which tungsten tip platforms suitable for electron tomography are created — recrystallisation, electrochemical etching and annealing — as well describing how usable tomography samples are created by evaporating metal onto the apexes of the tungsten tips and then annealing them. Several of these processes are excellent examples of how programmable microcontrollers can be used to automate sample creation. It has also described how tips are placed into the pucks used to manipulate samples in the TEAM 0.5 and the containers

in which these pucks are transported. Two coating techniques were tested as possible means of reducing sample oxidation during transport: bismuth coating and carbon coating. Bismuth coating was ultimately deemed unfeasible because it cannot be cleanly removed from the sample and is too heavy to be left on without significantly obscuring the sample region and reducing the signal-to-noise ratio. Carbon coating was deemed a potentially viable option as it was demonstrated that it can be applied in layers that should be sufficiently thin that they do not significantly obscure the sample. Going forwards, further experimentation would be required to assess how effective carbon coating is in preventing oxidation and ensure that oxidation can be prevented without significantly degrading the signal-to-noise ratio of the images.

Chapter 6

Using automated control software to enable high-throughput STEM

6.1 Introduction

As the stability and reliability of electron microscopes continues to improve, the demands on their human operators continues to diminish. At the same time, developments in computer vision — the study of how computers interpret and extract abstract information from images [107] — and increases in processing power mean that recognising complex features within micrographs is fairly straightforward even for ordinary desktop computers like those typically used to control electron microscopes. Simple tools like cross-correlation can be used to determine the relative placement of two overlapping images, or the placement of a high-magnification image within a low-magnification image, without using the sample stage's positioning system (which is unreliable as samples drift over time) [108], while several autofocusing techniques have been developed that are particular to the EM technique being used and the sample being studied. These include the beam-tilt-induced image displacement (BID) method described by Koster et al. for TEM [109] or the direct image contrast method described by Holburn & Smith for SEM [110], which has also been adapted for use in STEM [111].

With these fundamentals in place, computers are, in principle, already capable of carrying out any imaging task without human intervention, which has numerous practical and scientific benefits. On the practical side, automation frees up a scientist to carry out other tasks, enable remote operation or oversight from anywhere in the world and potentially enable even non-expert users to operate microscopes. In such a setup, an on-site technician is only required to align the microscope and transfer samples (both of which, ultimately, could be automated). Perhaps the greatest practical benefit is that it allows round-the clock use of microscopes without interruption. Not only does this maximise the use of microscope time, which is a precious commodity for most scientific institutions, it also greatly reduces the amount of human effort needed to acquire large numbers of images.

One field that has already recognised these practical benefits is 'connectomics': the study of the neurons and synaptic connections in the brain. Scientists involved in this research are attempting to map these connections on a large scale in the hope of improving understanding of how the brain works and hopefully leading to new treatments for brain diseases like Alzheimer's [112], but acquiring and processing the necessary data requires significant investments of both time and resources. The connectome of the *C. elegans* worm, for example, which consists of only 300 neurons and 7,000 connections, took over a decade to complete. A mammalian brain (like a human's), by contrast, consists of billions of connections, so any attempt to map such a brain would demand high-throughput, high-resolution microscopy techniques, as well as analytical methods commensurate with the size of the resultant datasets, which may be of the order of petabytes [112, 113]. Various automated acquisition protocols have already been devised for this purpose, including sample block-face scanning electron microscopy (SBEM), in which ultrathin slices of the sample block (i.e. brain) are shaved off after every image taken, and serial-section transmission electron microscopy (ss-TEM), in which slices are arranged laterally on a grid or tape [113, 114]. Over the last fifteen years, the speed of data acquisition using this method has increased roughly 200-fold, primarily by reducing the overhead times of moving the sample under the electron beam [114]. Even with these and other developments like multi-beam scanning, however, a full human brain would require 0.5-3.9 million electron beam years to map with current capabilities [114]. Nevertheless, this example illustrates the practical importance of careful experimental design and automation.

On the scientific side, automated systems are free from the subjective biases that influence a human operator's decision when selecting a region of interest to examine. They are also able to calculate precise stage movements without requiring the constant 'live feed' from the microscope that human operators typically rely on, which means that the beam can be blanked while moving between regions of interest, thereby reduce beam dose. This is an important consideration, especially for beam-sensitive samples like those studied in cryo-electron microscopy. In this system, samples are plunged into a cryogen like liquid ethane or liquid propane to flash-freeze them prior to placement in the microscope. This process creates a thin layer of vitreous (non-crystalline) ice that limits both localised heating and radiation damage induced by the beam [7, 115]. Even with this treatment, all other measures must be taken to minimise radiation damage, including using extremely low electron doses ($\sim$20-40 electrons/Å) [116], which significantly reduces the signal. For biological samples, where the structures of different particles of the same type are fairly uniform, thousands of randomly oriented single particles are imaged to compensate for this loss of signal and used to create an averaged computer model of a particle of that type [116]. These requirements — a high number of images and low electron dose — can both be addressed by automation and several software packages have been developed for this exact purpose, such as SerialEM [108] and Legonin [8]. Such systems can be configured to acquire data in the most beam-efficient manners possible, including ones that would be unfeasible for a human operator due to the time and/or complexity involved [8, 108, 116].

Another field in which beam efficiency is important is electron tomography, where potentially dozens of images are taken of the same sample at different tilt angles, resulting in a large cumulative beam dose on a single spot. This is likely to cause some level of radiation damage to even the hardiest of samples, not to mention generate significant contamination that will obscure the sample and lower the signal-to-noise ratio, making tomographic reconstruction more difficult. Additionally, the process of obtaining a tilt series is lengthy if performed manually, since it requires precise tilting and sample alignment to be effective [111]. Automated control software can be employed to mitigate both problems and has already been shown to be effective in the STEM tomography of a selection of plastic-embedded specimens, freeze-dried cryosections and inorganic materials [111].

The most important benefit of automation, however, is both practical and scientific: the en-

abling of studies that would be deemed practically unfeasible were they to be performed manually. The aforementioned case of mapping a mammalian brain is an excellent example of this; without automation, the time needed to acquire such a dataset would be prohibitively long. This is less true for cryo-electron microscopy and electron tomography, but neither of techniques would likely be used so widely were it not for advances in automation.

This chapter discusses the development of a suite of tools designed to automate the operation the TEAM 0.5 microscope at the National Center for Electron Microscopy in Berkeley, California, as well as their application in three ongoing experiments: acquisition of images of cadmium sulphide/calcium selenide (CdSe/CdS) core-shell nanoparticles, acquisition of images and CBED patterns of tellerium nanoparticles (TeNPs) and tomography of nickel-platinum nanoparticles (NiPtNPs). All the programs discussed in this chapter were written in Python 3.7 with the Numpy 1.18, Scipy 1.4, Pyqt 5.9 and Skimage library from Scikit-image 0.16.2 add-ons.

6.2 Software structure

The software described in this chapter plugs into the Wtip client and server software described in Chapter 3, as shown in Figure 6.1. The Wtip client is used for manual operation, such as early testing and selection of appropriate sample regions.

6.3 Core functions

In order to be considered truly automated, a microscope control system should, at the bare minimum, be able to acquire images, move the sample stage and find the optimal focus without any human intervention. Ideally, it should also be able to register its position relative to another image or region of interest. Stage movement is a simple command that instructs the TEAM 0.5 stage to move a certain distance in the x, y or z direction, with the use of piezo movers or steppers determined by the magnitude of the movement. Likewise, acquisition is handled by a single command that instructs the Wtip server to take an image with specified magnification, resolution and dwell time. Registration and focusing, on the other hand, are slightly more complicated operations.

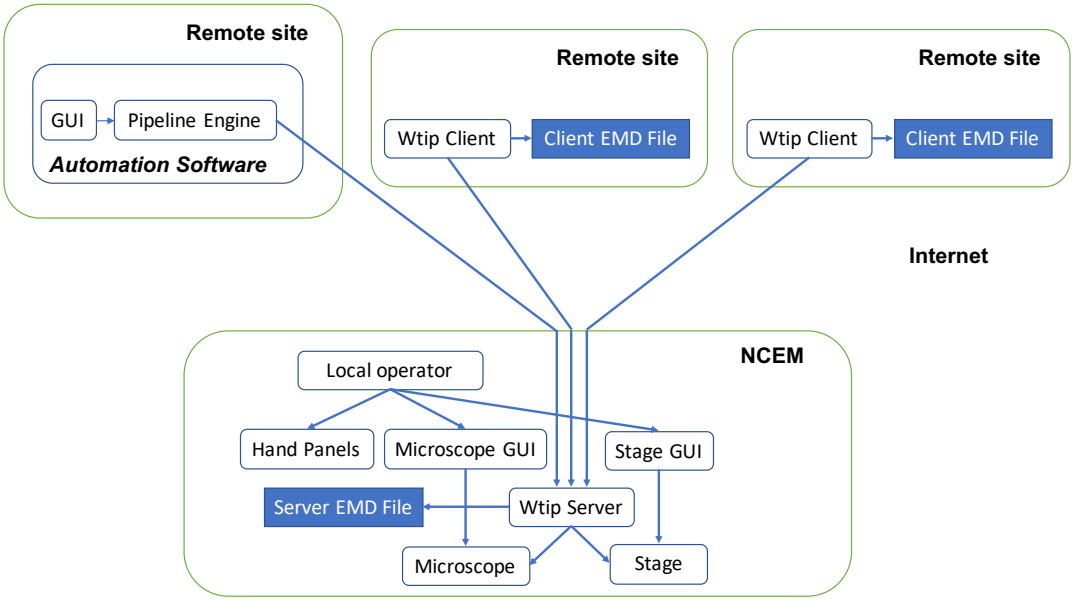


Figure 6.1: Software diagram showing connections between automation software (top-left), Wtip client and server and TEAM 0.5 microscope systems.

6.3.1 Registration

Registration — the alignment of two separate images of overlapping regions [117] — is performed via normalised cross-correlation, either between two overlapping images of the same magnification or between a low-magnification image and a high-magnification image taken with its boundaries, depending on the mode of operation. The two images being correlated must have the same pixel calibration (nm per pixel), which often involves average pooling to reduce the resolution of image with the greater spatial resolution (fewer nm per pixel). A general rule of thumb developed from testing is that the initial pixel calibrations should be no more than 8× different in size in order for this method to be effective. Image pairs are usually further binned to speed up processing.

6.3.2 Focusing

Automatic focusing is performed using the defocus slices described in Chapter 3. If the best focus is at one extreme of this focal series, the focus is adjusted to that point — either by moving the stage or adjusting the defocus, depending on the size of the move required — and the process is repeated again until a peak in focus is seen within the range of the focal series. If, at defocus =

0 nm, the distance from focus is more than 100 nm, the stage position is adjusted; otherwise, the defocus is changed.

6.3.3 Positioning-focusing-registration loop

The positioning-focusing-registration (posfocreg) loop is a core feature of all the programs described here. This is because the y and z motions of the piezo stage are coupled (due to both being controlled by the alpha module of the TEAM stage, as discussed in Chapter 3), meaning that stage movements made to change the focus may result in changes to lateral position and vice versa. A flowchart detailing the decisions taken within this loop is shown in Figure 6.2. In order to avoid an infinite loop, both the focusing and registration have tolerance levels, the values of which can be specified by the user. There are also loop counters that cause the programs to be aborted with an error after a specified maximum number of loops is reached. This error is then handled by the layer of software immediately above this one, which decides how to handle to respond to the error (e.g. retrying the image, skipping the image, aborting the program).

6.4 Automation pipeline and interface

The graphical user interface (GUI) developed for these experiments and shown in Figure 6.3 was created using pyqt5. Parameters for each experiment are input via the 'Pipeline Instructions' panel on the left of the interface and microscope images are displayed on the right of the interface.

The software is designed in a pipeline fashion, with each element completing its task and then calling the next in the sequence. The pipeline for each experiment consists of several steps, the number and function of which can be tailored to each individual experiment. There are four general methods that each step can use.

- Tiling: Takes images in a grid pattern. The starting position, height and width of the grid (in integer numbers of images) and the overlap between neighbouring images can all be specified, from which the method calculates and outputs the co-ordinates for each image that needs to be taken. Overlapping images can be stitched together to form a larger composite image. In

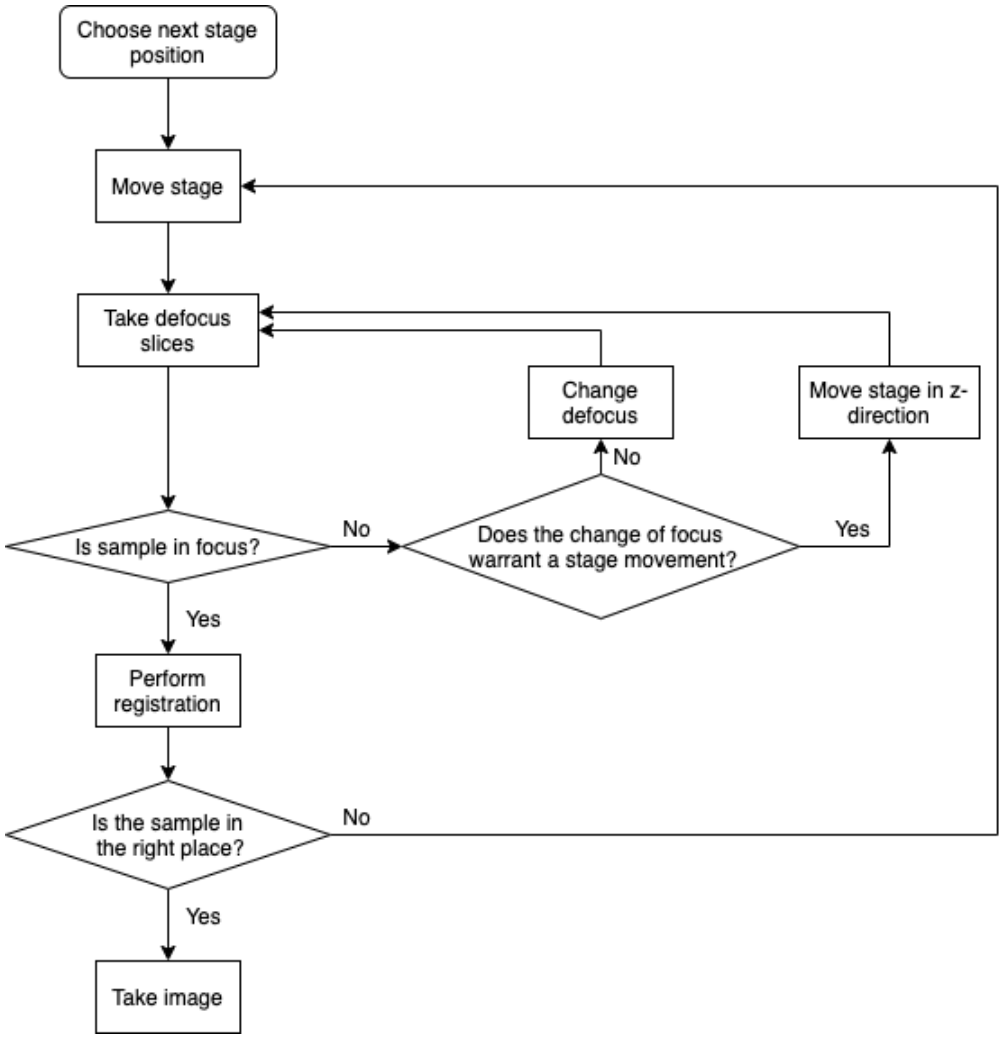


Figure 6.2: Positioning-focusing-registration loop flowchart.

practice, the first step in the pipeline always employs this method, even if the grid size is set to 1×1, in order to obtain an image against which subsequent steps can perform registration.

- ROI (region of interest): Defines a grid of potential regions of interest and selects those which satisfy the user-specified conditions concerning the intensities of the pixels within each region (a specified proportion of pixels have intensity values within a specified range).

- NP (nanoparticle): Selects individual nanoparticles within an image. Uses Laplacian-of-Gaussian-based detection method or custom particle detection method (see Section 6.6 for

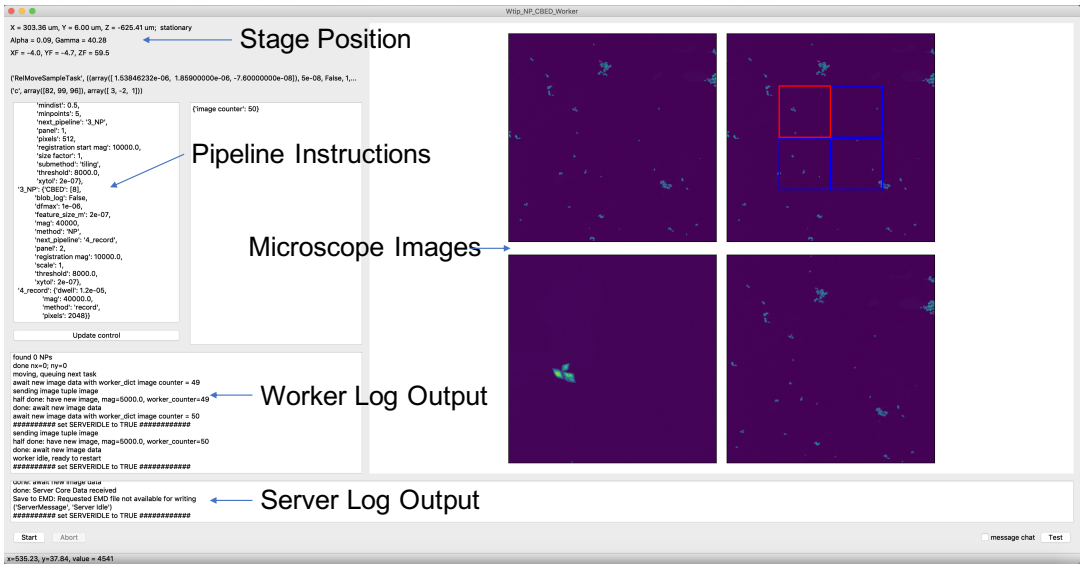


Figure 6.3: Graphical user interface.

details). Also acquires CBED patterns if specified.

- Record: Takes an image, the magnification, resolution and dwell time of which are all adjustable parameters.

Each step within the pipeline can be considered a sub-step of the previous step. For example, if the pipeline consists of two tiling steps, the first at 5k× magnification and the second at 10k× magnification, an image would be taken at the first position in the 5k× grid, then images would be taken at every position in the 10k× magnification grid before moving to the second position in the 5k× grid to take an image. This system reduces the amount of large stage movement required, which is both quicker and more reliable, since a sample may drift over time and reacquiring a previous position may be difficult.

As well as a main method, each step may have an additional submethod that contains extra instructions. The submethod used in each experiment described here is different and thus will be described in their respective sections, along with other function parameters such as magnification and image resolution. This structure is intended to maximise flexibility and allow functions developed for previous experiments to be transferred to new ones.

6.5 Cadmium selenide/cadmium sulphide core-shell nanoparticles (CdSe/CdS NPs)

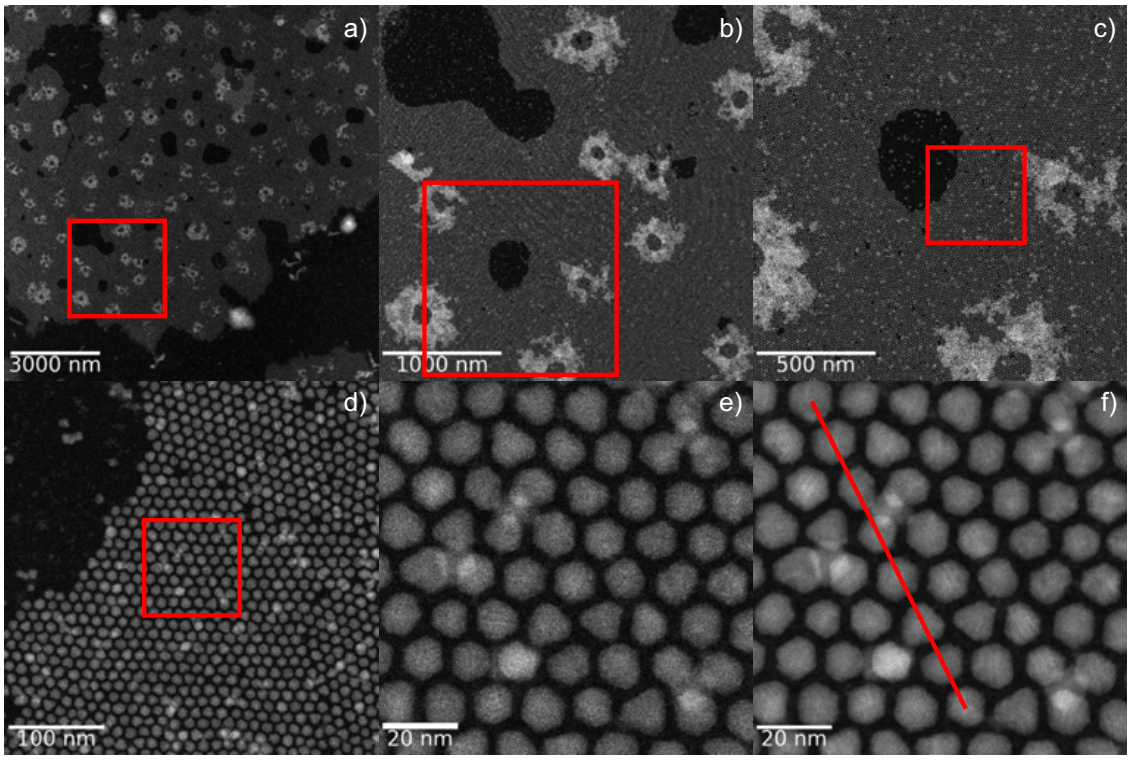


Figure 6.4: Image sequence of CdSe/CdS NP (red box denotes area of subsequent step). The core-shell structure of the nanoparticles is not easily visible due to the low Z value of S and Se compared to Cd. a) 5k× magnification (Step 1: Tiling, 1024×1024), b) 20k× magnification (Step 2: ROI, registration, 1024×1024)), c) 80k× magnification (Step 2: ROI, 1024×1024), d) 160k× magnification (Step 3: ROI, registration, 1024×1024), e) 640k× magnification (Step 3: ROI, 512×512), f) 640k× magnification (Step 4: Record, 2048×2048) (red line denotes line cut used to calculate spacing between particles).

The sample of CdSe/CdS nanoparticles used in this study was prepared by Justin Ondry, Alivisatos group, University of California, Berkeley. These were ligand-protected core-shell nanoparticles with cadmium selenide cores and cadmium sulphide shells arranged in close-packed monolayers, as shown in Figure 6.4e and f. These 'superlattices' are of great interest due to the tunability of their features (resulting from the tunability of the colloidal NPs that form them) [118]. However, while the organic surface ligands are required to control the size and shape of the NPs, as well

as achieve highly ordered arrays [119], they also seriously limit the coupling of electronic states between different NPs, which is highly undesirable for certain applications, like optoelectronics [118]. One way of solving this problem is to remove the ligands and fuse the nanocrystals together, which has been shown to improve the conductivity of arrays [120–122]. Of particular interest are crystallophically constant nanocrystal arrays, which can be created if the nanocrystal lattices share the same orientation [118, 122, 123]. Assessing this quantitatively and with reliable statistics, however, requires the acquisition of atomic-resolution images of thousands of particles. This was the goal of this experiment.

As can be seen in Figures 6.4a, b and c, the coverage of the particles was not uniform, with some regions having no coverage (darker regions) and some multiple overlapping layers of NPs (brighter regions). Such regions were of no interest and so the automation pipeline needed to be configured in such a way that such regions were avoided. This was achieved by using an intensity passband filter with a minimum and maximum threshold, the values of which could be manually selected by the user depending on the imaging conditions used (camera length, brightness and contrast, atomic number of material). Regions were deemed 'of interest' if a certain proportions of pixels within the specified area were between these values, as shown in Figure 6.5. This was implemented in the code in the form of the 'layer' submethod, which worked in conjunction with the 'ROI' method.

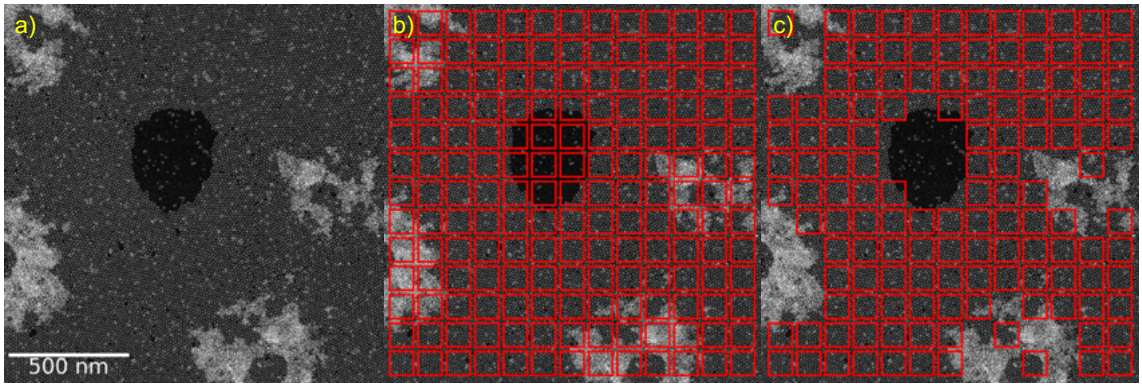


Figure 6.5: Region of interest selection using layer method. a) Original image (magnification = 40k×. b) Tiling positions (tiling magnification = 640k×, stride = 1.2). c) Tiling positions after passband filtering (lower threshold = 6000, upper threshold = 20000, pass fraction = 50%).

6.5.1 Pipeline implementation

The pipeline used in this experiment consisted of four steps. In the first step, a single 1024×1024 image (method = tiling, grid size = 1×1) was taken at a magnification of 5k× (image size = 12.6 µm × 12.6 µm). Only one image was taken because of the large number of particles present within this single image. As proven in Appendix C, assuming uniform coverage and hexagonal packing, the number of particles n within a given area A is roughly equal to:

$$n = \frac{A}{\sqrt{12}r^2} \tag{6.1}$$

where r is half the distance between the centres of two adjacent particles. The average spacing between particles (calculated from the line cut in Figure 6.4f) is 13.2±0.1 nm, so the average value of r is 6.1±0.1 nm. Using these values and the area of the image (12.6 µm × 12.6 µm), the value of n can be calculated as:

$$n = \frac{12600 \times 12600}{\sqrt{12} \times 6.1^2} \approx 1200000 \tag{6.2}$$

Binarising the image in Figure 6.4a to filter out empty regions or regions containing multiple layers of particles shows that ∼50% of the image is covered by monolayers, meaning that there are ∼600000 particles of interest within this single 5k× image alone, which is sufficient for this work.

The second step used the ROI method and the 'layer' submethod. The upper and lower thresholds for the passband filter in this step were 20000 and 4000 respectively and the proportion of pixels that needed to be in this range was 20%. Images were taken at 40k× magnification with a resolution of 1024×1024. Additional images were taken at each grid square at 20k× magnification for registration purposes to ensure that the final 40k× image was taken in the right place.

The third step also used the ROI method and the 'layer' submethod. The upper and lower thresholds for the passband filter in this step were 20000 and 4000 respectively and the proportion of pixels that needed to be in this range was 30%. Images were taken at 640k× magnification with a resolution of only 512×512, since areas deemed to be of interest were imaged again using longer dwell times and higher resolutions in the fourth and final step of the pipeline. This reduced both the electron beam dose and the overhead times involved in the experiment. Registration images

were taken at 160k× magnification with 1024×1024 resolution.

In the final step, areas deemed of interest in the previous step were imaged again at a magnification of 640k× (image size = 100 nm × 100 nm), a resolution of 2048×2048 and a dwell time of 12 µs. These images provide clear images of the atomic lattice structure of the CdSe/CdS NP lattice structure from which the desired information can be obtained, as shown in Figure 6.6.

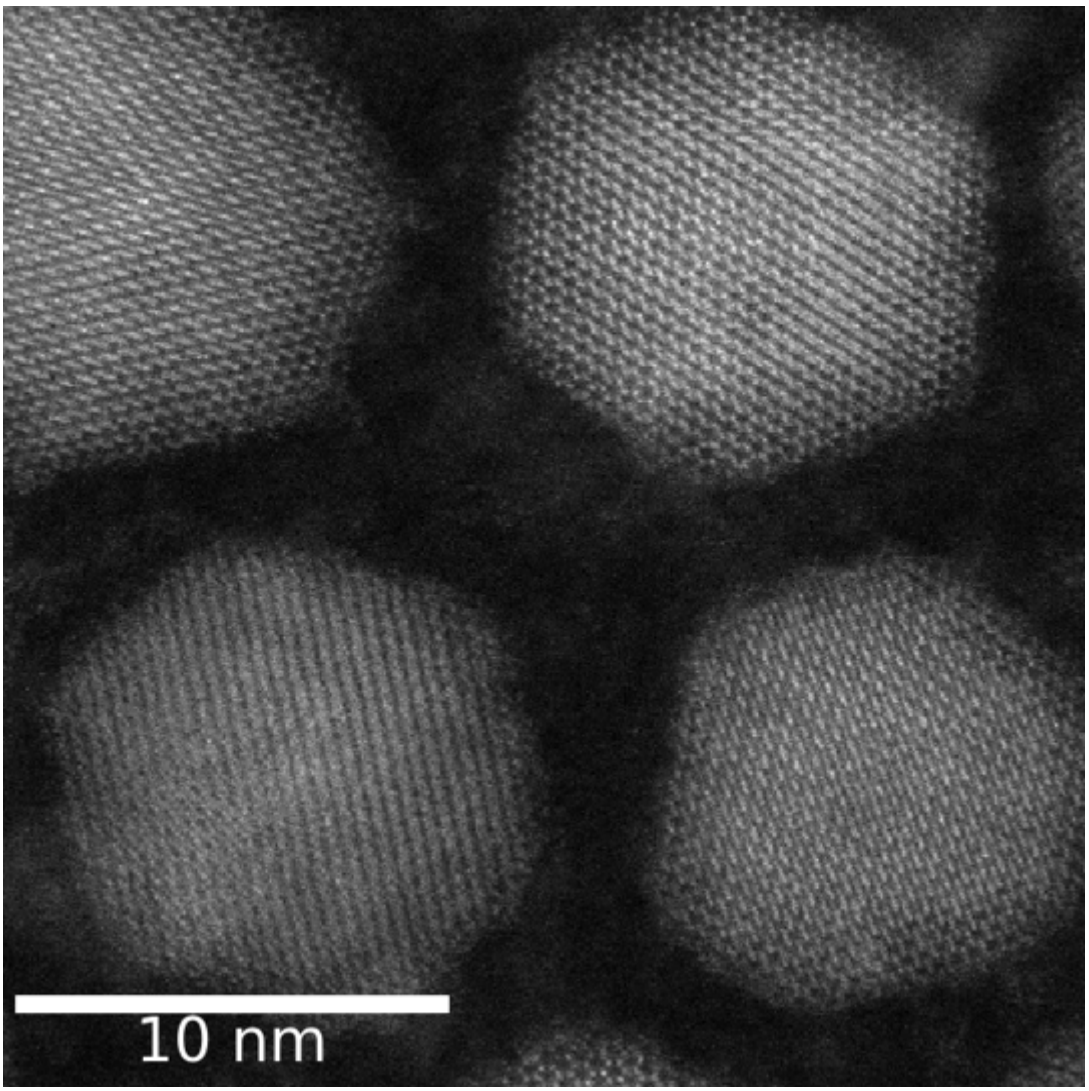


Figure 6.6: 512×512 cutout of 2048×2048 image of CdSe/CdS NPs

In one particular 8.5-hour period, 146 images (640k× magnification, 2048×2048 resolution)

images of CdSe/CdS NP lattice structure were acquired. A single 640k× magnification is 100 nm × 100 nm in size, which, using Equation 6.1, corresponds to 77.5 nanoparticles per image. While only about 50 of these are typically fully visible in one image, for this experiment, even a partial image can be counted towards the statistics so long as an FFT of its lattice structure can be obtained. Thus, ∼11000 data points were obtained during this period. The overheads for this session were aligning the microscope (∼1 hour), putting in the sample (∼30 minutes), finding a suitable region at 5k× (∼25 minutes) and determining suitable experimental parameters (∼20 minutes). In total, the session lasted ∼10.5 hours.

6.6 Tellurium nanoparticles (TeNPs)

In this study, samples of TeNPs prepared by Dr Assaf Ben-Moshe, Alivisatos group, University of California, Berkeley were used. These TeNPs have a chiral geometry, the handedness of which cannot be determined from a single STEM image, but can be determined from stereopairs as discussed in Chapter 7. The goal of this work was to develop a system that could automatically acquire images of isolated individual particles at a single angle. This process could then be repeated at a different tilt angle, with the sequence of registration images for each individual particle being used to pair particles at the different tilt angles, thereby generating stereopairs.

As shown in Figures 6.7a and b, the TeNPs in this sample were dispersed with low density across the holey carbon film of a TEM grid, some individually, some in agglomerations. Two methods were developed to distinguish between individual particles and agglomerates.

In the first method, Laplacian-of-Gaussian (LoG) blob detection was used to find the coordinates of Gaussian peaks with σ values in the range of 50-150 nm within an image (Figure 6.8b). The images were then binarised using a manually set threshold value and each separate 'island' of intensity within the binary image was labelled using flood-filling (all pixels in a given island are set to the same value) (Figure 6.8c). If two or more Gaussian peaks (particles) exist on the same 'island', this likely indicates the presence of an agglomeration and these peaks are subsequently ignored (Figure 6.8d).

In the second method, a maximum filter with a ring-shaped footprint, the diameter of which

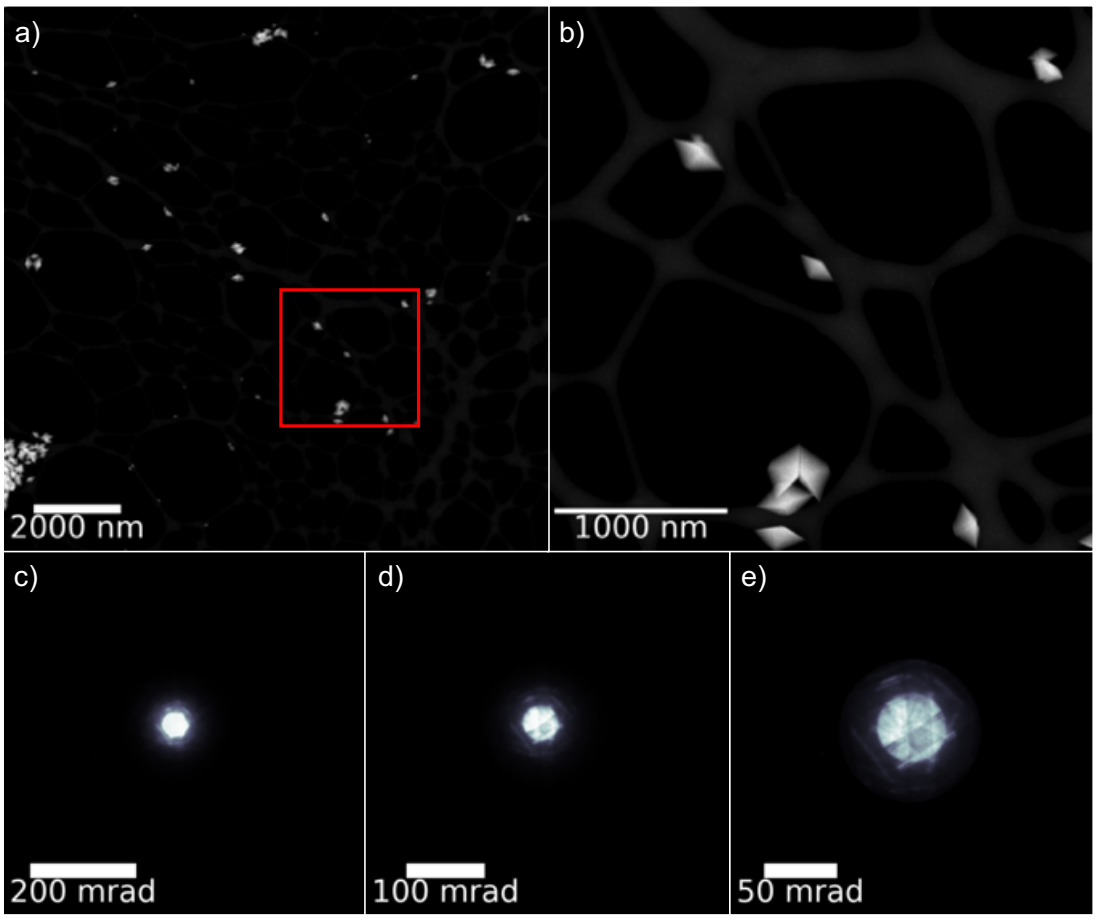


Figure 6.7: Image sequence for TeNPs (red box denotes area of subsequent step). a) 5k× magnification (Step 1: Tiling, 512×512), b) 20k× magnification (Step 2: ROI), c) CBED pattern (camera length = 4.3 cm), d) CBED pattern (camera length = 6.8 cm), e) CBED pattern (camera length = 13.5 cm).

equalled the feature size, was convolved with the image to produce a feature map, the intensity of which corresponded to the highest intensity in the ring (Figure 6.9b). Isolated particles appear as rings with a dark centre, whereas particles with close neighbours generate rings that overlap with the dark centres of the neighbour's rings. The original image (Figure 6.9c) and the feature map were then binarised, with the contrast of the latter being inverted (Figure 6.9d). When multiplied together, these generated a binarised map in which the bright regions corresponded to isolated particles. The product of this with the original image is shown in Figure 6.9e, from which the local maximum of each peak was identified, as shown in Figure 6.9f.

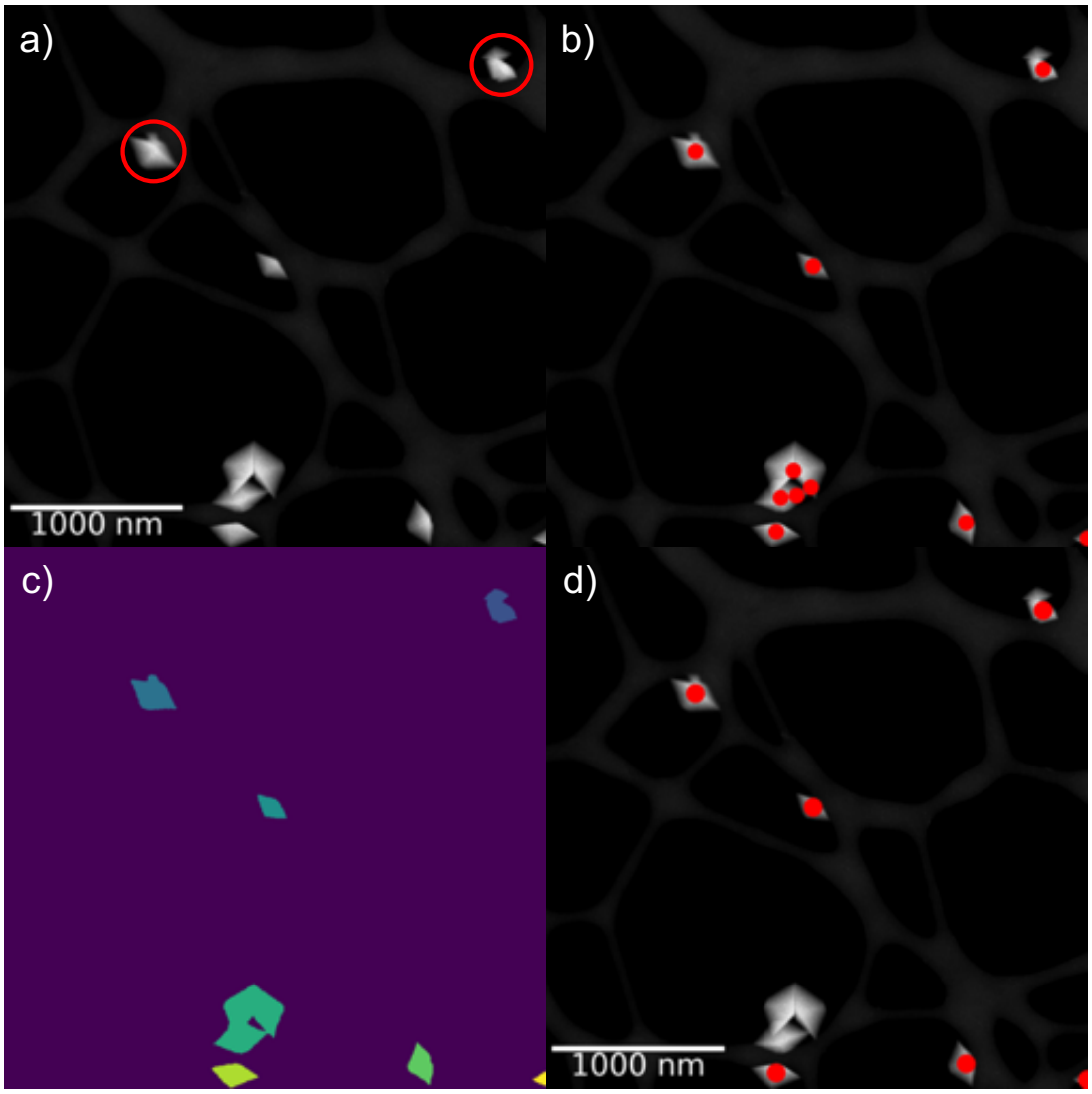


Figure 6.8: Laplacian-of-Gaussian-based particle detection method. a) Original image. b) Peaks identified by Laplacian-of-Gaussian blob detection. c) Flood-filled image. d) Identified individual particles.

A comparison of these two methods is shown in Figure 6.10, which demonstrates that, although the LoG-based method picks up more particles, it is also more likely to fail to filter out agglomerates like the one in the top of the image. This happens when the LoG blob detection only detects a single peak in an agglomeration, meaning that the subsequent flood-filling check fails to identify this as part of an agglomerate. For this reason, the custom method was used in the final experiments.

