MARK FRICKER

FUNGAL NETWORK ANALYSIS

PLANT SCIENCES, OXFORD
FIRST EDITION

PUBLISHED BY PLANT SCIENCES, OXFORD
FIRST EDITION

First Edition, April 2021

Current version, August 2021

Contents

Acknowledgements

This work was supported by a Visiting Fellowship to MDF at The Institute of Advanced Studies in Durham, The Leverhulme Trust (RPG-2015-437), The Human Frontier Science Program (RGP0053/2012), and The Oxford-Berlin Seedcorn Fund

The FungalNetworkAnalysis package is based on the phase congruency analysis originally developed for fungal and slime mold networks with Boguslaw Obara, University of Durham, available at:
http://community.dur.ac.uk/boguslaw.obara/research/software/

This package also uses the following matlab implementations:

Bioformats package for file import:
http://www.openmicroscopy.org/site/products/bio-formats

The phase congruency and skeleton extraction edgelink program by Peter Kovesi, University of Western Australia:
http://www.peterkovesi.com/matlabfns/

The anisotropic second-order Gaussian kernels by Carlos Lopez-Molina, Universidad Pública de Navarra, Pamplona:
http://www.kermit.ugent.be/software.php?navigatieId=0&categorieId=17

The Frangi 'Vesselness' filter by Marc Schrijver and D. Kroon, University of Twente:
https://uk.mathworks.com/matlabcentral/fileexchange/24409-hessian-based-frangi-vesselness-filter

The Matlab implementation of the Boost Graph Library by David Gleich:
https://uk.mathworks.com/matlabcentral/fileexchange/10922-matlabbgl

The Bresenham line drawing algorithm written by Aaron Wetzler, Haifa Technion, Israel:
https://uk.mathworks.com/matlabcentral/fileexchange/28190-bresenham-optimized-for-matlab

The CircStat toolbox writeen by Philipp Berens and Marc J. Velasco, Tuebingen:
P. Berens, CircStat: A Matlab Toolbox for Circular Statistics, Journal of Statistical Software, Volume 31, Issue 10, 2009. http://www.jstatsoft.org/v31/i10
https://uk.mathworks.com/matlabcentral/fileexchange/10676-circular-statistics-toolbox-directional-statistics

The export_fig package written by Oliver Woodford and Yair Altman:
https://uk.mathworks.com/matlabcentral/fileexchange/23629-export-fig

The screen capture package written by Yair Altman (2020):.
https://www.mathworks.com/matlabcentral/fileexchange/24323-screencapture-get-a-screen-capture-of-a-figure-frame-or-component

The colorcet.m package written by Peter Kovesi and downloaded from:
https://peterkovesi.com/projects/colourmaps/

Peter Kovesi. Good Colour Maps: How to Design Them. arXiv:1509.03700 [cs.GR] 2015
https://arxiv.org/abs/1509.03700

1
Installation

1.1 Overview

The code to run the network extraction and analysis is provided as open source under a GNU General Public License v3.0 from:

`https://zenodo.org`

The code is provided in two alternative formats:

- A MATLAB® app with a GUI interface that can be installed in MATLAB® on any platform with a screen resolution of 1600 x 900 or greater. Installation requires MATLAB® release 2020b or later, and the following Toolboxes:
  - Image Processing Toolbox™
  - Signal Processing Toolbox™
  - Mapping Toolbox™
  - Statistics and Machine Learning Toolbox™
  - Wavelet Toolbox™
  - Curve Fitting Toolbox™
  - Bioinformatics Toolbox™
  - Computer Vision Toolbox™
- A compiled standalone version that can be installed and run on Windows 10, 64 bit platforms and requires a screen resolution of 1600 x 900 or greater;

1.2 Installation of the MATLAB app

The MATLAB® app installer file, *FungalNetwork_extraction.mlappinstall*, contains everything necessary to install and run the App within the MATLAB® environment, including the source code, supporting data, information (such as product dependencies), and the app icon (Fig. 1.1).

Clicking on the *FungalNetwork_extraction.mlappinstall* link will initiate download of the app. Clicking the downloaded file should launch MATLAB® if it is not already running, and install the app in the app toolbar (Fig. 1.2). Once installation is complete, the program can be run by clicking on the icon in the toolbar.

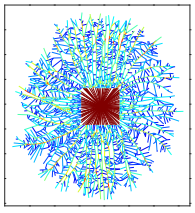


Figure 1.1: The program icon

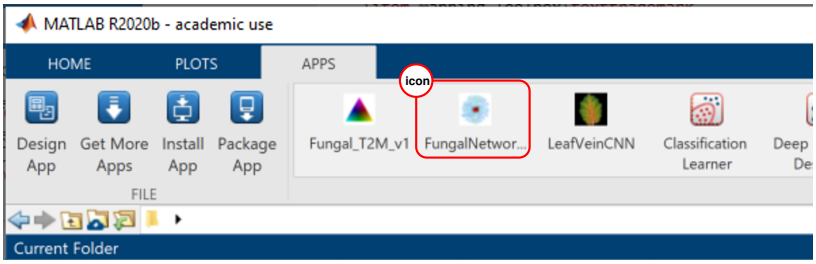


Figure 1.2: Location of the FungalNetwork app and icon in the APPS menu within MATLAB®

1.3 *Installation of the stand-alone program (Windows only)*

The software has been tested on Windows 10, and requires a minimum screen resolution of 1600 x 900. In addition, an appropriate version of the MATLAB® Compiler Runtime (MCR) is required to install the set of shared libraries that enable execution of the compiled MATLAB® application. The MCR should automatically download from the MathWorks website when the program is installed for the first time. Alternatively MCR can be downloaded from:

`http://www.mathworks.com/products/compiler/mcr`.

To install the MCR and standalone package, click the *FungalNetwork_extraction.exe* link to initiate the download. This may take several minutes as the package also includes the trained CNN networks and is around 1GB in size. Once download is complete, click the downloaded file to install the compiled MATLAB self-extracting *FungalNetwork_extraction.exe* program.

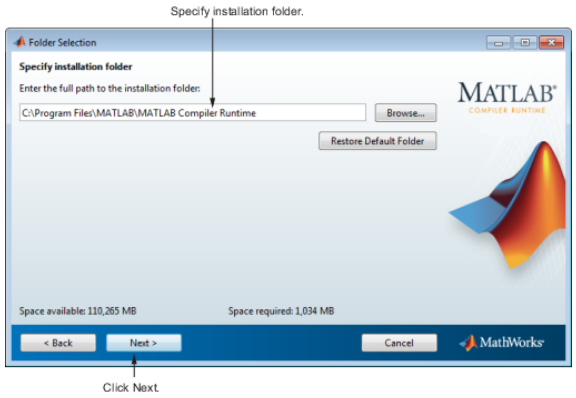
This extracts the MATLAB® Runtime Installer from the archive, along with all the files that make up the deployed MATLAB® environment. Once all the files have been extracted, the MATLAB® Runtime Installer starts automatically. When the MATLAB® Runtime Installer starts, it displays the following dialog box:

Read the information and then click **Next** to proceed with the installation.

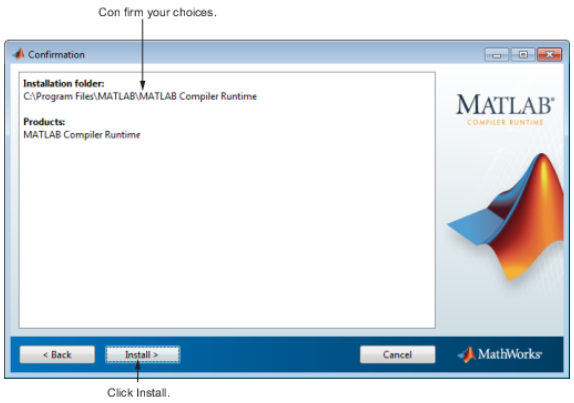
Specify the folder in which you want to install the MATLAB® runtime in the Folder Selection dialog box and click **Next**. Note on Windows systems, you can have multiple versions of the MATLAB®

runtime on your computer, but only one installation for any particular version. If you already have an existing installation, the MATLAB® runtime Installer does not display the Folder Selection dialog box because you can only overwrite the existing installation in the same folder.