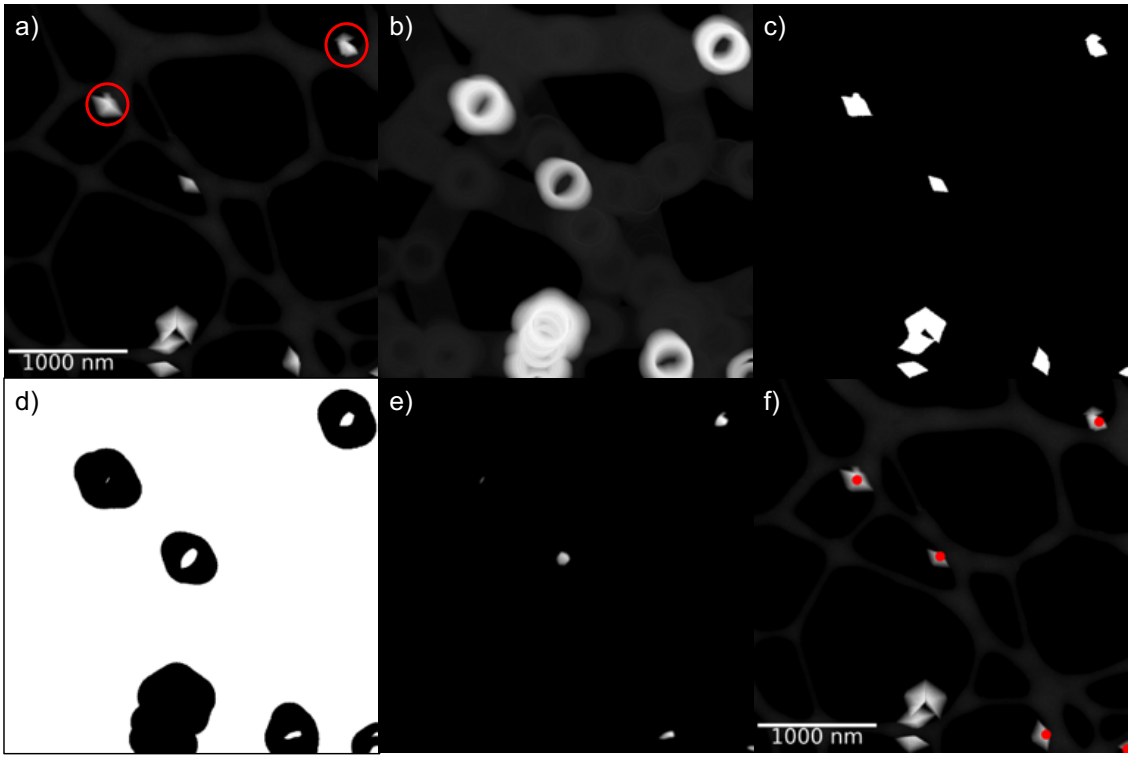


Figure 6.9: Custom particle detection method. a) Original image. b) Feature map generated by ring-shaped maximum filter. c) Binarised image. d) Binarised feature map. e) Product of binarised image and feature map multiplied by original image. f) Detected particles in original image.

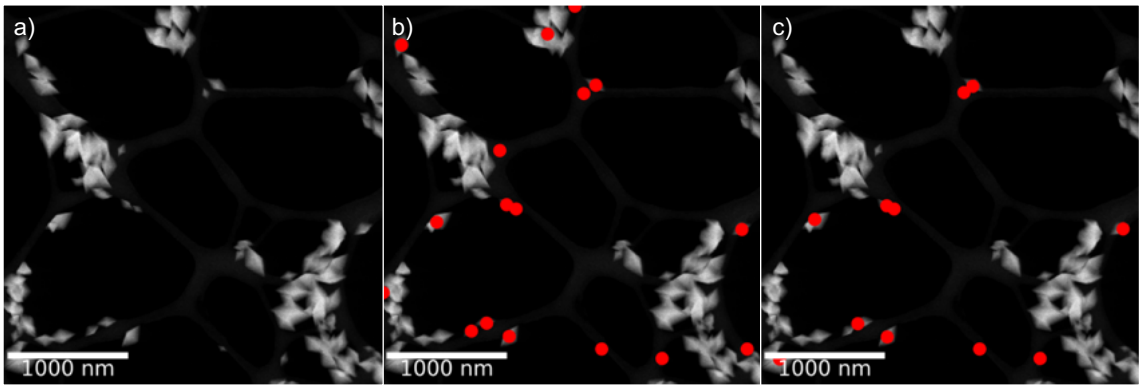


Figure 6.10: Comparison of LoG and custom particle detection methods. a) Original image. b) Peaks detected by LoG-based method. c) Peaks detected by custom method.

It should be noted, however, that neither method is 100% accurate, as evidenced by the particle circled in the Figures 6.9a and 6.8a, both of which are actually two particles, one much smaller

and dimmer than the other, and both of which are detected by both methods. The custom method would be incapable of distinguishing between individual and overlapping particles because it lacks any capability to assess the content of the particles, only to check that they are isolated from other particles in the vicinity. It is possible that the LoG-method could be configured to detect the overlapping particles by increasing the range of σ values that it searches for, which may enable it to detect both particles independently of one another. This would need considerable testing and optimisation, however, and would likely increase the number of false positives, since the use of smaller σ values would probably cause it to detect fluctuations in the background noise. This, in turn, could be mitigated by setting a threshold value for the intensity of the peaks, but this could cause other unforeseen problems if the function is not carefully optimised for each individual sample. Another solution would be to create an additional dedicated filter for this purpose, but such a filter may be difficult to program and would likely be too narrow in scope to transfer to a wider range of systems. A better solution would probably be to use some form of machine learning tool, such as a convolutional neural network, for this purpose, which would be applied as an additional filter to images of the particles detected by the algorithms. This concept is further explored in Section 6.6.2, but, if used, it would be better not to optimise the algorithms described here too strongly and let the CNN filter out unsuitable particles instead.

6.6.1 Pipeline implementation

The pipeline used in this experiment consisted of three steps. The first step was a 4×4 grid (method = tiling) of images was taken at 5k× magnification with 512×512 resolution. This size was selected because, unlike the CdSe/CdS NPs, the TeNPs are randomly dispersed and the fraction of particles suitable for analysis (isolated, non-overlapping) is comparatively small.

The second step used the ROI method and the 'tiling' submethod. The 'tiling' submethod is similar to the 'layer' submethod in that it generates a grid of positions within the boundaries of the previous 5k× image and the ROI method selects the squares within this grid that are of interest. In this case, however, it only requires a minimum threshold pixel intensity rather than a minimum and maximum threshold like the 'layer'. In this instance, the criteria for selection was regions in which more than 10 pixels had an intensity greater than 8000. This rule was used

instead of the percentage coverage used in the CdSe/CdS NPs experiment because the particles were highly dispersed with nothing close to a uniform coverage, so picking out individual points made more sense than looking for areas of high coverage. Images were taken at 20k× magnification with 1024×1024 resolution.

The final step used the 'NP' method to identify particles within the image. The feature size used here was 150 nm and the custom peak finding algorithm was used. Three CBED patterns were acquired at different camera lengths (4.3 cm, 6.8 cm and 13.5 cm) for each particle, as shown in Figure 6.7c, d and e. These were recorded both as a demonstration of the software's capabilities, but also to generate a library of CBED patterns matched to individual particles, which can be used as a dataset for the development of CBED analysis software in the future.

In one particular session, 2410 CBED patterns were acquired in a period of 17.5 hours, corresponding to 803 particles. The overheads for this session were aligning the microscope (∼1 hour), putting in the sample (∼30 minutes) and determining suitable experimental parameters (∼30 minutes). In total, the session lasted ∼19.5 hours.

6.6.2 Convolutional Neural Network

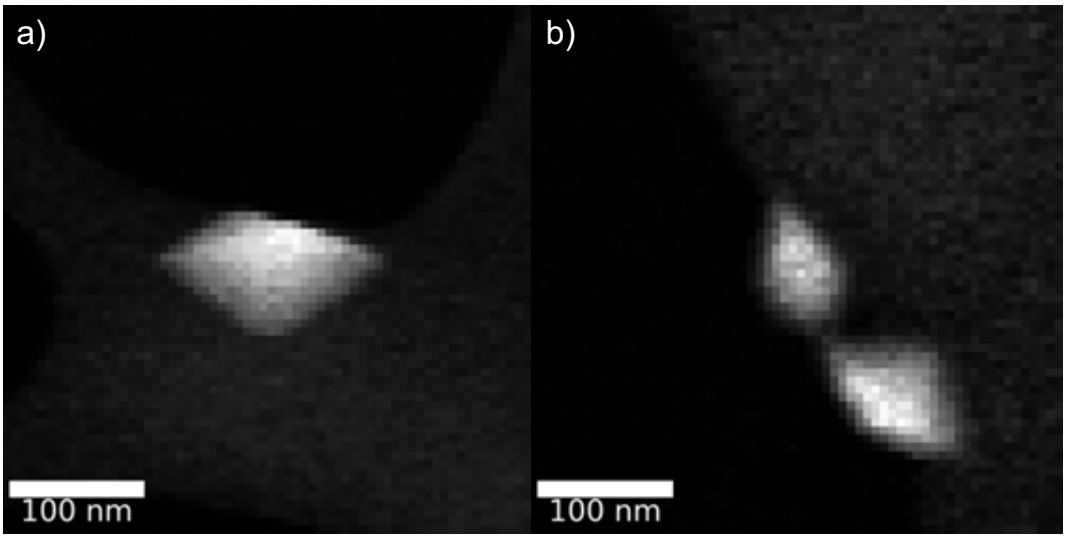


Figure 6.11: Example images of TeNPs used to train the CNN. a) 'Suitable' TeNP. b) 'Unsuitable' TeNP.

On top of the steps described above, a convolutional neural network (CNN) was trained to classify the particles identified by the particle-finding algorithms previously described. This was done using the same packages described in Chapter 7 and implemented into the automation software via Keras. The model structure used here was as follows:

- 2D Convolutional Layer (32 3×3 filters, stride = 1) (output = 30×30)) (ReLU activation)
- Max Pooling Layer (2×2 filter, stride = 2) (output = 15×15))
- Dropout Layer (dropout rate = 0.2)
- 2D Convolutional Layer (64 3×3 filters, stride = 1) (output = 13×13)) (ReLU activation)
- Max Pooling Layer (2×2 filter, stride = 2) (output = 6×6))
- Dropout Layer (dropout rate = 0.2)
- 2D Convolutional Layer (128 3×3 filters, stride = 1) (output = 4×4)) (ReLU activation)
- Max Pooling Layer (2×2 filter, stride = 2) (output = 2×2))
- Dropout Layer (dropout rate = 0.2)
- Flattening layer
- Dense Layer (64 nodes, ReLU activation)
- Dropout Layer (dropout rate = 0.2)
- Dense Layer (2 nodes, softmax activation)

For training, 1814 images were generated by cutting out 64×64 pixel regions surrounding the particle positions identified by the LoG-based method in 20k× magnification images of the TeNPs. These cutouts were manually labelled as 'suitable' or 'unsuitable', with cutouts being deemed suitable if the particles within them were isolated (not connected to any agglomerate), the geometric structure was clearly visible and they were oriented horizontally, as shown in Figure 6.11. In such an orientation, an alpha rotation of the TEAM 0.5 stage would produce a rotation of the particle solely about its long axis, so, if a CBED pattern reveals that the particle is close

to being on-axis, a small alpha rotation would bring the particle on-axis, meaning that its lattice structure could be examined alongside its geometric structure. 369 (20.3%) of these particles were deemed 'suitable' and the remaining 1445 (79.7%) were deemed 'unsuitable'. This original dataset was augmented 20 times by randomly mirroring images in the vertical and horizontal axes (neither of these transformations caused the labels to, since they did not change the absolute values of the orientation of the particles relative to the horizontal axis) and adding Gaussian noise ($\sigma = 500$). The images were then mean-pooled (filter size = 2×2, stride = 2) to reduce the size of the images to 32×32 to speed up training. The model was then trained on the augmented dataset and then tested on the original training dataset, achieving an accuracy of 100.0%.

This model was then incorporated into the code and, for each of the 803 particles imaged in the experiment described in Section 6.6.1, output its prediction as to whether each particle was 'suitable' or 'unsuitable'. These images were manually labelled (without prior reference to the CNN predictions), 183 being deemed 'suitable' and the remaining 626 being deemed 'unsuitable'. The model's predictions were then compared against the labelled dataset and found to agree in 679 of the 803 cases, a success rate of 84.6%. Of the 124 discrepancies, 56 were false positives (model predicted 'suitable' for 'unsuitable' particles) and 68 were false negatives (model predicted 'unsuitable' for 'suitable' particles). These results demonstrate that the model was attempting to learn how to correctly classify the particles rather than simply labelling all the particles as 'unsuitable', which would generate a classification accuracy of ∼80%. With further training and/or a larger training dataset, this test accuracy could be improved further.

This work demonstrates how CNNs could be integrated into the automation software and used in the process of selecting particles of interest. Looking forward, a user could acquire and label (manually or automatically) images of sample particles, apply augmentations suitable to the images to create a large dataset (∼20000 images) and train a model on this dataset. When applied to a new sample, the automation software would identify particles using either of the previously described algorithms, then generate appropriately sized cutouts of these particles from the image and pass these cutouts to the model, which would then output the probabilities of each particle belonging to each category that in had been trained on. These probabilities could then be used to filter out particles of little or no interest, either based on the label to which the highest probability is

assigned or by a setting a threshold probability for the label of interest. To illustrate the difference between these approaches, let there be three labels — A, B and C — to which the model assigns the probabilities 32%, 43% and 25%, respectively, for a given image and the label of interest is A. Using the first method, the image would be filtered out as the model predicts that the particle most closely resembles B. Using the second method with a threshold of 30%, the image would not be filtered out, since the probability assigned to label A is greater than the threshold, even though the probability assigned to label B is higher.

Using a CNN-based filter in this manner means that all the advantages of machine learning described in Chapters 4 and 7 can be incorporated in the automation software. In particular, the CNN filter can be used when the investment of time and effort required to create and optimise a filter by traditional coding techniques is too great, either because the task itself is too complex or because there are too few uses for the filter to make the investment worthwhile. An example of this is identifying images containing two overlapping particles, the problem plaguing the particle-finding algorithms previously discussed. Training a model on a dataset containing images of individual and images of overlapping particles would save a user having to program and optimise a filter that would only be applicable to one particular type of sample. Another potential example is particles of the same size but with slightly different shapes, where a user could train a program on example images of the different classes of particles without having to program a filter to recognise each individual case. It is also conceivable that filters of this nature could be retrained for use on different samples via transfer learning.

There are disadvantages to this approach, including the fact that acquiring and labelling the training data is time-consuming and training a network is computationally expensive. Once this is completed, however, running an image through a neural network would be fairly computationally inexpensive and could be easily incorporated into the automation software. The code for importing and applying the model would remain roughly the same even if different models were used, meaning that users could independently create their own neural networks and import them into the automation software with little to no additional effort required. Finally, this approach would enable users with limited programming abilities to create custom filters for specific tasks without having to learn new skills.

6.7 Alpha tomography

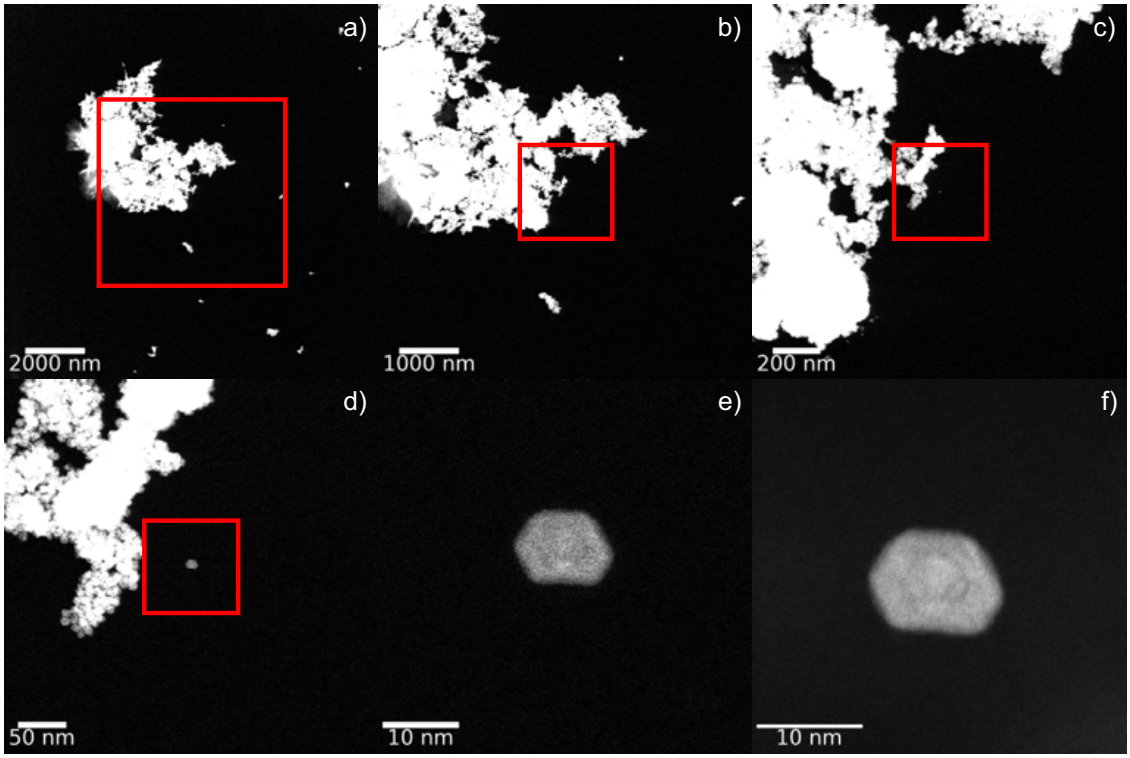


Figure 6.12: Magnification series of a single (manually selected) NiPt nanoparticle. a) 5k× magnification, b) 10k× magnification, c) 40k× magnification, d) 160k× magnification, e) 1.3M× magnification, f) 1.8M× magnification.

The sample of NiPt nanoparticles used in this study were prepared by Yongsoo Yang, Department of Physics, Korea Advanced Institute of Science and Technology (KAIST), Daejeon.

As described in Chapter 3, alpha tomography is a form of single-axis electron tomography in which the sample is tilted about the α axis of the TEAM stage. The tomography process itself has already been automated, but, at present, the process of finding potential candidate particles for tomography has not. As discussed in Chapter 3, in order to be suitable for alpha tomography, sample particles must have clearance in the vertical direction so that they are not shadowed at high tilt angles, but must also lie close to a large 'landmark' that is visible even at low magnification and strong defocus so that the sample particle can be relocated after a step in α, like the particle shown in Figure 6.12. The best places to start looking, therefore, are in the vicinities of the left-

and right-most points of large agglomerations.

In an effort to automate this process, a two-stage program was developed that searched for large agglomerates at low magnifications, identified their left- and right-most points, searched to the left and right of these points for particles, checked that they were not part of a large agglomerate and then ensured that they had sufficient clearance in the vertical direction so that they would not be shadowed at high tilt angles. This was done in two stages: one at low magnification and one at high magnification. The images on which these programs were developed and tested were of a sample of NiPt nanoparticles from which a tilt series had already been acquired.

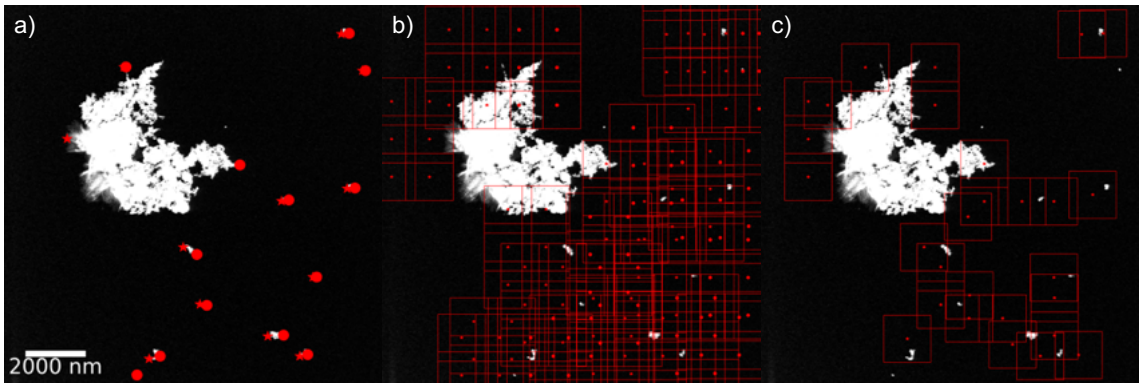


Figure 6.13: Steps for the first stage of the alpha tomography particle-searching program (5k× magnification). a) Original image with left- and right-most points of each large agglomerate denoted by stars and circles, respectively. b) Positions for higher magnification images placed around each of the previously identified points (grid size = 2×3, overlap = 20%). c) Final positions for higher magnification images after intensity and distancing filters applied.

The first stage started by binarising a low magnification (5k× in this example) image based on a manually selected threshold value, flood-filling to identify different 'islands' (agglomerates) (same process as described in Section 6.6) and then searching these islands for their left- and right-hand most pixels, to the left and right of which a grid of squares for higher magnification images (in this example, higher magnification = 40k×, grid size = 2×3, grid overlap = 20%) was set up. In order to be suitable, however, a grid square should:

- contain a 'landmark' that can be used for registration.
- not be too dominated by any landmark, or else any sample particle within it will likely be shadowed at higher tilt angles.

- have at least one column of pixels in which no landmark is present, so that there is a reasonable chance that a sample particle present will not be shadowed at higher angles.

To assess suitability, the grid squares were binarised using a manually set threshold value and both the number of bright pixels and number of columns in which there were no bright pixels counted. A grid square was deemed suitable if the number of bright pixels was between 0% and 50% of the total and there was at least one column that contained no bright pixels. Since, even after this filter, there were many overlapping grid squares, a distance filter was used to check whether the centres of two grid squares were close to each other (in this example, 'close' was defined as less than 40% of the side length of a grid square) and, if so, the two grid squares were replaced with one that was halfway between them. This process continued iteratively until no two grid squares were closer than the specified distance. Finally, all grid squares whose borders extended beyond those of the low magnification image were removed to ensure that accurate registration could be performed. The outputs of these steps are shown in Figure 6.13

In the second stage, particle-searching algorithms were applied to a higher magnification (40k×, in this example) image, as shown in Figure 6.14. As discussed in Chapter 3, the minimum vertical clearance h required for a particle of radius r is:

$$h = \frac{r}{\cos(70)} \tag{6.3}$$

but, as a precaution, a clearance of five times this amount is used in experiments. Therefore, around each identified particle, a section of the image with height = $5h$ and width = $5r$ (this was in case the alpha axis was not exactly horizontal in the image, which would mean the direction of the required clearance was not exactly vertical) was cut out, binarised and flood-filled to determine if any feature other than the particle of interest was present, as shown in Figure 6.15. If this was not the case, the particle was deemed suitable for tomography.

Both of the algorithms described in Section 6.6 were tested and found to be successful, although not without caveats. The custom method was more difficult to implement in this example than with the TeNPs because the particles were only 2-4 pixels in diameter in the image (the particle size was ∼10nm, the image size was 1.6 µm × 1.6 µm and the image resolution was 512×512). At

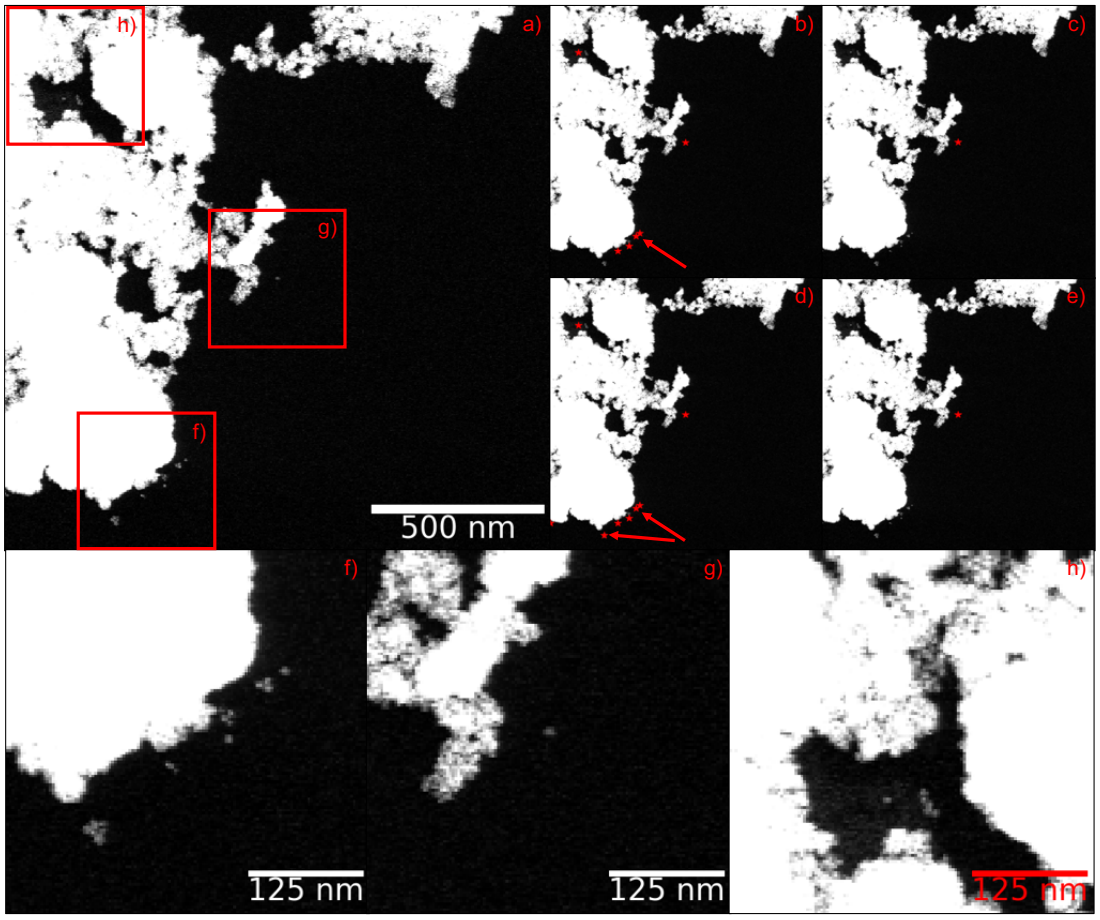


Figure 6.14: Particle-finding algorithm results for the second stage of the alpha tomography particle-searching program (40k× magnification). a) Original image. b) Particle-finding using custom method. c) Clearance-filtered peaks from b). d) Particle-finding using LoG method. e) Clearance-filtered peaks from d). (Arrows identify clusters of particles misidentified as single particles.). f)-h) Expanded regions of interest designated in a).

this scale, the ring-shaped maximum filter was found to be sensitive to small peaks in background noise and it was only by carefully setting the threshold level to a value significantly above the background intensity without exceeding the intensity of the particles that this program could be made to work. Using a higher-resolution and/or higher-magnification image would mitigate this problem. Additionally, while both algorithms successfully filtered out particles attached to the larger agglomerate present in the image, they also failed to filter out particles belonging to smaller agglomerates of 2-3 particles, as identified by the arrows in Figures 6.14b and c. The subsequent

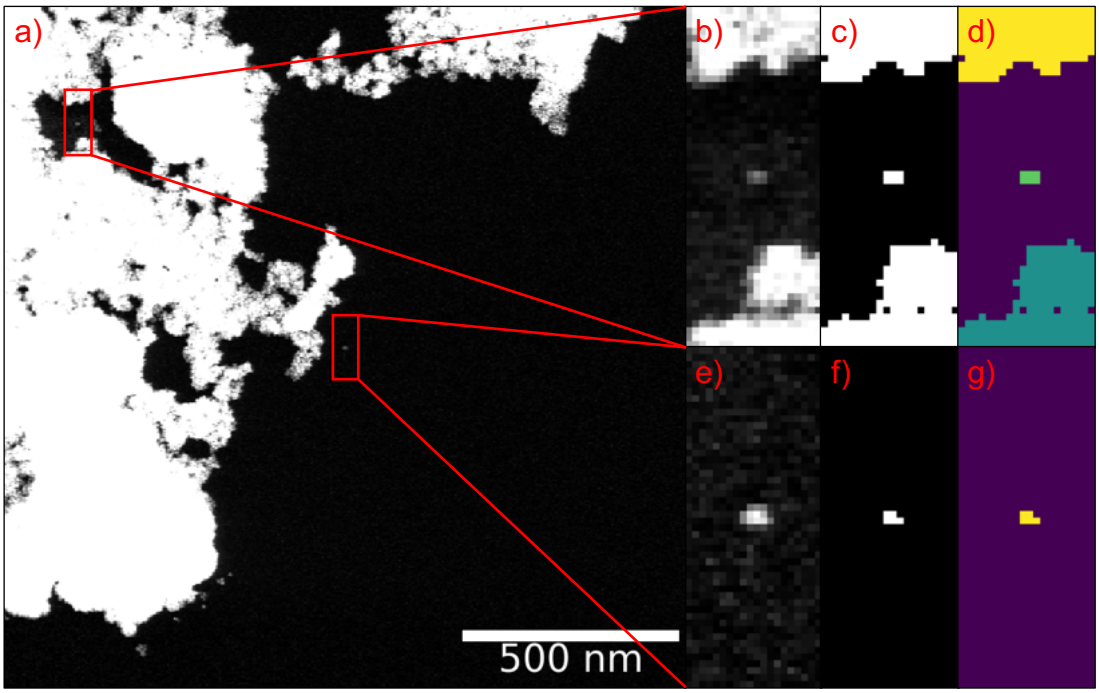


Figure 6.15: Clearance checking using the flood-filling algorithm. a) Original image with cutouts in b) and c) identified by boxes. b) Cutout of top-left particle in a). c) Binarised copy of a). d) Flood-filled copy of b), showing that the vertical clearance box is not clear and hence the particle is unsuitable for tomography. e) Cutout of centre particle in a). f) Binarised copy of e). g) Flood-filled copy of f), showing that the vertical clearance box is clear (aside from the particle) and hence the particle is suitable for tomography.

flood-filling also failed to filter out these particles since they appeared as single islands in the binarised images. One possible solution to this would be introduce a size filter, whereby 'islands' whose areas exceed a given threshold are excluded in the flood-filling stage. However, given the very small numbers of pixels involved, this would have very little margin of error and require precise selection of both the size and intensity threshold values. Using higher-resolution images would probably be a better solution, since it would become easier for the computer to recognise individual particles within small agglomerates. This is not a major concern, however, since this program was not developed to automate the selection of particles for tomography, only the search for possible candidates. At the end of its run, the program would output a selection images of possible candidates from which the operator could choose those of interest and rule out unsuitable candidates, such as small agglomerates.

Ultimately, although this program was tested on previously acquired experimental images and the functions incorporated into the control software, it has not, to date, been used to find any particle. This is because acquiring even a single tilt series is an extremely complex task with long overheads:

- Aligning the microscope (∼1 hour).
- Putting in the sample (∼30 minutes).
- Finding a suitable particle (∼1 hour).
- Setting up the experimental parameters (∼1 hour)
- Correcting stigmation (∼30 minutes). This requires finding an on-axis particle upon which the stigmation correction can be performed, which typically takes far longer than the correction itself. This step is generally repeated ∼5 times during the experiment to compensate for the rotation of the chopsticks.
- Beam showers (∼30 minutes). This step may have to be repeated up to 5-6 times during the experiment, depending on the buildup of contamination.
- Taking data (∼5 minutes per projection). This includes the time taken to rotate the stage, move the particle into the centre of the image and then move it into focus. Most tilt series involve ∼45-60 projections (∼4-6 hours).
- Manual interventions (∼10-20 minutes). These occur when the automated tilt series acquisition program runs into an error and stops. Correcting these errors typically involves manually finding the particle and focusing the microscope, then restarting the program.

A conservative estimate for the time taken to acquire a single tilt series is 16 hours, but it usually takes far longer. In order to achieve this, 48-hour sessions are booked on the TEAM 0.5 microscope with teams of users from three time zones (UK, USA and South Korea) operating in rotation and acquiring even a single tilt series is a challenge. Therefore, at present, testing and optimising this program would take up too much time, since, in general, only a single candidate particle needs to be found. If the speed and reliability of acquiring tilt series improves to the point

where several tilt series of different particles can be taken in a single session, this program may become more useful. A proposed pipeline for this experiment is as follows:

- Step 1: Take images at 5k× magnification using the 'tiling' method. A grid size of 1×1 will probably locate enough particles for a single session, but this can be expanded if necessary.
- Step 2: Use the 'ROI' method with a specialised submethod to find grid squares in the manner described above (40k× magnification, 1024×1024 resolution).
- Step 3: Use the 'NP' method with either of particle detection algorithms to locate isolated particles. A specialised submethod would be required to check their vertical clearance. A magnification series of the particle would then be taken.

As has already been mentioned, this program would work in conjunction with a human operator, who would start the program and, at its end of its run, assess the images generated and select those particles of interest. Multiple tilt series could then be queued up for acquisition, with the magnification series acquired by the program being used for registration.

6.8 Summary and Outlook

This chapter has introduced a new, versatile software program for automatically acquiring data from the TEAM 0.5 microscope. As well as fulfilling the basic functions of an automated microscope control system — position, autofocusing and registration — the program's pipeline architecture offers a simple and flexible approach to experimental design with a custom-built GUI that improves usability. The efficacy of this software was demonstrated in two experiments: the high-throughput acquisition of images of CdSe/CdS core-shell NPs and CBED patterns of TeNPs. In the former, atomic resolution images of ∼11000 CdSe/CdS NPs were acquired in a session lasting 10.5 hours. In the latter, CBED patterns of 803 TeNPs were acquired in a session lasting 17.5 hours. This experiment also served as a test bed for training a convolutional neural network filter that can be used to determine whether the TeNPs would retain their orientation in an image after an α-rotation. A series of functions were also developed and implemented into the framework for use in a third experiment: the search for candidate particles for alpha tomography. Going forwards, this

software could be used for any number of remote, high-throughput STEM experiments and bring studies that were previously deemed too tedious to carry out into the realm of feasibility.

Chapter 7

Using machine learning to determine the handedness of chiral nanoparticles

7.1 Introduction

With the size of datasets generated by electron microscopy predicted to increase with improved automation (potentially into the petabyte region [112]), the need to automate the analysis of these datasets grows ever more pressing. Countless image processing tools already exist for extracting information from such data, such as the Fast Fourier Transform (FFT) for examining the crystallinity of lattices [124], Laplacian of Gaussian for locating atomic columns in samples [125] and image segmentation routines for identifying organelles in micrographs of cells [126]. Selecting the correct tools and optimising their parameters to accomplish a given task, however, requires both a broad knowledge of the tools available and considerable testing, particularly if there is large variability within the data. A less labour-intensive option would be for a computer to develop its own mathematical model for extracting relevant information from datasets without human intervention. This concept is known as machine learning (ML).

Machine learning has already proven useful for solving many highly complex problems, such as image and speech recognition, behavioural prediction and controlling self-driving cars [72]. Increased accessibility to high-performance computing (HPC) and the development of flexible architectures for graphical processing units (GPU), whose inherent parallel processing capabilities can accelerate the training of ML algorithms by as much as 200-fold compared to conventional CPUs [72], have resulted in a surge of interest in ML in recent years for not only classification tasks, but also clustering (identifying groups of similar data), regression (estimation of the change in a dependent variable as a result of changes in other variables) and anomaly detection. Numerous algorithms exist to carry out these tasks, including, but certainly not limited to:

- Decision Trees: Systems that break down the larger problem into a sequence of smaller decisions to try to classify inputs [127].
- Evolutionary (or Genetic) Algorithms: Systems that mimic the process of natural selection in which randomly created models are culled based on their success in solving the problem, with those that work best being 'mutated' and 'cross-bred' to produce superior models [128].
- Support Vector Machines (SVM): Systems that attempt to find hyperplanes with the hyper-dimensional feature space of the input data, either for classification or clustering purposes [129].
- Perceptrons: Binary classification systems that sum the weighted products of all the inputs from a single data point to label these points as true or false ('active' or 'inactive'). The weights of these inputs are adjusted to find the optimal values for the entire training dataset [73].
- Artificial Neural Networks (ANN): Systems consisting of several layers of nodes that resemble individual perceptrons, with weighted connections to nodes in other layers. Unlike perceptrons, the nodes employ activation functions to make the decision process non-linear (see Chapter 4) [73].

Various ML techniques have been developed and applied to the analysis of electron microscopy data, especially biological data. Sorzano et al. (2009) demonstrated a custom-built ML algorithm

for identifying particles of interest within images, particularly of biological specimens [130]. Si et al. (2011) demonstrated a SVM-based approach (SSELearner) to identifying secondary structure elements (e.g. α-helices and β-sheets) of large protein complexes in medium-resolution (5-10Å) volumetric protein density maps generated from cryo-EM data [131]. Arganda-Carreras et al. (2017) reported the development of Trainable Weka Segmentation (TWS): an ML pixel classifier that sorts pixels from light and electron microscope images into different classes based on training labels provided by the user and uses this information to segment images. TWS has been cited over 100 times in scientific publications (according to Google Scholar) and, since it is both open source and ImageJ/Fiji-based, it can be applied in a number of different settings [132]. In a similar vein, Nunez-Iglesias et al. (2013) demonstrated a supervised ML algorithm that is used in image segmentation to decide whether two neighbouring 'superpixels' (small segments of the image) should merge into a larger segment or not [133]. It should be noted that this algorithm, although widely applicable, was originally developed to segment images of neuron membranes, an application of ML in EM for which even a cursory search of the available literature will return numerous results [134–136].

In recent years, however, artificial neural networks (ANNs) and their sub-variant, convolutional neural networks (CNNs), have emerged as the front-runners for general image classification tasks [10–12], which is, at least in part, due to the inherently 'deep' nature of their structure [73]. A ML algorithm is considered 'deep' if it uses multiple stages of non-linear feature extraction in its operation, rather than attempting to extract all features in a single stage. This results in hierarchical extraction of features, with earlier stages extracting simpler features (e.g. pixels in an image) and subsequent stages extracting more abstract features (e.g. edges, motifs) all the way up to final recognition of the subject of the image itself, which mimics current understanding of how the human brain works [11, 73]. Such networks have been proven effective for image classification tasks in numerous studies [10–12]. They also use more sequential operations and fewer parallel operations than broader networks, making them less reliant on parallel processing capabilities and thus better suited to being run on standard computers (with the trade-off of requiring greater processing time) [73]. Once again, there are numerous examples in literature of ANNs and CNNs being used to analyse EM data from biological samples, especially the segmentation of images of neuron

membranes [137–139]. Zhu et al. (2017) used a three-convolutional-layer CNN to analyse single particles picked out from cryo-EM micrographs and determine whether or not they were 'good' for the task at hand as a replacement for traditional template-matching techniques [140]. Something similar was achieved by Ogura and Sato (2004), who used non-convolutional neural network units with a single hidden layer to detect very small proteins (<400 kDa) (N.B. a neural network with a single hidden layer is not considered deep, as it does not enable hierarchical feature extraction over multiple stages) [141]. Li et al. (2016) also developed deep CNNs to detect secondary structure elements in proteins with better detection accuracies than those of the SVM method described in [131] prior to post-processing (SVM methods outperformed CNNs after post-processing, which the authors claimed was due to the effectiveness of the SVM post-processing algorithm designed to eliminate false positive results. A comparable algorithm for CNNs is listed as one of the authors' future research goals) [142]. Literature concerning the use of ML in EM for material science is much more limited. One example is Ziatdinov et al. (2017), which reports the use of deep, fully-convolutional 'encoder-decoder' neural networks to identify Si defects in graphene and W defects in $Mo_{1-x}W_xSe_2$ imaged using STEM and SPM. The encoder portion behaves like a conventional neural network to extract, while the decoder reverses the process to map these features back onto the original image. This work is notable for using simulated images generated by molecular dynamics simulations rather than labelled experimental data to train the models [13].

7.1.1 Chiral Nanoparticles

Chirality (derived from the Greek word 'kheir' meaning 'hand') is a geometric property of any three-dimensional shape that lacks a plane of symmetry, meaning that a mirror image of a chiral shape is always distinguishable from the original. These mirror images are known as 'enantiomers', which exhibit different physical and chemical behaviours when interacting with other chiral phenomena, such as other chiral molecules or circularly polarised light [143]. Chirality plays an important role in biology, since many organic molecules, including proteins, sugars and nucleic acids, are chiral [143, 144], as well as almost half of all drugs [145]. In optics, chiral materials are also of interest due to their ability to circularly polarize light (also known as optical activity) [146] and, conversely, their differential absorption of left- and right-circularly polarized light (also known as

circular dichroism) [143, 147].

In recent years, various studies have begun to explore the effects of chirality in inorganic materials, with nanoparticles being of particular interest due to the fact that they can more easily afford low symmetry structures like chiral ones than bulk materials [148]. Nanostructures with chiral geometries should possess much stronger chiroptical responses than those of chiral molecules due to their larger size [143] and many such nanostructures have already been synthesised via a number of techniques, including lithography [149], DNA origami [150] and vapour deposition [151]. One particularly notable result is the chemical synthesis of tellurium nanoparticles with both chiral geometry and chiral crystalline structure by Ben-Moshe et al. Through simulation, it was determined that the chiral geometry's contribution to optical activity was dominant over that of the chiral lattice structure [152].

The metric of the enantiomeric purity of a sample is known as 'enantiomeric excess', or *ee*, which is defined by [153]:

$$ee = \%\text{ left-handed particles} - \%\text{ right-handed particles} \tag{7.1}$$

Qualitative measurements of *ee* can be carried out using circular dichroism (CD) spectroscopy, where the difference between the absorption of left- and right-circularly polarised light is measured. In order to be used quantitatively, however, this process must first be calibrated to a known *ee*, which is not always feasible. Another option is to use electron microscopy, with scanning electron microscopy (SEM) typically being used to assess geometric chirality due to the topographical nature of its imaging, which makes it easy to interpret the overall shape of a particle, as shown in Figure 7.1. If the chiral particle is too small, however, the resolution of SEM may be insufficient to examine its geometric features (the resolution of SEM is also insufficient to examine lattice structure). The study of such particles requires the resolution of transmission electron microscopy (TEM) and scanning transmission electron microscopy (STEM), but neither technique is ideally suited for analysing chiral geometry. In TEM, the dominant contrast mechanism is usually phase contrast, meaning that its images will contain no information about the thickness of the sample at each point (and hence its geometry). In HAADF-STEM, Z-contrast is usually the dominant

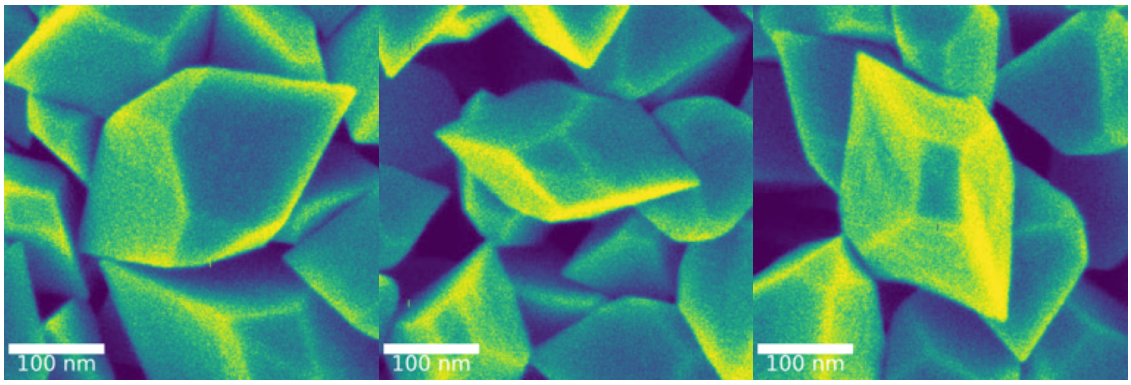


Figure 7.1: SEM images of tellurium nanoparticles. Sample created by Dr Assaf Ben-Moshe, Alivisatos group, University of California, Berkeley. Images courtesy of Alexander Mueller, National Center for Electron Microscopy.

contrast mechanism, so the intensity of each pixel will be related to the thickness of sample at the corresponding beam position, meaning that information about the geometry can be extracted. However, this technique also collects signal from all depths of the sample with (approximately) equal weighting (assuming that the sample is thin enough to be fully transparent to the electron beam), so, if a chiral particle is mirrored in a plane parallel to the image, the image of the mirror particle (which, by definition, will have the opposite handedness) will appear identical to the original. This means that a single STEM projection is insufficient to definitively determine the handedness of the chiral particle's geometry. However, if projections of two chiral particles with opposite handedness are identical at one particular angle, there is guaranteed to be another angle at which the projections appear different. A pair of images of the same subject taken from slightly different viewpoints is known as a stereopair.

This chapter details the development of deep convolutional neural networks (CNNs) for classifying the enantiomeric handedness of simulated chiral nanostructures based on stereopairs of STEM images. To the author's knowledge, this is the first study that explores the use of stereopairs in machine learning for electron microscopy (and possibly beyond), which opens up the possibility of employing more three-dimensional structural data in machine learning. The CNN architecture draws inspirations from the architectures of successful pre-existing image recognition CNNs like AlexNet, ZF Net and VGGNet (although these were all designed to classify single images rather than stereopairs). Although the work presented here is done using simulated data, the practicalities

of extending this work to real-world systems are considered and explored.

All the programs discussed in this chapter were written in Python 3.7 with the Numpy 1.18 and Scipy 1.4 library add-ons. The neural network sections were implemented using Keras via the TensorFlow 2.0 library.

7.2 Development of CNN for chiral geometry classification

The simulated chiral particles used in this study were hypothetical and not intended to represent any specific real-world system, but were loosely based on the tellurium nanoparticles described in [152] and [153] and shown in Figure 7.1. The structure of these particles was the intersected volume of two opposite-pointing pyramids that were rotated with respect to one another by a given angle (chiral twist), as demonstrated in Figure 7.2. Throughout most of this study (and unless explicitly stated otherwise), these particles had four-fold symmetry ($n = 4$) (square-based pyramids used) rather than the three-fold symmetry ($n = 3$) of the TeNPs. This was done because four-fold symmetry particles have a higher number of features (vertices, edges, facets) and because the projections of four-fold symmetry particles are always symmetric, as shown in Figure 7.3, which made the task of figuring out the chirality of a four-fold symmetry from the projections appear (at first glance, at least) more complicated. Unless explicitly stated otherwise, the chiral twist used throughout this study was $\pm 22.5°$, which is halfway between 0°, where a particle is achiral, and $\pm 45°$, where a four-fold symmetry particle is also achiral since such a particle with a chiral twist of +45° is identical to one with a chiral twist of -45° as shown in Figure 7.4. This is universally true for any two particles where the difference between their chiral twists is $360°/n$. Stereopairs of simulated chiral particles with four-fold symmetry generated in this fashion are shown in Figure 7.5.

To generate the stereopairs of images required to determine the handedness of the chiral particles, each particle was rotated by a fixed interval (stereo angle) and simulated STEM images generated for both positions. The images used for training were 64×64 pixels in size and the particles upon which the simulations were based are ∼300 nm in length, which equates to a resolution of 4.7 nm/pixel. Since this was far below atomic resolution, advanced simulation techniques like mul-

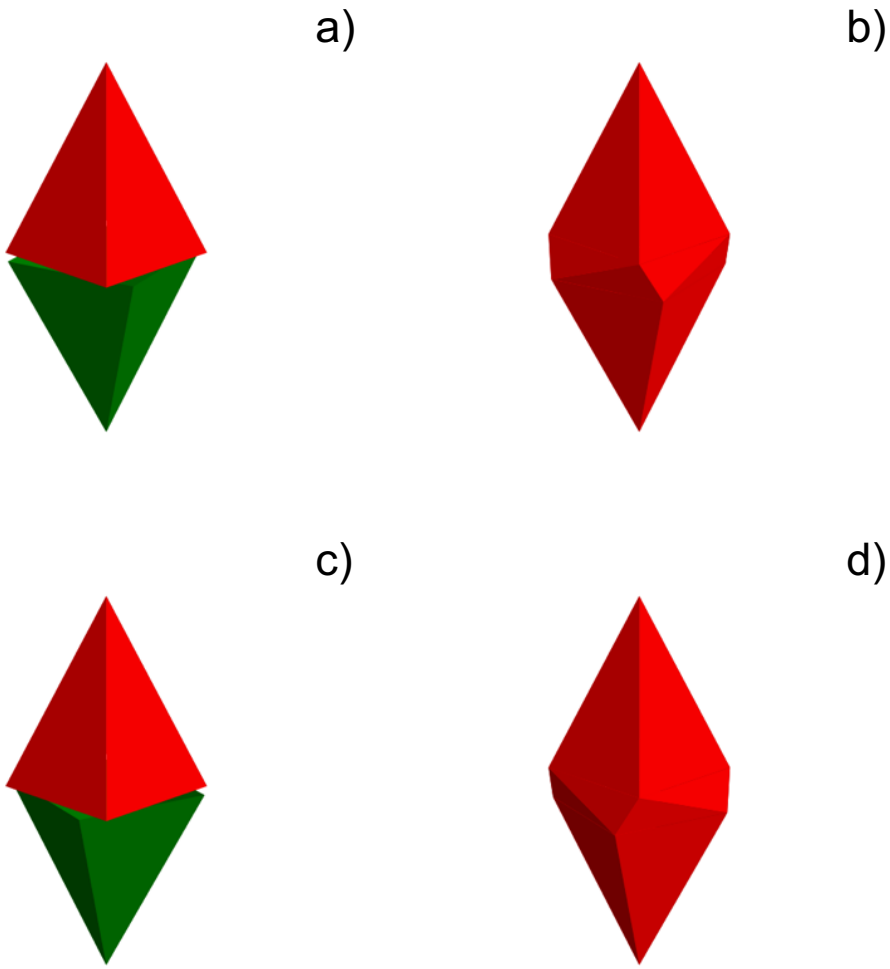


Figure 7.2: Steps for simulating chiral geometries. a) Opposite-pointing pyramids offset by +15.0°. b) Chiral particle with chiral twist = +15.0°. c) Opposite-pointing pyramids offset by -15.0°. b) Chiral particle with chiral twist = -15.0°.

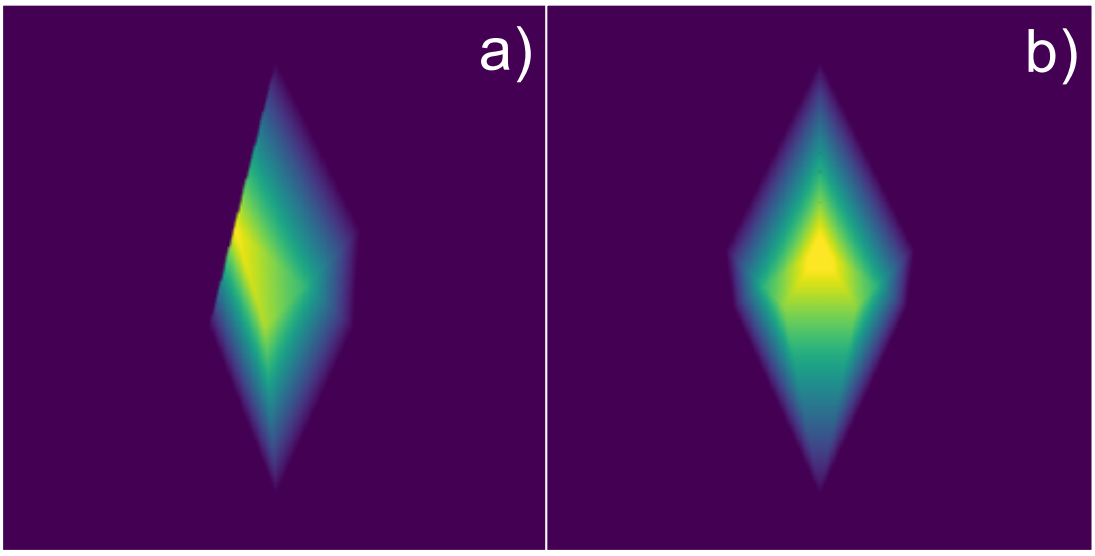


Figure 7.3: Simulated STEM projections of particles with a) 3-fold symmetry and b) 4-fold symmetry.

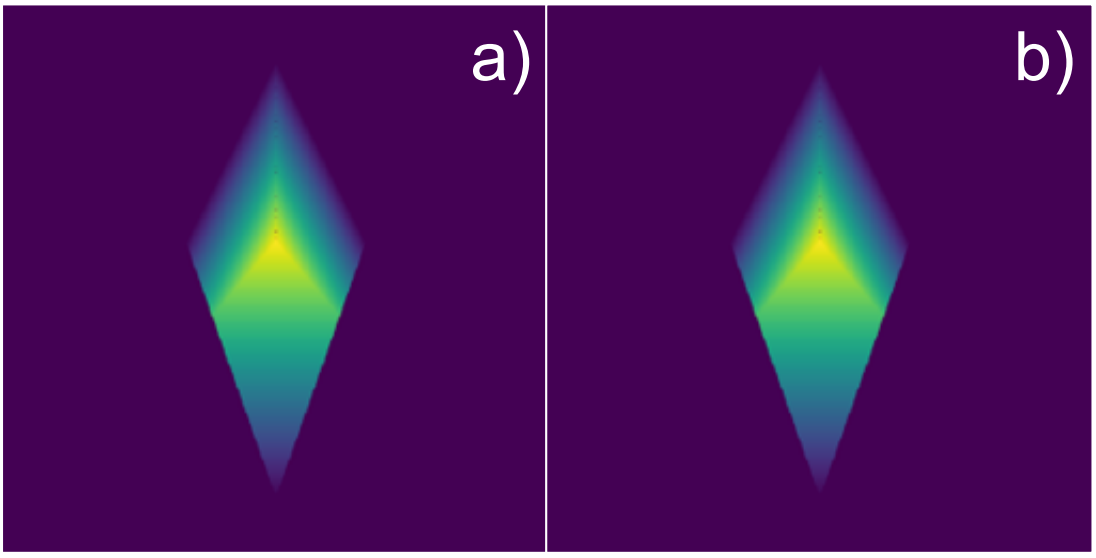


Figure 7.4: Simulated STEM projections of particles with chiral twists of a) +45° and b) -45°.

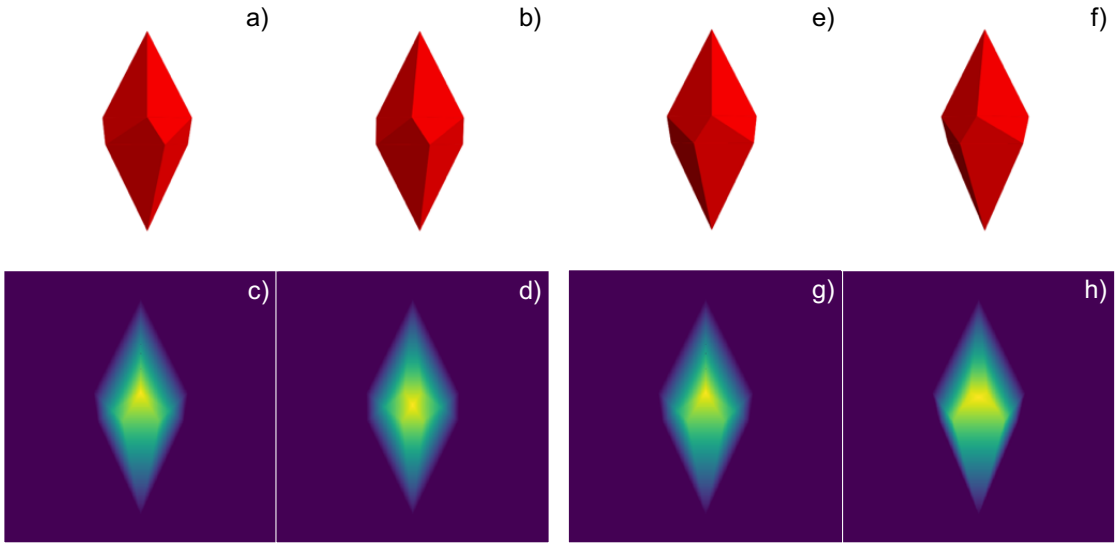


Figure 7.5: a) and b) Stereopair of chiral particles with chiral twist = +22.5° at stereo angles a) 0° and b) 10°. c) and d) Simulated STEM projections of a) and b), respectively. e) and f) Stereopair of chiral particles with chiral twist = -22.5° at stereo angles e) 0° and f) 10°. g) and h) Simulated STEM projections of e) and f), respectively.

tislice and Bloch wave were eschewed in favour of simpler density projections (see Section 7.2.1.1 for details), which were less computationally expensive and a very reasonable approximation of a STEM image at this resolution. The stereo angle used as a starting point in this study was 10°, which was chosen as it is well within the tilt range of the majority of electron microscope single-axis tilt holders. The axis of stereo rotation was aligned with the vertical axis of all simulated STEM images in this study, like those shown in Figures 7.3, 7.4 and 7.5.

It should be noted that certain stereo angles can be predicted to perform worse than others. Obviously, a stereo angle of 0° would generate two identical images, which, for the reasons already discussed, would make it impossible to determine the chirality of any orientation of particle. A stereo angle of any integer multiple of $360°/n$ would also cause difficulties if the particle's axis of rotational symmetry — in this case, the axis running between the apexes of the two original pyramids — were aligned with the stereo axis, since, at these stereo angles, the second projection in the stereopair would be indistinguishable from the first, as shown in Figure 7.6a-d. If these two axes are not aligned, however, other elements of rotation would make the two projections distinguishable, as shown in Figure 7.6e-h. These same conditions apply to stereo angles equal to

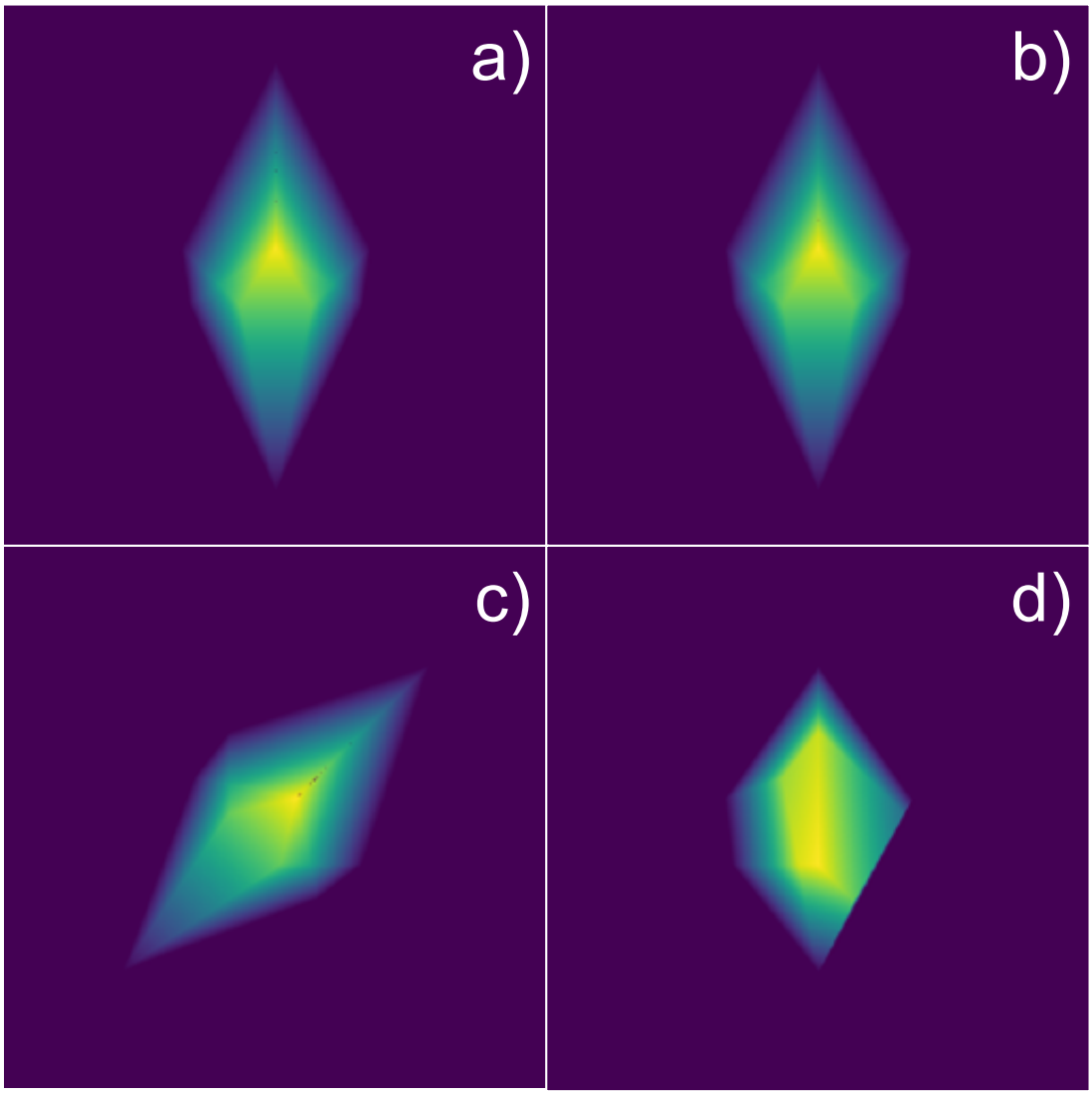


Figure 7.6: a) and b) Simulated STEM stereopair of vertically aligned four-fold symmetry chiral particle (chiral twist = 22.5°, z-rotation = 0°) at stereo angles a) 0° and b) 90°. c) and d) Simulated STEM stereopair of diagonally aligned four-fold symmetry chiral particle (chiral twist = 22.5°, z-rotation = 45°) at stereo angles a) 0° and b) 90°.

any integer multiple of $360°/2n$, too, although for slightly different reasons. If the axis of rotation symmetry were aligned with the stereo axis, the following is true:

- As discussed in Section 7.2, two stereo images of chiral particles (where the axis of rotational symmetry is aligned with the stereo axis) separated by a stereo angle of $360°/n$ are identical.

- If two stereopairs of the same enantiomer are taken in which the stereo angles of the first pair are -360°/2n and 0° and the stereo angles of the second are 0° and +360°/2n, the first image in the first pair and the second image in the second pair will look identical, since the separation between them is 360°/n. Additionally, the second image in the first pair and first image in the second pair will be identical because they are taken from the same stereo angle (0°). This means that one stereopair can be obtained from the other simply by reversing the ordering of the images.

- As will be demonstrated in Section 7.2.1.2.1, reversing the ordering of the images in a stereopair is equivalent to changing the chirality and adding 180° to the stereo angles of both images.

- Therefore, it would be impossible to distinguish between a stereopair of one enantiomer in which the first stereo angle is 0° and the second stereo angle is 360°/2n and a stereopair of the other enantiomer in which the first stereo angle is 180° and the second stereo angle is 180°+360°/2n. This is shown in Figure 7.7a-d.

Once again, if the axis of rotational symmetry were not aligned with the stereo axis, the other elements of rotation would make the two projections distinguishable, as shown in Figure 7.7e-h.

Two separate datasets were created for training and validation, with the size of the validation dataset fixed at 10% of the size of the training dataset. Once generated, both datasets were then augmented and pre-processed as described in sections 7.2.1.2 and 7.2.1.3, then used to train the CNN model, with 'positive'-handedness particle (+22.5°) labelled 1 and 'negative'-handedness particles (-22.5°) labelled 0. Once trained, a new test dataset of 1000 particles was generated, pre-processed and used to determine the model's true classification accuracy.

7.2.1 Simulation, Augmentation, Pre-Processing and Training

7.2.1.1 Simulation of chiral particle STEM images

The function that generated the geometries of these particles took the height (height of a single pyramid = half the height of a particle), radius (distance between centre and vertex of the pyramid

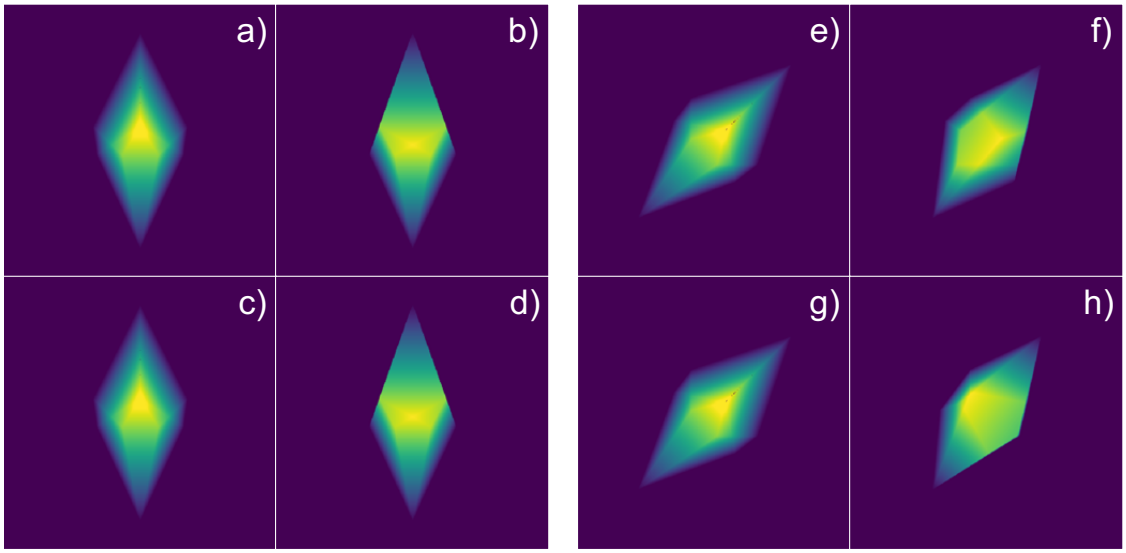


Figure 7.7: Comparison of stereopairs with stereo angle = $360°/2n$. a) and b) Stereopair of chiral particles ($n = 4$) with chiral twist = 22.5° and z-rotation = 0° (a) stereo rotation = 0°, b) stereo rotation = 45°). c) and d) Stereopair of chiral particles ($n = 4$) with chiral twist = -22.5° and z-rotation = 0° (c) stereo rotation = 180°, d) stereo rotation = 225°). e) and f) Stereopair of chiral particles ($n = 4$) with chiral twist = 22.5° and z-rotation = 45° (e) stereo rotation = 0°, f) stereo rotation = 45°). g) and h) Stereopair of chiral particles ($n = 4$) with chiral twist = -22.5° and z-rotation = 45° (g) stereo rotation = 180°, h) stereo rotation = 225°).

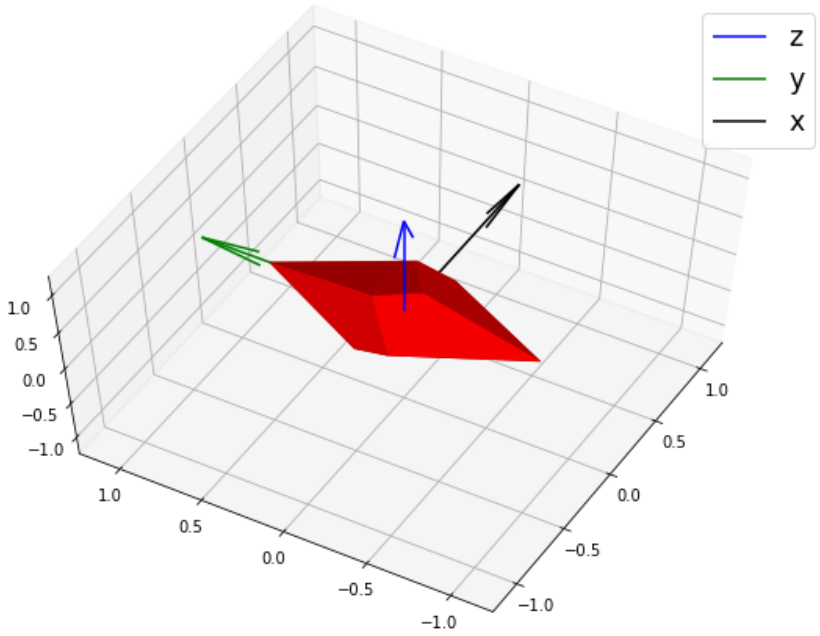


Figure 7.8: Rotation axes for the particle. These axes are stationary and do not rotate with the particle.

base)[1], chiral twist angle and n-fold rotational symmetry of the particles as input variables and calculated the triangle surface mesh of these particles, outputting a 2D numpy array containing the vertex coordinates for each triangle. A scale factor was used to change the size of the particle and the starting orientation of the particle was varied about the three axes of rotation shown in Figure 7.8.

In a realistic sample of monodisperse particles of this shape, one would expect that the particles would lie flat, on one of their faces or edges or at some point in between (i.e. not on one apex or the other). To mimic this, the x-rotation range was limited to ±arctan(radius/height) — the range achievable if the particles lied on their edges and the largest possible range of movement — while the range of y- and z-rotation remained unbounded. These rotation axes were stationary and did not rotate with the particle, so rotations were applied sequentially in the order z, x, y. A

[1]Due to the manner in which the faces are cut off to form the final chiral particle, the radius did not relate to any single measurable length in the final particle. It could, however, be calculated using the formula $r = h \times \tan(\theta)$, where r is the radius, h is the height and θ is the angle between the axis running between the apexes of the two original pyramids and one of the edges leading away from either of the apexes (one edge of the pyramid).

second triangular mesh was produced for the same particle after stereo rotation.

These meshes were passed to a STEM image simulation function developed by Dr Wolfgang Theis, which used ray-tracing to determine the depth of material through which the beam passed at every pixel position and then output this depth as an intensity. Once the image has been created (in the form of a 2D numpy array), a flat background intensity, appropriate levels of Gaussian blurring (to simulate non-zero electron beam spot size) and shot noise were added. After testing, the following pipeline was developed:

- Rescale the intensity of the original simulation (intensity upscale taken as input variable).
- Add a flat background (background intensity value taken as input variable).
- Apply Gaussian blurring (sigma value taken as input variable).
- Add shot noise to each pixel using the equation.

$$I_{shot} = I + a \times P_p(k = b \times I_p) \tag{7.2}$$

(a and b taken as input variables)

These images were then processed to reflect the properties of TEAM 0.5 STEM images, with all values below 0 and above 66535 cut off and values floored to integer values (in float format). The standard values used in this study for the different variables were as follows: intensity upscale = 50000 counts/unit thickness (units of the simulation space), background = 2000, Gaussian blur = 0.5 pixels, a = 10, b = 0.1. The effect of each step is shown in Figure 7.9.

7.2.1.2 Augmentation

As discussed in Chapter 4, neural networks trained on real data often increase the size of the dataset by applying various transformations to copies of the original data that change them into subtly different data points. Rotation — a common augmentation method for image data — could not be applied here since the stereo rotation axis had to remain common to all stereopairs. Mirroring could be applied, but with caveats due to the knock-on impact on handedness as described below.

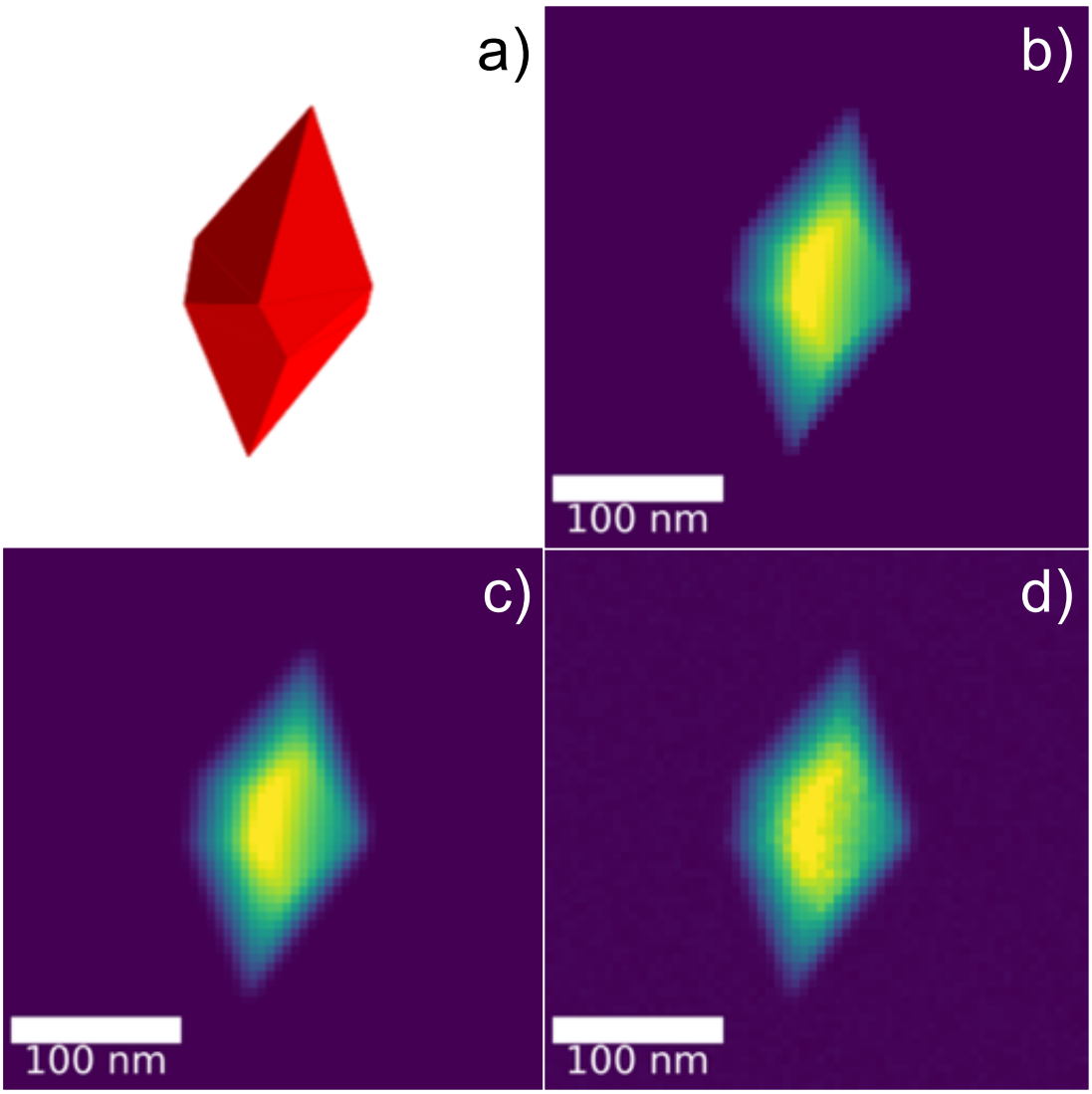


Figure 7.9: a) Simulation of chiral particle structure (height = 1.0, radius = 0.5, chiral twist angle = 22.5°, n-fold symmetry = 4, scale = 0.8, y-rotation = 10°, x-rotation = 31°, z-rotation = 10°). b) STEM image simulation with no shot noise or Gaussian blur. c) STEM image simulation with no shot noise and Gaussian blur = 0.5 pixels. d) STEM image simulation with shot noise (Poisson noise coefficients $a = 10.0$ and $b = 0.1$) and Gaussian blur = 0.5 pixels. Scale bars are an approximate representation of intended scale.

Translation was applied by rolling the image by a random number of pixels in both the x and y directions, although the range was carefully selected to prevent the particle from being 'cut in half' by this process (N.B. Since the particle was aligned to the centre of the image by effectively inverting this process during pre-processing (see Section 7.2.1.3), there was no advantage in using larger translation ranges for augmentation). Scale and shear transformations were applied independently in the x and y directions to produce slightly different shapes, with testing showing that changes of up to 20% in scale and up to 10% in shear produced images that were still reasonably representative of the original dataset. Additionally, the brightness and contrast of images were varied to improve the model's ability to handle changes in imaging conditions, with testing showing that adjustments of up to 20% to both brightness and contrast on the standard values outlined in Section 7.2.1.1 changed the images without significantly obscuring thickness information (due to pixel saturation). It should be noted that, throughout this work, the same brightness and contrast adjustments were applied to both images in a stereopair, rather than applying different adjustments to each in the pair, as it is assumed that both images in a stereopair would be acquired during the same session using the same imaging conditions. All of these augmentations were applied sequentially and a small amount of Gaussian noise was added to each pixel to reduce the chance of the neural network detecting that augmentation of previous data has taken place. The effect of each augmentation process is shown in Figure 7.10.

7.2.1.2.1 Mirroring When applied to a single image of a chiral particle, mirroring, by definition, reverses the handedness of the particle. When applied to a stereopair, however, the situation is more complicated and depends on the relative orientations of the stereo axis and the mirror plane.

Figure 7.11 shows two enantiomers of a 2D chiral structure with simplistic 1D density projections (heights of the red blocks correspond to projected thickness) for stereopair positions a, b and a, b + 180°, with the stereo axis pointing out of the figure. As can be seen from the table on the right hand side of the figure, mirroring the projections a, b in a stereopair individually in any plane parallel to the stereo axis (e.g. mirroring from left to right about the centre of the 1D projections), will produce the same stereopair as the projections a, b + 180° for the same chirality

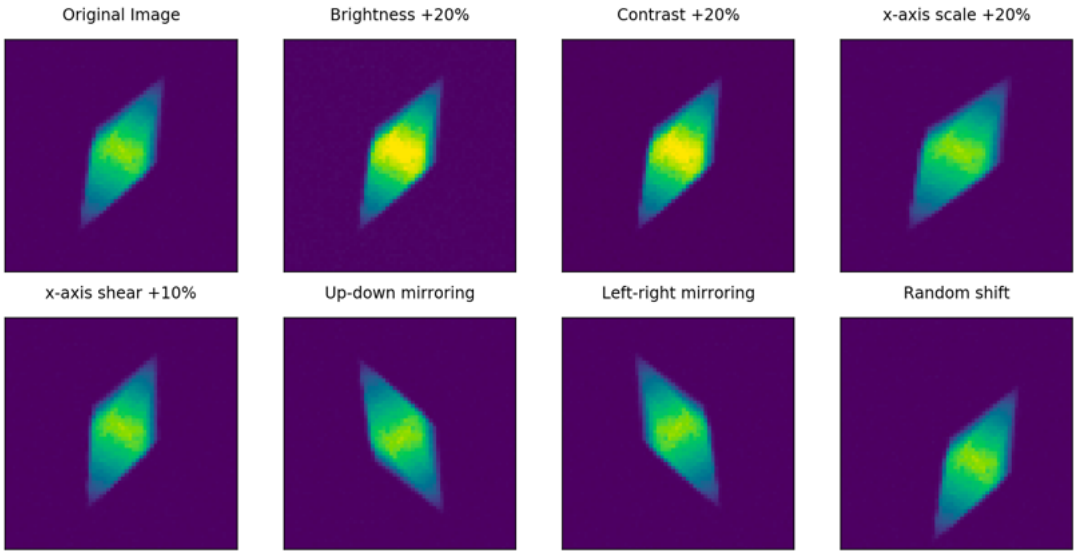


Figure 7.10: Effect of augmentations. Top row, left to right: original image, brightness +20%, contrast +20%, x-axis rescale +20%. Bottom row, left to right: x-shear +10%, up-down mirroring, left-right mirroring, random shift (x and y axis).

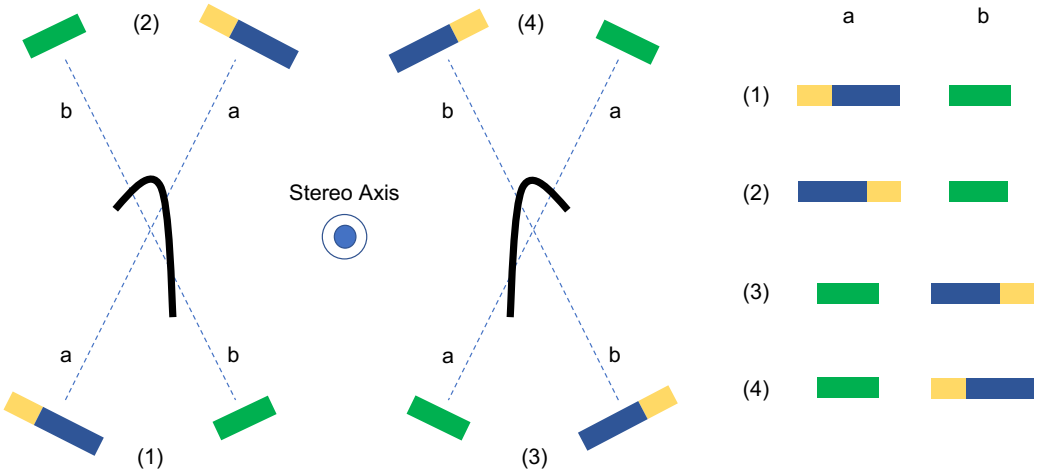


Figure 7.11: 1D density projections of two enantiomers of a 2D chiral structure (black) at stereopair positions a, b and a, b + 180°. The colour of the block roughly corresponds to the projected density (blue = low density, green = medium density, yellow = high density).

of structure, so mirroring the projections in this fashion will not change the chirality of the stereopair on its own. If the projections were mirrored in this way and the ordering of the stereopair a, b was reversed (producing b, a), however, this would produce a stereopair equivalent to a, b for the opposite chirality of structure, so reversing the ordering of a stereopair and mirroring the projections changes the chirality. Likewise, reversing the ordering of stereopair a, b alone would produce a stereopair equivalent to a, b + 180° for the opposite chirality of structure, so reversing the ordering of a stereopair changes the chirality.

If the mirror plane is perpendicular to the stereo axis (i.e. the plane of Figure 7.11 itself), none of these rules apply and mirroring the projections without reversing the ordering of the stereopair produces the opposite chirality. Therefore, for simplicity, all mirroring was performed perpendicular to the stereo axis in this study (aside from Section 7.4.2). Since the stereo axis was vertical in this study, this equated to mirroring the image along the horizontal axis.

7.2.1.3 Pre-Processing

Four pre-processing steps were applied to the simulated data before being fed into the neural network — background subtraction, normalisation, centre of intensity alignment and mean pooling — all of which would be equally applicable to experimental data. Since a flat background was used in these simulations, a simple median background subtraction method was used in this study. Images were normalised by rescaling the pixel intensities of the background-subtracted image from 0-65535 to 0-1, enabling this model to be applied to images of other memory sizes. The particle was then centered by binarising the image by its mean intensity, performing a centre of intensity calculation (akin to a centre of mass calculation) on the binarised image, then rolling the image axes so that the centre of the particle moves to the centre of the image, thus eliminating the impact of random shifts in particle position in different images. Finally, 2×2 mean pooling with a stride of 2 was used to reduce the size of the simulated image from 64×64 to 32×32 in order to speed up the training of the neural network. Additionally, the in-plane orientation of the particle (z-rotation) was measured here. This was important because, as will be discussed in section 7.2.2.3, it was found that successfully training networks on data with fully randomised orientations was difficult. Instead, networks had to be trained on subsets of the total dataset that contained particles with

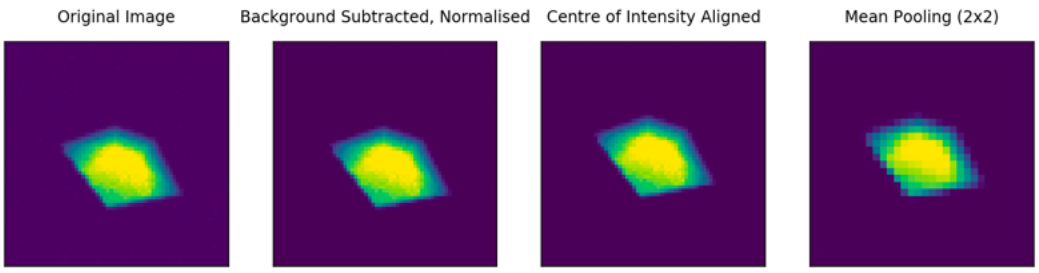


Figure 7.12: Pre-Processing Steps. Left to right: original image, background subtraction and normalisation, centre of intensity alignment, mean pooling (2×2).

a narrow range of z-orientations first, then retrained on the total dataset afterwards. The impact of each pre-processing step is shown in Figure 7.12.

7.2.1.4 CNN structure and training parameters

Although none of them was designed to work on stereopairs, AlexNet, ZF Net and VGG Net were all key sources of inspiration for the structure of the CNN used in this study. AlexNet provided the idea of using a repeating unit that consisting of a convolutional layer, a max pooling layer and a dropout layer. ZF Net provided the idea of maintaining the same convolutional filter size (in two dimensions) and allowing the downsizing effects of the convolutional and pooling layers to expand the effective receptive field. VGG Net provided the idea of doubling the number of filters with each convolutional layer to increase the number of features examined as the level of abstraction increases. The structure of the final CNN employed, to which the input was a 2×32×32 array (2 32×32 images per stereopair), was as follows:

- 3D Convolutional Layer (32 3×3×m_1 filters, stride = 1) (output = (2/m_1)×30×30)) (ReLU activation)
- Max Pooling Layer (2×2 filter, stride = 2) (output = (2/m_1)×15×15))
- Dropout Layer (dropout rate = 0.2)
- 3D Convolutional Layer (64 3×3×m_2 filters, stride = 1) (output = (2/(m_1×m_2))×13×13)) (ReLU activation)

- Max Pooling Layer (2×2 filter, stride = 2) (output = (2/(m_1×m_2))×6×6))
- Dropout Layer (dropout rate = 0.2)
- 3D Convolutional Layer (128 3×3×m_3 filters, stride = 1) (output = (2/(m_1×m_2×m_3))×4×4)) (ReLU activation)
- Max Pooling Layer (2×2 filter, stride = 2) (output = (2/(m_1×m_2×m_3))×2×2))
- Dropout Layer (dropout rate = 0.2)
- Flattening layer
- Dense Layer (64 nodes, ReLU activation)
- Dropout Layer (dropout rate = 0.2)
- Dense Layer (2 nodes, softmax activation)

where m_1, m_2 and m_3 were either 1 or 2. When m = 1, the convolutional operation was performed on each image separately. When m = 2, the convolutional operation was performed on both images simultaneously, effectively producing a single combined feature map. Four different network configurations were tested: ones with a combination element in the first (henceforth referred to as 211), second (121) and third (112) convolutional layers and one with no combination element at all (111) (the order of complexity from lowest to highest is 211 < 121 < 112 < 111). Since there were only two images in each starting pair, no more than one layer could have m = 2 as the output from this layer was always a single feature map, which would have had nothing to convolve with in a second m = 2 layer. All networks used sparse categorical cross-entropy as the loss function and Adam as the optimiser (learning rate = 0.001, $\beta_1 = 0.9$, $\beta_2 = 0.999$, ϵ = 1e-07).