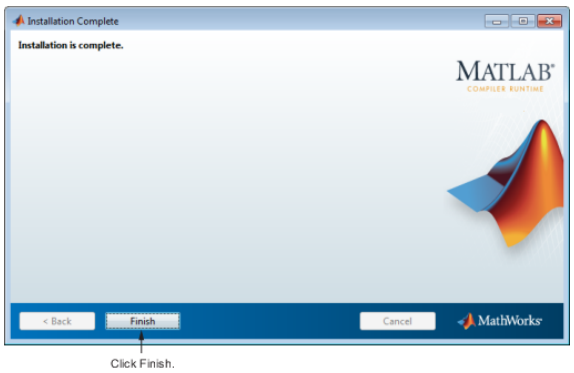
It is recommended to keep the default settings as this ensures the path to other program files is set automatically.



Confirm your choices and click **Install**. The MATLAB® Runtime Installer starts copying files into the installation folder. Installation takes about 10 minutes.



Click **Finish** to exit the installer.



MATLAB® Runtime Installer Readme File: A readme.txt file is included with the MATLAB Runtime Installer. This file, visible

when the MATLAB Runtime Installer is expanded, provides more detailed information about the installer and the switches that can be used with it.

1.4 *Installation of additional program files needed*

A number of additional files needed to provide high quality output need to be installed at the same time as the main program. The latest version of Java needs to be installed, and is available from:

`http://www.java.com/en/`

Output of images at full resolution uses *export_fig.m* originally written by Oliver Woodford (2008-2014) and now maintained by Yair Altman (2015-). This is included in the installation. However, when exporting to vector format (PDF or EPS) this function requires that ghostscript is installed on your system. Ghostscript can be downloaded from:

`http://www.ghostscript.com`.

When exporting images to eps and pdf formats, *export_fig* additionally requires pdftops, from the Xpdf suite of functions. This is included in the xpdf tools package and can be downloaded from:

`https://www.xpdfreader.com/download.html`

2

The Fungal Network Analysis GUI

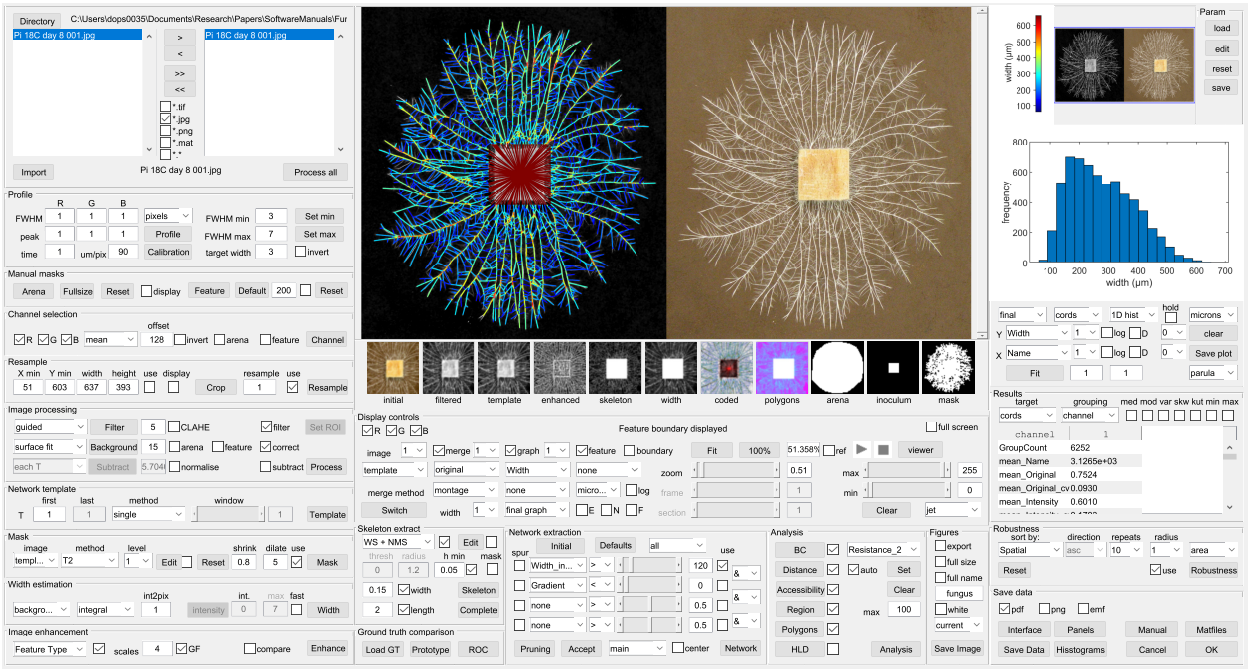


Figure 2.1: The fungal network analysis program

2.1 *The principle of fungal network analysis*

This manual describes a high-throughput automated image analysis system to detect and characterize large complex mycelial networks (Fig. 2.1). Images of fungal mycelia cab be loaded into a graphical user interface (GUI) and processed through a standard pipeline that includes noise reduction, background correction, network enhancement, segmentation, skeletonization, width estimation and network graph representation. As most biological networks have filaments, tubes or vessels that span a range of scales, with differing contrast and noise levels depending on the imaging method, we provide a number of different curvilinear feature enhancement methods, segmentation and skeletonisation algorithms to extract the network structure as a single-pixel wide binary skeleton.

Our preferred approach uses an intensity-independent edge enhancement technique using phase congruency developed by Peter Kovesi[1] and described in more detail in Chapter 9, along with the other enhancement techniques for comparison. Although the approaches described here were originally developed for fungal mycelial[2], we have applied the same approaches to other planar 2-D curvi-linear structure, including sub-cellular networks, such as the endoplasmic reticulum (ER)[3], or other macroscopic networks such as slime molds[4] or leaf venation patterns, although we have found in the latter case performance can be improved using deep learning techniques[5].

One of the most critical features of network analysis is to ensure that parts of the network do not become accidentally disconnected during extraction, as this may have a major impact on connectivity metrics, or prediction of functional flows on the network, and robustness measurements. Thus the approach used here initially over-segments the network to ensure connectivity, calculates an initial graph representation and then prunes the individual edges using a combination of metrics. The pruned skeleton is then re-converted to a graph representation, with nodes at the junctions or branch points, linked by edges that capture the curvi-linear structures.

Once in a graph format, a wide range of graph theoretic measures can be calculated (see for example Figure 2.2), or exported to other graph analysis packages such as iGraph in R. Alternatively, the graph can be used as the input to mathematical models that calculate flows on the network[6,7].

[1] P. Kovesi. Image features from phase congruency. *Videre: Journal of Computer Vision Research*, 1:1–26, 1999

[2] B. Obara, V. Grau, and M. D. Fricker. A bioimage informatics approach to automatically extract complex fungal networks. *Bioinformatics*, 28:2374–81, 2012

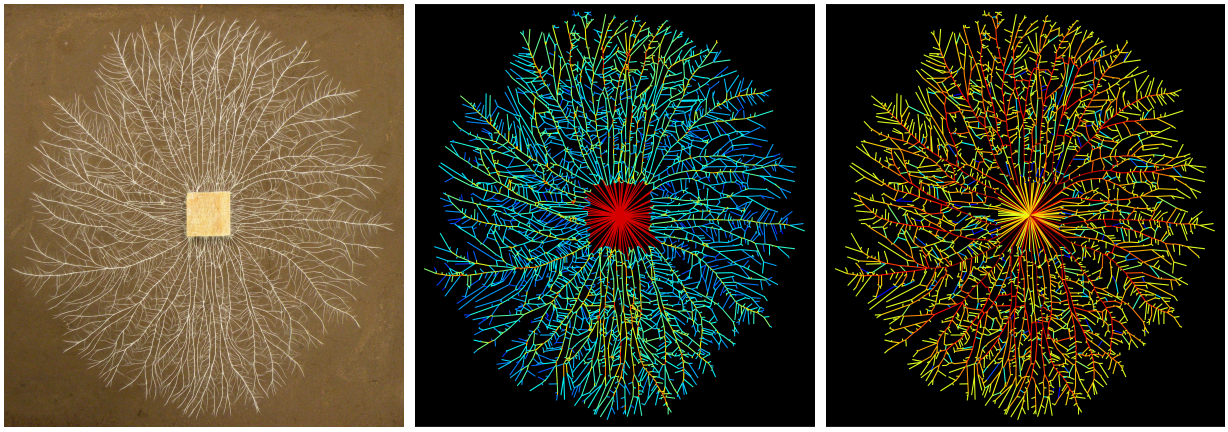
[3] C. Pain, V. Kriechbaumer, M. Kittelmann, C. Hawes, and M. Fricker. Quantitative analysis of plant er architecture and dynamics. *Nature Communications*, 10:984, 2019