7.2.2 Testing

At all stages of development and testing, the height of the particles was kept at a constant value of 1.0, with the radius being varied to produce different shapes where desired. All the particles had four-fold symmetry ($n = 4$) and a chiral twist angle of ±22.5°. The angle of stereo rotation

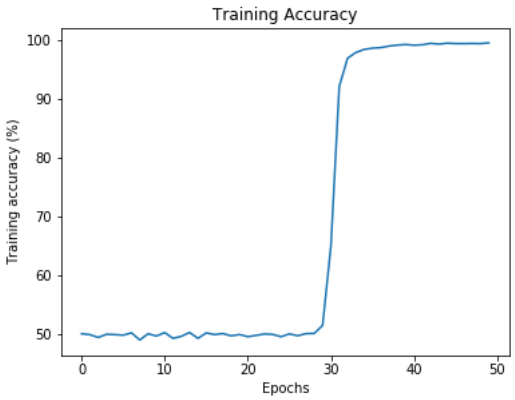


Figure 7.13: Graph of training accuracy vs number of epochs (Taken from Section 7.2.2.3, initial training, 111 network).

was 10°, with the axis of stereo rotation being kept parallel to the y-axis of the image. Training datasets of 20000 stereopairs and testing datasets of 1000 stereopairs were created for each test, with early training datasets containing images of 20000 different simulated stereopairs without any augmentation. All datasets contained a racemic mix of left-handed and right-handed particles and the range of y-rotation was ±180°. Ten networks of each configuration (211, 121, 112 and 111) were trained in order to obtain ensemble statistics, since each network was initialised with random weights and biases and therefore produced different outcomes. A training time of 50 epochs for each training stage was selected based on early testing in the development, which indicated that, within this time, most networks would either achieve their maximum potential accuracy or fail to train altogether (test accuracy ≈50%). No early stopping was used in order to assess the effects of overfitting and underfitting properly. The 'average convergence epoch' quoted here is the average number of epochs required for each network to achieve a validation accuracy of greater than 95% during training (for those networks that achieved 95% accuracy). It should be noted that, at every stage, the networks were trained on control datasets with a stereo angle of 0° (i.e. same orientation in both images of the pair) and these networks always failed to converge, demonstrating that a single projection contains insufficient information to determine the handedness of the simulated particles.

Initial tests showed that models trained on data with fully randomised orientations, scales,

radii and noise failed to train, so the development was broken down into stages with increasingly complex datasets. All the results in this section are separated into two categories of tables. The first category details the number of networks achieving 95% validation accuracy and the number of epochs required to reach this stage. While not as good a metric of performance as test accuracy, the speed at which a network converges can be a good indicator of suitability of a given network configuration, especially when the test accuracies are extremely similar. In later sections, where two training steps were required to accommodate different orientations, a table is given for each step. The second category details the number of networks achieving 55% test accuracy and the average test accuracy of all such networks. 55% is chosen as the minimum value because, before a network begins to converge, its accuracy will typically fluctuate around 47-53%, as shown in Figure 7.13. A network whose test accuracy exceeds 55% is typically one that has at least begun to converge on a final value.

7.2.2.1 Changing starting y-rotation

To start with, only the handedness and starting y-rotation of the simulated particles in the dataset were randomised, with all other variable values remaining fixed (scale = 0.8, radius = 0.5, x-rotation = 0°, z-rotation = 0°) and no noise being added to the images.

As shown in Tables 7.1 and 7.2, all of the 211, 121 and 112 networks successfully trained and achieved validation and test 100% accuracy, while the 111 networks failed to train at all. The number of epochs required to achieve 95% accuracy increased with increasing network complexity.

Table 7.1: Results of training with random starting y-rotation.

Network configuration	No. of networks achieving 95% validation accuracy (/10)	Average no. of epochs to achieve 95% validation accuracy
211	10	1.5±0.5
121	10	2.1±0.6
112	10	2.3±0.9
111	0	N/A

Table 7.2: Test accuracies of networks trained with random starting y-rotation.

Network Configuration	No. of networks with $> 55\%$ test accuracy (/10)	Average test accuracy of networks with $> 55\%$ test accuracy (%)
211	10	100.0±0.0
121	10	100.0±0.0
112	10	100.0±0.0
111	0	N/A

7.2.2.2 Changing shapes and sizes

In the next stage of development, the scale and radius were also randomised to assess the networks' capacity to train on different shapes and sizes of particles. The scale range was 0.7-0.9, the radius range was 0.3-0.7 and all other variable values remained fixed (x-rotation = 0°, z-rotation = 0°) with no noise being added to the images.

As shown in Tables 7.3 and 7.4, most of the 211, 121 and 112 networks successfully trained and those that did achieved 100% test accuracy (the others failed to train above ∼50%). Once again, the 111 networks failed to train at all. In this case, the 121 networks achieved 95% validation accuracy faster than the simpler 211 networks, which was an early indication that the 211 networks were less well-suited to the task, likely due to insufficient complexity.

Table 7.3: Results of training with random shapes and sizes.

Network configuration	No. of networks achieving 95% validation accuracy (/10)	Average no. of epochs to achieve 95% validation accuracy	Average test accuracy of converged networks (%)
211	9	3.2±0.4	100.0±0.0
121	10	2.7±0.5	100.0±0.0
112	8	4.7±3.0*	100.0±0.0
111	0	N/A	N/A

* 7 in 3-5 epochs, 1 in 12 epochs

Table 7.4: Test accuracies of networks trained with random shapes and sizes.

Network Configuration	No. of networks with > 55% test accuracy (/10)	Average test accuracy of networks with > 55% test accuracy (%)
211	9	100.0±0.0
121	10	100.0±0.0
112	8	100.0±0.0
111	0	N/A

7.2.2.3 Orientations

The next stage of development was to train the networks on datasets with randomly varying initial x- and z-rotations, but all four of the network configurations consistently failed to train on such datasets. This led to the use of 'curriculum learning', where networks are first trained on a comparatively simple dataset and then retrained on more complicated datasets covering incrementally larger volumes of parameter space [154]. In this test, the networks were first trained for 50 epochs on a dataset of 20000 stereopairs in which the z-rotation of the particles was fixed at 0° while the x-rotation was allowed to vary between ±arctan(radius/height) for the reasons discussed in Section 7.2.1.1, with all other variables being the same as before (scale range = 0.7-0.9, radius range = 0.3-0.7, no added noise). These networks were subsequently retrained for a further 50 epochs on a second dataset of 20000 stereopairs with a z-rotation range of +90°- -90° (all other variables kept the same), then tested on a 1000-stereopair dataset with the same conditions as the second dataset (N.B. No testing was performed between the two training sessions, hence why no test accuracy is given in the first table).

As shown in Table 7.5, even at the first stage, where the only difference from before is the introduction of varying x-rotation, the number of networks that successfully train decreased noticeably as the network complexity increased, showing that underfitting (see Chapter 4) becomes a significant issue at this level. Once again, the 121 networks achieved 95% in fewer epochs than 211 networks, both in initial training and retraining, as shown in Tables 7.5 and 7.6. All the networks that successfully trained in the first stage successfully retrained in the second stage, while those that failed to train in the first stage also failed in the second stage. A single 111 network also successfully trained on this occasion, demonstrating that the architecture is a viable solution to this task, but that it is prone to underfitting. As shown in Table 7.7, the average test accuracies

of the 121 and 112 networks were slightly higher than those of the 211 and 111 networks, but the differences were fairly marginal, making it difficult to assess their relative performance at this juncture.

Table 7.5: Results of initial training with random y- and x-rotation.

Network configuration	No. of networks achieving 95% validation accuracy (/10)	Average no. of epochs to achieve 95% validation accuracy
211	10	7.1±5.1
121	9	4.9±1.5
112	4	6.2±1.3
111	1	31.0±N/A

Table 7.6: Results of retraining with random z-rotation.

Network configuration	No. of networks achieving 95% validation accuracy (/10)	Average no. of epochs to achieve 95% validation accuracy
211	10	16.0±4.5
121	9	8.0±0.7
112	4	9.3±2.6
111	1	16.0±N/A

Table 7.7: Test accuracies of networks trained with random z-rotation.

Network Configuration	No. of networks with $>$ 55% test accuracy (/10)	Average test accuracy of networks with $>$ 55% test accuracy (%)
211	10	98.7±0.6
121	9	99.6±0.3
112	4	99.6±0.4
111	1	98.2±N/A

7.2.2.4 Noise

The same process as before was repeated with noise added (intensity upscale = 50000, background = 2000, Gaussian blurring = 0.5 pixels, a = 10, b = 0.1).

As shown in Tables 7.8, 7.9 and 7.10, most of the statistics strongly resemble the same procedure without noise. The most significant difference is the number of epochs required by the 211 to train in both stages, which is significantly higher than before, suggesting that this configuration is particularly sensitive to noise.

Table 7.8: Results of initial training with pixel noise.

Network configuration	No. of networks achieving 95% validation accuracy (/10)	Average no. of epochs to achieve 95% validation accuracy
211	10	8.8±4.8
121	10	5.0±1.2
112	4	6.5±3.1
111	1±N/A	32.0±N/A

Table 7.9: Results of retraining with pixel noise.

Network configuration	No. of networks achieving 95% validation accuracy (/10)	Average no. of epochs to achieve 95% validation accuracy
211	10	24.0±10.0
121	10	9.3±5.0
112	4	8.5±1.0
111	1	21.0±N/A

Table 7.10: Test accuracies of networks trained with pixel noise.

Network Configuration	No. of networks with $> 55\%$ test accuracy (/10)	Average test accuracy of networks with $> 55\%$ test accuracy (%)
211	10	98.0±1.0
121	10	99.5±0.3
112	4	99.4±0.2
111	1	98.2±N/A

7.2.2.5 Augmentation

In this stage of development, the two 20000-stereopair datasets were replaced with 500-stereopair datasets that were augmented 40 times to create 20000-stereopair datasets. The brightness and contrast were altered randomly in a range of ±20%, the images were rolled by up to 10 pixels along both axes, the images were rescaled randomly and independently in both the x and y directions in a range of ±20% and a random shear of up to ±10% was added independently in both directions. For comparison, a second set of networks was trained on 500-stereopair datasets in which the images were copied 40 times to create 20000-stereopair datasets without any augmentation processes being used. The same simulation conditions as before were used throughout (scale range = 0.7-0.9, radius

range = 0.3-0.7, x-rotation = $\pm$arctan(radius/height), z-rotation = 0° (1st set), $\pm$90° (2nd set), intensity upscale = 50000, background = 2000, Gaussian blurring = 0.5 pixels, a = 10, b = 0.1).

As shown in Tables 7.11, 7.12 and 7.13, networks of all four types successfully trained, achieving test accuracies that, while marginally lower than before, were still reasonably high. The average test accuracy was significantly higher for networks trained on augmented data than networks trained on copied data, the results of which are shown in Tables 7.14, 7.15 and 7.16. This demonstrates that the augmentation scheme used here is an effective method of overcoming the limitations of small datasets and superior to simply copying the data.

Table 7.11: Results of initial augmented training (500 original simulated stereopairs, 40× augmentation).

Network configuration	No. of networks achieving 95% validation accuracy (/10)	Average no. of epochs to achieve 95% validation accuracy
211	10	7.8±6.1
121	10	3.8±1.3
112	9	6.0±4.3
111	9	5.8±3.7

Table 7.12: Results of augmented retraining (500 original simulated stereopairs, 40× augmentation).

Network configuration	No. of networks achieving 95% validation accuracy (/10)	Average no. of epochs to achieve 95% validation accuracy
211	9	26.0±12.5
121	10	11.0±2.4
112	9	13.7±2.9
111	7	12.1±2.4

Table 7.13: Test accuracies of networks trained on augmented data (500 original simulated stereopairs, 40× augmentation).

Network Configuration	No. of networks with > 55% test accuracy (/10)	Average test accuracy of trained networks (%)
211	10	95.9±0.8
121	10	98.0±0.5
112	9	97.4±0.7
111	7	97.6±0.5

Table 7.14: Results of multiplicative initial training (500 original simulated stereopairs, 40× multiplication).

Network configuration	No. of networks achieving 95% validation accuracy (/10)	Average no. of epochs to achieve 95% validation accuracy
211	10	3.2±0.8
121	10	2.3±0.5
112	10	1.9±0.3
111	10	2.1±0.3

Table 7.15: Results of multiplicative retraining (500 original simulated stereopairs, 40× multiplication)

Network configuration	No. of networks achieving 95% validation accuracy (/10)	Average no. of epochs to achieve 95% validation accuracy
211	0	N/A±N/A
121	0	N/A±N/A
112	0	N/A±N/A
111	0	N/A±N/A

Table 7.16: Test accuracies of networks trained on multiplied data (500 original simulated stereopairs, 40× multiplication).

Network Configuration	No. of networks with > 55% test accuracy (/10)	Average test accuracy of trained networks (%)
211	10	76.6±1.4
121	10	87.9±1.4
112	10	88.5±1.9
111	10	88.4±2.4

One result of particular note is the fact that 70% of the 111 networks successfully trained on the augmented data, which indicates that the aforementioned underfitting problem that 111 networks suffer can be overcome by training on datasets with lower variance. This phenomenon was further explored by training ten 111 networks on 500-stereopair datasets augmented in the same manner as before, then retrained on a dataset with 20000 original stereopairs with 1× augmentation. As shown in Table 7.17, nine of the ten networks tested achieved 95% in the first stage. All of these networks achieved >95% validation accuracy in subsequent stages and >55% test accuracy in the final stage. The average test accuracy of these nine networks was 99.3±0.3%, which is higher than

97.6±0.5% achieved by the 111 networks that had only completed the first two steps of training (as shown in Table 7.13). This demonstrates that the 111 configuration is viable for this application and can be used to train on large datasets with high variance of data, so long as networks of this type are trained on lower-variance datasets first.

Table 7.17: Validation data results for 111 networks trained on 500 original simulated stereopairs, 40× augmentation, then retrained on 20000 original simulated stereopairs, 1× augmentation.

Training Stage	No. of networks achieving 95% validation accuracy (/10)	Average no. of epochs to achieve 95% validation accuracy
500 original stereopairs × 40 augmentation (10°)	9	6.0±3.9
500 original stereopairs × 40 augmentation (90°)	9	12.3±4.0
20000 original stereopairs × 1 augmentation (90°)	9	1.0±0.0

7.3 Emulating 'realistic' experiments

Unlike the simulated datasets used during development, a 'realistic' dataset would likely only contain a relatively small number of stereopairs (e.g. 1000), the z-orientation of which would completely randomised. As learned during development, the key to successfully training a network on such a dataset is to first train it on data with a narrow range of z-orientations and then retrain it on a larger range, so the z-orientation measurement was added to the list of pre-processing steps to facilitate the easy creation of sub-datasets of different angular ranges (±10°, ±20°, etc.). Since these sub-datasets contain far fewer stereopairs than the overall dataset (e.g. ∼11% for ±10°), a process dubbed 'dynamic augmentation' was devised in which each sub-dataset was augmented in the same manner as before (brightness and contrast adjustment range = ±20%, shift range = ±10 pixels, rescaling range = ±20%, shear range = ±10%) to create a dataset of approximately 20000 stereopairs. This technique was compared against 'single augmentation', where the initial dataset was augmented to create 20000 stereopairs first and sub-datasets of this augmented dataset were created subsequently, meaning that the sub-datasets contained far fewer than 20000 stereopairs (∼2000 for ±10°). Two different training regimes were tested for both techniques. In the first,

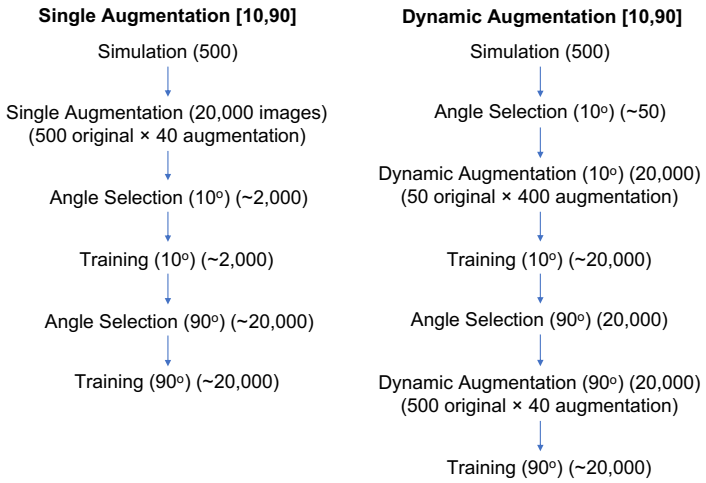


Figure 7.14: Flowchart of single and dynamic augmentation processes with size of dataset at each stage in brackets (for the case of 500 original simulated stereopairs).

each network was trained on a dataset with an angular range of $\pm 10^\circ$ followed by another dataset with an angular range of $\pm 90^\circ$ (henceforth referred to as [10,90]). In the second, each network was trained on a series of datasets in which the angular ranges increased from $\pm 10^\circ$ to $\pm 90^\circ$ in increments of $\pm 10^\circ$ (henceforth referred to as [10:10:90]). A flowchart demonstrating the single and dynamic augmentation processes for [10,90] and the effect on the size of the dataset at each stage is shown in Figure 7.14. In the case of 20000 original simulated stereopairs, the same augmentation procedures were applied to each stereopair, but the size of the dataset was not increased.

The training datasets in these experiments consisted of stereopairs of four-fold symmetry particles ($n = 4$) with randomised shape, size and orientation in the x, y and z directions, with noise being added to all images (height = 1.0, chiral twist angle = $\pm 22.5^\circ$, radius range = 0.3-0.7, scale range = 0.7-0.9, y-rotation range = -180° - +180°, x-rotation range = $\pm$arctan(radius/height), z-rotation range = $\pm 90^\circ$, intensity upscale = 50000, background = 2000, Gaussian blurring = 0.5 pixels, a = 10, b = 0.1). These datasets contained 125, 250, 500, 1000 and 20000 stereopairs, augmented in the manners described. The validation and testing datasets contained 1000 stereopairs generated in the same manner as the training datasets. As before, the stereo angle was 10°.

As shown in Figures 7.15 and 7.16, the test accuracy of all network types increased as the number of original simulated stereopairs in the dataset increased (and hence the level of augmentation

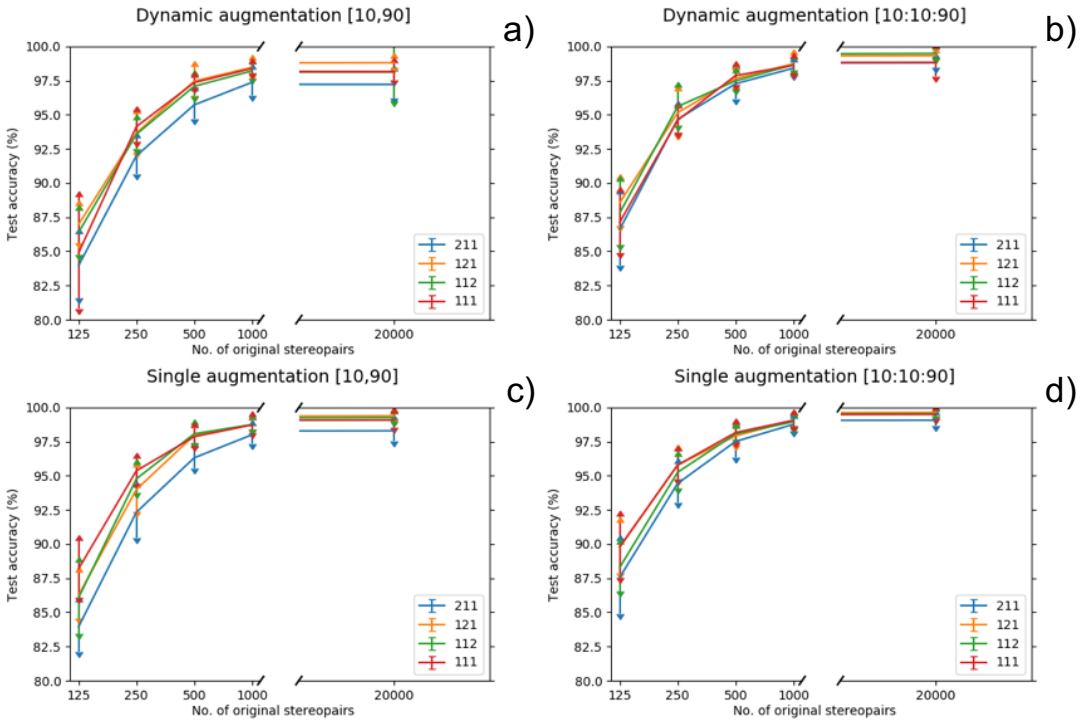


Figure 7.15: Comparison of average test accuracies of each network configuration for a) dynamic augmentation [10,90], b) dynamic augmentation [10:10:90], c) single augmentation [10,90] and d) single augmentation [10:10:90].

decreased), demonstrating the advantages of having larger amounts of 'experimental' data to train on, but also demonstrating that smaller amounts of 'experimental' data can be used to train networks, albeit to lower standards. The difference in performance between the dynamic and single augmentation schemes was almost negligible, but the best average test accuracy for each number of original simulated stereopairs was achieved using single augmentation rather than dynamic augmentation. This difference ranged between 0.1% for 20000 original simulated stereopairs and 1.3% for 125 original simulated stereopairs, as shown in Appendix D. The test accuracies of the networks trained using the longer regime [10:10:90] were significantly better than those trained using the shorter regime [10,90]. This can be explained by the fact that the range of z-orientations upon which the data is trained expands more slowly and/or the fact that the total number of epochs is much greater for the [10:10:90] scheme (450 epochs) than the [10,90] scheme (100 epochs). The results also showed that the test accuracy roughly correlated to the complexity of the network

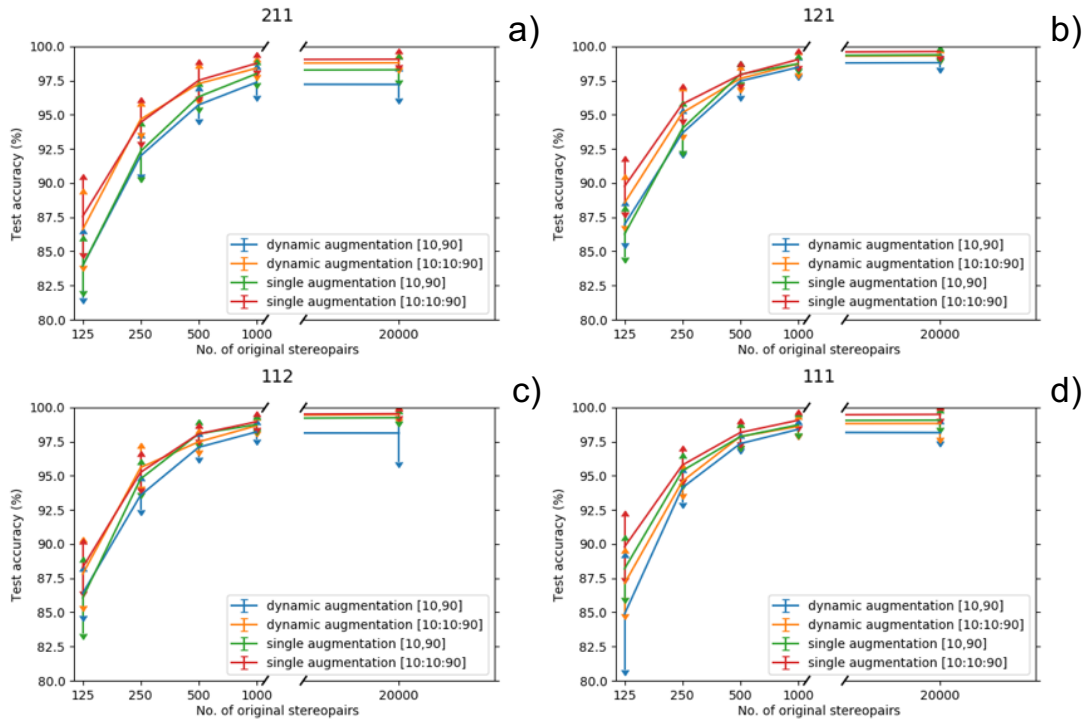


Figure 7.16: Comparison of average test accuracies of each augmentation type for a) 211 networks, b) 121 networks, c) 112 networks and d) 111 networks.

type, with 111 (most complex) networks usually slightly outperforming the other networks and 211 networks (least complex) consistently performing the worst, especially in more highly augmented datasets. The difference in performance between the three most complex network types (111, 112 and 121), however, was fairly marginal and the predicted correlation between complexity and performance did not hold in all cases.

One other notable result is that the probability of successful network training (see Figure 7.17) was significantly greater for 1000 original stereopairs and below than for 20000 original stereopairs. This can be explained by the lower variability of the smaller original datasets (even after going through the augmentation process), which makes it easier for the networks to train. This could be exploited, however, to train more complex 112 and 111 networks, whose probabilities of successfully training were higher in smaller, heavily augmented datasets than larger, lightly augmented original datasets and whose test accuracies were noticeably higher for datasets containing very

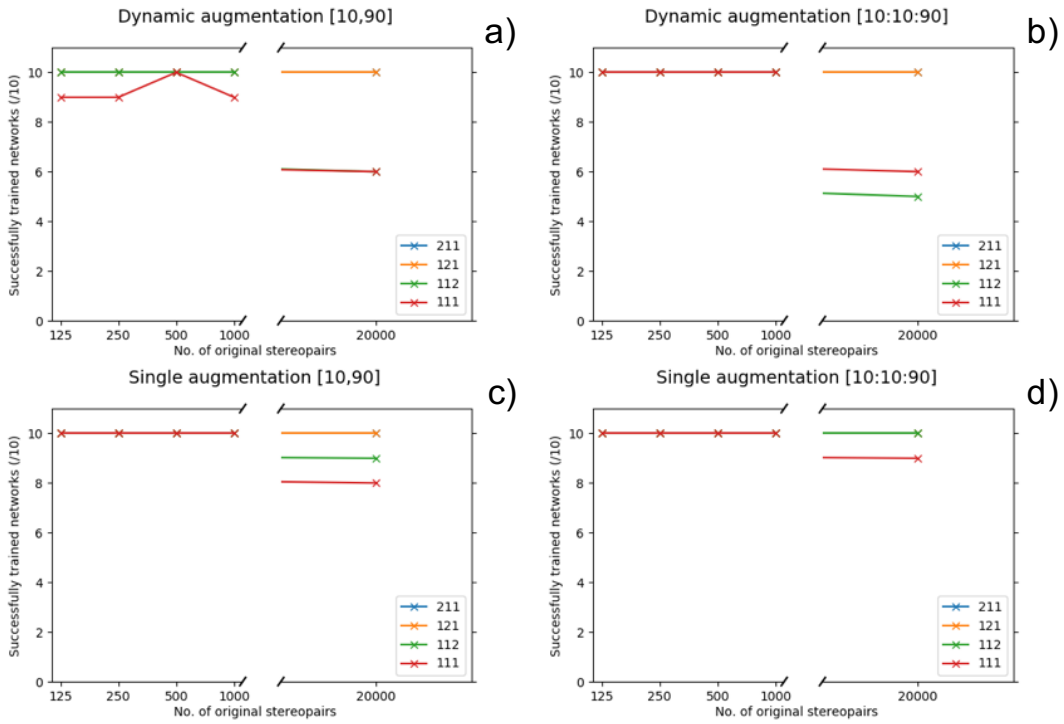


Figure 7.17: Comparison of number of successfully trained networks of each configuration using a) dynamic augmentation [10,90], b) dynamic augmentation [10:10:90], c) single augmentation [10,90] and d) single augmentation [10:10:90].

small numbers of original stereopairs.

7.3.1 Varying augmentation levels

To examine the effect of varying the level of augmentation on test accuracy, ten networks of each type were trained using datasets containing 250 original simulated stereopairs that were augmented by different amounts: 40, 80 (standard), 160 and 320. Dynamic augmentation and the [10,90] training regime were used for these tests with all simulation and augmentation variables remaining the same as before. As shown in Figure 7.18, the test accuracy generally increased with augmentation factor for all network configurations, which indicated that increasing the augmentation factors increased the variability of data within the dataset and thus improved the quality of training. This effect was particular noticeable for the simplest network configuration: 211.

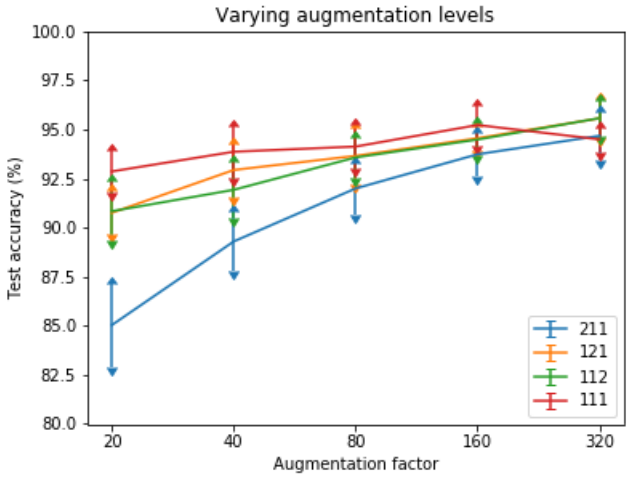


Figure 7.18: Average test accuracy of trained networks vs augmentation levels.

7.3.2 Varying stereo angle

To examine the effect of varying stereo angle on test accuracy, a series of networks were trained on datasets containing simulated stereopairs with different stereo angles (5° to 45° in increments of 5°). To see how this would be affected by different particle shapes, this was replicated using different chiral twists — 15°, 22.5° and 30° — and three different levels of symmetry — three-fold ($n = 3$), four-fold ($n = 4$) and five-fold ($n = 5$). The training datasets contained 250 original simulated stereopairs that were then augmented to create datasets of 20000 stereopairs. Once again, dynamic augmentation and the [10,90] training regime were used and all other simulation and augmentation variables were kept the same as before. Ten networks of each type were trained for each combination of stereo angle, chiral twist and n-fold symmetry.

As shown in Figure 7.19, there was a strong initial increase in test accuracy in all cases as the stereo angle increased above 0°, followed by dips in the test accuracy at ∼60° for $n = 3$, ∼45° for $n = 4$ and ∼35-40° for $n = 5$, then modest increases after this (followed, in the $n = 5$ case, by another dip). The nadirs of the first dips corresponded to ($360°/2n$) for all three cases — 60° ($n = 3$), 45° ($n = 4$) and 36° ($n = 5$) — at which angle it becomes impossible to distinguish between two enantiomers if the axis of rotational symmetry is aligned with the stereo axis for the reasons discussed in Section 7.2. Thus, determining the handedness of particles close to this orientation is

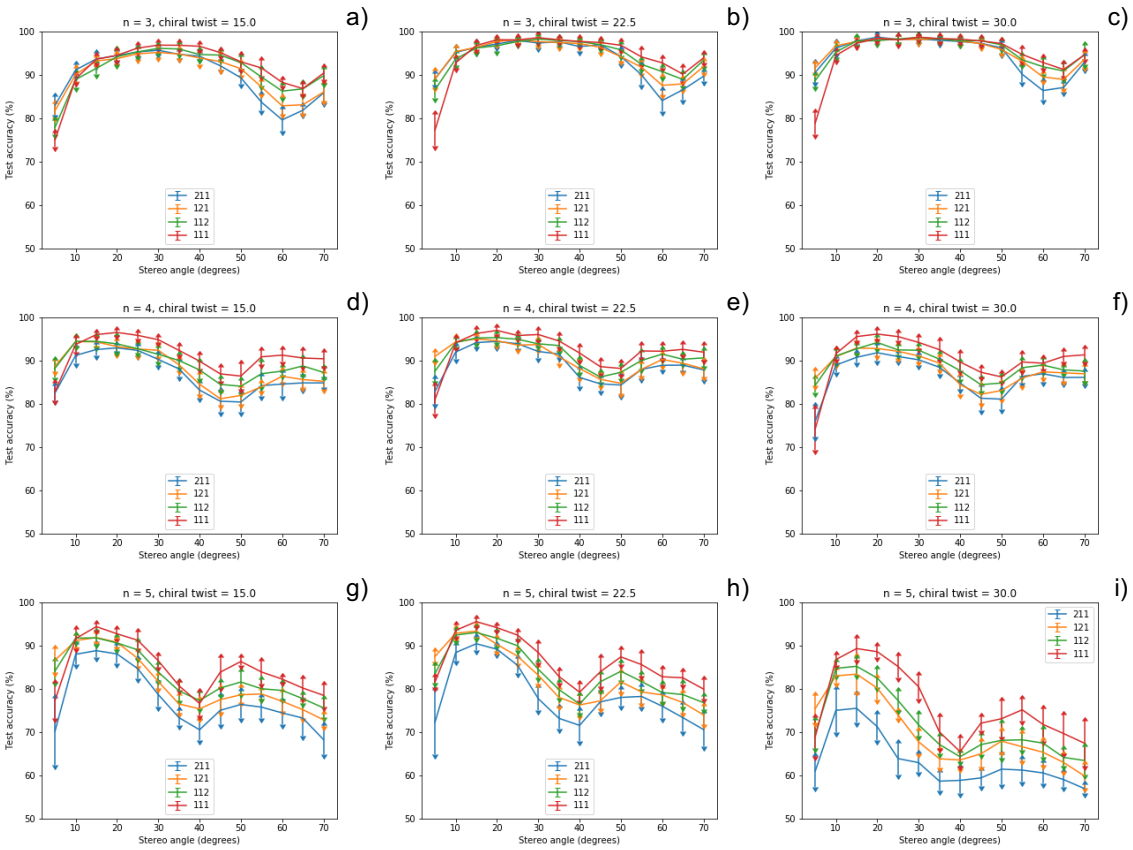


Figure 7.19: Average test accuracy vs stereo angle for a) $n = 3$, chiral twist = 15.0°, b) $n = 3$, chiral twist = 22.5°, c) $n = 3$, chiral twist = 30.0°, d) $n = 4$, chiral twist = 15.0°, e) $n = 4$, chiral twist = 22.5°, f) $n = 4$, chiral twist = 30.0°, g) $n = 5$, chiral twist = 15.0°, h) $n = 5$, chiral twist = 22.5° and i) $n = 5$, chiral twist = 30.0°.

difficult, while determining the handedness of particles further away from this orientation is easier, resulting in a slight reduction in test accuracy, but not a total fall-off to 50% (i.e. no better than random chance) as seen when the stereo angle is 0°.

Even disregarding these pronounced dips, there was still a clear overall decrease in test accuracy with increasing stereo angle in the $n = 5$ networks, with the $n = 4$ networks exhibiting similar behaviour to a lesser degree. This effect was not clearly visible in the $n = 3$ case, although this may be due to the range of stereo angles used. This fall-off in classification accuracy may indicate that the CNNs perceive stereopairs in much the same way as human brains, which use the small disparities between the two images of a stereopair with a small stereo angle to measure features [155]. As the stereo angle increases, the differences between the images increase and it becomes harder to interpret these disparities, resulting in a drop in classification accuracy.

There was little clear evidence of a correlation between chiral twist and test accuracy. The only notable exception was for the five-fold particles with a chiral twist of 30°, where the test accuracy was generally lower than in the cases of 15° and 22.5°. This may be because this chiral twist was close to $360°/2n$ for $n = 5$ — 36° — at which point the particle becomes achiral, as discussed in Section 7.2. There was a notable increase in test accuracy as the level of symmetry decreased (which is a promising sign for the future application of this system to real stereopairs of three-fold symmetry TeNPs). This may have been because particles with higher symmetry have more features (faces, edges, vertices), which may make them more complex to classify. The higher density of features also results in more features obscuring one another in the projections, as shown in Figure 7.20. Using higher resolution images may mitigate this issue.

7.4 Weak Labelling

All the methods previously described assume that the training dataset is 100% correctly labelled. In order to achieve this, a dataset containing both enantiomers of a chiral particle would have to be manually labelled, which would be prohibitively laborious, or two pure samples of each enantiomer would need to be synthesised, which tends to be difficult to guarantee. While ensuring 100% *ee* can be practically impossible, however, synthesising samples with high *ee* values (as estimated by

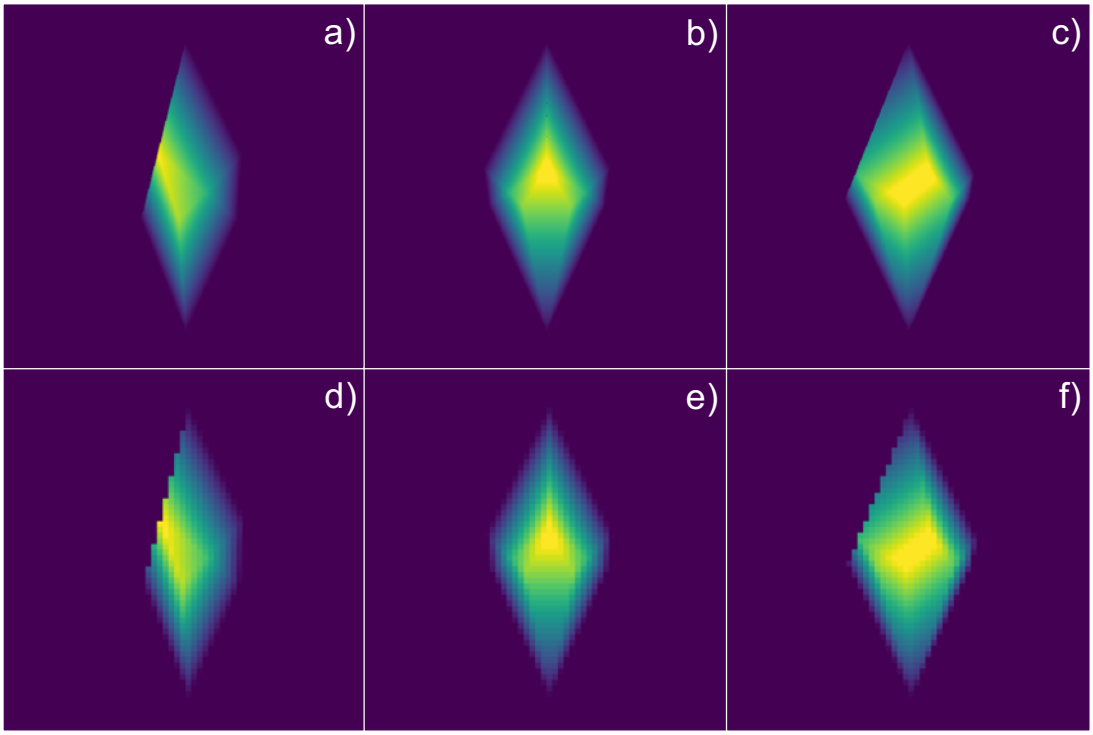


Figure 7.20: Comparison of STEM simulations of chiral particles with different symmetries and resolutions. a) 3-fold symmetry, 512×512 resolution. b) 4-fold symmetry, 512×512 resolution. c) 5-fold symmetry, 512×512 resolution. d) 3-fold symmetry, 64×64 resolution. e) 4-fold symmetry, 64×64 resolution. f) 5-fold symmetry, 64×64 resolution.

CD spectroscopy) is not [152]. Such samples could be used to train neural networks by employing a 'weakly supervised' approach.

In weak supervision, the set of labels for the training dataset may be incomplete (not all the data points have labels), inexact (the labels may only contain some of the information needed to categorise the data points) or inaccurate (some of the labels may not reflect the ground truth) [156]. Both of the methods discussed here involve the third case. The first method is to create an impure sample of each enantiomer and assign the majority label to all of the data taken from each sample, regardless of whether or not it is true for the individual particle. The second method is to do this for a sample of one enantiomer, then mirror all the data to create a second dataset. Due to the fundamental nature of chirality, all of the data in the second dataset will have the opposite handedness. This method is significantly simpler than the first, not just because it

only involves the creation of a single sample, but also because it is common for the quality of different enantiomeric nanocrystals to be different, even when produced using the same method and enantiomeric chemicals of the same supposed purity. This is because, in reality, the stock chemicals used may have slightly different concentrations depending on the enantiomer as well as unlisted impurities, both of which will affect the reaction rates and hence the quality of the enantiomeric nanocrystals produced. It is this weak labelling method that is tested here.

In these tests, simulated datasets of stereopairs containing 80% of one enantiomer and 20% of the other enantiomer were created, with all stereopairs in this set given the majority label. These datasets contained different numbers of original stereopairs — 125, 250, 500, 1000, 2000, 5000 and 10000 — and were then mirrored to create a second dataset, the stereopairs in which were given what was previously considered the minority label. These combined datasets were then augmented to create datasets containing 20000 stereopairs. Early testing revealed that this approach could not handle fully randomised orientation, size and shape all at the same time, so only orientation and size were randomised with shape being fixed (radius = 0.5), since images of the TeNPs upon which this work is based showed that the shape of the particles tends to remain roughly constant. This was reflected in the augmentation procedure, with the independent x and y shear and rescaling operations replaced by a uniform rescaling operation, the range of which was $\pm 10\%$. The brightness and contrast were also varied by $\pm 20\%$ as before and all other simulation conditions were kept the same as at the start of Section 7.3 (height = 1.0, chiral twist angle = $\pm 22.5^{\circ}$, scale range = 0.7-0.9, y-rotation range = $\pm 180^{\circ}$, x-rotation range = $\pm$arctan(radius/height), z-rotation range = $\pm 90^{\circ}$, intensity upscale = 50000, background = 2000, Gaussian blurring = 0.5 pixels, a = 10, b = 0.1). The datasets were then randomly shuffled before training. As before, the validation dataset consisted of 1000 stereopairs generated in the same manner and weakly labelled. Two test datasets, one containing 500 stereopairs of one enantiomer and the other containing 500 stereopairs of the other, were also generated in the same manner, but were 100% correctly labelled. This was done to test for any bias in the classification accuracy between the two enantiomers. All four network types (211, 121, 112 and 111) were tested using single and dynamic augmentation for both the [10,90] and [10:10:90] z-orientation training regimes described in Section 7.3. Ten of each type of network were trained and the results averaged.