[4] M.D. Fricker, D. Akita, L.L.M. Heaton, N. Jones, B. Obara, and T. Nakagaki. Automated analysis of physarum network structure and dynamics. *J. Phys. D*, 50:254005, 2017

[5] H. Xu, B. Blonder, M. Jodra, Y. Malhi, and M. Fricker. Automated and accurate segmentation of leaf venation networks via deep learning. *New Phytologist*, 229:631–648, 2020

[6] L.L.M. Heaton, E. López, P.K. Maini, M.D. Fricker, and N.S. Jones. Growth-induced mass flows in fungal networks. *Proc. R. Soc. B*, 277:3265–3274, 2010

[7] L.L.M. Heaton, E. López, P.K. Maini, M.D. Fricker, and N.S. Jones. Advection, diffusion, and delivery over a network. *Phys Rev E*, 86:021905, 2012



(a) fungal mycelium

(b) extracted network, colour coded by edge thickness

(c) network coded by log betweenness centrality

Figure 2.2: Typical network extraction from a fungal mycelium of *Phallus impudicus* grown on compressed soil.

2.2 *Overview of the main interface*

The main interface (Fig. 2.1) sets out the workflow for network extraction and analysis. Starting at the top-left, the processing sequence moves vertically down and then horizontally across, with

key stages of the process separated out into individual panels. Typically, the button at the right-hand bottom of each panel runs the processing steps for that stage, using the parameter settings given by the controls to the left. If there are multiple steps in the stage, each one can be run individually using its own specific button. Depending on the methods chosen, some controls are unavailable and will be greyed out. The images produced at each stage can be selected and inspected in the main display window, whilst the results for any of the parameters can be displayed in the plotting window. Results are saved in Excel format, whilst images and graphs can be exported in pdf, png or emf format. Perhaps the most useful button is the **Process all** button in the top-left hand panel, which will run through all the images selected using the corresponding set of saved parameters, and then save all the data.

2.3 *Saving and editing parameter settings*

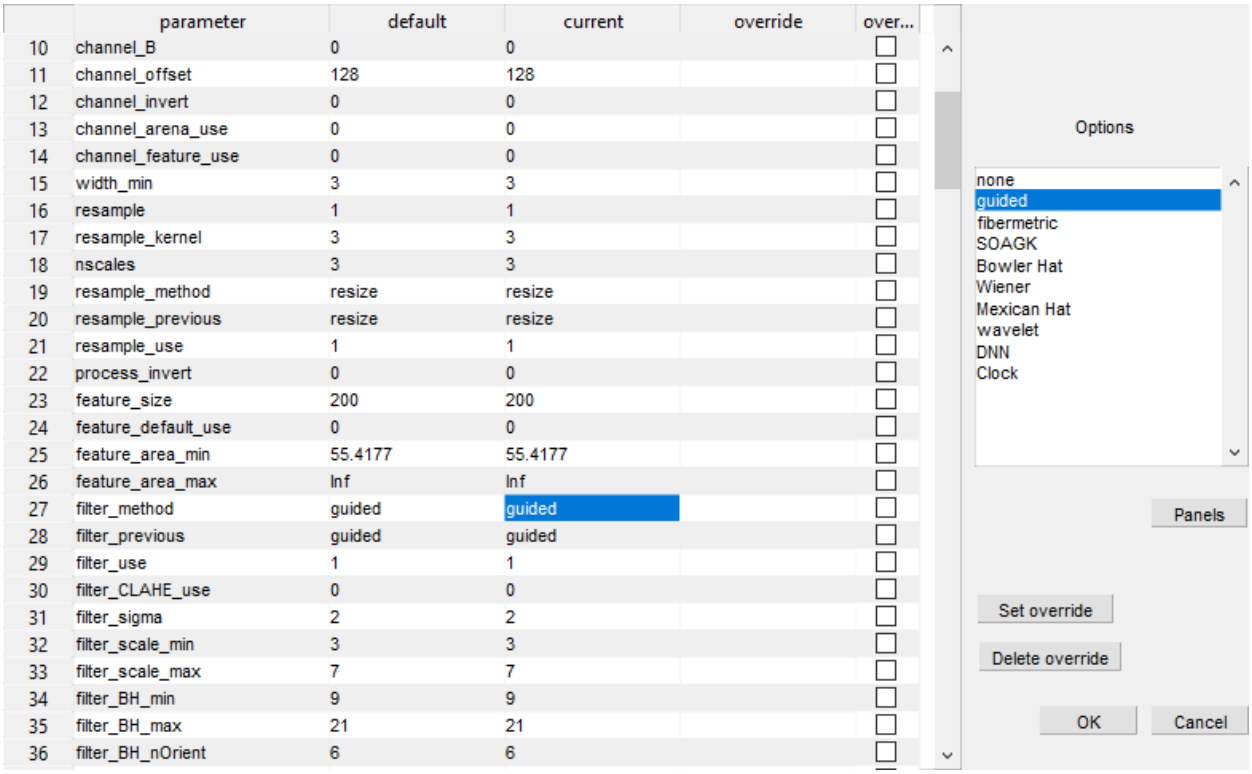
A number of key parameters for each method are accessible from the main interface. Any changes to these parameters are automatically saved as a *"filename_param.mat"* file in the *"..\processed data\parameters"* directory. The next time the image is loaded the saved parameters are re-applied. These parameter files can also be stored with the images to ensure complete transparency and reproducibility of the data analysis pipeline.

Figure 2.3: Controls to load, edit and save all the parameters

For several of the methods there are additional parameters that have default settings that are unlikely to be changed. However, these values can be edited using the **Edit** button in the **Param** panel (Fig. 2.3) in the top-right corner of the GUI. This opens up a table showing the default setting and the current choice for all the underlying parameters (Fig. 2.4). If the parameter selected has a number of different options set by a drop-down menu, these are displayed in the **options** box alongside. The value of the parameter can be changed by typing the new value into the current column, or selecting the option from the list box if it is a string variable.

Clicking the **OK** button returns to the main interface and updates the parameters. The **Reset** button will return all parameters to their default value. This should be used with care if considerable time has already been spent tailoring parameters to a particular image.

Adjacent to the **current** column, there is a column marked **override**. Values can be entered in this column that are distinct from either the default or current value, and will be used instead of the set value once the **Set override** button is pressed. These override values are saved to a *"override.mat"* file in the current directory, and will be applied to any image processed from that directory. This provides a straightforward way of testing out the effect of a different set of parameters across all images in an experiment. The **Delete override** button removes the saved *"override.mat"* (or it can be manually deleted from the directory) and processing will then



	parameter	default	current	override	over...
10	channel_B	0	0		
11	channel_offset	128	128		
12	channel_invert	0	0		
13	channel_arena_use	0	0		
14	channel_feature_use	0	0		
15	width_min	3	3		
16	resample	1	1		
17	resample_kernel	3	3		
18	nscales	3	3		
19	resample_method	resize	resize		
20	resample_previous	resize	resize		
21	resample_use	1	1		
22	process_invert	0	0		
23	feature_size	200	200		
24	feature_default_use	0	0		
25	feature_area_min	55.4177	55.4177		
26	feature_area_max	Inf	Inf		
27	filter_method	guided	guided		
28	filter_previous	guided	guided		
29	filter_use	1	1		
30	filter_CLAHE_use	0	0		
31	filter_sigma	2	2		
32	filter_scale_min	3	3		
33	filter_scale_max	7	7		
34	filter_BH_min	9	9		
35	filter_BH_max	21	21		
36	filter_BH_nOrient	6	6		

Figure 2.4: Interface to edit any hidden parameters

revert to the current values.

In some instances it is desirable to update the current settings to the override values for some or all of the parameters modified. This can be achieved by ticking the corresponding **overwrite** checkbox. Any image file processed from this directory will now have the corresponding parameter values over-written.

An alternative approach is to save a particular set of parameters using the **Save** button in the main interface, which prompts the user to provide a different filename. These settings can be re-loaded from this file and applied to any another image using the **Load** button. It should be noted that certain settings, such as the position of regions-of-interest (ROIs), remain specific to a particular experiment, and are not modified using controls in the parameter editing window.

3
Loading images

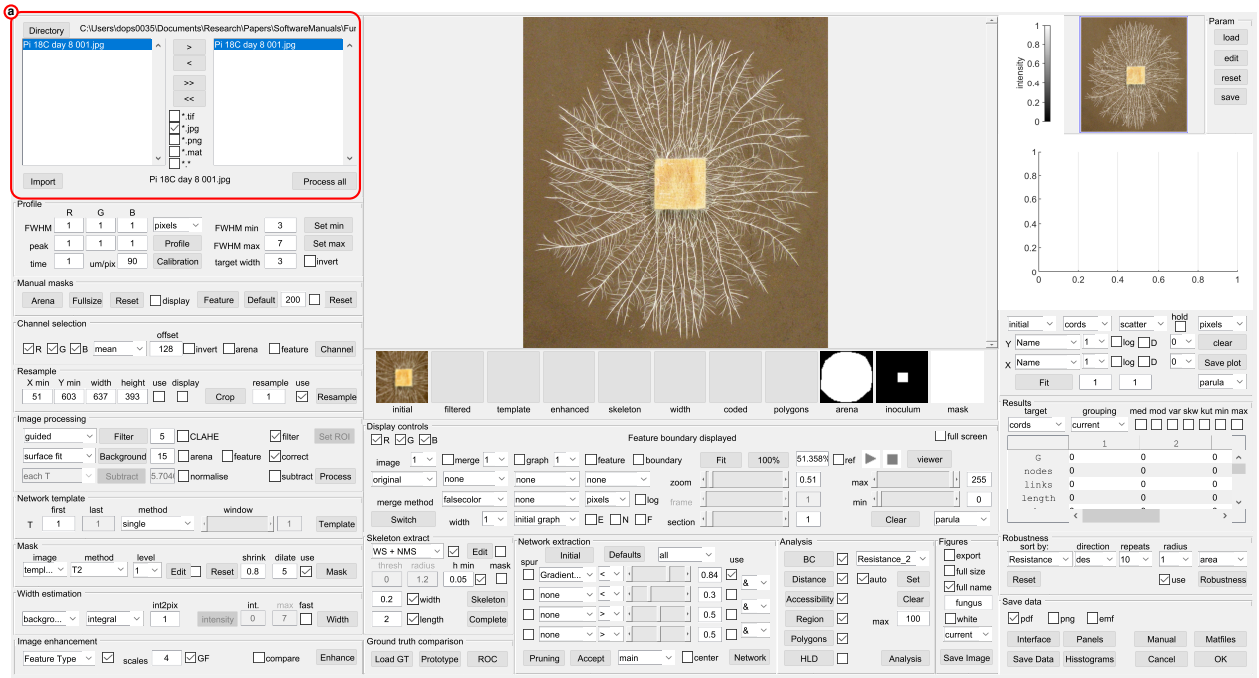


Figure 3.1: (a) The Image load panel on the main interface

3.1 The Image load panel

The Fungal Network GUI automatically displays the name of any image files the default ***.jpg** extension present in the current directory, in the left-hand list box with when the program is started (Fig. 3.2). The working directory can be changed using the **Directory** button, and additional file types can be displayed using the ***.tif**, ***.png** or ***.*** checkboxes.

A single image can be selected for processing in the left-hand list box using a left mouse-click to highlight the name in the list followed by clicking the **>** arrow, or by double-clicking the name. Multiple files in any combination can be selected using *Ctrl + left click* to highlight the files, followed by the **>** arrow. Alternatively, all the files can be selected with the **>>** arrow and will appear in the

It should be noted that whatever image format is used, images should be saved with no compression or lossless compression

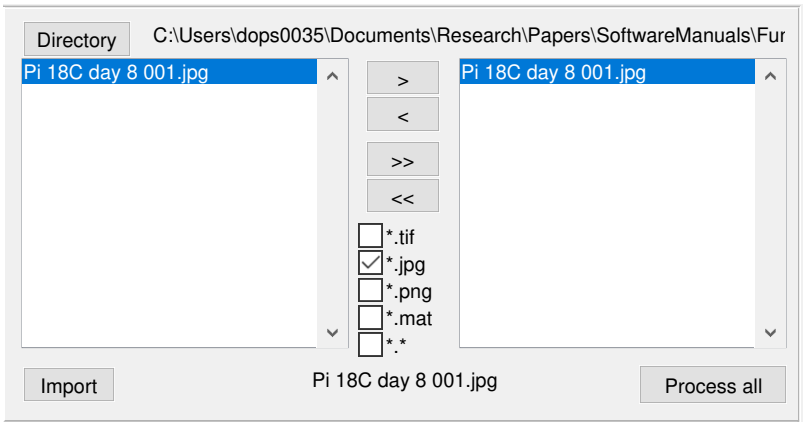


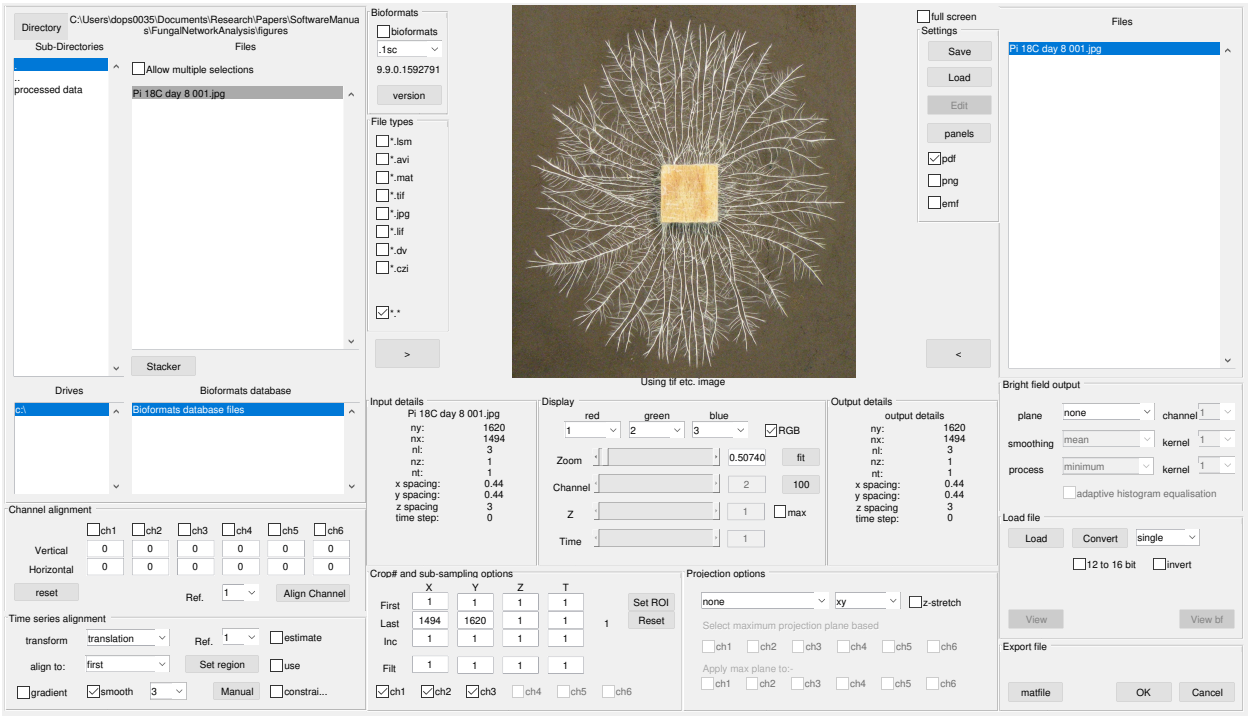
Figure 3.2: Image load panel: Images in a range of file formats can be imported for processing.

right-hand list box. Individual files, or all the files can be removed using the **<** and **<<** arrows, respectively.