Since 20% of the training labels were incorrect, the sign of a successfully trained network was one that achieved a training (or validation) accuracy of 80%, after which it should achieve a test accuracy of 100%. Overfitting was therefore a significant danger since the training process attempts to reach 100% training accuracy, which would result in 80% test accuracy at best. This was borne out by the results represented in Figure 7.22, which showed that the test accuracy increased to ∼100% as the final training accuracy decreased from 100% to 80%. As shown in Figure 7.23, there was no significant bias in favour of either the majority or minority handedness set. As expected, overfitting was more of an issue when using datasets with smaller numbers of original stereopairs and decreased as the variance of the datasets increased. Once again, the choice of single and dynamic augmentation made little difference to the final test accuracy, with dynamic augmentation outperforming single augmentation in some instances and vice versa. There was, however, a more notable impact on the success rate, with single augmentation resulting in a higher number of successfully trained networks than dynamic augmentation, particularly with higher numbers of original stereopairs in the dataset, as shown in Figure 7.21.

7.4.1 Varying weak labelling levels

To examine the impact of varying levels of weak labelling on the test accuracy, ten neural networks of each configuration were trained on datasets of 20000 stereopairs containing the same range of original simulated stereopairs as before — 125, 250, 500, 1000, 2000, 5000 and 10000 — with 20%, 10%, 5% and 0% weak labelling. Single augmentation and the [10,90] training regime were used and all other simulation and augmentation conditions were kept the same as before. As shown in Figure 7.24, the test accuracy increased with decreasing weak labelling across all network configurations and numbers of original stereopairs, with especially notable improvements for networks trained on datasets with smaller numbers of original stereopairs. This can be explained by the fact that the magnitude of any overfitting that took place was more strongly limited when the proportion of incorrectly labelled data was smaller.

To compare the weak labelling method with the 100% correctly labelled method, ten neural networks of each configuration were trained using the 100% correctly labelled method on datasets with the same range of original simulated stereopairs used in all the weak labelling method —

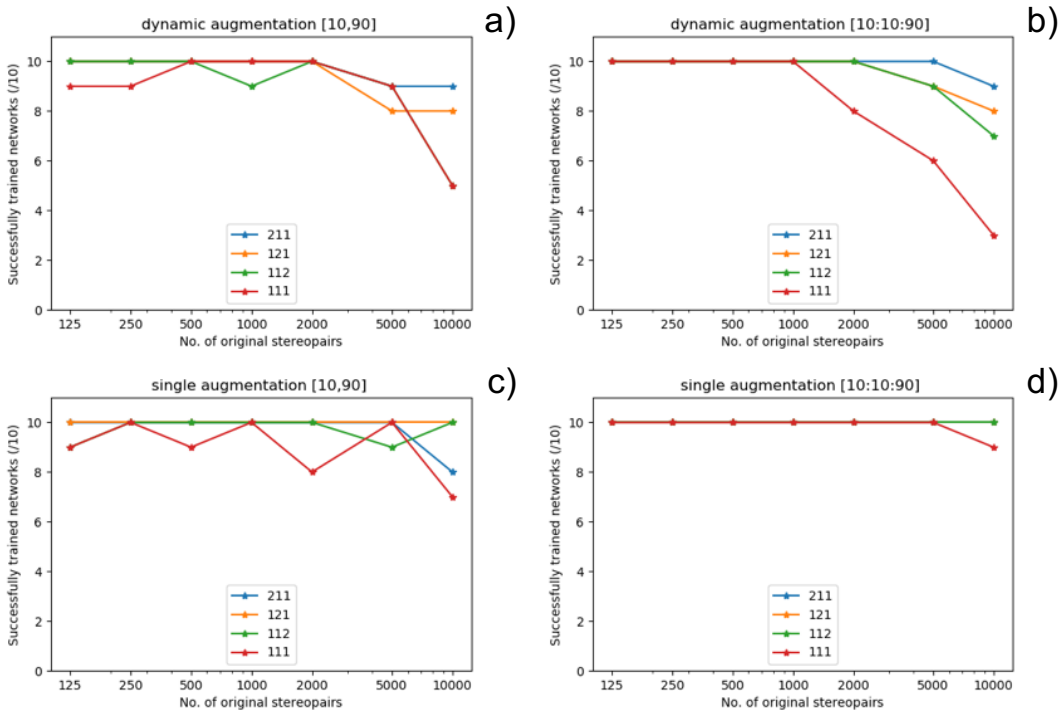


Figure 7.21: Number of successfully trained networks vs number of original stereopairs (20% weak labelling) for a) dynamic augmentation [10,90], b) dynamic augmentation [10:10:90], c) single augmentation [10,90] and d) single augmentation [10:10:90].

125, 250, 500, 1000, 2000, 5000 and 10000 — with the shape of the particles fixed (height = 1.0, radius = 0.5) and the same augmentation parameters as used in the weak labelling method to avoid changes of shape (brightness and contrast rescale = 20%, uniform rescaling = 10%). All other simulation parameters were the same as in Section 7.4. As shown in Figure 7.25, the results are comparable.

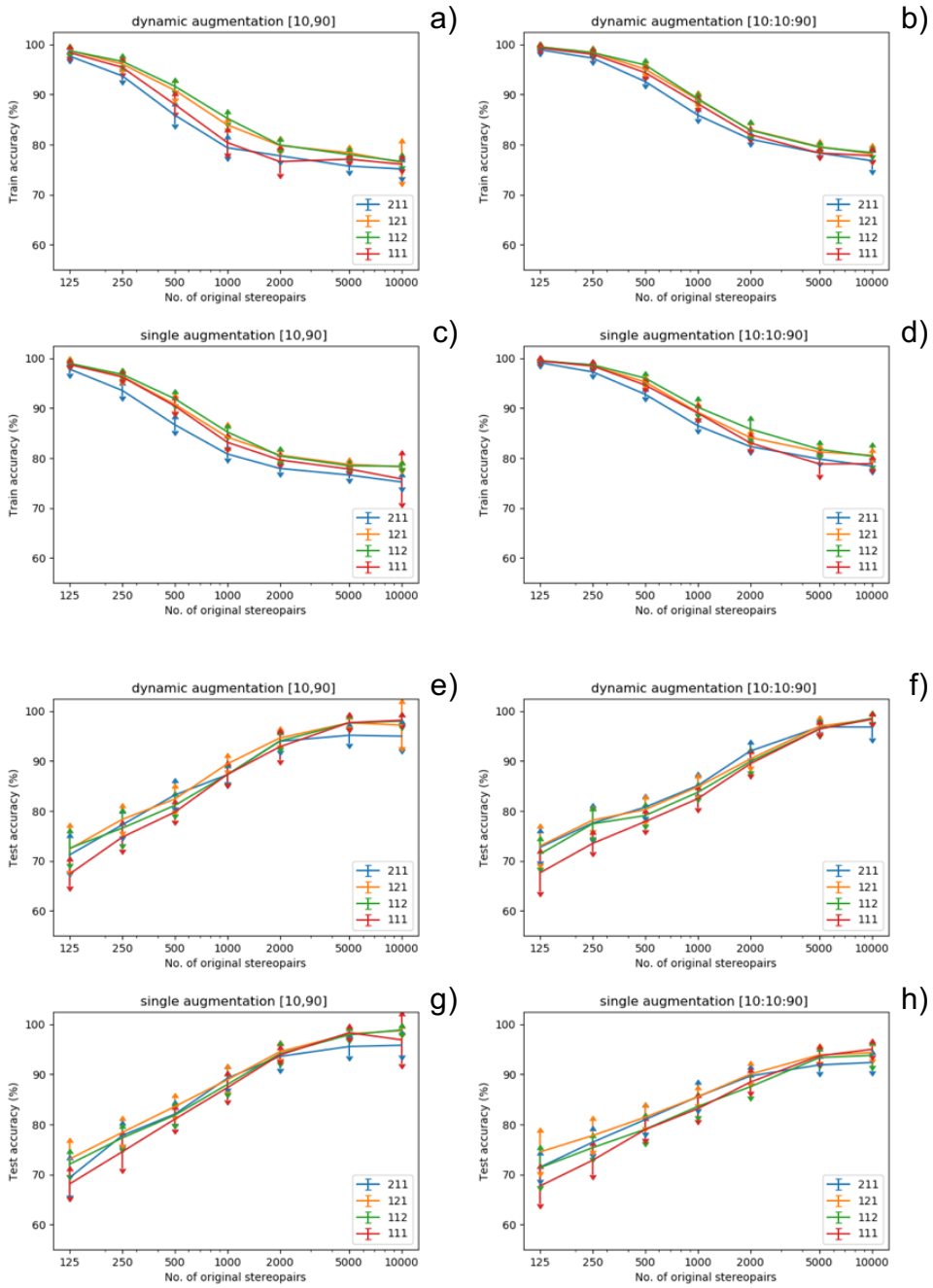


Figure 7.22: Average final training (a-d) and test (e-h) accuracy of trained networks vs number of original stereopairs (20% weak labelling). a) dynamic augmentation [10,90] training accuracy. b) dynamic augmentation [10:10:90] training accuracy. c) single augmentation [10,90] training accuracy. d) single augmentation [10:10:90] training accuracy. e) dynamic augmentation [10,90] test accuracy. f) dynamic augmentation [10:10:90] test accuracy. g) single augmentation [10,90] test accuracy. h) single augmentation [10:10:90] test accuracy.

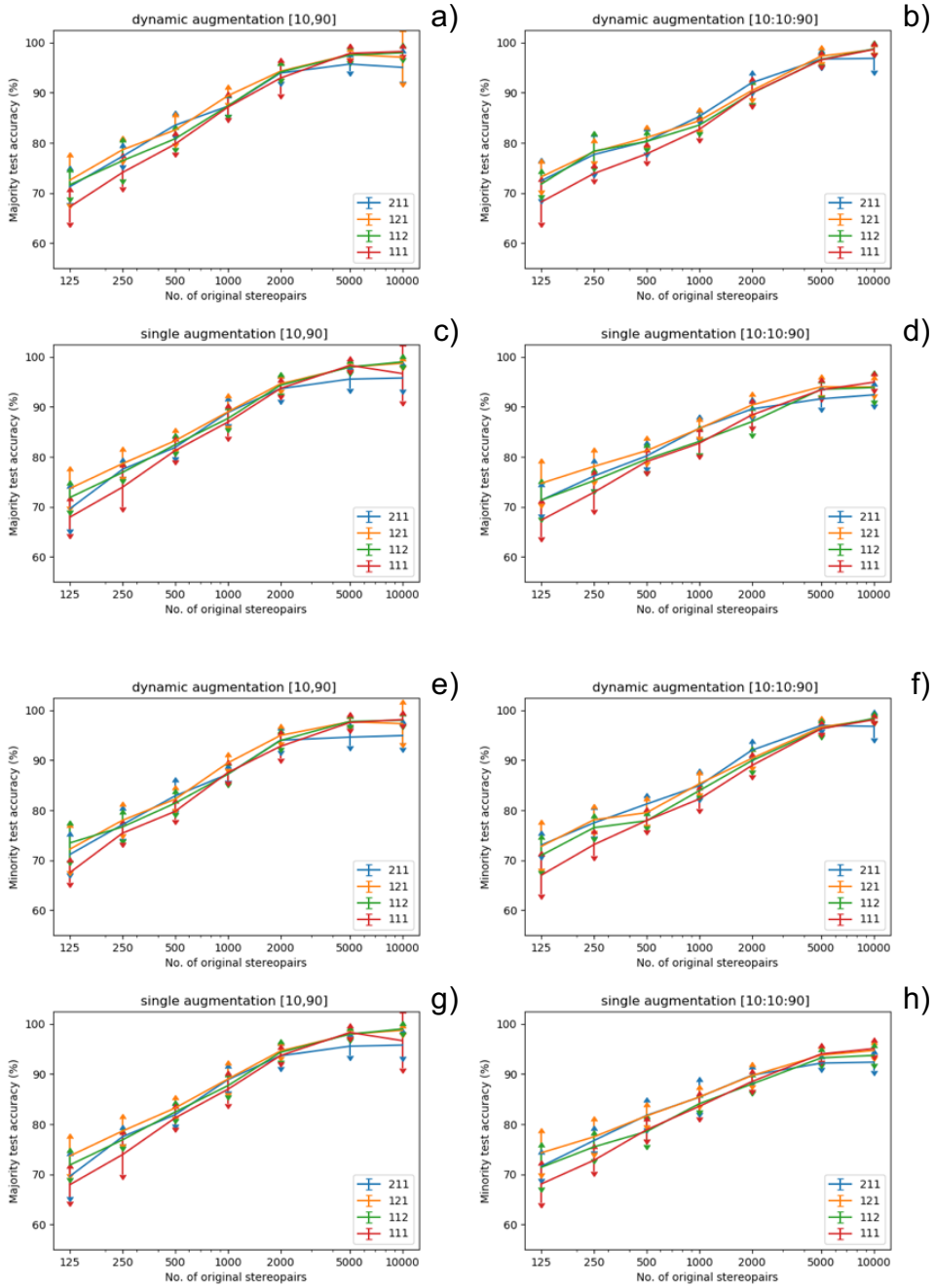


Figure 7.23: Average majority (a-d) and minority (e-h) test accuracy of trained networks vs number of original stereopairs (20% weak labelling). a) dynamic augmentation [10,90] majority test accuracy. b) dynamic augmentation [10:10:90] majority test accuracy. c) single augmentation [10,90] majority test accuracy. d) single augmentation [10:10:90] majority test accuracy. e) dynamic augmentation [10,90] minority test accuracy. f) dynamic augmentation [10:10:90] minority test accuracy. g) single augmentation [10,90] minority test accuracy. h) single augmentation [10:10:90] minority test accuracy.

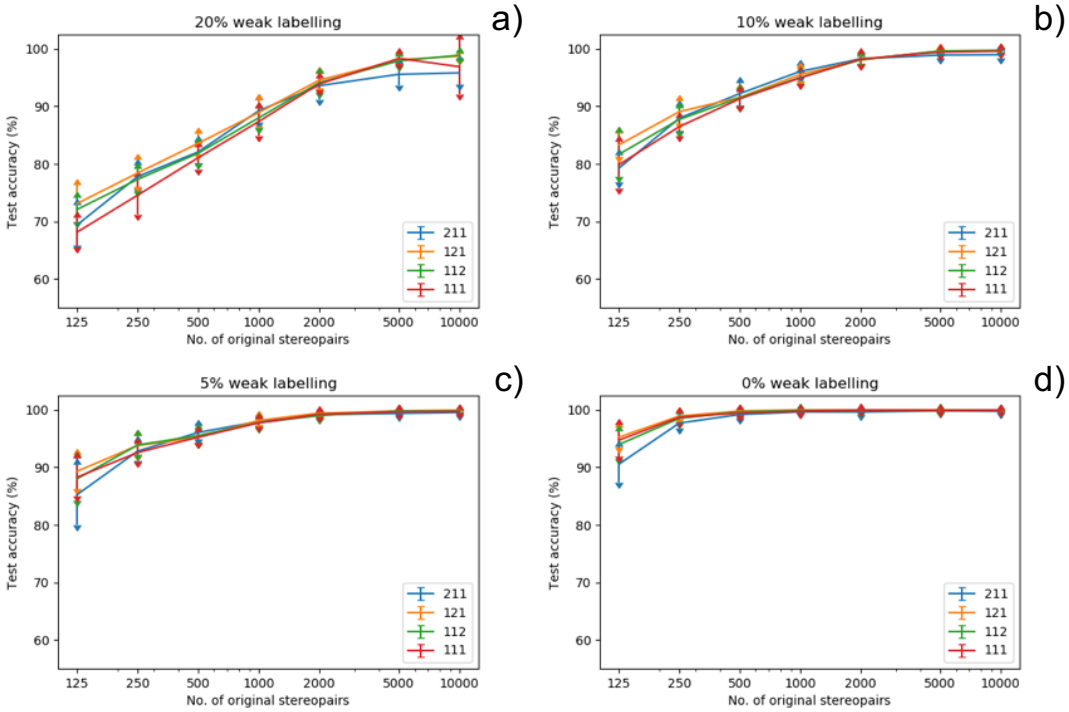


Figure 7.24: Average test accuracy of trained networks vs number of original stereopairs for a) 20% weak labelling, b) 10% weak labelling, c) 5% weak labelling and d) 0% weak labelling.

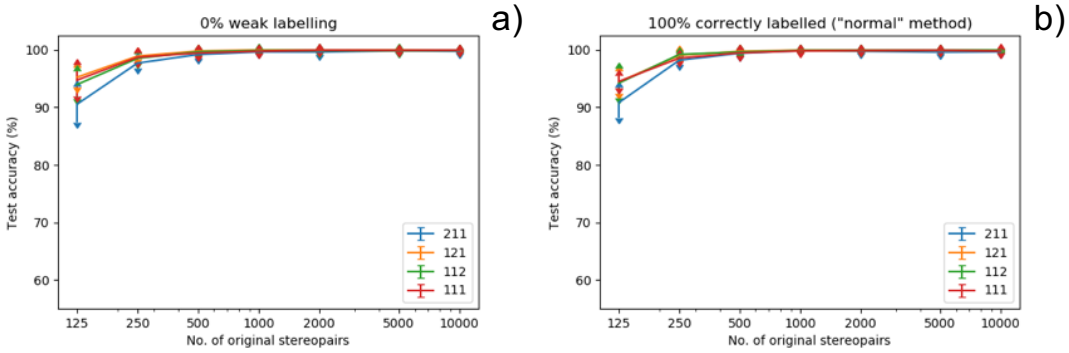


Figure 7.25: Average test accuracy of trained networks vs number of original stereopairs for a) 0% weak labelling and b) 'normal' method (100% correctly labelled data).

7.4.2 Different image conditions

Table 7.18: Test accuracies of 20% weak labelling, single augmentation training (1000 original simulated stereopairs, 10× augmentation) [10,90] with left-to-right sloped backgrounds in the second images of the original stereopairs. Stereopair reordering and left-to-right mirroring used in formation of minority handedness dataset.

Network Configuration	Average train accuracy of networks (>55%) (%)	Number of successfully trained networks (>55%)	Average majority test accuracy (>55%) (%)	Average minority test accuracy (>55%) (%)	Average test accuracy (>55%) (%)
211	100.0±0.0	10	99.6±0.8	0.8±1.9	50.2±0.8
121	100.0±0.0	10	99.4±1.3	1.2±2.1	50.3±0.7
112	100.0±0.0	10	93.4±15.2	7.2±17.2	50.3±1.3
111	100.0±0.0	10	90.6±14.3	12.4±15.5	51.5±4.2

One potential difficulty with the weak labelling method described above is that it assumes that the object of interest is the only feature that changes between the two images of the stereopair, which is far from guaranteed. A buildup of contamination, for example, will cause differences in background intensity. Beam focus and/or alignment may differ between the two, especially if the second image in the stereopair is taken a long time after the first, which may be the case if the operator chooses to image a number of particles at one stereo angle before imaging them again at the second. If this is the case and the ordering of the images is reversed in creating the minority dataset, the CNN may train on these features learn to distinguish the ordering of images rather than the chirality of the particles.

To demonstrate this, weakly labelled datasets were created containing stereopairs in which the first image had a flat background of ∼2000 counts while the second had a sloped background with ∼10000 counts on the left falling linearly to ∼2000 counts on the right, as shown in Figure 7.26. This mimics a real phenomenon that occurs when the slow scan direction of STEM is not properly synchronised with the frequency of the mains power supply and the beam hovers at the start of each row until a peak in the power supply, resulting in non-uniform beam dose and hence high contamination along one edge of the image. These stereopairs were all labelled the majority handedness, then the ordering reversed and the images mirrored along the stereo axis to create the minority dataset. The training datasets contained 1000 original stereopairs with an 80-20 mix of

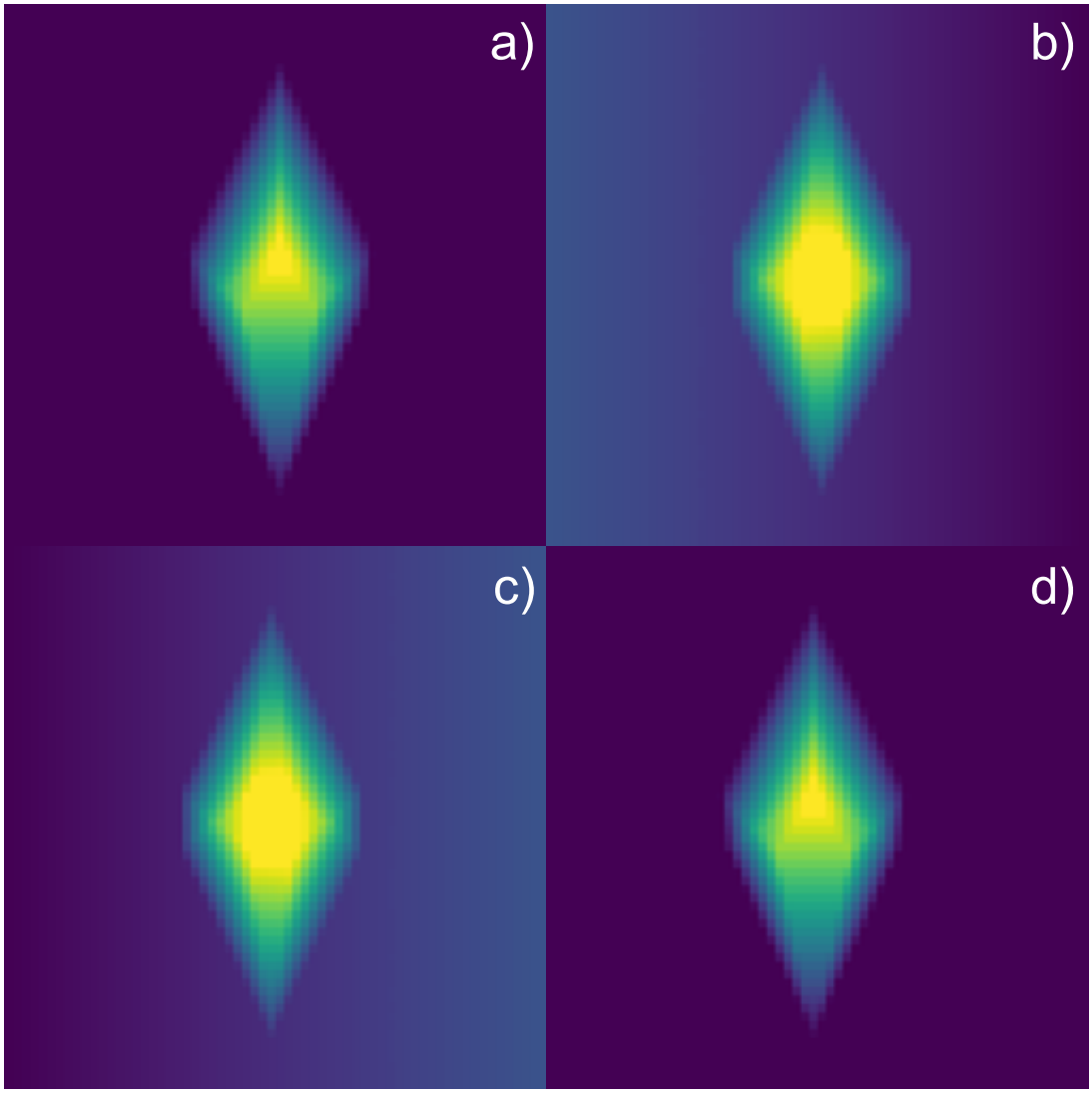


Figure 7.26: Simulated particles (height = 1.0, radius = 0.5, n = 4, chiral twist = 22.5°, stereo angle = 10°, scale = 0.8, y-rotation = 0°, x-rotation = 0°, z-rotation = 0°) with sloped background (left to right). a) and b) Original stereopair. c) and d) Original stereopair mirrored along stereo axis with ordering reversed.

enantiomers that were augmented to create datasets of 20000 stereopairs. Single augmentation and [10,90] training regime were used and the same simulation and augmentation conditions as before used here. The test data consisted of 500 correctly labelled stereopairs of each handedness, all of which had the same flat backgrounds in the first images and sloped backgrounds in the second images.

Ten networks of each type were trained in this manner, all of which achieved 100% training accuracy as opposed to the desired 80% accuracy. As shown in Table 7.18, the average test accuracy for the majority handedness dataset was ∼100% across all network types, while the average test accuracy for the minority handedness dataset was ∼0%. This is consistent with the neural network training to classify the stereopairs based on the ordering of the images (via the different backgrounds) rather than classifying them based on the chirality of the particles. There were some exceptions to this in the 112 and 111 networks trained, as evidenced by the large standard deviations on the average majority and minority test accuracies of these network types. Two 112 networks achieved majority test accuracies of <95% (86% and 54%) and these same networks achieved minority test accuracies >5% (16% and 52%, respectively). Three 111 networks majority test accuracies of <95% (68%, 70% and 72%) and these same networks achieved minority test accuracies >5% (28%, 44% and 20% respectively) with two more achieving minority test accuracies >5% as well (10% and 22%). The cause of these anomalies is unknown, but may indicate that the more complex 112 and 111 networks are slightly more resilient to this error. Nevertheless, the overall test accuracies of all of these networks are less 60%, meaning that they are still effectively useless.

This experiment was repeated under the same conditions with the minority dataset being created only by changing the ordering of stereopair (no mirroring), the results of which are shown in Table 7.19, from which all the same conclusions may be drawn. In this case, five 112 networks achieved majority test accuracies of <95% (66%, 64%, 86%, 90% and 46%), with the same five achieving minority test accuracies of >5% (30%, 34%, 16%, 9% and 56%, respectively). Four 111 networks achieved majority test accuracies of <95% (73%, 57%, 62% and 89%), with the same five achieving minority test accuracies of >5% (23%, 39%, 32% and 10%, respectively). Once again, the overall test accuracies were all below 60% for all networks.

Table 7.19: Test accuracies of 20% weak labelling, single augmentation training (1000 original simulated stereopairs, 10× augmentation) [10,90] with left-to-right sloped backgrounds in the second images of the original stereopairs. Stereopair reordering only used in formation of minority handedness dataset.

Network Configuration	Average train accuracy of networks (>55%) (%)	Number of successfully trained networks (>55%)	Average majority test accuracy (>55%) (%)	Average minority test accuracy (>55%) (%)	Average test accuracy (>55%) (%)
211	100.0±0.0	10	99.0±1.5	1.4±2.2	50.2±0.5
121	100.0±0.0	10	99.6±1.0	0.5±1.2	50.0±0.1
112	100.0±0.0	10	85.1±19.7	14.5±19.6	49.8±0.8
111	100.0±0.0	10	88.1±17.4	10.4±15.3	49.2±1.1

In this case, the left-to-right sloped background would not have caused any issue had the normal mirroring system (perpendicular to the stereo axis with ordering maintained) been used, since there would have been no difference between the backgrounds of the original and mirrored stereopairs. Had the scan rotation been changed by ±90° so that the background sloped from top to bottom [2], however, mirroring in the normal fashion would also result in a background that was distinguishable between the two stereopairs, as shown in Figure 7.27. This would be the case, for example, if the sample were tilted about the α-axis of the TEAM 0.5 stage and the same scan rotation used in the alpha tomography experiments described in Chapters 3 and 6. As has been shown, however, careful choice of the scan rotation and mirroring directions — slow scan axis parallel to the stereo axis (vertical, in this case), mirror plane perpendicular to the stereo axis (horizontal, in this case) — can easily eliminate this problem. This example highlights the importance of eliminating, both by careful monitoring and experimental design, any features in the images that the model could use to distinguish between the original stereopair and the mirrored stereopair.

[2] In reality, if the scan rotation had been changed, the entire image itself would have been rotated, not the direction of the background slope. This would also change the orientation of the stereo axis within the image, however, so, for ease and clarity of description, this will be discussed as if the background has rotated so that the direction of the stereo axis within the image (vertical) is maintained.

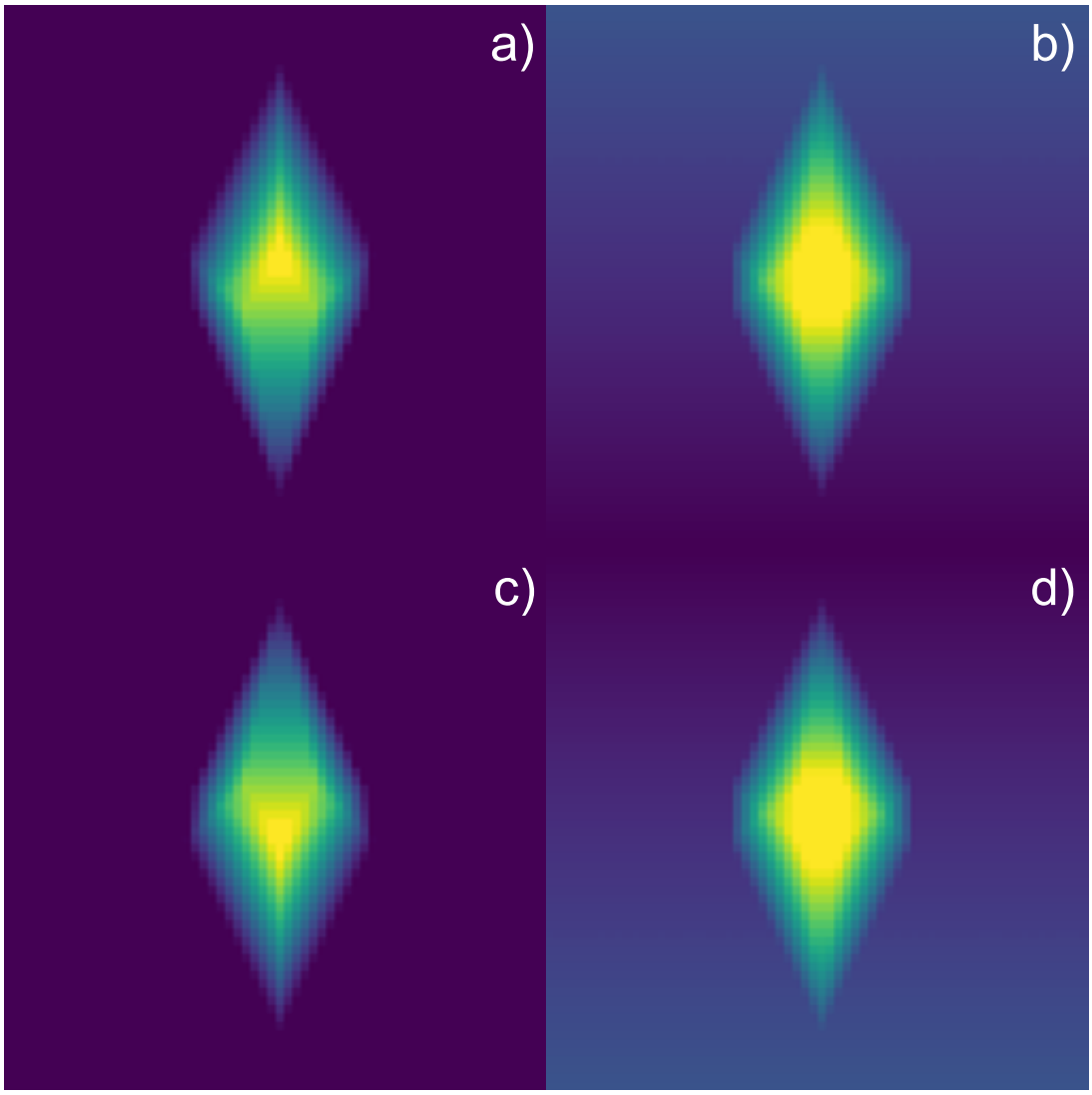


Figure 7.27: Simulated particles (height = 1.0, radius = 0.5, $n = 4$, chiral twist = 22.5°, stereo angle = 10°, scale = 0.8, y-rotation = 0°, x-rotation = 0°, z-rotation = 0°) with sloped background (top to bottom). a) and b) Original stereopair. c) and d) Original stereopair mirrored perpendicular to the stereo axis with ordering maintained.

7.5 Interpreting and using model results

The ultimate goal of this work would be to use these models to determine the *ee* of new samples. Since these models are not 100% accurate, however, careful interpretation of the results would be required. To demonstrate this, consider a sample of 100 chiral particles, 80 of which are of one

enantiomer and 20 of the other, to which a model with 95% accuracy is applied. This would mean that the model would correctly identify 76 of the first enantiomer correctly and 4 incorrectly as the second enantiomer, and 19 of the second enantiomer correctly and 1 incorrectly as the first enantiomer, giving an overall result of 77-23 as opposed to 80-20. If the sample were a 60-40 mix, the model would give a result of 59-41. Only with a 50-50 mix would the model give an accurate result of 50-50. This bias towards racemic would exists for all values of *ee*, but could easily be compensated for by knowing the test accuracy of the model. This would require testing the model with a 100% correctly labelled dataset, as has been done in this work. It would not, however, require 100% correctly labelling the training dataset, meaning that the weakly labelled approach to training would be viable. This assumes that the accuracy with which the model classifies both enantiomers is the same, which could easily be tested by using on separate test datasets for each enantiomer, as was done in Section 7.4 to prove that the weakly labelled method proposed there did not result in an inherent bias.

7.6 Summary and Outlook

The chapter has demonstrated how machine learning — specifically convolutional neural networks — can be used to determine the handedness of inorganic nanoparticles with chiral geometries from stereopairs of scanning transmission electron microscope images. The need for screening datasets according to the orientation of the particles within the image was identified, along with the improvement in classification accuracy afforded by increasing the range of orientations within the dataset more slowly as training progresses. Two augmentation schemes — 'single' and 'dynamic' — were tested and found to result in negligible differences in classification accuracy. As predicted by the literature, the complexity of the neural network was found to have a positive correlation to classification accuracy and a negative correlation to the probability of successful training. Additionally, the expected positive correlation between test accuracy and number of different original data points was quantitatively measured and it was determined that a minimum of 250 original stereopairs is required to generate models with >95% classification accuracy if there are 20000 stereopairs in the augmented dataset. Increasing the size of the augmented dataset for the same

number of original stereopairs was also found to increase the test accuracy. The relationship between the stereo angle of the stereopairs in the training dataset and the classification accuracy of the final network was assessed and found to increase from 0-$\sim$20°, then slowly fall away, with sharper dips around integer multiples of $360°/2n$. This relationship was found to be reasonably independent of chiral twist, except when the chiral twist is $\approx 360°/2n$. Additionally, it was found that the classification accuracy was negatively correlated to the level of symmetry of the chiral particle.

In addition to this work, which was performed on 100% correctly labelled data, a weakly labelled training scheme for neural networks that exploits the nature of chirality to remove the need to manually label each stereopair in the training dataset was tested. The classification accuracy was found to increase as the percentage of incorrectly labelled data decreased. The importance of maintaining constant imaging conditions between images in a stereopair if this method is employed was also demonstrated.

With regard to the specific goal of measuring enantiomeric excess, it has been shown that the results from a model with <100% classification accuracy will slightly underestimate the *ee*, which can be easily compensated for if the actual accuracy of the model is already known.

Overall, this chapter has demonstrated how stereo imaging may be used in conjunction with machine learning to produce results that cannot be obtained from single projections. It has also shown how simulation data can be used to devise and assess training workflows for machine learning without needing to label large quantities of experimental data in advance.

With these simulations indicating that the proposed method is feasible, the logical next step is to train a neural network of this type on real STEM images of TeNPs. This would likely involve the mass acquisition of images via the automation system proposed in Chapter 6 on a sample with high enantiomeric excess, which would be used in conjunction with the weakly-labelled approach here to generate a model. A handful of TeNP stereopairs have already been acquired from the TeNP sample used in Chapter 6, an example of which is shown in Figure 7.28. Should that prove successful, this tool could start to see widespread use in the quantitative determination of *ee* of nanoparticles with chiral geometry.

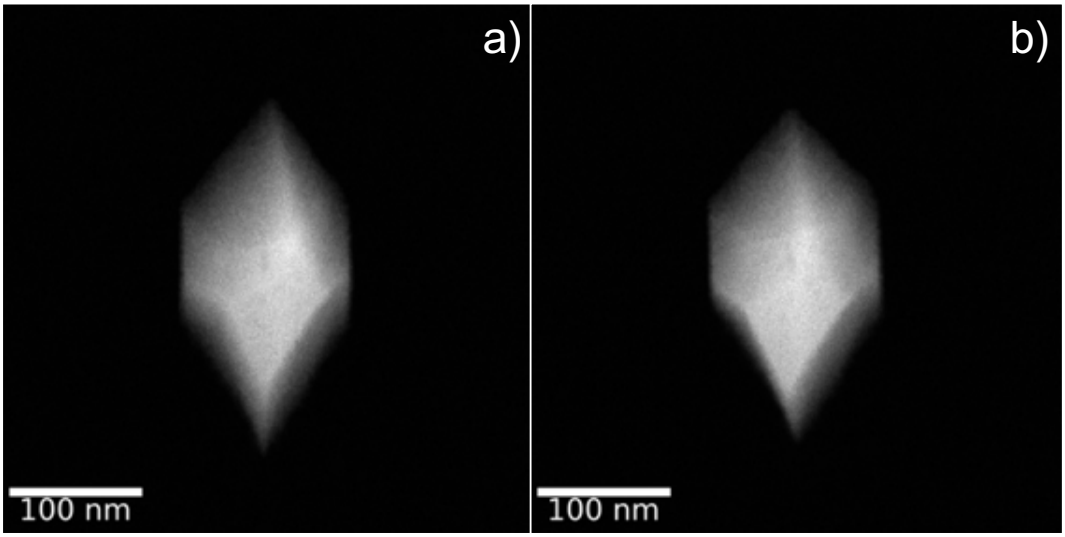


Figure 7.28: Stereopair of TeNP taken at a) $\alpha = 0°$ and b) $\alpha = 10°$. N.B. The images have been rotated by 90° from their original positions so that the stereo axis is consistent with the other examples given in this chapter (i.e vertical).

Chapter 8

Conclusion

This thesis presents various ways in which recent advances in automation and machine learning can be applied to the field of STEM. The procedures needed to create recrystallised tungsten tips for use as sample platforms in atomic-resolution electron tomography are discussed in Chapter 5, along with two experiments that coated the samples in materials that could reduce their oxidation: bismuth and carbon. It was shown that bismuth is not a viable solution as it cannot be cleanly removed, whereas carbon is a potential option, so long as it can be shown that a layer thin enough that it does not significantly reduce the signal-to-noise ratio of the sample can also prevent oxidation. This work advances the field of tungsten tip electron tomography and demonstrates how automation can be applied to the preparation of samples via the use of programmable microcontrollers.

A versatile automated STEM control system was introduced in Chapter 6 that was demonstrated to be capable of running STEM experiments without human oversight or intervention. This software was used in the high-throughput acquisition of images CdSe/CdS core-shell nanoparticles, in which high-resolution images of ∼11000 particles were taken in a 10.5-hour session, and the acquisition of CBED patterns of chiral tellurium nanoparticles, in which patterns of ∼800 particles were acquired in a 17.5-hour session. The functions and experimental pipeline for a third experiment — the search for suitable candidate particles for alpha tomography — were developed but not tested. The use of CNNs for the purposes of image analysis within the software framework

was also demonstrated. This work extends the use of automation in acquiring STEM data and demonstrates its utility in the field of materials science.

In Chapter 7, the use of CNNs for the purpose of classifying the handedness of simulated chiral particles with chiral geometry from simulated stereopairs of STEM projections was demonstrated. To the author's knowledge, this work is the first example of applying stereo imaging to machine learning, which may open the door to more three-dimensional data being used in ML. Alongside this, a 'weak labelling' approach was also demonstrated that exploits in the intrinsic nature of chirality to remove the need to manually label large quantities of data. This approach offers a simple and effective approach to training CNNs that are capable of quantitatively and precisely measuring the enantiomeric excess of samples of particles with geometric chirality, which is of potentially huge significance in fields that exploit nanomaterials such properties, such as optoelectronics.

The various processes presented in this thesis demonstrate the power of automation and machine learning and their potential to revolutionise the field of electron microscopy. Any future progress in these combined fields will be the result of dedicated computer-proficient electron microscopists willing to invest time in developing standard procedures for such work. This work will open up hitherto unimaginable avenues for scientific advancement that would otherwise remain inaccessible.

List of References

1. Ruska, E. The development of the electron microscope and of electron microscopy. *Bioscience Reports* **7,** 607–629 (1987).
2. Rojas, R., Darius, F., Goktekin, C. & Heyne, G. The reconstruction of Konrad Zuse's Z3. *IEEE Annals of the History of Computing* **27,** 23–32 (2005).
3. Müller, E. W. & Bahadur, K. Field Ionization of Gases at a Metal Surface and the Resolution of the Field Ion Microscope. *Phys. Rev.* **102,** 624–631 (3 May 1956).
4. Crewe, A. V., Wall, J. & Langmore, J. Visibility of single atoms. *Science* **168,** 1338–1340 (1970).
5. Haider, M. *et al.* Electron microscopy image enhanced. *Nature* **392,** 768–769 (Apr. 1998).
6. Krivanek, O., Dellby, N. & Lupini, A. Towards sub-Å electron beams. *Ultramicroscopy* **78,** 1–11 (1999).
7. Ercius, P., Alaidi, O., Rames, M. J. & Ren, G. Electron Tomography: A Three-Dimensional Analytic Tool for Hard and Soft Materials Research. *Advanced Materials* **27,** 5638–5663 (2015).
8. Suloway, C. *et al.* Automated molecular microscopy: the new Leginon system. *Journal of structural biology* **151,** 41–60 (2005).
9. Mastronarde, D. N. Dual-Axis Tomography: An Approach with Alignment Methods That Preserve Resolution. *Journal of structural biology* **120,** 343–352 (1997).
10. Krizhevsky, A., Sutskever, I. & Hinton, G. E. ImageNet Classification with Deep Convolutional Neural Networks. *Advances in neural information processing systems* **25,** 1097–1105 (2012).
11. Zeiler, M. D. & Fergus, R. *Visualizing and understanding convolutional networks* in *European conference on computer vision* (2014), 818–833.
12. Simonyan, K. & Zisserman, A. Very Deep Convolutional Networks for Large-Scale Image Recognition. *arXiv preprint arXiv:1409.1556* (2014).
13. Ziatdinov, M. *et al.* Deep learning of atomically resolved scanning transmission electron microscopy images: chemical identification and tracking local transformations. *ACS nano* **11,** 12742–12752 (2017).
14. Scherzer, O. The Theoretical Resolution Limit of the Electron Microscope. *Journal of Applied Physics* **20,** 20–29 (1949).
15. Rayleigh, L. On the theory of optical images, with special reference to the microscope. *Journal of the Royal Microscopical Society* **23,** 447–473 (1903).

16. Weyland, M. & Muller, D. A. Tuning the convergence angle for optimum STEM performance. *arXiv preprint arXiv:2008.12870* (2020).

17. De Broglie, L. Waves and quanta. *Nature* **112,** 540–540 (1923).

18. Davisson, C. J. & Germer, L. H. Reflection of Electrons by a Crystal of Nickel. *Proceedings of the National Academy of Sciences of the United States of America* **14,** 317 (1928).

19. Tanaka, N. *Scanning Transmission Electron Microscopy of Nanomaterials* (IMPERIAL COLLEGE PRESS, 2014).

20. Ul-Hamid, A. *A beginners' guide to scanning electron microscopy* (Springer, 2018).

21. Linck, M. *et al.* Chromatic aberration correction for atomic resolution TEM imaging from 20 to 80 kV. *Physical review letters* **117,** 076101 (2016).

22. Sunaoshi, T., Kaji, K., Orai, Y., Schamp, C. & Voelkl, E. STEM/SEM, chemical analysis, atomic resolution and surface imaging at 30 kV with no aberration correction for nanomaterials on graphene support. *Microscopy and Microanalysis* **22,** 604–605 (2016).

23. Schatten, H. *Scanning electron microscopy for the life sciences* (Cambridge University Press, 2012).

24. Korde, V. B. & Patil, N. M. Domain imaging in Fe-doped KNbO3 single crystal via trinocular microscopy and scanning electron microscopy. *Materials Chemistry and Physics* **226,** 230–234 (2019).

25. Liang, J., Wang, Z., Xie, H., Shi, H. & Li, X. In situ scanning electron microscopy analysis of effect of temperature on small fatigue crack growth behavior of nickel-based single-crystal superalloy. *International Journal of Fatigue* **128,** 105195 (2019).

26. Krumeich, F. *Characterization Techniques in Heterogeneous Catalysis: Electron Microscopy I* Accessed: 2020-09-15. 2019. `https://ethz.ch/content/dam/ethz/special-interest/chab/icb/van-bokhoven-group-dam/coursework/Characterization-Techniques/2019/EM-ScriptI.pdf`.

27. Walther, T. in *Microscopy Methods in Nanomaterials Characterization* (eds Thomas, S., Thomas, R., Zachariah, A. K. & Mishra, R. K.) 105–134 (Elsevier, 2017).

28. Klein, N. D., Hurley, K. R., Feng, Z. V. & Haynes, C. L. Dark field transmission electron microscopy as a tool for identifying inorganic nanoparticles in biological matrices. *Analytical Chemistry* **87,** 4356–4362 (2015).

29. Reimer, L. *Transmission electron microscopy: physics of image formation and microanalysis* (Springer, 2013).

30. Krajnak, M., McGrouther, D., Maneuski, D., O'Shea, V. & McVitie, S. Pixelated detectors and improved efficiency for magnetic imaging in STEM differential phase contrast. *Ultramicroscopy* **165,** 42–50 (2016).

31. Ophus, C. Four-dimensional scanning transmission electron microscopy (4D-STEM): From scanning nanodiffraction to ptychography and beyond. *Microscopy and Microanalysis* **25,** 563–582 (2019).

32. *JEOL Glossary of TEM Terms: ABF* Accessed: 2020-08-10. `https://www.jeol.co.jp/en/words/emterms/search_result.html?keyword=ABF-STEM`.

33. *JEOL Glossary of TEM Terms: HAADF* Accessed: 2020-08-10. `https://www.jeol.co.jp/en/words/emterms/search_result.html?keyword=HAADF-STEM`.

34. Gnanasekaran, K., de With, G. & Friedrich, H. Quantification and optimization of ADF-STEM image contrast for beam-sensitive materials. *Royal Society open science* **5,** 171838 (2018).

35. Hartel, P., Rose, H. & Dinges, C. Conditions and reasons for incoherent imaging in STEM. *Ultramicroscopy* **63,** 93–114 (1996).

36. Aert, S. V. *et al.* Quantitative atomic resolution mapping using high-angle annular dark field scanning transmission electron microscopy. *Ultramicroscopy* **109,** 1236–1244 (2009).

37. Scherzer, O. Über einige fehler von elektronenlinsen. *Zeitschrift für Physik* **101,** 593–603 (1936).

38. Erni, R., Rossell, M. D., Kisielowski, C. & Dahmen, U. Atomic-Resolution Imaging with a Sub-50-pm Electron Probe. *Phys. Rev. Lett.* **102,** 096101 (9 Mar. 2009).

39. Goodhew, P. J. & Humphreys, J. *Electron microscopy and analysis* (CRC Press, 2000).

40. Ade, G. & Lauer, R. A NEW METHOD FOR COMA-FREE ALIGNMENT OF TRANSMISSION ELECTRON MICROSCOPES AND DIGITAL DETERMINATION OF ABERRATION COEFFICIENTS. *Scanning Microscopy* **11,** 251–256 (1997).

41. Hosokawa, F., Sawada, H., Kondo, Y., Takayanagi, K. & Suenaga, K. Development of Cs and Cc correctors for transmission electron microscopy. *Microscopy* **62,** 23–41 (2013).

42. Egerton, R., Li, P. & Malac, M. Radiation damage in the TEM and SEM. *Micron* **35,** 399–409 (2004).

43. Egerton, R., McLeod, R., Wang, F. & Malac, M. Basic questions related to electron-induced sputtering in the TEM. *Ultramicroscopy* **110,** 991–997 (2010).

44. Egerton, R. Control of radiation damage in the TEM. *Ultramicroscopy* **127,** 100–108 (2013).

45. Asano, S., Engel, B. D. & Baumeister, W. In Situ Cryo-Electron Tomography: A Post-Reductionist Approach to Structural Biology. *Journal of Molecular Biology* **428,** 332–343 (2016).

46. Arslan, I., Tong, J. R. & Midgley, P. A. Reducing the missing wedge: High-resolution dual axis tomography of inorganic materials. *Ultramicroscopy* **106,** 994–1000 (2006).

47. Ke, X. *et al.* Three-Dimensional Analysis of Carbon Nanotube Networks in Interconnects by Electron Tomography without Missing Wedge Artifacts. *Microscopy and Microanalysis* **16,** 210–217 (2010).

48. Bals, S., Goris, B., Liz-Marzán, L. M. & Van Tendeloo, G. Three-dimensional characterization of noble-metal nanoparticles and their assemblies by electron tomography. *Angewandte Chemie International Edition* **53,** 10600–10610 (2014).

49. Tong, H. *et al.* Peptide-conjugation induced conformational changes in human IgG1 observed by optimized negative-staining and individual-particle electron tomography. *Scientific Reports* **3,** 1089 (2013).

50. Zhang, X. *et al.* 3D structural fluctuation of IgG1 antibody revealed by individual particle electron tomography. *Scientific reports* **5,** 1–14 (2015).

51. Zhang, L. & Ren, G. IPET and FETR: experimental approach for studying molecular structure dynamics by cryo-electron tomography of a single-molecule structure. *PloS one* **7,** e30249 (2012).

52. Lei, D. *et al.* Three-dimensional structural dynamics of DNA origami Bennett linkages using individual-particle electron tomography. *Nature communications* **9,** 1–8 (2018).

53. *Grid-tech catalog* Accessed: 2020-08-10. 2020. http://www.grid-tech.com/catalog.htm.

54. Poon, T. *Tungsten Tip and Atomic Site Tomography* PhD thesis (University of Birmingham, May 2019).

55. *TEAM 0.5 Website* Accessed: 2020-09-06. https://foundry.lbl.gov/instrumentation/team-0-5/.

56. Kisielowski, C., Erni, R. & Freitag, B. Object-defined Resolution Below 0.5 Å in Transmission Electron Microscopy-Recent Advances on the TEAM 0.5 Instrument. *Microscopy and Microanalysis* **14,** 78–79 (2008).

57. Ercius, P., Boese, M., Duden, T. & Dahmen, U. Operation of TEAM I in a user environment at NCEM. *Microscopy and Microanalysis* **18,** 676 (2012).

58. Scott, M. *et al.* Electron tomography at 2.4-ångström resolution. *Nature* **483,** 444–447 (2012).

59. Chen, C.-C. *et al.* Three-dimensional imaging of dislocations in a nanoparticle at atomic resolution. *Nature* **496,** 74–77 (2013).

60. Yang, Y. *et al.* Deciphering chemical order/disorder and material properties at the single-atom level. *Nature* **542,** 75–79 (2017).

61. Yuan, Y. *et al.* Direct observation of 3D atomic packing in monatomic amorphous materials. *arXiv preprint arXiv:2007.03881* (2020).

62. Zou, J., Han, Y. & So, S.-S. in *Artificial Neural Networks: Methods and Applications* 14–22 (Humana Press, Totowa, NJ, 2009).

63. Suk, H.-I. in *Deep Learning for Medical Image Analysis* (eds Zhou, S. K., Greenspan, H. & Shen, D.) 3–24 (Academic Press, 2017).

64. Stewart, M. G., Popov, V. I., Kraev, I. V., Medvedev, N. & Davies, H. A. in *The Synapse: Structure and Function* 1–20 (Elsevier, 2014).

65. Sharma, S. *Activation Functions in Neural Networks* Accessed: 2020-04-17. Feb. 2019. https://towardsdatascience.com/activation-functions-neural-networks-1cbd9f8d91d6.

66. Glorot, X., Bordes, A. & Bengio, Y. *Deep Sparse Rectifier Neural Networks* in *Proceedings of the fourteenth international conference on artificial intelligence and statistics* (2011), 315–323.

67. Gupta, D. *Activation Functions: Fundamentals Of Deep Learning* Accessed: 2020-09-15. July 2020. https://www.analyticsvidhya.com/blog/2020/01/fundamentals-deep-learning-activation-functions-when-to-use-them/.

68. S. Mohamed, I. *Detection and Tracking of Pallets using a Laser Rangefinder and Machine Learning Techniques* PhD thesis (European Master on Advanced Robotics+ (EMARO+), University of Genova, Italy, Sept. 2017).

69. Hopfield, J. J. Neural networks and physical systems with emergent collective computational abilities. *Proceedings of the national academy of sciences* **79,** 2554–2558 (1982).

70. Kohonen, T. *Self-organization and associative memory* (Springer Science & Business Media, 2012).

71. Zhu, X., Vondrick, C., Fowlkes, C. C. & Ramanan, D. Do we need more training data? *International Journal of Computer Vision* **119,** 76–92 (2016).

72. Verbraeken, J. *et al.* A Survey on Distributed Machine Learning. *ACM Computing Surveys (CSUR)* **53,** 1–33 (2020).

73. LeCun, Y. & Ranzato, M. *Deep Learning Tutorial* in *Tutorials in International Conference on Machine Learning (ICML13)* (2013).

74. Goodfellow, I., Bengio, Y. & Courville, A. *Deep Learning* (MIT Press, 2017).

75. Liu, Y., Starzyk, J. A. & Zhu, Z. Optimizing number of hidden neurons in neural networks. *EeC* **1,** 6 (2007).

76. Bengio, Y. in *Neural networks: Tricks of the trade* 437–478 (Springer, 2012).

77. Kingma, D. & Ba, J. Adam: A Method for Stochastic Optimization. *International Conference on Learning Representations* (Dec. 2014).

78. Saha, S. *A Comprehensive Guide to Convolutional Neural Networks - the ELI5 way* Accessed: 2020-04-17. Dec. 2018. `https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53`.

79. Srinivas, S. *et al.* in *Deep Learning for Medical Image Analysis* (eds Zhou, S. K., Greenspan, H. & Shen, D.) 25–52 (Academic Press, 2017).

80. *Imagenet Large Scale Visual Recognition Challenge 2012 (ILSVRC-2012)* Accessed: 2020-09-15. 2020. `http://image-net.org/challenges/LSVRC/2012/results.html`.

81. *Imagenet Large Scale Visual Recognition Challenge 2013 (ILSVRC-2013)* Accessed: 2020-09-15. 2020. `http://www.image-net.org/challenges/LSVRC/2013/results.php`.

82. Dictionary, O. E. Oxford english dictionary. *Simpson, JA & Weiner, ESC* (1989).

83. Penczek, P. A. in *Methods in enzymology* 1–33 (Elsevier, 2010).

84. Masood, S., Dawam, D., Bhat, T. R., Kumar, P. & Mufti, G. Godfrey Hounsfield and the development of computed tomography (CT). *The Journal of Urology* **179,** 311–312 (2008).

85. Bailey, D. L., Maisey, M. N., Townsend, D. W. & Valk, P. E. *Positron emission tomography* (Springer, 2005).

86. Holly, T. A. *et al.* Single photon-emission computed tomography. *Journal of nuclear cardiology* **17,** 941–973 (2010).

87. Hart, R. G. Electron Microscopy of Unstained Biological Material: The Polytropic Montage. *Science* **159,** 1464–1467 (1968).

88. De Rosier, D. & Klug, A. Reconstruction of three dimensional structures from electron micrographs. *Nature* **217,** 130–134 (1968).

89. Loos, J. *et al.* Electron tomography on micrometer-thick specimens with nanometer resolution. *Nano letters* **9,** 1704–1708 (2009).

90. Aoyama, K., Takagi, T., Hirase, A. & Miyazawa, A. STEM tomography for thick biological specimens. *Ultramicroscopy* **109,** 70–80 (2008).

91. Kawase, N., Kato, M., Nishioka, H. & Jinnai, H. Transmission electron microtomography without the "missing wedge" for quantitative structural analysis. *Ultramicroscopy* **107,** 8–15 (2007).

92. Friedrich, H., de Jongh, P. E., Verkleij, A. J. & de Jong, K. P. Electron tomography for heterogeneous catalysts and related nanostructured materials. *Chemical reviews* **109,** 1613–1629 (2009).

93. Friedrich, H. *et al.* Quantitative structural analysis of binary nanocrystal superlattices by electron tomography. *Nano letters* **9,** 2719–2724 (2009).

94. Van Aert, S., Batenburg, K. J., Rossell, M. D., Erni, R. & Van Tendeloo, G. Three-dimensional atomic imaging of crystalline nanoparticles. *Nature* **470,** 374–377 (2011).

95. Xu, R. *et al.* Three-dimensional coordinates of individual atoms in materials revealed by electron tomography. *Nature materials* **14,** 1099–1103 (2015).

96. Przybilla, T. *et al.* Transfer of Individual Micro-and Nanoparticles for High-Precision 3D Analysis Using 360°Electron Tomography. *Small Methods* **2,** 1700276 (2018).

97. Greiner, M. & Kruse, P. Recrystallization of tungsten wire for fabrication of sharp and stable nanoprobe and field-emitter tips. *Review of Scientific Instruments* **78,** 026104 (2007).

98. Snow, D. B. The recrystallization of commercially pure and doped tungsten wire drawn to high strain. *Metallurgical Transactions A* **10,** 815–821 (1979).

99. Humphreys, F. J. & Hatherly, M. *Recrystallization and related annealing phenomena* (Elsevier, 2012).

100. Rieck, G. Growth and preferred orientations of crystals in tungsten wires. *Acta Metallurgica* **6,** 360–366 (1958).

101. Whang, S. in *Nanostructured Metals and Alloys* (ed Whang, S. H.) xxi–xxxv (Woodhead Publishing, 2011).

102. Leber, S., Tavernelli, J., White, D. & Hehemann, R. Fracture modes in tungsten wire. *Journal of the Less Common Metals* **48,** 119–133 (1976).

103. Ju, B.-F., Chen, Y.-L. & Ge, Y. The art of electrochemical etching for preparing tungsten probes with controllable tip profile and characteristic parameters. *Review of Scientific Instruments* **82,** 013707 (2011).

104. Cricenti, A., Paparazzo, E., Scarselli, M., Moretto, L. & Selci, S. Preparation and characterization of tungsten tips for scanning tunneling microscopy. *Review of scientific instruments* **65,** 1558–1560 (1994).

105. Sanderson, R. T. *Bismuth* Accessed: 2020-09-15. Jan. 2020. `https://www.britannica.com/science/bismuth`.

106. Zhou, J., Yang, Y., Ercius, P. & Miao, J. Atomic electron tomography in three and four dimensions. *MRS Bulletin* **45,** 290–297 (2020).

107. Szeliski, R. *Computer vision: algorithms and applications* (Springer Science & Business Media, 2010).

108. Schorb, M., Haberbosch, I., Hagen, W. J., Schwab, Y. & Mastronarde, D. N. Software tools for automated transmission electron microscopy. *Nature methods* **16,** 471–477 (2019).

109. Koster, A., Van Den Bos, A. & Van Der Mast, K. An autofocus method for a TEM. *Ultramicroscopy* **21,** 209–222 (1987).

110. Holburn, D. & Smith, K. Topographical analysis in the SEM using an automatic focusing technique. *Journal of Microscopy* **127,** 93–103 (1982).

111. Feng, J., Somlyo, A. P., Somlyo, A. V. & Shao, Z. Automated electron tomography with scanning transmission electron microscopy. *Journal of microscopy* **228,** 406–412 (2007).

112. Beyer, J. *et al.* Exploring the connectome: Petascale volume visualization of microscopy data streams. *IEEE computer graphics and applications* **33,** 50–61 (2013).

113. Lichtman, J. W., Pfister, H. & Shavit, N. The big data challenges of connectomics. *Nature neuroscience* **17,** 1448–1454 (2014).

114. Motta, A., Schurr, M., Staffler, B. & Helmstaedter, M. Big data in nanoscale connectomics, and the greed for training labels. *Current opinion in neurobiology* **55,** 180–187 (2019).

115. Adrian, M., Dubochet, J., Lepault, J. & McDowall, A. W. Cryo-electron microscopy of viruses. *Nature* **308,** 32–36 (1984).

116. Nogales, E. The development of cryo-EM into a mainstream structural biology technique. *Nature methods* **13,** 24–27 (2016).

117. Malek, S., Mosleh, M., Dhillon, S. K. & Milow, P. in *Encyclopedia of Bioinformatics and Computational Biology* (eds Ranganathan, S., Gribskov, M., Nakai, K. & Schönbach, C.) 993–1010 (Academic Press, Oxford, 2019).

118. Ondry, J. C., Philbin, J. P., Lostica, M., Rabani, E. & Alivisatos, A. P. Resilient Pathways to Atomic Attachment of Quantum Dot Dimers and Artificial Solids from Faceted CdSe Quantum Dot Building Blocks. *ACS nano* **13,** 12322–12344 (2019).

119. Kovalenko, M. V., Bodnarchuk, M. I. & Talapin, D. V. Nanocrystal superlattices with thermally degradable hybrid inorganic- organic capping ligands. *Journal of the American Chemical Society* **132,** 15124–15126 (2010).

120. Whitham, K. *et al.* Charge transport and localization in atomically coherent quantum dot solids. *Nature materials* **15,** 557–563 (2016).

121. Walravens, W. *et al.* Chemically triggered formation of two-dimensional epitaxial quantum dot superlattices. *ACS nano* **10,** 6861–6870 (2016).

122. Sandeep, C. S. *et al.* Epitaxially connected PbSe quantum-dot films: controlled neck formation and optoelectronic properties. *ACS nano* **8,** 11499–11511 (2014).

123. Evers, W. H. *et al.* High charge mobility in two-dimensional percolative networks of PbSe quantum dots connected by atomic bonds. *Nature communications* **6,** 1–8 (2015).

124. Ghosh, J. *et al.* Solid-State Synthesis of Stable and Color Tunable Cesium Lead Halide Perovskite Nanocrystals and Mechanism of High-Performance Photodetection in Monolayer MoS2/CsPbBr3 Vertical Heterojunction. *Journal of Materials Chemistry C* (2020).

125. Laanait, N., Ziatdinov, M., He, Q. & Borisevich, A. Identifying local structural states in atomic imaging by computer vision. *Advanced structural and chemical imaging* **2,** 1–11 (2016).

126. Perez, A. J. *et al.* A workflow for the automatic segmentation of organelles in electron microscopy image stacks. *Frontiers in neuroanatomy* **8,** 126 (2014).

127. Safavian, S. R. & Landgrebe, D. A survey of decision tree classifier methodology. *IEEE transactions on systems, man, and cybernetics* **21,** 660–674 (1991).

128. Gong, Y.-J. *et al.* Distributed evolutionary algorithms and their models: A survey of the state-of-the-art. *Applied Soft Computing* **34,** 286–300 (2015).

129. Suykens, J. A. & Vandewalle, J. Least squares support vector machine classifiers. *Neural processing letters* **9,** 293–300 (1999).

130. Sorzano, C. *et al.* Automatic particle selection from electron micrographs using machine learning techniques. *Journal of structural biology* **167,** 252–260 (2009).

131. Si, D., Ji, S., Nasr, K. A. & He, J. A Machine Learning Approach for the Identification of Protein Secondary Structure Elements from Electron Cryo-Microscopy Density Maps. *Biopolymers* **97,** 698–708 (2012).

132. Arganda-Carreras, I. *et al.* Trainable Weka Segmentation: a machine learning tool for microscopy pixel classification. *Bioinformatics* **33,** 2424–2426 (2017).

133. Nunez-Iglesias, J., Kennedy, R., Parag, T., Shi, J. & Chklovskii, D. B. Machine learning of hierarchical clustering to segment 2D and 3D images. *PloS one* **8,** e71715 (2013).

134. Kreshuk, A. *et al.* Automated detection and segmentation of synaptic contacts in nearly isotropic serial electron microscopy images. *PloS one* **6,** e24899 (2011).

135. Kaynig-Fittkau, V. S. *Machine learning approaches for neuron geometry extraction and synapse detection in electron microscopy images* PhD thesis (ETH Zurich, 2011).

136. Navlakha, S., Suhan, J., Barth, A. L. & Bar-Joseph, Z. A high-throughput framework to detect synapses in electron microscopy images. *Bioinformatics* **29,** i9–i17 (2013).

137. Jurrus, E. *et al.* Detection of neuron membranes in electron microscopy images using a serial neural network architecture. *Medical image analysis* **14,** 770–783 (2010).

138. Ciresan, D., Giusti, A., Gambardella, L. M. & Schmidhuber, J. *Deep neural networks segment neuronal membranes in electron microscopy images* in *Advances in neural information processing systems* (2012), 2843–2851.

139. Quan, T. M., Hildebrand, D. G. & Jeong, W.-K. Fusionnet: A deep fully residual convolutional neural network for image segmentation in connectomics. *arXiv preprint arXiv:1612.05360* (2016).

140. Zhu, Y., Ouyang, Q. & Mao, Y. A deep convolutional neural network approach to single-particle recognition in cryo-electron microscopy. *BMC bioinformatics* **18,** 1–10 (2017).

141. Ogura, T. & Sato, C. Automatic particle pickup method using a neural network has high accuracy by applying an initial weight derived from eigenimages: a new reference free method for single-particle analysis. *Journal of structural biology* **145,** 63–75 (2004).

142. Li, R., Si, D., Zeng, T., Ji, S. & He, J. *Deep convolutional neural networks for detecting secondary structures in protein density maps from cryo-electron microscopy* in *2016 IEEE International Conference on Bioinformatics and Biomedicine (BIBM)* (2016), 41–46.

143. Ben-Moshe, A., Govorov, A. O. & Markovich, G. Enantioselective synthesis of intrinsically chiral mercury sulfide nanocrystals. *Angewandte Chemie* **125,** 1313–1317 (2013).

144. Vinegrad, E., Hananel, U., Markovich, G. & Cheshnovsky, O. Determination of handedness in a single chiral nanocrystal via circularly polarized luminescence. *ACS nano* **13,** 601–608 (2018).

145. Gübitz, G. & Schmid, M. G. Chiral separation principles in chromatographic and electromigration techniques. *Molecular biotechnology* **32,** 159–179 (2006).

146. Gautier, C. & Bürgi, T. Chiral Gold Nanoparticles. *ChemPhysChem* **10,** 483–492 (2009).

147. Berova, N., Nakanishi, K. & Woody, R. W. *Circular dichroism: principles and applications* (John Wiley & Sons, 2000).

148. Kuno, J., Kawai, T. & Nakashima, T. The effect of surface ligands on the optical activity of mercury sulfide nanoparticles. *Nanoscale* **9,** 11590–11595 (2017).

149. Soukoulis, C. M. & Wegener, M. Past achievements and future challenges in the development of three-dimensional photonic metamaterials. *Nature photonics* **5,** 523–530 (2011).

150. Kuzyk, A. *et al.* DNA-based self-assembly of chiral plasmonic nanostructures with tailored optical response. *Nature* **483,** 311–314 (2012).

151. Mark, A. G., Gibbs, J. G., Lee, T.-C. & Fischer, P. Hybrid nanocolloids with programmed three-dimensional shape and material composition. *Nature materials* **12,** 802–807 (2013).

152. Ben-Moshe, A. *et al.* Enantioselective control of lattice and shape chirality in inorganic nanostructures using chiral biomolecules. *Nature communications* **5,** 1–9 (2014).

153. Hananel, U., Ben-Moshe, A., Diamant, H. & Markovich, G. Spontaneous and directed symmetry breaking in the formation of chiral nanocrystals. *Proceedings of the National Academy of Sciences* **116,** 11159–11164 (2019).

154. Bengio, Y., Louradour, J., Collobert, R. & Weston, J. *Curriculum learning* in *Proceedings of the 26th annual international conference on machine learning* (2009), 41–48.

155. Marr, D. & Poggio, T. A computational theory of human stereo vision. *Proceedings of the Royal Society of London. Series B. Biological Sciences* **204,** 301–328 (1979).

156. Zhou, Z.-H. A brief introduction to weakly supervised learning. *National Science Review* **5,** 44–53 (2018).

157. Chang, H.-C. & Wang, L.-C. A simple proof of Thue's Theorem on circle packing. *arXiv preprint arXiv:1009.4322* (2010).

Appendices

A GUI Overview Panels 4-7

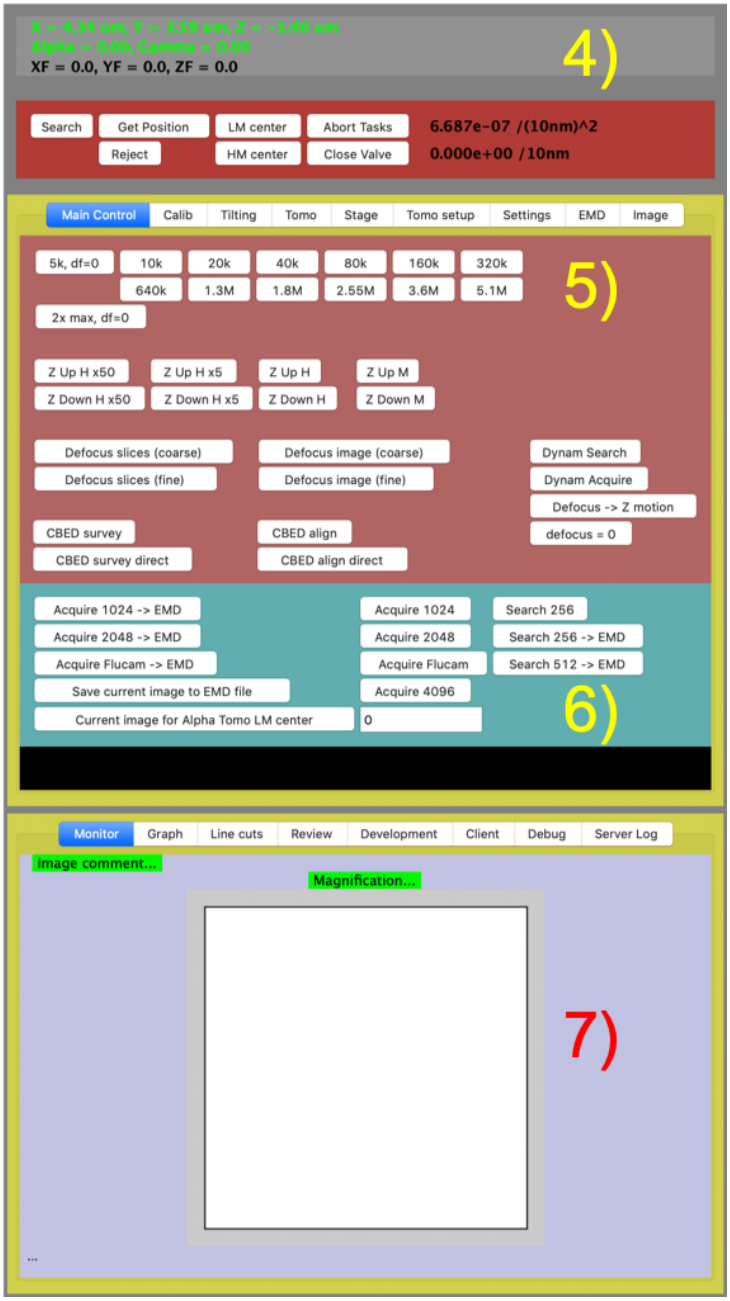


Figure 8.1: Panels 4-7 in Figure 3.3. 4) Stage position bar. 5) 'Main control' panel with magnification, z stepper movement, defocus slices (see Section 3.2.1) and CBED acquisition controls. 6) Image save controls. 7) Monitor window.

B Backpropagation

The formula for backpropagation is:

$$l = f(o(a(\theta))) \tag{8.1}$$

$$\frac{dl}{d(\theta)} = \frac{\partial l}{\partial o}\frac{\partial o}{\partial a}\frac{\partial a}{\partial(\theta)} \tag{8.2}$$

where l is the loss, o is the output from the node in the previous layer, a is the activation function of the node in the previous layer, θ is a trainable parameter (a weight or bias) [73].

C Hexagonal packing derivation

The packing fraction η for hexagonal packed circles on a 2D plane is [157]:

$$\eta = \frac{\pi}{\sqrt{12}} = 0.90690 \tag{8.3}$$

The area of the hexagon A_h associated with each circle is therefore:

$$A_h = \frac{\pi r^2}{\eta} = \pi r^2 / \frac{\pi}{\sqrt{12}} = \sqrt{12} r^2 \tag{8.4}$$

where r is the radius of a circle or half the distance between two adjacent circle centres. The number of circles n in a given area A is therefore:

$$n = \frac{A}{A_h} = \frac{A}{\sqrt{12} r^2} \tag{8.5}$$

and the number of particles in a given area A.

D 'Realistic' Experiments

D.1 Tables for dynamic augmentation [10,90]

D.1.1 125×160

Table 8.1: Results of initial dynamic augmentation training (125 original simulated stereopairs, 160× augmentation)

Network configuration	No. of networks achieving 95% validation accuracy (/10)	Average no. of epochs to achieve 95% validation accuracy
211	0	N/A
121	0	N/A
112	1	10.0±N/A
111	0	N/A

Table 8.2: Results of dynamic augmentation retraining (125 original simulated stereopairs, 160× augmentation)

Network configuration	No. of networks achieving 95% validation accuracy (/10)	Average no. of epochs to achieve 95% validation accuracy
211	0	N/A
121	0	N/A
112	0	N/A
111	0	N/A

Table 8.3: Test accuracies of dynamic augmentation training (125 original simulated stereopairs, 160× augmentation).

Network Configuration	No. of networks with > 55% test accuracy (/10)	Average test accuracy of trained networks (%)
211	10	83.7±2.5
121	10	87.1±1.5
112	10	86.4±1.6
111	9	85.2±4.2

D.1.2 250×80

Table 8.4: Results of initial dynamic augmentation training (250 original simulated stereopairs, 80× augmentation)

Network configuration	No. of networks achieving 95% validation accuracy (/10)	Average no. of epochs to achieve 95% validation accuracy
211	3	17.0±8.9
121	8	16.8±11.7
112	7	11.9±10.4
111	5	6.4±9.3

Table 8.5: Results of dynamic augmentation retraining (250 original simulated stereopairs, 80× augmentation)

Network configuration	No. of networks achieving 95% validation accuracy (/10)	Average no. of epochs to achieve 95% validation accuracy
211	1	45.0±N/A
121	5	26.6±11.4
112	2	26.0±7.1
111	6	27.0±14.6

Table 8.6: Test accuracies of dynamic augmentation training (250 original simulated stereopairs, 80× augmentation).

Network Configuration	No. of networks with $>$ 55% test accuracy (/10)	Average test accuracy of trained networks (%)
211	10	92.0±1.3
121	10	93.7±1.4
112	10	93.6±1.0
111	9	94.1±1.1

D.1.3 500×40

Table 8.7: Results of initial dynamic augmentation training (500 original simulated stereopairs, 40× augmentation)

Network configuration	No. of networks achieving 95% validation accuracy (/10)	Average no. of epochs to achieve 95% validation accuracy
211	10	9.5±9.8
121	10	3.9±2.3
112	10	2.7±1.6
111	10	2.5±0.7

Table 8.8: Results of dynamic augmentation retraining (500 original simulated stereopairs, 40× augmentation)

Network configuration	No. of networks achieving 95% validation accuracy (/10)	Average no. of epochs to achieve 95% validation accuracy
211	10	23.9±5.5
121	10	10.3±1.5
112	10	10.5±3.2
111	10	10.0±2.8

Table 8.9: Test accuracies of dynamic augmentation training (500 original simulated stereopairs, 40× augmentation).

Network Configuration	No. of networks with $>$ 55% test accuracy (/10)	Average test accuracy of trained networks (%)
211	10	95.7±1.0
121	10	97.5±1.0
112	10	97.1±0.8
111	10	97.4±0.4

D.1.4 1000×20

Table 8.10: Results of initial dynamic augmentation training (1000 original simulated stereopairs, 20× augmentation)

Network configuration	No. of networks achieving 95% validation accuracy (/10)	Average no. of epochs to achieve 95% validation accuracy
211	10	5.3±1.5
121	10	2.6±0.7
112	10	2.4±0.7
111	10	2.5±0.5

Table 8.11: Results of dynamic augmentation retraining (1000 original simulated stereopairs, 20× augmentation)

Network configuration	No. of networks achieving 95% validation accuracy (/10)	Average no. of epochs to achieve 95% validation accuracy
211	10	17.2±4.1
121	10	9.1±2.2
112	10	8.4±0.8
111	9	7.1±2.6

Table 8.12: Test accuracies of dynamic augmentation training (1000 original simulated stereopairs, 20× augmentation).

Network Configuration	No. of networks with $>$ 55% test accuracy (/10)	Average test accuracy of trained networks (%)
211	10	97.4±0.9
121	10	98.5±0.5
112	10	98.2±0.5
111	9	98.4±0.3

D.1.5 20000×1

Table 8.13: Results of initial dynamic augmentation training (20000 original simulated stereopairs, 1× augmentation)

Network configuration	No. of networks achieving 95% validation accuracy (/10)	Average no. of epochs to achieve 95% validation accuracy
211	10	9.7±3.8
121	10	8.0±5.9
112	6	11.3±12.9
111	6	11.2±8.4

Table 8.14: Results of dynamic augmentation retraining (20000 original simulated stereopairs, 10× augmentation)

Network configuration	No. of networks achieving 95% validation accuracy (/10)	Average no. of epochs to achieve 95% validation accuracy
211	10	16.1±3.4
121	10	9.7±2.8
112	6	13.0±10.0
111	6	11.3±3.8

Table 8.15: Test accuracies of dynamic augmentation training (20000 original simulated stereopairs, 1× augmentation).

Network Configuration	No. of networks with > 55% test accuracy (/10)	Average test accuracy of trained networks (%)
211	10	97.2±1.0
121	10	98.9±0.4
112	6	98.1±2.1
111	6	98.2±0.6

D.1.6 250×20

Table 8.16: Results of initial dynamic augmentation training (250 original simulated stereopairs, 20× augmentation)

Network configuration	No. of networks achieving 95% validation accuracy (/10)	Average no. of epochs to achieve 95% validation accuracy
211	0	N/A±N/A
121	3	17.3±11.9
112	6	27.8±11.8
111	5	24.4±16.5

Table 8.17: Results of dynamic augmentation retraining (250 original simulated stereopairs, 20× augmentation)

Network configuration	No. of networks achieving 95% validation accuracy (/10)	Average no. of epochs to achieve 95% validation accuracy
211	0	N/A±N/A
121	0	N/A±N/A
112	0	N/A±N/A
111	2	35.0±4.2

Table 8.18: Test accuracies of dynamic augmentation training (250 original simulated stereopairs, 20× augmentation).

Network Configuration	No. of networks with > 55% test accuracy (/10)	Average test accuracy of trained networks (%)
211	10	85.2±2.1
121	10	90.6±1.0
112	10	90.6±1.2
111	6	93.1±0.9

D.1.7 250×40

Table 8.19: Results of initial dynamic augmentation training (250 original simulated stereopairs, 40× augmentation)

Network configuration	No. of networks achieving 95% validation accuracy (/10)	Average no. of epochs to achieve 95% validation accuracy
211	1	11.0±N/A
121	3	18.3±21.5
112	6	15.3±13.6
111	8	8.8±9.5

Table 8.20: Results of dynamic augmentation retraining (250 original simulated stereopairs, 40× augmentation)

Network configuration	No. of networks achieving 95% validation accuracy (/10)	Average no. of epochs to achieve 95% validation accuracy
211	0	N/A±N/A
121	2	37.5±10.6
112	3	39.0±3.6
111	4	33.2±15.8

Table 8.21: Test accuracies of dynamic augmentation training (250 original simulated stereopairs, 40× augmentation).

Network Configuration	No. of networks with $>$ 55% test accuracy (/10)	Average test accuracy of trained networks (%)
211	10	89.3±1.5
121	10	93.0±1.3
112	10	92.1±1.3
111	10	94.0±1.2

D.1.8 250×160

Table 8.22: Results of initial dynamic augmentation training (250 original simulated stereopairs, 160× augmentation)

Network configuration	No. of networks achieving 95% validation accuracy (/10)	Average no. of epochs to achieve 95% validation accuracy
211	2	13.5±6.4
121	6	17.3±14.3
112	7	4.9±2.4
111	7	6.0±4.5

Table 8.23: Results of dynamic augmentation retraining (250 original simulated stereopairs, 160× augmentation)

Network configuration	No. of networks achieving 95% validation accuracy (/10)	Average no. of epochs to achieve 95% validation accuracy
211	4	26.0±15.1
121	7	23.7±12.9
112	9	26.3±10.2
111	9	18.1±10.3

Table 8.24: Test accuracies of dynamic augmentation training (250 original simulated stereopairs, 160× augmentation).

Network Configuration	No. of networks with $> 55\%$ test accuracy (/10)	Average test accuracy of trained networks (%)
211	10	93.7±1.1
121	10	94.6±0.5
112	10	94.5±0.8
111	10	95.2±1.0

D.1.9 250×320

Table 8.25: Results of initial dynamic augmentation training (250 original simulated stereopairs, 320× augmentation)

Network configuration	No. of networks achieving 95% validation accuracy (/10)	Average no. of epochs to achieve 95% validation accuracy
211	4	25.8±19.0
121	7	3.6±1.7
112	9	6.3±6.2
111	7	1.6±1.0

Table 8.26: Results of dynamic augmentation retraining (250 original simulated stereopairs, 320× augmentation)

Network configuration	No. of networks achieving 95% validation accuracy (/10)	Average no. of epochs to achieve 95% validation accuracy
211	7	22.0±11.6
121	10	12.5±5.3
112	10	11.6±5.5
111	6	15.3±7.6

Table 8.27: Test accuracies of dynamic augmentation training (250 original simulated stereopairs, 320× augmentation).

Network Configuration	No. of networks with $>$ 55% test accuracy (/10)	Average test accuracy of trained networks (%)
211	10	94.6±1.2
121	10	95.6±0.9
112	10	95.5±0.8
111	10	94.5±0.6

D.2 Tables for dynamic augmentation [10:10:90]

D.2.1 125×160

Table 8.28: Test accuracies of dynamic augmentation training (125 original simulated stereopairs, 160× augmentation) [10:10:90]

Network Configuration	No. of networks with > 55% test accuracy (/10)	Average test accuracy of trained networks (%)
211	10	86.6±2.6
121	10	88.5±1.7
112	10	87.8±2.3
111	10	87.1±2.2

D.2.2 250×80

Table 8.29: Test accuracies of dynamic augmentation training (250 original simulated stereopairs, 80× augmentation) [10:10:90]

Network Configuration	No. of networks with > 55% test accuracy (/10)	Average test accuracy of trained networks (%)
211	10	94.7±1.0
121	10	95.1±1.6
112	10	95.6±1.4
111	10	94.6±0.9

D.2.3 500×40

Table 8.30: Test accuracies of dynamic augmentation training (500 original simulated stereopairs, 40× augmentation) [10:10:90]

Network Configuration	No. of networks with > 55% test accuracy (/10)	Average test accuracy of trained networks (%)
211	10	97.3±1.1
121	10	97.7±0.6
112	10	97.5±0.6
111	10	97.9±0.7

D.2.4 1000×20

Table 8.31: Results of dynamic augmentation training (1000 original simulated stereopairs, 20× augmentation) [10:10:90]

Network Configuration	No. of networks with > 55% test accuracy (/10)	Average test accuracy of trained networks (%)
211	10	98.4±0.5
121	10	98.7±0.7
112	10	98.7±0.4
111	10	98.6±0.5

D.2.5 20000×1

Table 8.32: Results of dynamic augmentation training (20000 original simulated stereopairs, 1× augmentation) [10:10:90]

Network Configuration	No. of networks with > 55% test accuracy (/10)	Average test accuracy of trained networks (%)
211	10	98.8±0.3
121	10	99.3±0.2
112	4	99.5±0.3
111	5	98.8±1.0

D.3 Tables for single augmentation [10,90]

D.3.1 125×160

Table 8.33: Results of initial single augmentation training (125 original simulated stereopairs, 160× augmentation)

Network configuration	No. of networks achieving 95% validation accuracy (/10)	Average no. of epochs to achieve 95% validation accuracy
211	0	N/A
121	0	N/A
112	2	37.5±3.5
111	2	35.0±2.8

Table 8.34: Results of single augmentation retraining (125 original simulated stereopairs, 160× augmentation)

Network configuration	No. of networks achieving 95% validation accuracy (/10)	Average no. of epochs to achieve 95% validation accuracy
211	0	N/A
121	0	N/A
112	0	N/A
111	0	N/A

Table 8.35: Test accuracies of single augmentation training (125 original simulated stereopairs, 160× augmentation).

Network Configuration	No. of networks with $>$ 55% test accuracy (/10)	Average test accuracy of trained networks (%)
211	10	83.9±1.8
121	10	86.6±2.1
112	10	86.1±2.6
111	10	88.2±2.1

D.3.2 250×80

Table 8.36: Results of initial single augmentation training (250 original simulated stereopairs, 80× augmentation)

Network configuration	No. of networks achieving 95% validation accuracy (/10)	Average no. of epochs to achieve 95% validation accuracy
211	2	36.0±11.3
121	6	22.8±8.2
112	9	22.6±11.6
111	6	18.0±6.0

Table 8.37: Results of single augmentation retraining (250 original simulated stereopairs, 80× augmentation)

Network configuration	No. of networks achieving 95% validation accuracy (/10)	Average no. of epochs to achieve 95% validation accuracy
211	0	N/A
121	5	23.6±7.4
112	5	28.4±9.6
111	8	12.4±6.9

Table 8.38: Test accuracies of single augmentation training (250 original simulated stereopairs, 80× augmentation).