Many different file-types can be opened directly using the Bio-Formats program[1] called by the load function. Alternatively, greater control of the format of the imported image is available through the **Import** button. This opens up a new window to load, align, crop and filter the image, prior to network analysis, and is described in Chapter 15.

[1] M. Linkert, C. T. Rueden, C. Allan, J. M. Burel, W. Moore, A. Patterson, B. Loranger, J. Moore, C. Neves, D. Macdonald, A. Tarkowska, C. Sticco, E. Hill, M. Rossner, K. W. Eliceiri, and J. R. Swedlow. Metadata matters: access to image data in the real world. *J Cell Biol*, 189:777–82, 2010

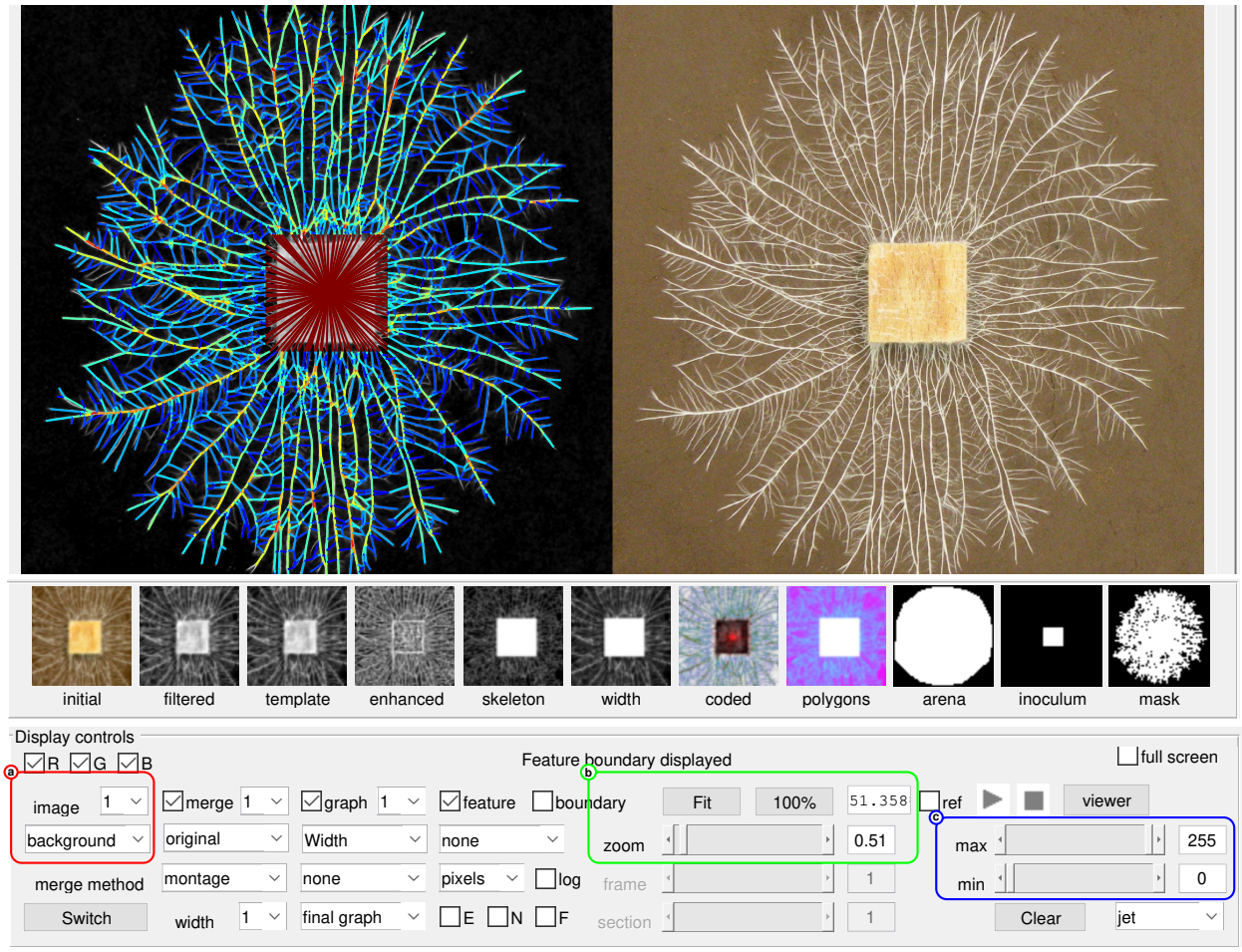
Figure 3.3: The Image import GUI described in Chapter 15



3.2 *Image display controls*

Once a filename is displayed in the right hand list box, the image is automatically loaded and displayed in the main window (Fig. 3.4). At the same time, a thumbnail for the *initial* image is also displayed in the **thumbnail** shortcut bar immediately underneath the main display. In addition, if the *arena* and *inoculum* binary images have already been defined, these are loaded in and the corresponding thumbnails displayed in the **thumbnail** bar (Fig. 3.4). Thumbnails for other key steps in the processing sequence are displayed once the appropriate step has completed successfully, and can be used subsequently to switch rapidly between different images. Alternatively, all the images available for display can be accessed from the **image** drop down menu (Fig. 3.4(a)). The adjacent drop down menu is used to select the channel for display, but should be left a 1 for the majority of processing described here.

Figure 3.4: Main image display window, thumbnail shortcut bar and associated controls to select the image for display (a), adjust the zoom (b), or image contrast (c)



If multiple files have been selected for processing, the last one in the sequence is shown. Any image can be viewed by clicking on it's filename in the right-hand list box.

A number of options to adjust the image displayed are available in the **Display controls** panel. The other controls that are relevant

at this stage adjust the image size and contrast, through the **zoom** (Fig. 3.4(b)), **white level** and **black level** sliders on the right of the display controls panel (Fig. 3.4(c)). The **Fit** button resizes the image to ensure all of it is visible, whilst the **100%** button gives a 1:1 image:screen pixel scaling. The other commands in the **Display controls** panel will be covered at a later stage in the manual.