Network Configuration	No. of networks with > 55% test accuracy (/10)	Average test accuracy of trained networks (%)
211	10	92.3±1.8
121	10	94.0±1.6
112	10	94.8±1.0
111	10	95.4±0.9

D.3.3 500×40

Table 8.39: Results of initial single augmentation training (500 original simulated stereopairs, 40× augmentation)

Network configuration	No. of networks achieving 95% validation accuracy (/10)	Average no. of epochs to achieve 95% validation accuracy
211	6	29.5±7.0
121	9	19.1±3.5
112	10	14.8±3.9
111	10	18.5±9.1

Table 8.40: Results of single augmentation retraining (500 original simulated stereopairs, 40× augmentation)

Network configuration	No. of networks achieving 95% validation accuracy (/10)	Average no. of epochs to achieve 95% validation accuracy
211	10	29.1±7.7
121	10	9.7±3.3
112	10	8.6±4.2
111	10	6.9±1.0

Table 8.41: Test accuracies of single augmentation training (500 original simulated stereopairs, 40× augmentation).

Network Configuration	No. of networks with > 55% test accuracy (/10)	Average test accuracy of trained networks (%)
211	10	96.3±0.8
121	10	98.0±0.6
112	10	98.1±0.7
111	10	97.9±0.7

D.3.4 1000×20

Table 8.42: Results of initial single augmentation training (1000 original simulated stereopairs, 20× augmentation)

Network configuration	No. of networks achieving 95% validation accuracy (/10)	Average no. of epochs to achieve 95% validation accuracy
211	10	33.8±8.7
121	10	21.8±5.8
112	10	19.9±4.2
111	10	22.2±7.2

Table 8.43: Results of single augmentation retraining (1000 original simulated stereopairs, 20× augmentation)

Network configuration	No. of networks achieving 95% validation accuracy (/10)	Average no. of epochs to achieve 95% validation accuracy
211	10	17.6±2.6
121	10	6.5±0.8
112	10	6.1±1.2
111	10	5.8±1.0

Table 8.44: Test accuracies of single augmentation training (1000 original simulated stereopairs, 20× augmentation).

Network Configuration	No. of networks with > 55% test accuracy (/10)	Average test accuracy of trained networks (%)
211	10	98.0±0.7
121	10	98.7±0.3
112	10	98.8±0.4
111	10	98.7±0.6

D.3.5 20000×1

Table 8.45: Results of initial single augmentation training (20000 original simulated stereopairs, 1× augmentation)

Network configuration	No. of networks achieving 95% validation accuracy (/10)	Average no. of epochs to achieve 95% validation accuracy
211	8	35.8±7.3
121	10	25.6±9.8
112	7	30.9±11.0
111	5	35.4±9.4

Table 8.46: Results of single augmentation retraining (20000 original simulated stereopairs, 1× augmentation)

Network configuration	No. of networks achieving 95% validation accuracy (/10)	Average no. of epochs to achieve 95% validation accuracy
211	10	19.1±4.2
121	10	6.5±1.2
112	9	8.6±4.2
111	8	8.5±5.8

Table 8.47: Test accuracies of single augmentation training (20000 original simulated stereopairs, 1× augmentation).

Network Configuration	No. of networks with > 55% test accuracy (/10)	Average test accuracy of trained networks (%)
211	10	98.3±0.8
121	10	99.4±0.3
112	9	99.3±0.3
111	8	99.1±0.5

D.4 Tables for single augmentation [10:10:90]

D.4.1 125×160

Table 8.48: Test accuracies of single augmentation training (125 original simulated stereopairs, 160× augmentation) [10:10:90]

Network Configuration	No. of networks with > 55% test accuracy (/10)	Average test accuracy of trained networks (%)
211	10	87.6±2.7
121	10	89.8±1.9
112	10	88.3±1.7
111	10	89.8±2.3

D.4.2 250×80

Table 8.49: Test accuracies of single augmentation training (250 original simulated stereopairs, 80× augmentation) [10:10:90]

Network Configuration	No. of networks with > 55% test accuracy (/10)	Average test accuracy of trained networks (%)
211	10	94.4±1.4
121	10	95.8±1.1
112	10	95.3±1.2
111	10	95.8±1.0

D.4.3 500×40

Table 8.50: Test accuracies of single augmentation training (500 original simulated stereopairs, 40× augmentation) [10:10:90]

Network Configuration	No. of networks with > 55% test accuracy (/10)	Average test accuracy of trained networks (%)
211	10	97.5±1.2
121	10	97.9±0.6
112	10	98.1±0.4
111	10	98.2±0.7

D.4.4 1000×20

Table 8.51: Test accuracies of single augmentation training (1000 original simulated stereopairs, 20× augmentation) [10:10:90]

Network Configuration	No. of networks with > 55% test accuracy (/10)	Average test accuracy of trained networks (%)
211	10	98.8±0.5
121	10	99.0±0.4
112	10	99.0±0.4
111	10	99.1±0.4

D.4.5 20000×1

Table 8.52: Test accuracies of single augmentation training (20000 original simulated stereopairs, 1× augmentation) [10:10:90]

Network Configuration	No. of networks with > 55% test accuracy (/10)	Average test accuracy of trained networks (%)
211	10	99.1±0.4
121	10	99.6±0.3
112	10	99.5±0.2
111	9	99.5±0.3

E Varying stereo angles

E.1 Tables for $n = 3$

E.1.1 Chiral twist = 15.0°

E.1.1.1 $n = 3$, Chiral twist = 15.0°, Stereo angle = 5°

Table 8.53: Test data results for $n = 3$, chiral twist = 15.0°, stereo angle = 5° (dynamic augmentation, [10,90], 250 original simulated stereopairs, 80× augmentation

Network Configuration	No. of networks with > 55% test accuracy (/10)	Average test accuracy of trained networks (%)
211	10	83.0±2.2
121	10	81.7±1.6
112	10	77.9±1.7
111	10	75.0±1.7

E.1.1.2 $n = 3$, Chiral twist = 15.0°, Stereo angle = 10°

Table 8.54: Test data results for $n = 3$, chiral twist = 15.0°, stereo angle = 10° (dynamic augmentation, [10,90], 250 original simulated stereopairs, 80× augmentation

Network Configuration	No. of networks with > 55% test accuracy (/10)	Average test accuracy of trained networks (%)
211	10	91.1±1.3
121	10	90.1±1.6
112	10	88.9±1.7
111	10	89.1±0.8

E.1.1.3 $n = 3$, Chiral twist = 15.0°, Stereo angle = 15°

Table 8.55: Test data results for $n = 3$, chiral twist = 15.0°, stereo angle = 15° (dynamic augmentation, [10,90], 250 original simulated stereopairs, 80× augmentation

Network Configuration	No. of networks with > 55% test accuracy (/10)	Average test accuracy of trained networks (%)
211	10	93.6±1.4
121	10	93.2±0.8
112	10	91.5±1.3
111	10	93.6±0.9

E.1.1.4 $n = 3$, Chiral twist = 15.0°, Stereo angle = 20°

Table 8.56: Test data results for $n = 3$, chiral twist = 15.0°, stereo angle = 20° (dynamic augmentation, [10,90], 250 original simulated stereopairs, 80× augmentation

Network Configuration	No. of networks with > 55% test accuracy (/10)	Average test accuracy of trained networks (%)
211	10	94.4±1.1
121	10	93.7±1.1
112	10	94.2±1.7
111	10	94.5±1.1

E.1.1.5 $n = 3$, Chiral twist = 15.0°, Stereo angle = 25°

Table 8.57: Test data results for $n = 3$, chiral twist = 15.0°, stereo angle = 25° (dynamic augmentation, [10,90], 250 original simulated stereopairs, 80× augmentation

Network Configuration	No. of networks with > 55% test accuracy (/10)	Average test accuracy of trained networks (%)
211	10	95.3±1.0
121	10	94.9±1.2
112	10	95.3±0.8
111	10	96.2±0.8

E.1.1.6 $n = 3$, Chiral twist = 15.0°, Stereo angle = 30°

Table 8.58: Test data results for $n = 3$, chiral twist = 15.0°, stereo angle = 30° (dynamic augmentation, [10,90], 250 original simulated stereopairs, 80× augmentation

Network Configuration	No. of networks with > 55% test accuracy (/10)	Average test accuracy of trained networks (%)
211	10	95.7±0.7
121	10	95.1±1.0
112	10	96.1±0.8
111	10	96.8±0.8

E.1.1.7 $n = 3$, Chiral twist = 15.0°, Stereo angle = 35°

Table 8.59: Test data results for $n = 3$, chiral twist = 15.0°, stereo angle = 35° (dynamic augmentation, [10,90], 250 original simulated stereopairs, 80× augmentation

Network Configuration	No. of networks with > 55% test accuracy (/10)	Average test accuracy of trained networks (%)
211	10	94.7±0.8
121	10	94.8±1.0
112	10	96.0±1.1
111	10	96.8±0.8

E.1.1.8 $n = 3$, Chiral twist = 15.0°, Stereo angle = 40°

Table 8.60: Test data results for $n = 3$, chiral twist = 15.0°, stereo angle = 40° (dynamic augmentation, [10,90], 250 original simulated stereopairs, 80× augmentation

Network Configuration	No. of networks with > 55% test accuracy (/10)	Average test accuracy of trained networks (%)
211	10	94.3±0.7
121	10	93.9±1.4
112	10	94.7±1.0
111	10	96.6±0.6

E.1.1.9 $n = 3$, Chiral twist = 15.0°, Stereo angle = 45°

Table 8.61: Test data results for $n = 3$, chiral twist = 15.0°, stereo angle = 45° (dynamic augmentation, [10,90], 250 original simulated stereopairs, 80× augmentation

Network Configuration	No. of networks with > 55% test accuracy (/10)	Average test accuracy of trained networks (%)
211	10	92.1±1.0
121	10	93.1±1.8
112	10	94.5±0.9
111	9	95.1±0.6

E.1.2 Chiral twist = 22.5°

E.1.2.1 $n = 3$, Chiral twist = 22.5°, Stereo angle = 5°

Table 8.62: Test data results for $n = 3$, chiral twist = 22.5°, stereo angle = 5° (dynamic augmentation, [10,90], 250 original simulated stereopairs, 80× augmentation

Network Configuration	No. of networks with > 55% test accuracy (/10)	Average test accuracy of trained networks (%)
211	10	89.1±1.4
121	10	89.0±1.9
112	10	86.6±2.0
111	10	77.3±3.6

E.1.2.2 $n = 3$, Chiral twist = 22.5°, Stereo angle = 10°

Table 8.63: Test data results for $n = 3$, chiral twist = 22.5°, stereo angle = 10° (dynamic augmentation, [10,90], 250 original simulated stereopairs, 80× augmentation

Network Configuration	No. of networks with > 55% test accuracy (/10)	Average test accuracy of trained networks (%)
211	10	95.2±1.0
121	10	95.4±1.0
112	10	93.6±1.4
111	10	92.7±1.1

E.1.2.3 $n = 3$, Chiral twist = 22.5°, Stereo angle = 15°

Table 8.64: Test data results for $n = 3$, chiral twist = 22.5°, stereo angle = 15° (dynamic augmentation, [10,90], 250 original simulated stereopairs, 80× augmentation

Network Configuration	No. of networks with > 55% test accuracy (/10)	Average test accuracy of trained networks (%)
211	10	96.3±1.1
121	10	96.3±0.9
112	10	96.2±1.0
111	10	96.8±0.8

E.1.2.4 $n = 3$, Chiral twist = 22.5°, Stereo angle = 20°

Table 8.65: Test data results for $n = 3$, chiral twist = 22.5°, stereo angle = 20° (dynamic augmentation, [10,90], 250 original simulated stereopairs, 80× augmentation

Network Configuration	No. of networks with > 55% test accuracy (/10)	Average test accuracy of trained networks (%)
211	10	97.1±0.7
121	10	97.7±0.6
112	10	96.7±1.3
111	10	98.1±0.5

E.1.2.5 $n = 3$, Chiral twist = 22.5°, Stereo angle = 25°

Table 8.66: Test data results for $n = 3$, chiral twist = 22.5°, stereo angle = 25° (dynamic augmentation, [10,90], 250 original simulated stereopairs, 80× augmentation

Network Configuration	No. of networks with > 55% test accuracy (/10)	Average test accuracy of trained networks (%)
211	10	97.9±0.5
121	10	98.0±0.4
112	10	97.7±0.7
111	10	98.1±0.7

E.1.2.6 $n = 3$, Chiral twist = 22.5°, Stereo angle = 30°

Table 8.67: Test data results for $n = 3$, chiral twist = 22.5°, stereo angle = 30° (dynamic augmentation, [10,90], 250 original simulated stereopairs, 80× augmentation

Network Configuration	No. of networks with > 55% test accuracy (/10)	Average test accuracy of trained networks (%)
211	10	97.3±1.1
121	10	97.6±0.7
112	10	98.2±0.6
111	10	98.6±0.8

E.1.2.7 $n = 3$, Chiral twist = 22.5°, Stereo angle = 35°

Table 8.68: Test data results for $n = 3$, chiral twist = 22.5°, stereo angle = 35° (dynamic augmentation, [10,90], 250 original simulated stereopairs, 80× augmentation

Network Configuration	No. of networks with > 55% test accuracy (/10)	Average test accuracy of trained networks (%)
211	10	97.6±0.9
121	10	97.5±0.9
112	10	97.9±0.6
111	10	98.1±0.4

E.1.2.8 $n = 3$, Chiral twist = 22.5°, Stereo angle = 40°

Table 8.69: Test data results for $n = 3$, chiral twist = 22.5°, stereo angle = 40° (dynamic augmentation, [10,90], 250 original simulated stereopairs, 80× augmentation

Network Configuration	No. of networks with > 55% test accuracy (/10)	Average test accuracy of trained networks (%)
211	10	96.5±0.9
121	10	97.2±0.6
112	10	97.7±0.4
111	10	97.7±0.9

E.1.2.9 $n = 3$, Chiral twist = 22.5°, Stereo angle = 45°

Table 8.70: Test data results for $n = 3$, chiral twist = 22.5°, stereo angle = 45° (dynamic augmentation, [10,90], 250 original simulated stereopairs, 80× augmentation

Network Configuration	No. of networks with > 55% test accuracy (/10)	Average test accuracy of trained networks (%)
211	10	97.0±0.7
121	10	96.4±1.0
112	10	97.0±0.9
111	10	97.5±0.7

E.1.3 Chiral twist = 30.0°

E.1.3.1 $n = 3$, Chiral twist = 30.0°, Stereo angle = 5°

Table 8.71: Test data results for $n = 3$, chiral twist = 30.0°, stereo angle = 5° (dynamic augmentation, [10,90], 250 original simulated stereopairs, 80× augmentation

Network Configuration	No. of networks with > 55% test accuracy (/10)	Average test accuracy of trained networks (%)
211	10	90.5±2.2
121	10	91.6±1.0
112	10	88.6±1.3
111	10	78.8±2.6

E.1.3.2 $n = 3$, Chiral twist = 30.0°, Stereo angle = 10°

Table 8.72: Test data results for $n = 3$, chiral twist = 30.0°, stereo angle = 10° (dynamic augmentation, [10,90], 250 original simulated stereopairs, 80× augmentation

Network Configuration	No. of networks with > 55% test accuracy (/10)	Average test accuracy of trained networks (%)
211	10	96.2±1.3
121	10	96.7±0.7
112	10	95.3±1.2
111	10	94.5±1.1

E.1.3.3 $n = 3$, Chiral twist = 30.0°, Stereo angle = 15°

Table 8.73: Test data results for $n = 3$, chiral twist = 30.0°, stereo angle = 15° (dynamic augmentation, [10,90], 250 original simulated stereopairs, 80× augmentation

Network Configuration	No. of networks with > 55% test accuracy (/10)	Average test accuracy of trained networks (%)
211	10	97.8±0.6
121	10	97.7±0.7
112	10	97.8±0.9
111	10	97.3±0.5

E.1.3.4 $n = 3$, Chiral twist = 30.0°, Stereo angle = 20°

Table 8.74: Test data results for $n = 3$, chiral twist = 30.0°, stereo angle = 20° (dynamic augmentation, [10,90], 250 original simulated stereopairs, 80× augmentation

Network Configuration	No. of networks with > 55% test accuracy (/10)	Average test accuracy of trained networks (%)
211	10	98.7±0.6
121	10	98.4±0.5
112	10	97.9±0.6
111	10	98.4±0.4

E.1.3.5 $n = 3$, Chiral twist = 30.0°, Stereo angle = 25°

Table 8.75: Test data results for $n = 3$, chiral twist = 30.0°, stereo angle = 25° (dynamic augmentation, [10,90], 250 original simulated stereopairs, 80× augmentation

Network Configuration	No. of networks with > 55% test accuracy (/10)	Average test accuracy of trained networks (%)
211	10	98.2±0.7
121	10	98.2±0.8
112	10	98.3±0.7
111	10	98.0±0.9

E.1.3.6 $n = 3$, Chiral twist = 30.0°, Stereo angle = 30°

Table 8.76: Test data results for $n = 3$, chiral twist = 30.0°, stereo angle = 30° (dynamic augmentation, [10,90], 250 original simulated stereopairs, 80× augmentation

Network Configuration	No. of networks with > 55% test accuracy (/10)	Average test accuracy of trained networks (%)
211	10	98.2±0.8
121	10	98.1±0.6
112	10	98.6±0.6
111	10	98.7±0.3

E.1.3.7 $n = 3$, Chiral twist = 30.0°, Stereo angle = 35°

Table 8.77: Test data results for $n = 3$, chiral twist = 30.0°, stereo angle = 35° (dynamic augmentation, [10,90], 250 original simulated stereopairs, 80× augmentation

Network Configuration	No. of networks with > 55% test accuracy (/10)	Average test accuracy of trained networks (%)
211	10	98.0±0.7
121	10	98.3±0.6
112	10	98.3±0.5
111	10	98.4±0.6

E.1.3.8 $n = 3$, Chiral twist = 30.0°, Stereo angle = 40°

Table 8.78: Test data results for $n = 3$, chiral twist = 30.0°, stereo angle = 40° (dynamic augmentation, [10,90], 250 original simulated stereopairs, 80× augmentation

Network Configuration	No. of networks with > 55% test accuracy (/10)	Average test accuracy of trained networks (%)
211	10	97.7±0.5
121	10	97.8±0.7
112	10	97.8±0.6
111	10	98.2±0.6

E.1.3.9 $n = 3$, Chiral twist = 30.0°, Stereo angle = 45°

Table 8.79: Test data results for $n = 3$, chiral twist = 30.0°, stereo angle = 45° (dynamic augmentation, [10,90], 250 original simulated stereopairs, 80× augmentation

Network Configuration	No. of networks with > 55% test accuracy (/10)	Average test accuracy of trained networks (%)
211	10	97.3±0.7
121	10	97.2±0.7
112	10	97.9±0.6
111	10	97.8±0.6

E.2 Tables for $n = 4$

E.2.1 Chiral twist = 15.0°

E.2.1.1 $n = 4$, Chiral twist = 15.0°, Stereo angle = 5°

Table 8.80: Test data results for $n = 4$, chiral twist = 15.0°, stereo angle = 5° (dynamic augmentation, [10,90], 250 original simulated stereopairs, 80× augmentation

Network Configuration	No. of networks with > 55% test accuracy (/10)	Average test accuracy of trained networks (%)
211	10	82.5±1.8
121	10	88.7±1.2
112	10	88.2±2.0
111	9	82.8±2.0

E.2.1.2 $n = 4$, Chiral twist = 15.0°, Stereo angle = 10°

Table 8.81: Test data results for $n = 4$, chiral twist = 15.0°, stereo angle = 10° (dynamic augmentation, [10,90], 250 original simulated stereopairs, 80× augmentation

Network Configuration	No. of networks with > 55% test accuracy (/10)	Average test accuracy of trained networks (%)
211	10	91.2±1.9
121	10	94.4±1.0
112	10	94.4±1.0
111	10	93.7±1.6

E.2.1.3 $n = 4$, Chiral twist = 15.0°, Stereo angle = 15°

Table 8.82: Test data results for $n = 4$, chiral twist = 15.0°, stereo angle = 15° (dynamic augmentation, [10,90], 250 original simulated stereopairs, 80× augmentation

Network Configuration	No. of networks with > 55% test accuracy (/10)	Average test accuracy of trained networks (%)
211	10	92.5±1.3
121	10	94.3±0.9
112	10	94.4±1.2
111	9	96.0±0.4

E.2.1.4 $n = 4$, Chiral twist = 15.0°, Stereo angle = 20°

Table 8.83: Test data results for $n = 4$, chiral twist = 15.0°, stereo angle = 20° (dynamic augmentation, [10,90], 250 original simulated stereopairs, 80× augmentation

Network Configuration	No. of networks with > 55% test accuracy (/10)	Average test accuracy of trained networks (%)
211	10	92.9±0.8
121	10	93.2±1.5
112	10	93.9±1.8
111	9	96.5±0.7

E.2.1.5 $n = 4$, Chiral twist = 15.0°, Stereo angle = 25°

Table 8.84: Test data results for $n = 4$, chiral twist = 15.0°, stereo angle = 25° (dynamic augmentation, [10,90], 250 original simulated stereopairs, 80× augmentation

Network Configuration	No. of networks with > 55% test accuracy (/10)	Average test accuracy of trained networks (%)
211	10	92.3±1.2
121	10	92.7±1.6
112	10	92.7±1.0
111	10	95.8±0.7

E.2.1.6 $n = 4$, Chiral twist = 15.0°, Stereo angle = 30°

Table 8.85: Test data results for $n = 4$, chiral twist = 15.0°, stereo angle = 30° (dynamic augmentation, [10,90], 250 original simulated stereopairs, 80× augmentation

Network Configuration	No. of networks with > 55% test accuracy (/10)	Average test accuracy of trained networks (%)
211	10	90.3±1.4
121	10	92.4±1.3
112	10	91.4±1.6
111	9	94.8±0.6

E.2.1.7 $n = 4$, Chiral twist = 15.0°, Stereo angle = 35°

Table 8.86: Test data results for $n = 4$, chiral twist = 15.0°, stereo angle = 35° (dynamic augmentation, [10,90], 250 original simulated stereopairs, 80× augmentation

Network Configuration	No. of networks with > 55% test accuracy (/10)	Average test accuracy of trained networks (%)
211	10	88.1±1.9
121	10	89.0±1.3
112	10	90.0±1.4
111	10	92.4±1.0

E.2.1.8 $n = 4$, Chiral twist = 15.0°, Stereo angle = 40°

Table 8.87: Test data results for $n = 4$, chiral twist = 15.0°, stereo angle = 40° (dynamic augmentation, [10,90], 250 original simulated stereopairs, 80× augmentation

Network Configuration	No. of networks with > 55% test accuracy (/10)	Average test accuracy of trained networks (%)
211	10	83.2±1.7
121	10	84.5±2.3
112	10	87.8±1.9
111	10	89.8±1.6

E.2.1.9 $n = 4$, Chiral twist = 15.0°, Stereo angle = 45°

Table 8.88: Test data results for $n = 4$, chiral twist = 15.0°, stereo angle = 45° (dynamic augmentation, [10,90], 250 original simulated stereopairs, 80× augmentation

Network Configuration	No. of networks with > 55% test accuracy (/10)	Average test accuracy of trained networks (%)
211	10	80.6±2.5
121	10	81.1±1.5
112	10	84.5±1.4
111	9	86.9±1.7

E.2.2 Chiral twist = 22.5°

E.2.2.1 $n = 4$, Chiral twist = 22.5°, Stereo angle = 5°

Table 8.89: Test data results for $n = 4$, chiral twist = 22.5°, stereo angle = 5° (dynamic augmentation, [10,90], 250 original simulated stereopairs, 80× augmentation

Network Configuration	No. of networks with > 55% test accuracy (/10)	Average test accuracy of trained networks (%)
211	10	82.9±3.0
121	10	91.0±1.0
112	10	87.5±2.2
111	9	81.0±3.3

E.2.2.2 $n = 4$, Chiral twist = 22.5°, Stereo angle = 10°

Table 8.90: Test data results for $n = 4$, chiral twist = 22.5°, stereo angle = 10° (dynamic augmentation, [10,90], 250 original simulated stereopairs, 80× augmentation

Network Configuration	No. of networks with > 55% test accuracy (/10)	Average test accuracy of trained networks (%)
211	10	92.0±1.5
121	10	94.3±1.1
112	10	94.2±1.0
111	9	94.1±1.1

E.2.2.3 $n = 4$, Chiral twist = 22.5°, Stereo angle = 15°

Table 8.91: Test data results for $n = 4$, chiral twist = 22.5°, stereo angle = 15° (dynamic augmentation, [10,90], 250 original simulated stereopairs, 80× augmentation

Network Configuration	No. of networks with > 55% test accuracy (/10)	Average test accuracy of trained networks (%)
211	10	94.1±1.0
121	10	95.0±0.7
112	10	95.2±1.1
111	8	96.3±1.1

E.2.2.4 $n = 4$, Chiral twist = 22.5°, Stereo angle = 20°

Table 8.92: Test data results for $n = 4$, chiral twist = 22.5°, stereo angle = 20° (dynamic augmentation, [10,90], 250 original simulated stereopairs, 80× augmentation

Network Configuration	No. of networks with > 55% test accuracy (/10)	Average test accuracy of trained networks (%)
211	10	94.4±1.1
121	10	94.6±1.0
112	10	95.3±1.0
111	9	97.0±0.8

E.2.2.5 $n = 4$, Chiral twist = 22.5°, Stereo angle = 25°

Table 8.93: Test data results for $n = 4$, chiral twist = 22.5°, stereo angle = 25° (dynamic augmentation, [10,90], 250 original simulated stereopairs, 80× augmentation

Network Configuration	No. of networks with > 55% test accuracy (/10)	Average test accuracy of trained networks (%)
211	10	93.9±1.1
121	10	93.6±1.0
112	10	94.9±0.6
111	10	95.8±0.6

E.2.2.6 $n = 4$, Chiral twist = 22.5°, Stereo angle = 30°

Table 8.94: Test data results for $n = 4$, chiral twist = 22.5°, stereo angle = 30° (dynamic augmentation, [10,90], 250 original simulated stereopairs, 80× augmentation

Network Configuration	No. of networks with > 55% test accuracy (/10)	Average test accuracy of trained networks (%)
211	10	92.1±1.4
121	10	93.7±1.1
112	10	93.9±0.5
111	9	96.0±1.1

E.2.2.7 $n = 4$, Chiral twist = 22.5°, Stereo angle = 35°

Table 8.95: Test data results for $n = 4$, chiral twist = 22.5°, stereo angle = 35° (dynamic augmentation, [10,90], 250 original simulated stereopairs, 80× augmentation

Network Configuration	No. of networks with > 55% test accuracy (/10)	Average test accuracy of trained networks (%)
211	10	91.5±0.8
121	10	90.9±1.4
112	10	93.5±1.3
111	10	94.5±1.3

E.2.2.8 $n = 4$, Chiral twist = 22.5°, Stereo angle = 40°

Table 8.96: Test data results for $n = 4$, chiral twist = 22.5°, stereo angle = 40° (dynamic augmentation, [10,90], 250 original simulated stereopairs, 80× augmentation

Network Configuration	No. of networks with > 55% test accuracy (/10)	Average test accuracy of trained networks (%)
211	10	86.1±2.1
121	10	88.1±2.0
112	10	88.9±1.5
111	7	91.7±1.6

E.2.2.9 $n = 4$, Chiral twist = 22.5°, Stereo angle = 45°

Table 8.97: Test data results for $n = 4$, chiral twist = 22.5°, stereo angle = 45° (dynamic augmentation, [10,90], 250 original simulated stereopairs, 80× augmentation

Network Configuration	No. of networks with > 55% test accuracy (/10)	Average test accuracy of trained networks (%)
211	10	84.6±1.5
121	10	85.7±1.4
112	10	86.2±1.0
111	9	88.6±1.4

E.2.3 Chiral twist = 30.0°

E.2.3.1 $n = 4$, Chiral twist = 30.0°, Stereo angle = 5°

Table 8.98: Test data results for $n = 4$, chiral twist = 30.0°, stereo angle = 5° (dynamic augmentation, [10,90], 250 original simulated stereopairs, 80× augmentation

Network Configuration	No. of networks with > 55% test accuracy (/10)	Average test accuracy of trained networks (%)
211	10	76.0±3.7
121	10	85.9±1.7
112	10	84.2±1.7
111	10	74.1±4.7

E.2.3.2 $n = 4$, Chiral twist = 30.0°, Stereo angle = 10°

Table 8.99: Test data results for $n = 4$, chiral twist = 30.0°, stereo angle = 10° (dynamic augmentation, [10,90], 250 original simulated stereopairs, 80× augmentation

Network Configuration	No. of networks with > 55% test accuracy (/10)	Average test accuracy of trained networks (%)
211	10	88.9±1.7
121	10	90.9±1.8
112	10	91.1±1.4
111	9	91.6±1.1

E.2.3.3 $n = 4$, Chiral twist = 30.0°, Stereo angle = 15°

Table 8.100: Test data results for $n = 4$, chiral twist = 30.0°, stereo angle = 15° (dynamic augmentation, [10,90], 250 original simulated stereopairs, 80× augmentation

Network Configuration	No. of networks with > 55% test accuracy (/10)	Average test accuracy of trained networks (%)
211	10	90.7±1.1
121	10	92.9±1.2
112	10	92.7±0.9
111	10	95.5±0.8

E.2.3.4 $n = 4$, Chiral twist = 30.0°, Stereo angle = 20°

Table 8.101: Test data results for $n = 4$, chiral twist = 30.0°, stereo angle = 20° (dynamic augmentation, [10,90], 250 original simulated stereopairs, 80× augmentation

Network Configuration	No. of networks with > 55% test accuracy (/10)	Average test accuracy of trained networks (%)
211	10	91.8±1.8
121	10	92.8±1.6
112	10	94.2±1.3
111	9	96.1±1.0

E.2.3.5 $n = 4$, Chiral twist = 30.0°, Stereo angle = 25°

Table 8.102: Test data results for $n = 4$, chiral twist = 30.0°, stereo angle = 25° (dynamic augmentation, [10,90], 250 original simulated stereopairs, 80× augmentation

Network Configuration	No. of networks with > 55% test accuracy (/10)	Average test accuracy of trained networks (%)
211	10	90.9±0.9
121	10	92.1±1.5
112	10	92.5±1.0
111	9	95.5±1.0

E.2.3.6 $n = 4$, Chiral twist = 30.0°, Stereo angle = 30°

Table 8.103: Test data results for $n = 4$, chiral twist = 30.0°, stereo angle = 30° (dynamic augmentation, [10,90], 250 original simulated stereopairs, 80× augmentation

Network Configuration	No. of networks with > 55% test accuracy (/10)	Average test accuracy of trained networks (%)
211	10	90.2±1.3
121	10	91.1±1.4
112	10	92.4±1.1
111	9	94.2±1.3

E.2.3.7 $n = 4$, Chiral twist = 30.0°, Stereo angle = 35°

Table 8.104: Test data results for $n = 4$, chiral twist = 30.0°, stereo angle = 35° (dynamic augmentation, [10,90], 250 original simulated stereopairs, 80× augmentation

Network Configuration	No. of networks with > 55% test accuracy (/10)	Average test accuracy of trained networks (%)
211	10	88.4±1.5
121	10	89.4±1.6
112	10	90.3±1.2
111	10	92.5±2.2

E.2.3.8 $n = 4$, Chiral twist = 30.0°, Stereo angle = 40°

Table 8.105: Test data results for $n = 4$, chiral twist = 30.0°, stereo angle = 40° (dynamic augmentation, [10,90], 250 original simulated stereopairs, 80× augmentation

Network Configuration	No. of networks with > 55% test accuracy (/10)	Average test accuracy of trained networks (%)
211	10	84.8±1.5
121	10	84.4±2.1
112	10	87.6±2.1
111	8	89.7±2.0

E.2.3.9 $n = 4$, Chiral twist = 30.0°, Stereo angle = 45°

Table 8.106: Test data results for $n = 4$, chiral twist = 30.0°, stereo angle = 45° (dynamic augmentation, [10,90], 250 original simulated stereopairs, 80× augmentation

Network Configuration	No. of networks with > 55% test accuracy (/10)	Average test accuracy of trained networks (%)
211	10	81.3±2.7
121	10	82.2±2.0
112	10	84.4±1.4
111	9	87.3±1.0

E.3 Tables for $n = 5$

E.3.1 Chiral twist = 15.0°

E.3.1.1 $n = 5$, Chiral twist = 15.0°, Stereo angle = 5°

Table 8.107: Test data results for $n = 5$, chiral twist = 15.0°, stereo angle = 5° (dynamic augmentation, [10,90], 250 original simulated stereopairs, 80× augmentation

Network Configuration	No. of networks with > 55% test accuracy (/10)	Average test accuracy of trained networks (%)
211	10	70.2±7.7
121	10	86.6±2.7
112	10	84.1±2.5
111	9	77.3±4.1

E.3.1.2 $n = 5$, Chiral twist = 15.0°, Stereo angle = 10°

Table 8.108: Test data results for $n = 5$, chiral twist = 15.0°, stereo angle = 10° (dynamic augmentation, [10,90], 250 original simulated stereopairs, 80× augmentation

Network Configuration	No. of networks with > 55% test accuracy (/10)	Average test accuracy of trained networks (%)
211	10	88.0±2.4
121	10	91.1±1.3
112	10	91.7±0.7
111	9	91.6±2.2

E.3.1.3 $n = 5$, Chiral twist = 15.0°, Stereo angle = 15°

Table 8.109: Test data results for $n = 5$, chiral twist = 15.0°, stereo angle = 15° (dynamic augmentation, [10,90], 250 original simulated stereopairs, 80× augmentation

Network Configuration	No. of networks with > 55% test accuracy (/10)	Average test accuracy of trained networks (%)
211	10	88.8±1.5
121	10	91.8±0.9
112	10	91.8±1.5
111	8	94.4±0.7

E.3.1.4 $n = 5$, Chiral twist = 15.0°, Stereo angle = 20°

Table 8.110: Test data results for $n = 5$, chiral twist = 15.0°, stereo angle = 20° (dynamic augmentation, [10,90], 250 original simulated stereopairs, 80× augmentation

Network Configuration	No. of networks with > 55% test accuracy (/10)	Average test accuracy of trained networks (%)
211	10	88.0±2.2
121	10	90.8±0.9
112	10	90.6±1.5
111	7	92.7±1.1

E.3.1.5 $n = 5$, Chiral twist = 15.0°, Stereo angle = 25°

Table 8.111: Test data results for $n = 5$, chiral twist = 15.0°, stereo angle = 25° (dynamic augmentation, [10,90], 250 original simulated stereopairs, 80× augmentation

Network Configuration	No. of networks with > 55% test accuracy (/10)	Average test accuracy of trained networks (%)
211	10	84.7±2.3
121	10	87.0±1.8
112	10	89.0±1.9
111	8	91.2±1.9

E.3.1.6 $n = 5$, Chiral twist = 15.0°, Stereo angle = 30°

Table 8.112: Test data results for $n = 5$, chiral twist = 15.0°, stereo angle = 30° (dynamic augmentation, [10,90], 250 original simulated stereopairs, 80× augmentation

Network Configuration	No. of networks with > 55% test accuracy (/10)	Average test accuracy of trained networks (%)
211	10	78.6±2.7
121	10	81.7±1.6
112	10	83.9±1.9
111	9	86.5±1.3

E.3.1.7 $n = 5$, Chiral twist = 15.0°, Stereo angle = 35°

Table 8.113: Test data results for $n = 5$, chiral twist = 15.0°, stereo angle = 35° (dynamic augmentation, [10,90], 250 original simulated stereopairs, 80× augmentation

Network Configuration	No. of networks with > 55% test accuracy (/10)	Average test accuracy of trained networks (%)
211	10	73.3±1.8
121	10	76.5±2.1
112	10	79.5±2.1
111	9	80.7±1.0

E.3.1.8 $n = 5$, Chiral twist = 15.0°, Stereo angle = 40°

Table 8.114: Test data results for $n = 5$, chiral twist = 15.0°, stereo angle = 40° (dynamic augmentation, [10,90], 250 original simulated stereopairs, 80× augmentation

Network Configuration	No. of networks with > 55% test accuracy (/10)	Average test accuracy of trained networks (%)
211	10	70.5±2.3
121	10	75.3±2.2
112	10	77.3±1.9
111	9	76.6±2.8

E.3.1.9 $n = 5$, Chiral twist = 15.0°, Stereo angle = 45°

Table 8.115: Test data results for $n = 5$, chiral twist = 15.0°, stereo angle = 45° (dynamic augmentation, [10,90], 250 original simulated stereopairs, 80× augmentation

Network Configuration	No. of networks with > 55% test accuracy (/10)	Average test accuracy of trained networks (%)
211	10	75.0±3.2
121	10	77.5±2.8
112	9	80.2±1.5
111	9	83.8±3.5

E.3.2 Chiral twist = 22.5°

E.3.2.1 $n = 5$, Chiral twist = 22.5°, Stereo angle = 5°

Table 8.116: Test data results for $n = 5$, chiral twist = 22.5°, stereo angle = 5° (dynamic augmentation, [10,90], 250 original simulated stereopairs, 80× augmentation

Network Configuration	No. of networks with > 55% test accuracy (/10)	Average test accuracy of trained networks (%)
211	10	72.3±7.4
121	9	87.5±1.2
112	10	83.5±2.0
111	7	81.6±1.3

E.3.2.2 n = 5, Chiral twist = 22.5°, Stereo angle = 10°

Table 8.117: Test data results for n = 5, chiral twist = 22.5°, stereo angle = 10° (dynamic augmentation, [10,90], 250 original simulated stereopairs, 80× augmentation

Network Configuration	No. of networks with > 55% test accuracy (/10)	Average test accuracy of trained networks (%)
211	10	88.4±1.9
121	10	92.8±1.1
112	10	92.4±1.2
111	7	93.5±1.3

E.3.2.3 n = 5, Chiral twist = 22.5°, Stereo angle = 15°

Table 8.118: Test data results for n = 5, chiral twist = 22.5°, stereo angle = 15° (dynamic augmentation, [10,90], 250 original simulated stereopairs, 80× augmentation

Network Configuration	No. of networks with > 55% test accuracy (/10)	Average test accuracy of trained networks (%)
211	10	90.4±1.5
121	10	93.3±1.3
112	10	93.0±1.2
111	9	95.5±0.8

E.3.2.4 n = 5, Chiral twist = 22.5°, Stereo angle = 20°

Table 8.119: Test data results for n = 5, chiral twist = 22.5°, stereo angle = 20° (dynamic augmentation, [10,90], 250 original simulated stereopairs, 80× augmentation

Network Configuration	No. of networks with > 55% test accuracy (/10)	Average test accuracy of trained networks (%)
211	10	89.1±0.9
121	10	90.2±1.5
112	10	91.7±1.6
111	7	94.1±0.6

E.3.2.5 $n = 5$, Chiral twist = 22.5°, Stereo angle = 25°

Table 8.120: Test data results for $n = 5$, chiral twist = 22.5°, stereo angle = 25° (dynamic augmentation, [10,90], 250 original simulated stereopairs, 80× augmentation

Network Configuration	No. of networks with > 55% test accuracy (/10)	Average test accuracy of trained networks (%)
211	10	85.4±1.7
121	10	87.5±1.6
112	10	89.9±1.0
111	8	92.4±0.8

E.3.2.6 $n = 5$, Chiral twist = 22.5°, Stereo angle = 30°

Table 8.121: Test data results for $n = 5$, chiral twist = 22.5°, stereo angle = 30° (dynamic augmentation, [10,90], 250 original simulated stereopairs, 80× augmentation

Network Configuration	No. of networks with > 55% test accuracy (/10)	Average test accuracy of trained networks (%)
211	10	77.7±2.7
121	10	83.1±1.9
112	10	84.5±1.8
111	9	88.4±2.3

E.3.2.7 $n = 5$, Chiral twist = 22.5°, Stereo angle = 35°

Table 8.122: Test data results for $n = 5$, chiral twist = 22.5°, stereo angle = 35° (dynamic augmentation, [10,90], 250 original simulated stereopairs, 80× augmentation

Network Configuration	No. of networks with > 55% test accuracy (/10)	Average test accuracy of trained networks (%)
211	10	73.1±2.4
121	10	78.0±1.9
112	10	79.8±1.9
111	9	82.8±1.4

E.3.2.8 $n = 5$, Chiral twist = 22.5°, Stereo angle = 40°

Table 8.123: Test data results for $n = 5$, chiral twist = 22.5°, stereo angle = 40° (dynamic augmentation, [10,90], 250 original simulated stereopairs, 80× augmentation

Network Configuration	No. of networks with > 55% test accuracy (/10)	Average test accuracy of trained networks (%)
211	10	71.5±3.5
121	10	76.2±2.3
112	10	76.5±2.0
111	9	79.2±2.0

E.3.2.9 $n = 5$, Chiral twist = 22.5°, Stereo angle = 45°

Table 8.124: Test data results for $n = 5$, chiral twist = 22.5°, stereo angle = 45° (dynamic augmentation, [10,90], 250 original simulated stereopairs, 80× augmentation

Network Configuration	No. of networks with > 55% test accuracy (/10)	Average test accuracy of trained networks (%)
211	10	76.9±2.3
121	10	77.2±1.7
112	10	81.6±1.8
111	9	83.9±3.4

E.3.3 Chiral twist = 30.0°

E.3.3.1 $n = 5$, Chiral twist = 30.0°, Stereo angle = 5°

Table 8.125: Test data results for $n = 5$, chiral twist = 30.0°, stereo angle = 5° (dynamic augmentation, [10,90], 250 original simulated stereopairs, 80× augmentation

Network Configuration	No. of networks with > 55% test accuracy (/10)	Average test accuracy of trained networks (%)
211	4	60.9±3.6
121	10	75.4±3.1
112	10	69.5±3.3
111	9	68.9±4.5

E.3.3.2 $n = 5$, Chiral twist = 30.0°, Stereo angle = 10°

Table 8.126: Test data results for $n = 5$, chiral twist = 30.0°, stereo angle = 10° (dynamic augmentation, [10,90], 250 original simulated stereopairs, 80× augmentation

Network Configuration	No. of networks with > 55% test accuracy (/10)	Average test accuracy of trained networks (%)
211	10	75.0±5.0
121	10	83.0±1.5
112	10	84.7±1.6
111	7	86.7±1.3

E.3.3.3 $n = 5$, Chiral twist = 30.0°, Stereo angle = 15°

Table 8.127: Test data results for $n = 5$, chiral twist = 30.0°, stereo angle = 15° (dynamic augmentation, [10,90], 250 original simulated stereopairs, 80× augmentation

Network Configuration	No. of networks with > 55% test accuracy (/10)	Average test accuracy of trained networks (%)
211	10	75.5±3.2
121	10	83.4±4.1
112	10	85.2±1.9
111	9	89.3±2.6

E.3.3.4 $n = 5$, Chiral twist = 30.0°, Stereo angle = 20°

Table 8.128: Test data results for $n = 5$, chiral twist = 30.0°, stereo angle = 20° (dynamic augmentation, [10,90], 250 original simulated stereopairs, 80× augmentation