4
Profile measurements

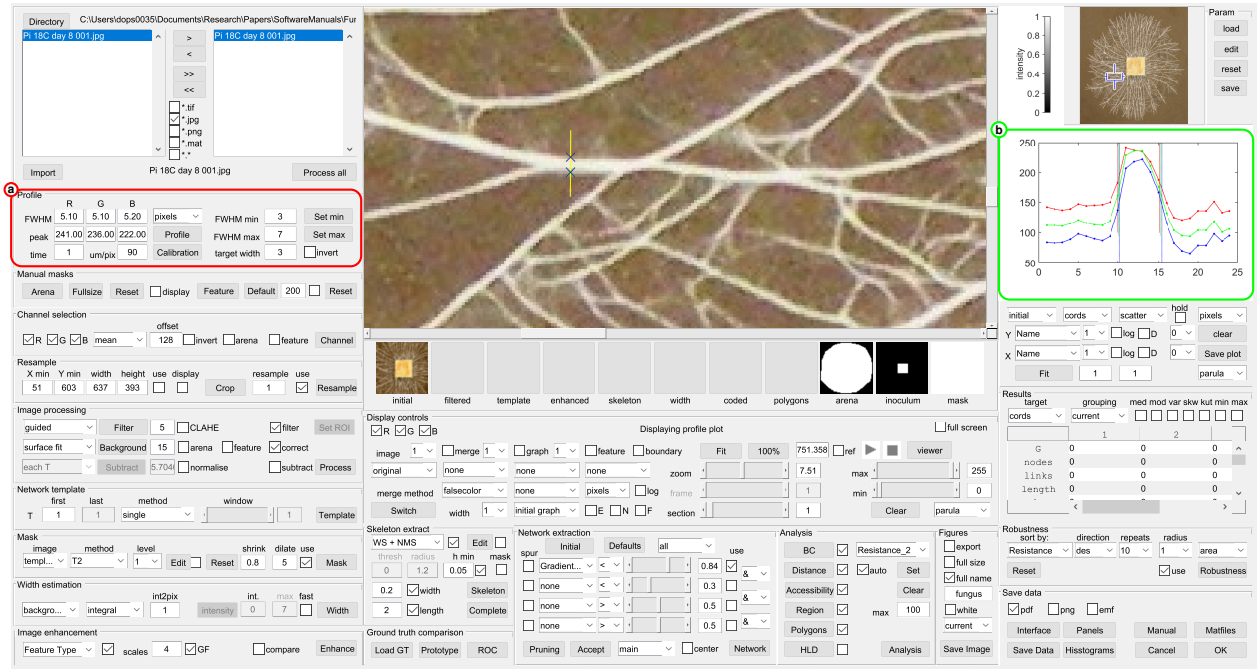


Figure 4.1: (a) Controls for profile measurements. (b) Profile plot

Images of the mycelium may have been collected at different pixel resolutions, depending on the camera or microscope settings, and may span different width scales depending on the species and growth conditions.

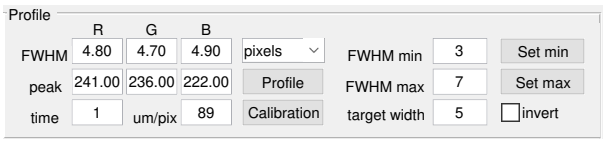


Figure 4.2: Image profile controls: these allow the user to measure the physical size of structures in the image from transects that automatically calculate the full-width half-maximum (FWHM) of the underlying object. These are used to estimate the minimum and maximum widths of the hyphae or cords to standardise all subsequent processing steps

To standardise all the subsequent processing steps, it is useful to define the expected minimum and maximum width of the hyphae or cords manually using a transect drawn on the image, using the **Set min** and **Set max** buttons in the **Profile** panel (Fig. 4.2). The image can be subsequently resampled to ensure that the minimum hypha/cord width is scaled to the width set in the **target** textbox.

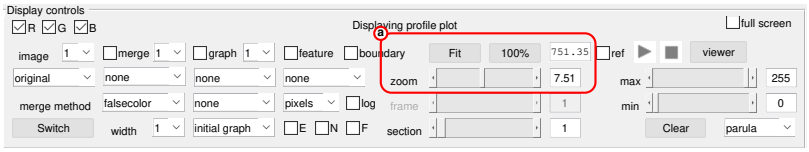


Figure 4.3: Zoom controls in the Display panel

It is helpful to zoom into the image using the **zoom** slider in the **Display controls** panel (Fig. 4.3) so that individual cords or hyphae are clearly visible in the main image window (Fig. 4.4). The current region is also shown in the image overview panel as a blue and white rectangle (Fig. 4.5). The region of the image can be changed using the scroll bars on the main display, or by dragging the box in the overview window.

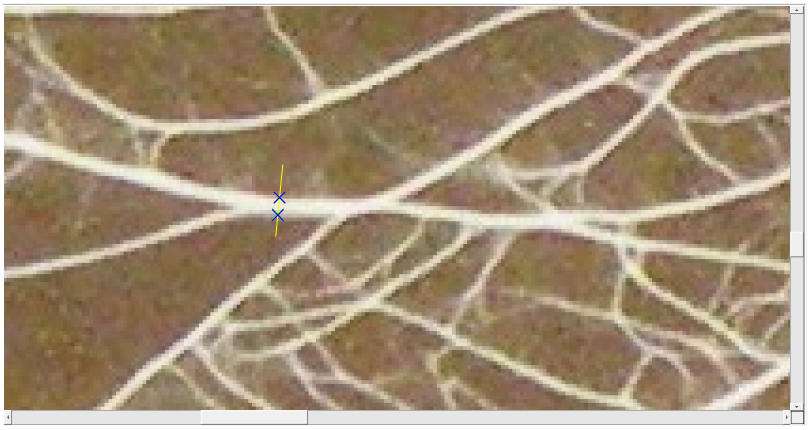


Figure 4.4: Zoomed in image in main display window

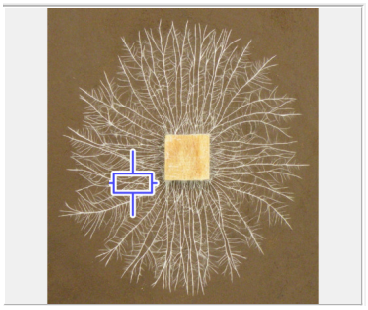


Figure 4.5: Zoomed region shown in the overview panel

When either the **Set min** or **Set max** buttons are clicked in the **Profile** panel, the user is prompted to draw a two-point transect on the image across a cord that, by eye, appears to be close to the smallest or largest hypha/cord width, respectively. On completion of the second mouse click, the line is plotted on the image (Fig. 4.4), and a graph of the transect is displayed in the **graph** panel (Fig. 4.6), with the line colour reflecting the intensity values of the original RGB channels.

In addition, the full-width at half-maximum (FWHM) peak intensity is automatically calculated and displayed as:

- two dotted vertical lines on the graph;
- the estimated pixel width in the **FWHM min** and **FWHM max** text boxes, respectively;
- the estimated pixel width and peak intensity values in the **FWHM** and **peak** text boxes for the **R**,**G** and **B** channels;
- a yellow line on the image with markers indicating the FWHM for the sampled hypha

Values for *FWHM min* and *FWHM max* are always given in pixels. However, the units displayed for FWHM for the individual RGB

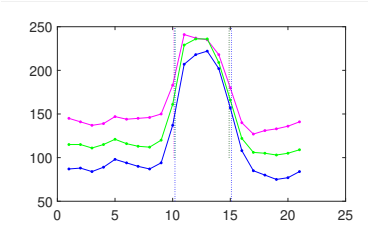


Figure 4.6: Profile measurements: the graph shows the intensity profile along a user-defined transect drawn on the image, along with the full-width half-maximum (FWHM) automatically calculated from the peak height for each channel

channels can be changed using the adjacent drop-down menu. The peak intensities are given in intensity units, ranging from 0 to 255 for an 8-bit image. Additional profiles can be drawn at any stage without updating the **FWHM min** and **FWHM max** text boxes by using the **Profile** button.

The value of $FWHM_{min}$ is used to calculate a resampling factor needed to ensure that the minimum apparent hyphal width matches the **target** width (typically 5 pixels wide, depending on the application) to reduce pixelation errors later on. Likewise, $FWHM_{max}$ is used to determine the number of scales to use in the subsequent image enhancement steps to ensure both that the largest hyphae/cords are correctly segmented, and also that structures above this limit are excluded. As these scaling factors are approximate, it is sufficient to round the values of $FWHM_{min}$ down, and $FWHM_{max}$ up to convenient integer values (usually odd numbers to ensure that the processing kernels are centered on the pixel of interest).

The **Calibration** button prompts the user to define the physical scale between two measurement points on the image, which then updates the adjacent textbox to give the pixel spacing in **micron per pixel**. Alternatively, the pixel size can be entered manually in the text box, if the information is available from the original image file. Likewise, the time interval between frames in time-series images can be entered in the **time** text box.

5
Manually defined masks

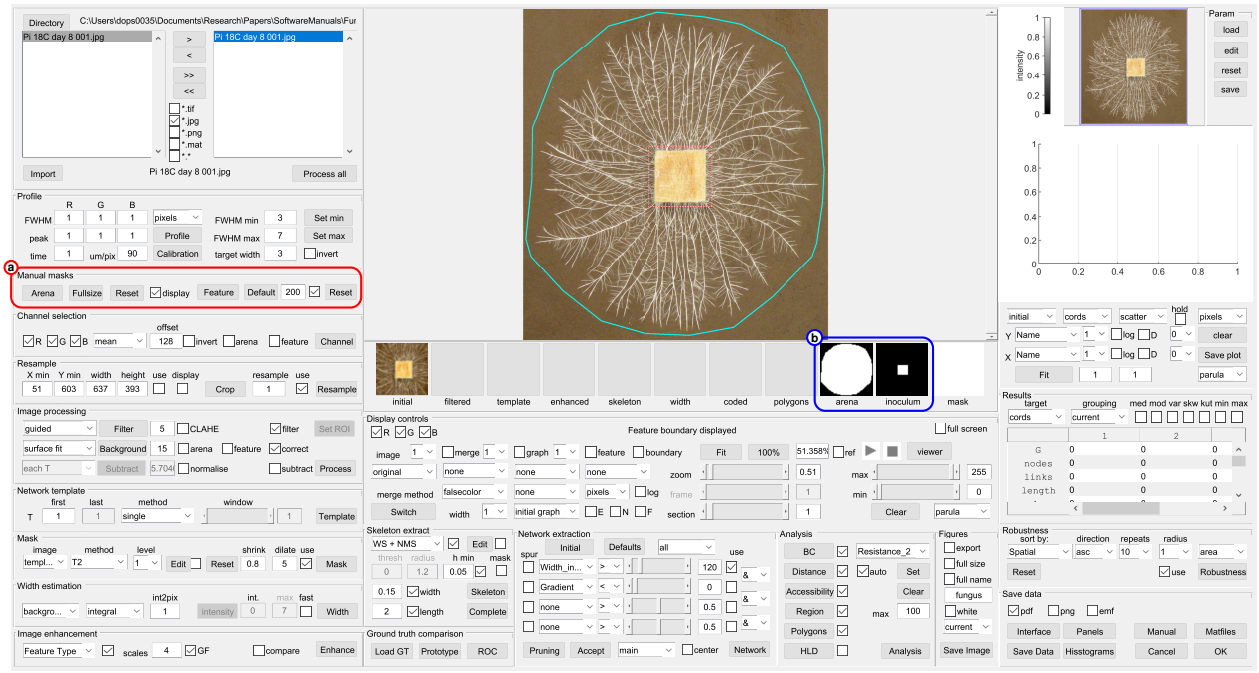


Figure 5.1: defining the arena and inoculum (a), and the corresponding icons in the thumbnail bar (b)

To improve subsequent processing steps, it is useful to define an *arena* around the colony that excludes extraneous features, such as the edge of the culture dish for example. In addition, it is not usually possible to define the network within the *inoculum*, so this has to be treated somewhat differently to the growing mycelium. Our approach is to represent the network within the inoculum as a single node that is well connected to all hyphae/cords that intersect the boundary. Whilst it is possible to use automatic segmentation to define the inoculum, in practice we have found it quicker and easier to define it manually. In addition, rather than constrain the boundary to exactly fit the inoculum, it is better to increase it by some margin to ignore the dense, fine growth that often occurs immediately adjacent to the inoculum that cannot be segmented cleanly. We also recommend excluding shadowed regions next to wood block inocula that can interfere with width estimation.

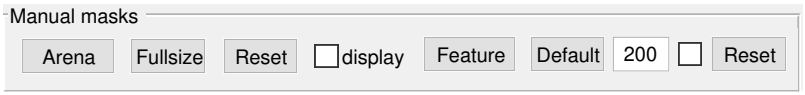


Figure 5.2: Manual mask control panel.

The **Manual mask** panel (Fig. 5.2) has a set of controls to define the arena and other features such as the inoculum. The **Arena** button and the **Feature** button open a second interface to the binary editing package (Fig. 5.3) described in detail in Chapter 16.

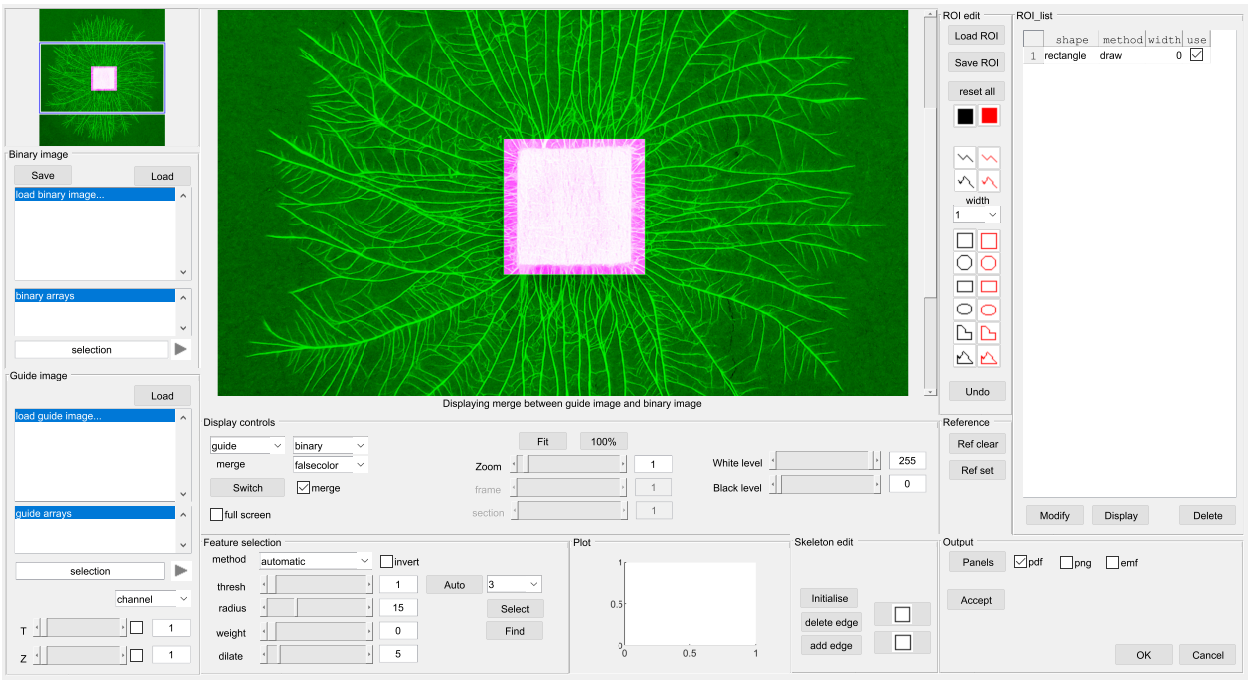


Figure 5.3: The binary editing GUI showing the inoculum mask

In brief, there are a set of drawing tools similar to standard image editing packages to add regions-of-interest (ROIs) in sequence. *Black* tools add ROIs, whilst *red* tools, substract ROIs. These ROIs are saved along with the resultant binary image and returned to the main GUI when editing is finished. The icons for the masks are shown in the **Thumbnail** bar.

The checkboxes in the **Manual masks** panel toggle display of the *arena* in cyan, or the *inoculum* (dashed red/white) on the main image display. In addition, there are buttons to automatically set a **Fullsize** arena, or **Reset** the arena to an empty image. Likewise, for the **Feature** (inoculum) the **Default** button adds a square ROI with the size set to the value, in pixels, in the adjacent text box, without invoking the Binary Editing GUI. This can be resized and re-positioned, before double-clicking to set the mask.

Once a ROI or set of ROIs has been delineated, they can be subsequently re-loaded and edited in the Binary Editing GUI using the **Arena** or **Feature** button as appropriate (Fig. 5.4).