Network Configuration	No. of networks with > 55% test accuracy (/10)	Average test accuracy of trained networks (%)
211	10	71.2±3.1
121	10	80.1±2.5
112	9	82.5±2.2
111	8	88.6±1.3

E.3.3.5 $n = 5$, Chiral twist = 30.0°, Stereo angle = 25°

Table 8.129: Test data results for $n = 5$, chiral twist = 30.0°, stereo angle = 25° (dynamic augmentation, [10,90], 250 original simulated stereopairs, 80× augmentation

Network Configuration	No. of networks with > 55% test accuracy (/10)	Average test accuracy of trained networks (%)
211	10	63.8±3.6
121	10	73.9±0.8
112	10	77.3±2.2
111	6	85.1±2.6

E.3.3.6 $n = 5$, Chiral twist = 30.0°, Stereo angle = 30°

Table 8.130: Test data results for $n = 5$, chiral twist = 30.0°, stereo angle = 30° (dynamic augmentation, [10,90], 250 original simulated stereopairs, 80× augmentation

Network Configuration	No. of networks with > 55% test accuracy (/10)	Average test accuracy of trained networks (%)
211	10	62.9±2.1
121	10	67.7±2.6
112	10	71.6±2.5
111	8	80.4±2.8

E.3.3.7 $n = 5$, Chiral twist = 30.0°, Stereo angle = 35°

Table 8.131: Test data results for $n = 5$, chiral twist = 30.0°, stereo angle = 35° (dynamic augmentation, [10,90], 250 original simulated stereopairs, 80× augmentation

Network Configuration	No. of networks with > 55% test accuracy (/10)	Average test accuracy of trained networks (%)
211	10	58.6±2.2
121	10	63.8±2.0
112	10	67.1±2.0
111	10	70.0±3.7

E.3.3.8 n = 5, Chiral twist = 30.0°, Stereo angle = 40°

Table 8.132: Test data results for n = 5, chiral twist = 30.0°, stereo angle = 40° (dynamic augmentation, [10,90], 250 original simulated stereopairs, 80× augmentation

Network Configuration	No. of networks with > 55% test accuracy (/10)	Average test accuracy of trained networks (%)
211	10	58.8±2.9
121	10	63.5±2.0
112	10	64.2±1.1
111	9	65.4±3.4

E.3.3.9 n = 5, Chiral twist = 30.0°, Stereo angle = 45°

Table 8.133: Test data results for n = 5, chiral twist = 30.0°, stereo angle = 45° (dynamic augmentation, [10,90], 250 original simulated stereopairs, 80× augmentation

Network Configuration	No. of networks with > 55% test accuracy (/10)	Average test accuracy of trained networks (%)
211	9	59.3±1.8
121	10	64.9±2.9
112	9	67.0±2.2
111	8	72.0±1.6

F Weak Labelling

F.1 Tables for 20% weak labelling, dynamic augmentation [10,90]

F.1.1 125×80×2

Table 8.134: Test accuracies of 20% weak labelling, dynamic augmentation training (125 original simulated stereopairs, 80× augmentation) [10,90]

Network Configuration	Average train accuracy of networks (>55%) (%)	Number of successfully trained networks (>55%)	Average majority test accuracy (>55%) (%)	Average minority test accuracy (>55%) (%)	Average test accuracy (>55%) (%)
211	97.7±0.6	10	71.3±3.3	71.2±3.7	71.2±3.4
121	98.8±0.3	10	72.6±4.6	72.2±4.3	72.4±4.3
112	98.8±0.5	10	71.6±2.5	73.5±3.5	72.6±3.1
111	98.4±0.7	9	67.2±3.1	67.5±2.0	67.4±2.5

F.1.2 250×40×2

Table 8.135: Test accuracies of 20% , dynamic augmentation training (250 original simulated stereopairs, 40× augmentation) [10,90]

Network Configuration	Average train accuracy of networks (>55%) (%)	Number of successfully trained networks (>55%)	Average majority test accuracy (>55%) (%)	Average minority test accuracy (>55%) (%)	Average test accuracy (>55%) (%)
211	93.7±0.9	10	77.3±1.8	77.1±2.9	77.2±2.3
121	96.1±0.9	10	78.6±1.9	78.0±2.8	78.3±2.3
112	96.6±0.7	10	76.4±3.7	76.7±2.6	76.6±3.1
111	95.4±1.1	9	74.1±2.9	75.4±1.8	74.8±2.4

F.1.3 500×20×2

Table 8.136: Test accuracies of 20% weak labelling, dynamic augmentation training (500 original simulated stereopairs, 20× augmentation) [10,90]

Network Configuration	Average train accuracy of networks (>55%) (%)	Number of successfully trained networks (>55%)	Average majority test accuracy (>55%) (%)	Average minority test accuracy (>55%) (%)	Average test accuracy (>55%) (%)
211	85.9±1.8	10	83.5±2.0	82.9±2.9	83.2±2.4
121	90.9±1.5	10	82.6±2.6	82.2±1.9	82.4±2.2
112	91.7±0.8	10	80.8±1.8	81.4±1.8	81.1±1.8
111	88.0±1.7	10	79.8±1.7	79.8±1.6	79.8±1.6

F.1.4 1000×10×2

Table 8.137: Test accuracies of 20% weak labelling, dynamic augmentation training (1000 original simulated stereopairs, 10× augmentation) [10,90]

Network Configuration	Average train accuracy of networks (>55%) (%)	Number of successfully trained networks (>55%)	Average majority test accuracy (>55%) (%)	Average minority test accuracy (>55%) (%)	Average test accuracy (>55%) (%)
211	79.4±1.7	10	87.3±1.7	87.2±1.2	87.3±1.4
121	83.9±0.6	10	89.4±1.3	89.5±1.2	89.4±1.2
112	85.3±0.8	9	87.4±1.7	87.2±1.8	87.3±1.7
111	80.4±2.1	10	87.1±2.1	87.5±1.7	87.3±1.9

F.1.5 2000×5×2

Table 8.138: Test accuracies of 20% weak labelling, dynamic augmentation training (2000 original simulated stereopairs, 5× augmentation) [10,90]

Network Configuration	Average train accuracy of networks (>55%) (%)	Number of successfully trained networks (>55%)	Average majority test accuracy (>55%) (%)	Average minority test accuracy (>55%) (%)	Average test accuracy (>55%) (%)
211	77.8±0.9	10	93.9±1.7	94.0±2.0	94.0±1.8
121	79.8±0.9	10	94.3±1.3	95.0±1.3	94.6±1.3
112	79.9±0.7	10	94.1±1.3	94.0±1.4	94.0±1.3
111	76.6±2.4	10	92.9±3.2	92.8±2.3	92.9±2.7

F.1.6 5000×2×2

Table 8.139: Test accuracies of 20% weak labelling, dynamic augmentation training (5000 original simulated stereopairs, 2× augmentation) [10,90]

Network Configuration	Average train accuracy of networks (>55%) (%)	Number of successfully trained networks (>55%)	Average majority test accuracy (>55%) (%)	Average minority test accuracy (>55%) (%)	Average test accuracy (>55%) (%)
211	75.7±1.0	9	95.7±1.5	94.6±1.8	95.2±1.7
121	78.4±0.7	8	97.7±0.6	97.7±0.8	97.7±0.7
112	78.0±0.6	9	97.5±1.2	97.8±0.6	97.7±1.0
111	77.1±0.4	9	97.9±1.0	97.6±1.2	97.7±1.1

F.1.7 10000×1×2

Table 8.140: Test accuracies of 20% weak labelling, dynamic augmentation training (10000 original simulated stereopairs, 1× augmentation) [10,90]

Network Configuration	Average train accuracy of networks (>55%) (%)	Number of successfully trained networks (>55%)	Average majority test accuracy (>55%) (%)	Average minority test accuracy (>55%) (%)	Average test accuracy (>55%) (%)
211	75.1±1.5	9	95.1±2.9	95.0±2.3	95.0±2.5
121	76.4±3.8	8	97.1±5.0	97.4±3.8	97.2±4.3
112	76.6±0.9	5	98.0±1.0	98.0±0.8	98.0±0.9
111	76.1±0.9	5	98.2±0.6	98.2±0.9	98.2±0.7

F.2 Tables for 20% weak labelling, dynamic augmentation [10:10:90]

F.2.1 125×80×2

Table 8.141: Test accuracies of 20% weak labelling, dynamic augmentation training (125 original simulated stereopairs, 80× augmentation) [10:10:90]

Network Configuration	Average train accuracy of networks (>55%) (%)	Number of successfully trained networks (>55%)	Average majority test accuracy (>55%) (%)	Average minority test accuracy (>55%) (%)	Average test accuracy (>55%) (%)
211	99.0±0.4	10	72.4±3.7	73.0±1.9	72.7±2.9
121	99.6±0.1	10	73.2±2.7	72.8±4.4	73.0±3.5
112	99.5±0.1	10	71.8±2.1	71.0±3.1	71.4±2.7
111	99.3±0.2	10	68.2±4.1	67.0±3.8	67.6±3.9

F.2.2 250×40×2

Table 8.142: Test accuracies of 20% weak labelling, dynamic augmentation training (250 original simulated stereopairs, 40× augmentation) [10:10:90]

Network Configuration	Average train accuracy of networks (>55%) (%)	Number of successfully trained networks (>55%)	Average majority test accuracy (>55%) (%)	Average minority test accuracy (>55%) (%)	Average test accuracy (>55%) (%)
211	97.3±0.5	10	77.7±3.6	77.5±2.7	77.6±3.1
121	98.4±0.4	10	78.2±1.8	78.1±2.2	78.2±1.9
112	98.4±0.3	10	78.4±3.1	76.5±1.8	77.4±2.6
111	98.1±0.3	10	73.9±1.2	73.1±2.2	73.5±1.7

F.2.3 500×20×2

Table 8.143: Test accuracies of 20% weak labelling, dynamic augmentation training (500 original simulated stereopairs, 20× augmentation) [10:10:90]

Network Configuration	Average train accuracy of networks (>55%) (%)	Number of successfully trained networks (>55%)	Average majority test accuracy (>55%) (%)	Average minority test accuracy (>55%) (%)	Average test accuracy (>55%) (%)
211	92.6±0.6	10	80.3±2.1	81.2±1.3	80.8±1.8
121	95.1±0.7	10	81.0±1.7	79.5±2.4	80.3±2.2
112	95.9±0.4	10	80.3±1.3	77.9±1.1	79.1±1.7
111	94.4±0.8	10	77.8±1.5	77.9±1.8	77.9±1.6

F.2.4 1000×10×2

Table 8.144: Test accuracies of 20% weak labelling, dynamic augmentation training (1000 original simulated stereopairs, 10× augmentation) [10:10:90]

Network Configuration	Average train accuracy of networks (>55%) (%)	Number of successfully trained networks (>55%)	Average majority test accuracy (>55%) (%)	Average minority test accuracy (>55%) (%)	Average test accuracy (>55%) (%)
211	85.9±0.7	10	85.3±0.7	84.9±2.5	85.1±1.8
121	89.0±0.9	10	84.4±1.7	85.3±1.8	84.8±1.8
112	89.2±0.4	10	83.5±1.3	83.9±0.9	83.7±1.1
111	88.2±0.8	10	82.7±1.8	82.2±1.9	82.5±1.8

F.2.5 2000×5×2

Table 8.145: Test accuracies of 20% weak labelling, dynamic augmentation training (2000 original simulated stereopairs, 5× augmentation) [10:10:90]

Network Configuration	Average train accuracy of networks (>55%) (%)	Number of successfully trained networks (>55%)	Average majority test accuracy (>55%) (%)	Average minority test accuracy (>55%) (%)	Average test accuracy (>55%) (%)
211	81.1±0.6	10	92.0±1.6	92.0±1.3	92.0±1.4
121	83.0±1.0	10	90.4±1.5	90.4±1.5	90.4±1.5
112	82.9±1.2	10	89.9±1.8	89.9±1.9	89.9±1.8
111	82.0±0.7	8	90.0±2.2	89.0±1.8	89.5±2.0

F.2.6 5000×2×2

Table 8.146: Test accuracies of 20% weak labelling, dynamic augmentation training (5000 original simulated stereopairs, 2× augmentation) [10:10:90]

Network Configuration	Average train accuracy of networks (>55%) (%)	Number of successfully trained networks (>55%)	Average majority test accuracy (>55%) (%)	Average minority test accuracy (>55%) (%)	Average test accuracy (>55%) (%)
211	78.3±0.3	10	96.7±1.3	97.0±0.7	96.9±1.0
121	79.6±0.5	9	97.3±1.2	96.7±1.2	97.0±1.2
112	79.5±0.3	9	96.7±1.0	96.3±1.2	96.5±1.1
111	78.3±0.5	6	96.6±1.0	96.3±0.7	96.4±0.8

F.2.7 10000×1×2

Table 8.147: Test accuracies of 20% weak labelling, dynamic augmentation training (10000 original simulated stereopairs, 1× augmentation) [10:10:90]

Network Configuration	Average train accuracy of networks (>55%) (%)	Number of successfully trained networks (>55%)	Average majority test accuracy (>55%) (%)	Average minority test accuracy (>55%) (%)	Average test accuracy (>55%) (%)
211	76.8±1.7	9	96.9±2.3	96.8±2.4	96.8±2.3
121	78.1±1.2	8	98.6±0.6	98.2±0.5	98.4±0.6
112	78.4±0.5	7	98.8±0.8	98.4±0.5	98.6±0.7
111	77.8±0.8	3	98.7±0.8	98.1±0.2	98.4±0.6

F.3 Tables for 20% weak labelling, single augmentation [10,90]

F.3.1 125×80×2

Table 8.148: Test accuracies of 20% weak labelling, single augmentation training (125 original simulated stereopairs, 80× augmentation) [10,90]

Network Configuration	Average train accuracy of networks (>55%) (%)	Number of successfully trained networks (>55%)	Average majority test accuracy (>55%) (%)	Average minority test accuracy (>55%) (%)	Average test accuracy (>55%) (%)
211	97.8±0.8	10	69.6±4.0	69.2±3.2	69.4±3.5
121	99.0±0.4	10	73.7±3.5	72.5±3.0	73.1±3.2
112	99.0±0.2	9	71.9±2.5	72.3±2.0	72.1±2.2
111	98.8±0.2	9	67.9±3.3	68.3±2.0	68.1±2.6

F.3.2 250×40×2

Table 8.149: Test accuracies of 20% weak labelling, single augmentation training (250 original simulated stereopairs, 40× augmentation) [10,90]

Network Configuration	Average train accuracy of networks (>55%) (%)	Number of successfully trained networks (>55%)	Average majority test accuracy (>55%) (%)	Average minority test accuracy (>55%) (%)	Average test accuracy (>55%) (%)
211	93.6±1.1	10	77.5±1.5	78.1±2.5	77.8±2.0
121	96.4±0.7	10	78.6±2.5	78.2±2.2	78.4±2.3
112	96.8±0.3	10	76.9±1.3	77.7±2.5	77.3±2.0
111	96.3±0.4	10	73.9±3.9	75.2±2.6	74.6±3.3

F.3.3 500×20×2

Table 8.150: Test accuracies of 20% weak labelling, single augmentation training (500 original simulated stereopairs, 20× augmentation) [10,90]

Network Configuration	Average train accuracy of networks (>55%) (%)	Number of successfully trained networks (>55%)	Average majority test accuracy (>55%) (%)	Average minority test accuracy (>55%) (%)	Average test accuracy (>55%) (%)
211	86.7±1.2	10	81.9±2.0	82.2±2.0	82.1±1.9
121	90.8±1.3	10	83.2±1.7	83.9±2.0	83.6±1.8
112	91.9±1.0	10	82.4±1.4	81.3±1.9	81.9±1.7
111	90.4±1.2	9	81.3±1.9	80.8±2.2	81.1±2.0

F.3.4 1000×10×2

Table 8.151: Test accuracies of 20% weak labelling, single augmentation training (1000 original simulated stereopairs, 10× augmentation) [10,90]

Network Configuration	Average train accuracy of networks (>55%) (%)	Number of successfully trained networks (>55%)	Average majority test accuracy (>55%) (%)	Average minority test accuracy (>55%) (%)	Average test accuracy (>55%) (%)
211	80.8±0.8	10	88.9±2.2	89.6±1.8	89.2±2.0
121	84.2±2.1	10	89.0±2.7	88.8±1.8	88.9±2.3
112	85.3±0.8	10	87.7±1.8	88.3±1.5	88.0±1.7
111	83.2±1.0	10	87.0±2.9	87.7±1.9	87.4±2.4

F.3.5 2000×5×2

Table 8.152: Test accuracies of 20% weak labelling, single augmentation training (2000 original simulated stereopairs, 5× augmentation) [10,90]

Network Configuration	Average train accuracy of networks (>55%) (%)	Number of successfully trained networks (>55%)	Average majority test accuracy (>55%) (%)	Average minority test accuracy (>55%) (%)	Average test accuracy (>55%) (%)
211	77.9±0.8	10	93.7±2.1	93.6±2.7	93.6±2.4
121	80.6±0.7	10	94.6±1.1	94.5±1.5	94.5±1.3
112	80.4±1.0	10	94.3±1.6	93.9±1.6	94.1±1.6
111	79.6±0.7	8	93.7±1.3	94.1±0.8	93.9±1.1

F.3.6 5000×2×2

Table 8.153: Test accuracies of 20% weak labelling, single augmentation training (5000 original simulated stereopairs, 2× augmentation) [10,90]

Network Configuration	Average train accuracy of networks (>55%) (%)	Number of successfully trained networks (>55%)	Average majority test accuracy (>55%) (%)	Average minority test accuracy (>55%) (%)	Average test accuracy (>55%) (%)
211	76.6±0.7	10	95.6±1.8	95.6±2.1	95.6±1.9
121	78.7±0.5	10	98.0±0.6	98.2±1.0	98.1±0.8
112	78.5±0.3	9	98.0±0.8	97.9±0.7	97.9±0.7
111	77.8±0.4	10	98.3±0.9	98.4±1.0	98.4±0.9

F.3.7 10000×1×2

Table 8.154: Test accuracies of 20% weak labelling, single augmentation training (10000 original simulated stereopairs, 1× augmentation) [10,90]

Network Configuration	Average train accuracy of networks (>55%) (%)	Number of successfully trained networks (>55%)	Average majority test accuracy (>55%) (%)	Average minority test accuracy (>55%) (%)	Average test accuracy (>55%) (%)
211	75.2±1.0	8	95.8±2.3	95.9±1.9	95.8±2.0
121	78.3±0.4	10	98.8±0.4	98.8±0.5	98.8±0.5
112	78.3±0.2	10	99.0±0.6	98.8±0.3	98.9±0.5
111	75.8±4.8	7	96.7±5.6	97.1±4.2	96.9±4.7

F.4 Tables for 20% weak labelling, single augmentation [10:10:90]

F.4.1 125×80×2

Table 8.155: Test accuracies of 20% weak labelling, single augmentation training (125 original simulated stereopairs, 80× augmentation) [10:10:90]

Network Configuration	Average train accuracy of networks (>55%) (%)	Number of successfully trained networks (>55%)	Average majority test accuracy (>55%) (%)	Average minority test accuracy (>55%) (%)	Average test accuracy (>55%) (%)
211	99.1±0.2	10	71.3±2.7	71.6±2.4	71.5±2.5
121	99.6±0.1	10	74.7±4.0	74.3±4.0	74.5±3.9
112	99.5±0.1	10	71.3±3.4	71.4±4.0	71.4±3.6
111	99.5±0.1	10	67.3±3.4	68.1±3.8	67.7±3.5

F.4.2 250×40×2

Table 8.156: Test accuracies of 20% weak labelling, single augmentation training (250 original simulated stereopairs, 40× augmentation) [10:10:90]

Network Configuration	Average train accuracy of networks (>55%) (%)	Number of successfully trained networks (>55%)	Average majority test accuracy (>55%) (%)	Average minority test accuracy (>55%) (%)	Average test accuracy (>55%) (%)
211	97.3±0.4	10	76.2±2.6	76.7±2.0	76.5±2.3
121	98.5±0.2	10	78.1±2.9	77.5±3.2	77.8±3.0
112	98.7±0.2	10	75.2±1.6	75.5±2.4	75.4±2.0
111	98.5±0.4	10	72.9±3.5	72.8±2.3	72.8±2.9

F.4.3 500×20×2

Table 8.157: Test accuracies of 20% weak labelling, single augmentation training (500 original simulated stereopairs, 20× augmentation) [10:10:90]

Network Configuration	Average train accuracy of networks (>55%) (%)	Number of successfully trained networks (>55%)	Average majority test accuracy (>55%) (%)	Average minority test accuracy (>55%) (%)	Average test accuracy (>55%) (%)
211	92.8±0.6	10	80.2±2.2	81.8±2.6	81.0±2.5
121	95.3±0.5	10	81.2±2.1	81.7±1.8	81.5±1.9
112	96.0±0.6	10	79.5±2.3	78.5±2.6	79.0±2.5
111	94.7±0.5	10	79.1±1.9	78.9±1.8	79.0±1.8

F.4.4 1000×10×2

Table 8.158: Test accuracies of 20% weak labelling, single augmentation training (1000 original simulated stereopairs, 10× augmentation) [10:10:90]

Network Configuration	Average train accuracy of networks (>55%) (%)	Number of successfully trained networks (>55%)	Average majority test accuracy (>55%) (%)	Average minority test accuracy (>55%) (%)	Average test accuracy (>55%) (%)
211	86.5±0.8	10	85.7±1.9	85.4±3.1	85.6±2.5
121	89.2±1.2	10	85.6±1.5	85.4±1.5	85.5±1.5
112	90.2±1.2	10	83.1±2.3	84.1±1.3	83.6±1.9
111	89.0±1.0	10	82.7±2.3	83.6±2.2	83.1±2.2

F.4.5 2000×5×2

Table 8.159: Test accuracies of 20% weak labelling, single augmentation training (2000 original simulated stereopairs, 5× augmentation) [10:10:90]

Network Configuration	Average train accuracy of networks (>55%) (%)	Number of successfully trained networks (>55%)	Average majority test accuracy (>55%) (%)	Average minority test accuracy (>55%) (%)	Average test accuracy (>55%) (%)
211	82.3±0.8	10	89.6±1.4	89.8±1.3	89.7±1.3
121	84.1±0.7	10	90.4±1.7	89.7±1.8	90.0±1.8
112	85.8±1.8	10	87.0±2.3	88.0±1.5	87.5±1.9
111	83.1±1.2	10	88.4±2.3	88.5±1.5	88.5±1.9

F.4.6 5000×2×2

Table 8.160: Test accuracies of 20% weak labelling, single augmentation training (5000 original simulated stereopairs, 2× augmentation) [10:10:90]

Network Configuration	Average train accuracy of networks (>55%) (%)	Number of successfully trained networks (>55%)	Average majority test accuracy (>55%) (%)	Average minority test accuracy (>55%) (%)	Average test accuracy (>55%) (%)
211	79.8±0.5	10	91.6±1.7	92.2±0.9	91.9±1.3
121	81.2±0.6	10	94.0±1.6	93.8±1.3	93.9±1.4
112	81.8±1.0	10	93.5±1.3	93.2±1.2	93.4±1.2
111	78.8±2.2	10	93.4±1.5	94.0±1.3	93.7±1.4

F.4.7 10000×1×2

Table 8.161: Test accuracies of 20% weak labelling, single augmentation training (10000 original simulated stereopairs, 1× augmentation) [10:10:90]

Network Configuration	Average train accuracy of networks (>55%) (%)	Number of successfully trained networks (>55%)	Average majority test accuracy (>55%) (%)	Average minority test accuracy (>55%) (%)	Average test accuracy (>55%) (%)
211	78.4±0.8	10	92.4±1.8	92.4±1.6	92.4±1.7
121	80.6±0.5	10	94.0±1.4	94.7±1.2	94.4±1.3
112	80.4±1.8	10	93.9±2.5	93.7±1.5	93.8±2.0
111	78.9±0.9	9	95.0±1.2	95.1±1.2	95.0±1.2

F.5 Tables for 10% weak labelling, single augmentation [10,90]

F.5.1 125×80×2

Table 8.162: Test accuracies of 10% weak labelling, single augmentation training (125 original simulated stereopairs, 80× augmentation) [10,90]

Network Configuration	Average train accuracy of networks (>55%) (%)	Number of successfully trained networks (>55%)	Average majority test accuracy (>55%) (%)	Average minority test accuracy (>55%) (%)	Average test accuracy (>55%) (%)
211	98.4±0.5	10	78.6±2.0	79.9±2.6	79.3±2.3
121	99.3±0.2	10	83.2±2.0	83.4±2.1	83.3±2.0
112	99.2±0.2	10	81.8±4.0	81.5±4.0	81.7±3.9
111	99.0±0.3	9	79.6±4.6	80.1±3.7	79.9±4.0

F.5.2 250×40×2

Table 8.163: Test accuracies of 10% weak labelling, single augmentation training (250 original simulated stereopairs, 40× augmentation) [10,90]

Network Configuration	Average train accuracy of networks (>55%) (%)	Number of successfully trained networks (>55%)	Average majority test accuracy (>55%) (%)	Average minority test accuracy (>55%) (%)	Average test accuracy (>55%) (%)
211	96.6±0.4	10	88.0±2.0	87.9±2.3	88.0±2.1
121	98.1±0.3	10	88.8±2.1	89.4±2.0	89.1±2.0
112	98.1±0.4	10	88.0±2.4	87.5±1.8	87.7±2.1
111	97.8±0.2	10	86.3±1.8	86.7±1.5	86.5±1.6

F.5.3 500×20×2

Table 8.164: Test accuracies of 10% weak labelling, single augmentation training (500 original simulated stereopairs, 20× augmentation) [10,90]

Network Configuration	Average train accuracy of networks (>55%) (%)	Number of successfully trained networks (>55%)	Average majority test accuracy (>55%) (%)	Average minority test accuracy (>55%) (%)	Average test accuracy (>55%) (%)
211	92.8±0.7	10	92.1±1.7	92.3±2.3	92.2±1.9
121	95.6±0.7	10	91.6±1.4	91.5±1.7	91.6±1.5
112	95.5±0.5	10	91.3±1.6	91.8±1.3	91.5±1.4
111	94.8±0.3	8	90.9±1.2	91.7±1.3	91.3±1.3

F.5.4 1000×10×2

Table 8.165: Test accuracies of 10% weak labelling, single augmentation training (1000 original simulated stereopairs, 10× augmentation) [10,90]

Network Configuration	Average train accuracy of networks (>55%) (%)	Number of successfully trained networks (>55%)	Average majority test accuracy (>55%) (%)	Average minority test accuracy (>55%) (%)	Average test accuracy (>55%) (%)
211	89.6±0.3	10	96.3±1.0	95.8±1.0	96.1±1.0
121	91.9±0.7	10	95.4±1.4	95.7±0.7	95.5±1.1
112	91.8±0.5	10	95.1±1.1	94.9±0.9	95.0±1.0
111	90.8±0.6	10	94.7±0.9	95.2±0.8	95.0±0.9

F.5.5 2000×5×2

Table 8.166: Test accuracies of 10% weak labelling, single augmentation training (2000 original simulated stereopairs, 5× augmentation) [10,90]

Network Configuration	Average train accuracy of networks (>55%) (%)	Number of successfully trained networks (>55%)	Average majority test accuracy (>55%) (%)	Average minority test accuracy (>55%) (%)	Average test accuracy (>55%) (%)
211	88.0±0.4	10	98.1±0.9	98.5±0.6	98.3±0.7
121	89.9±0.3	10	98.2±0.7	98.0±0.7	98.1±0.7
112	89.7±0.5	10	98.2±0.7	98.0±0.5	98.1±0.6
111	88.7±0.4	10	98.6±0.7	97.9±0.9	98.2±0.9

F.5.6 5000×2×2

Table 8.167: Test accuracies of 10% weak labelling, single augmentation training (5000 original simulated stereopairs, 2× augmentation) [10,90]

Network Configuration	Average train accuracy of networks (>55%) (%)	Number of successfully trained networks (>55%)	Average majority test accuracy (>55%) (%)	Average minority test accuracy (>55%) (%)	Average test accuracy (>55%) (%)
211	87.5±0.4	10	98.7±0.4	99.1±0.5	98.9±0.5
121	88.9±0.3	10	99.6±0.3	99.7±0.3	99.7±0.3
112	88.8±0.3	9	99.7±0.3	99.6±0.2	99.6±0.3
111	88.1±0.3	8	99.5±0.4	99.4±0.4	99.5±0.4

F.5.7 10000×1×2

Table 8.168: Test accuracies of 10% weak labelling, single augmentation training (10000 original simulated stereopairs, 1× augmentation) [10,90]

Network Configuration	Average train accuracy of networks (>55%) (%)	Number of successfully trained networks (>55%)	Average majority test accuracy (>55%) (%)	Average minority test accuracy (>55%) (%)	Average test accuracy (>55%) (%)
211	87.2±0.6	10	99.2±0.4	98.8±0.8	99.0±0.6
121	88.7±0.3	9	99.7±0.2	99.6±0.3	99.7±0.2
112	88.4±0.4	9	99.7±0.2	99.8±0.2	99.8±0.2
111	88.0±0.3	7	99.6±0.4	99.6±0.3	99.6±0.3

F.6 Tables for 5% weak labelling, single augmentation [10,90]

F.6.1 125×80×2

Table 8.169: Test accuracies of 5% weak labelling, single augmentation training (125 original simulated stereopairs, 80× augmentation) [10,90]

Network Configuration	Average train accuracy of networks (>55%) (%)	Number of successfully trained networks (>55%)	Average majority test accuracy (>55%) (%)	Average minority test accuracy (>55%) (%)	Average test accuracy (>55%) (%)
211	98.8±0.4	10	85.8±4.9	84.7±5.9	85.3±5.3
121	99.5±0.1	10	89.4±3.2	89.2±2.9	89.3±3.0
112	99.5±0.2	10	88.0±2.8	88.0±4.7	88.0±3.8
111	99.4±0.3	8	88.5±3.2	88.1±3.7	88.3±3.3

F.6.2 250×40×2

Table 8.170: Test accuracies of 5% weak labelling, single augmentation training (250 original simulated stereopairs, 40× augmentation) [10,90]

Network Configuration	Average train accuracy of networks (>55%) (%)	Number of successfully trained networks (>55%)	Average majority test accuracy (>55%) (%)	Average minority test accuracy (>55%) (%)	Average test accuracy (>55%) (%)
211	97.8±0.2	10	92.6±1.7	93.1±1.7	92.8±1.7
121	98.9±0.1	10	93.7±1.9	93.9±1.7	93.8±1.8
112	98.8±0.4	10	94.0±1.4	93.8±1.9	93.9±1.6
111	98.6±0.2	10	92.5±2.0	92.6±1.2	92.6±1.6

F.6.3 500×20×2

Table 8.171: Test accuracies of 5% weak labelling, single augmentation training (500 original simulated stereopairs, 20× augmentation) [10,90]

Network Configuration	Average train accuracy of networks (>55%) (%)	Number of successfully trained networks (>55%)	Average majority test accuracy (>55%) (%)	Average minority test accuracy (>55%) (%)	Average test accuracy (>55%) (%)
211	95.7±0.5	10	96.1±0.9	96.1±1.5	96.1±1.2
121	97.8±0.3	10	95.3±1.1	95.2±0.9	95.2±1.0
112	97.5±0.3	10	95.8±1.0	95.2±1.2	95.5±1.1
111	96.8±0.5	10	95.3±1.1	95.2±0.8	95.2±1.0

F.6.4 1000×10×2

Table 8.172: Test accuracies of 5% weak labelling, single augmentation training (1000 original simulated stereopairs, 10× augmentation) [10,90]

Network Configuration	Average train accuracy of networks (>55%) (%)	Number of successfully trained networks (>55%)	Average majority test accuracy (>55%) (%)	Average minority test accuracy (>55%) (%)	Average test accuracy (>55%) (%)
211	94.1±0.5	10	98.0±0.7	98.0±0.7	98.0±0.7
121	95.7±0.3	10	98.2±0.8	98.2±0.7	98.2±0.7
112	95.4±0.3	10	97.9±0.8	97.7±0.7	97.8±0.7
111	94.8±0.4	10	97.7±0.7	97.8±0.6	97.7±0.6

F.6.5 2000×5×2

Table 8.173: Test accuracies of 5% weak labelling, single augmentation training (2000 original simulated stereopairs, 5× augmentation) [10,90]

Network Configuration	Average train accuracy of networks (>55%) (%)	Number of successfully trained networks (>55%)	Average majority test accuracy (>55%) (%)	Average minority test accuracy (>55%) (%)	Average test accuracy (>55%) (%)
211	93.5±0.3	10	99.1±0.3	99.2±0.6	99.2±0.5
121	94.6±0.2	10	99.5±0.3	99.4±0.2	99.4±0.3
112	94.4±0.4	10	99.1±0.4	98.9±0.4	99.0±0.4
111	93.7±0.3	10	99.4±0.4	99.2±0.5	99.3±0.5

F.6.6 5000×2×2

Table 8.174: Test accuracies of 5% weak labelling, single augmentation training (5000 original simulated stereopairs, 2× augmentation) [10,90]

Network Configuration	Average train accuracy of networks (>55%) (%)	Number of successfully trained networks (>55%)	Average majority test accuracy (>55%) (%)	Average minority test accuracy (>55%) (%)	Average test accuracy (>55%) (%)
211	92.8±0.2	10	99.3±0.5	99.5±0.3	99.4±0.4
121	94.2±0.2	10	99.8±0.2	99.9±0.2	99.8±0.2
112	93.8±0.3	10	99.9±0.2	99.7±0.3	99.8±0.2
111	93.0±1.3	8	99.6±0.4	99.8±0.2	99.7±0.4

F.6.7 10000×1×2

Table 8.175: Test accuracies of 5% weak labelling, single augmentation training (10000 original simulated stereopairs, 1× augmentation) [10,90]

Network Configuration	Average train accuracy of networks (>55%) (%)	Number of successfully trained networks (>55%)	Average majority test accuracy (>55%) (%)	Average minority test accuracy (>55%) (%)	Average test accuracy (>55%) (%)
211	92.7±0.4	10	99.6±0.3	99.6±0.5	99.6±0.4
121	94.0±0.3	10	100.0±0.1	99.9±0.1	99.9±0.1
112	93.8±0.2	10	99.9±0.2	99.8±0.2	99.9±0.2
111	93.3±0.3	6	99.8±0.2	99.7±0.3	99.8±0.2

F.7 Tables for 0% weak labelling, single augmentation [10,90]

F.7.1 125×80×2

Table 8.176: Test accuracies of 0% weak labelling, single augmentation training (125 original simulated stereopairs, 80× augmentation) [10,90]

Network Configuration	Average train accuracy of networks (>55%) (%)	Number of successfully trained networks (>55%)	Average majority test accuracy (>55%) (%)	Average minority test accuracy (>55%) (%)	Average test accuracy (>55%) (%)
211	99.3±0.1	10	90.1±3.3	91.0±3.1	90.6±3.2
121	99.7±0.1	10	95.4±1.9	95.1±1.7	95.3±1.7
112	99.6±0.2	10	93.9±2.8	94.1±2.2	94.0±2.4
111	99.5±0.3	10	94.8±2.6	94.7±3.1	94.7±2.8

F.7.2 250×40×2

Table 8.177: Test accuracies of 0% weak labelling, single augmentation training (250 original simulated stereopairs, 40× augmentation) [10,90]

Network Configuration	Average train accuracy of networks (>55%) (%)	Number of successfully trained networks (>55%)	Average majority test accuracy (>55%) (%)	Average minority test accuracy (>55%) (%)	Average test accuracy (>55%) (%)
211	99.1±0.1	10	97.8±1.0	97.7±0.9	97.7±0.9
121	99.7±0.1	10	99.0±0.7	98.8±0.4	98.9±0.6
112	99.6±0.1	9	98.6±0.8	98.5±0.8	98.5±0.8
111	99.4±0.1	10	98.7±0.7	98.8±0.9	98.7±0.8

F.7.3 500×20×2

Table 8.178: Test accuracies of 0% weak labelling, single augmentation training (500 original simulated stereopairs, 20× augmentation) [10,90]

Network Configuration	Average train accuracy of networks (>55%) (%)	Number of successfully trained networks (>55%)	Average majority test accuracy (>55%) (%)	Average minority test accuracy (>55%) (%)	Average test accuracy (>55%) (%)
211	98.9±0.3	10	99.1±0.7	99.3±0.5	99.2±0.6
121	99.6±0.1	10	99.8±0.2	99.9±0.2	99.8±0.2
112	99.5±0.1	10	99.8±0.3	99.6±0.3	99.7±0.3
111	99.2±0.1	9	99.6±0.5	99.4±0.4	99.5±0.5

F.7.4 1000×10×2

Table 8.179: Test accuracies of 0% weak labelling, single augmentation training (1000 original simulated stereopairs, 10× augmentation) [10,90]

Network Configuration	Average train accuracy of networks (>55%) (%)	Number of successfully trained networks (>55%)	Average majority test accuracy (>55%) (%)	Average minority test accuracy (>55%) (%)	Average test accuracy (>55%) (%)
211	98.7±0.1	10	99.7±0.3	99.6±0.2	99.7±0.2
121	99.5±0.1	10	100.0±0.1	100.0±0.1	100.0±0.1
112	99.4±0.1	10	99.9±0.1	99.9±0.3	99.9±0.2
111	99.0±0.1	10	99.8±0.2	99.8±0.2	99.8±0.2

F.7.5 2000×5×2

Table 8.180: Test accuracies of 0% weak labelling, single augmentation training (2000 original simulated stereopairs, 5× augmentation) [10,90]

Network Configuration	Average train accuracy of networks (>55%) (%)	Number of successfully trained networks (>55%)	Average majority test accuracy (>55%) (%)	Average minority test accuracy (>55%) (%)	Average test accuracy (>55%) (%)
211	98.5±0.1	10	99.6±0.5	99.7±0.3	99.6±0.4
121	99.4±0.1	10	100.0±0.1	100.0±0.0	100.0±0.0
112	99.3±0.2	10	99.9±0.3	100.0±0.1	99.9±0.2
111	98.9±0.2	8	99.9±0.2	100.0±0.0	99.9±0.1

F.7.6 5000×2×2

Table 8.181: Test accuracies of 0% weak labelling, single augmentation training (5000 original simulated stereopairs, 2× augmentation) [10,90]

Network Configuration	Average train accuracy of networks (>55%) (%)	Number of successfully trained networks (>55%)	Average majority test accuracy (>55%) (%)	Average minority test accuracy (>55%) (%)	Average test accuracy (>55%) (%)
211	98.5±0.2	10	99.9±0.1	99.9±0.2	99.9±0.2
121	99.5±0.1	10	100.0±0.0	100.0±0.0	100.0±0.0
112	98.8±1.3	10	99.9±0.3	99.9±0.3	99.9±0.3
111	98.8±0.1	8	99.9±0.1	100.0±0.1	99.9±0.1

F.7.7 10000×1×2

Table 8.182: Test accuracies of 0% weak labelling, single augmentation training (10000 original simulated stereopairs, 1× augmentation) [10,90]

Network Configuration	Average train accuracy of networks (>55%) (%)	Number of successfully trained networks (>55%)	Average majority test accuracy (>55%) (%)	Average minority test accuracy (>55%) (%)	Average test accuracy (>55%) (%)
211	98.4±0.2	10	99.8±0.3	99.7±0.3	99.8±0.3
121	99.4±0.1	10	100.0±0.1	100.0±0.1	100.0±0.1
112	99.2±0.1	10	99.9±0.1	100.0±0.1	100.0±0.1
111	98.7±0.2	7	99.9±0.1	100.0±0.1	100.0±0.1

G Illustrated Acknowledgements

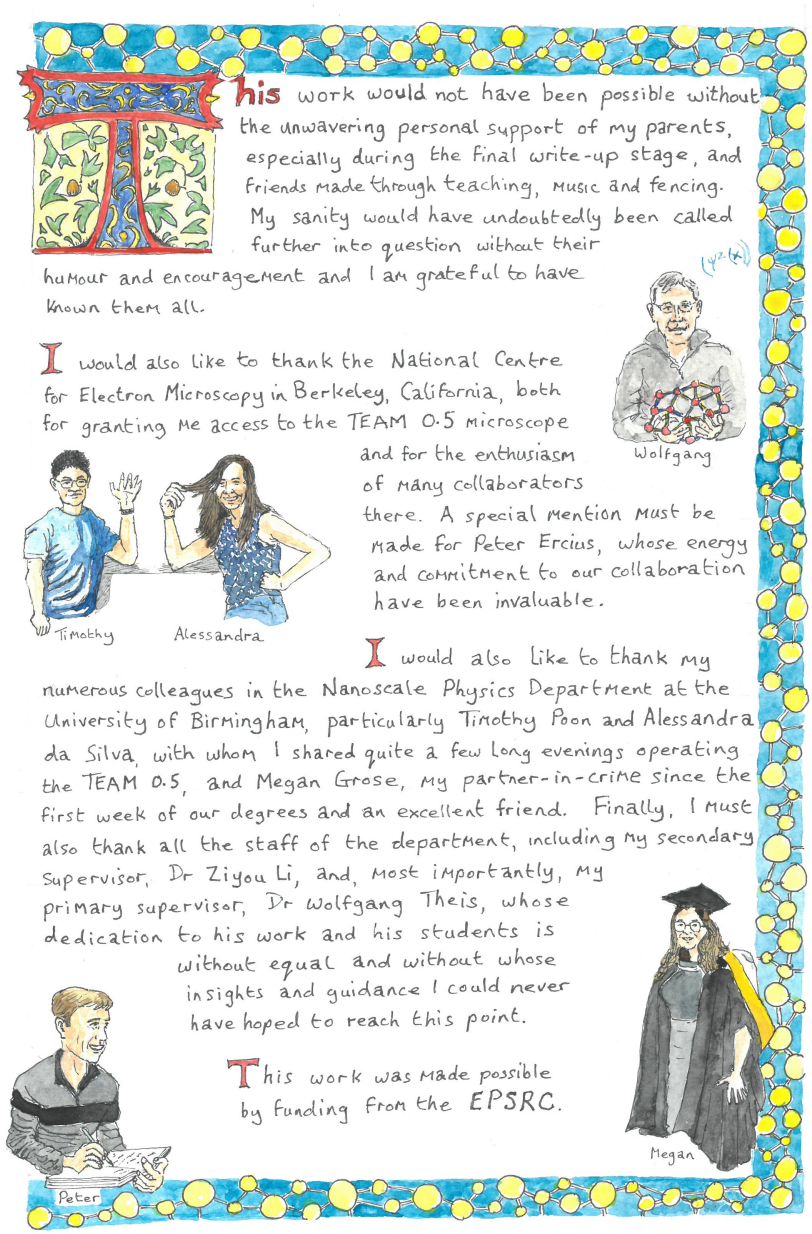


Figure 8.2: Illustrated acknowledgements. Text by author, illustrations courtesy of Dr Michael Pattison.



Courtesy of Dr Michael Pattison