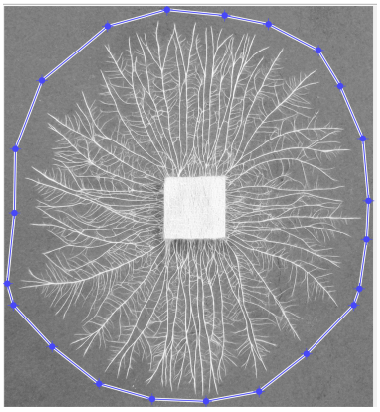


Figure 5.4: Adjustment of the arena ROI

6
Image Processing

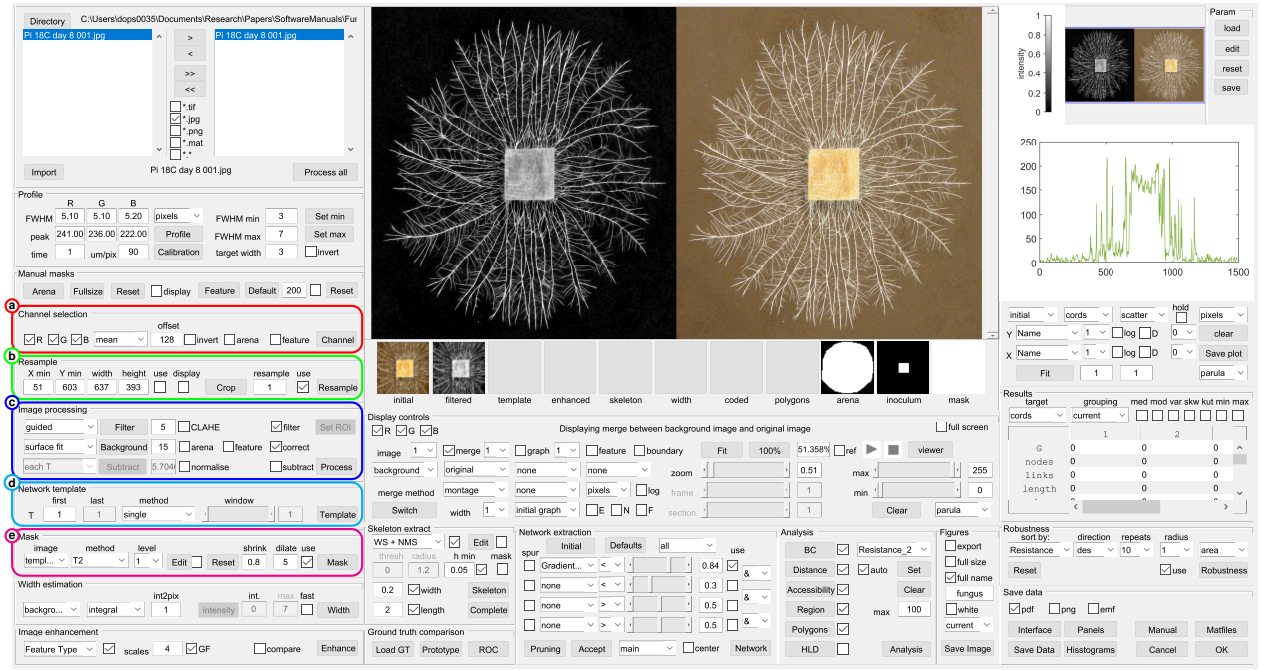


Figure 6.1: Controls for channel selection (a), resampling (b), image processing (c), template construction (d), and masking (e)

6.1 *Overview*

Before attempting to extract the skeleton, a series of processing steps can be employed to convert the image to a single-channel 8-bit format using the **Channel selection** panel (Fig. 10.1(a)); crop and resample the image, usually by upsampling, to ensure that the hyphae/cords are processed in a consistent manner using the **Resample** panel (Fig. 10.1(b)); followed by filtering to remove noise and background correction to standardise the intensity scaling using the controls in the **Image processing** panel (Fig. 10.1(c)). The **Network template** (Fig. 10.1(d)) panel is used to collapse time-series images into a single image to allow extraction of a single consistent network across frames, but has no effect on single images. The **Mask** controls (Fig. 10.1(e)) set up a region within the

arena that encompasses the mycelium and is used to filter out noise or spurious features later in the processing.

6.2 Channel selection

Typical image inputs from macro-photography systems are in 8-bit RGB format, whilst microscopy-based systems may be 8-bit or 16-bit greyscale images, possibly with multiple channels depending on the imaging mode employed. The expectation is that the network extraction will operate on an 8-bit image, which requires selection of which channels or combination of channels should be processed.

Figure 6.2: The Channel selection control panel:

The **Channel selection** panel allows choice of which **RGB** channels to include via the corresponding checkboxes and the adjacent drop-down menu. The method includes use of single channels, the *mean* or *max* value, or specific operations involving two channels including differencing (e.g. R-B) or ratioing (e.g. R\B or R^2\B). Operations involving subtraction require an offset, using the **offset** textbox, to avoid negative values, whilst both subtraction and division operations require scaling. This is achieved automatically using the minimum and maximum values in the image. Nevertheless, the **arena** and **feature** checkboxes may need to be checked to ensure that regions outside the arena or in the inoculum are excluded from the scaling calculation, so that extraneous features do not affect the resultant image intensities.

6.3 Resampling

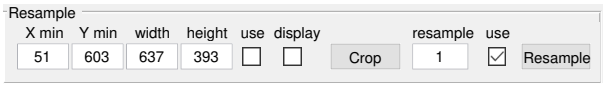


Figure 6.3: The Channel selection control panel:

The **Crop** button prompts the user to define a rectangular ROI that will be used to crop the image if the **use** checkbox is also ticked. Alternatively, the pixel co-ordinates can be entered directly in the **X min**, **Y min**, **width** and **height** textboxes. The **display** checkbox overlays the selected ROI on the image.

There is also the option to oversample the image by a scaling factor and bi-linear interpolation, set using the **resample** text box. This can improve the subsequent image enhancement steps and assignment of the centreline of thin edges, but at the cost of longer processing times.

The value of $FWHM_{min}$ estimated from the **Profile** measurements is automatically used to calculate the resampling factor needed to ensure that the minimum hyphal width matches the

size set by the **target** text box in the **Profile** panel. Likewise, the $FWHM_{max}$ value is used to determine the number of scales to use in the subsequent steps to ensure both that the largest hyphae are correctly segmented, and set the size of many of the processing and filtering kernels. As the scaling required is approximate, it is sufficient to set the values of $FWHM_{min}$ and $FWHM_{max}$ to integers, ideally odd values below or above the measured value, respectively. The resample **use** checkbox is active by default. If the box is unchecked, no resampling takes place, but it is possible that subsequent steps do not perform as expected if the hyphae/cords are not within the expected size range. Once the **Resample** button is clicked, the initial image is resampled and displayed. The *filtered* thumbnail is updated to show a small icon taken from the centre of the resampled image.

6.4 Filtering and background subtraction

Following completion of the required steps in the **Resampling** panel, the controls in the **Image processing** panel are automatically enabled, to provide a number of options to reduce noise in the image and correct or subtract the background (Fig. 6.4).

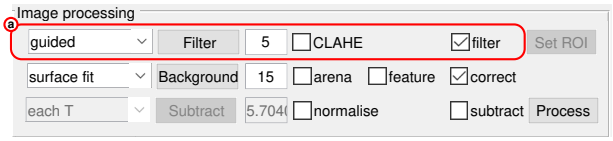


Figure 6.4: Filtering controls in the Image processing panel: Provides options to reduce noise in the image with linear or adaptive filtering techniques

The **Filter** button (Fig. 6.4(a)) can be used to apply different types of filtering to reduce noise and, in some cases, enhance ridge-like structures in the image. Filtering can be toggled on or off using the **filter** checkbox. The filters available include:

- *'Guided'* (default) : applies the Matlab edge-preserving smoothing filter that is guided by the intensities in the original image[1]. The size of the square neighbourhood used in the filter is set by the adjacent text box (in integer pixels). Typical values range from 3-7. Note: if the filter size is set to an even number there will be a 1-pixel shift in the image and this is not recommended.
- *'fibermetric'* : applies the Matlab fibermetric implementation of the 'Vesselness' enhancement filter developed by Frangi[2] that highlights ridge structures based on the eigenvectors of the local image gradient (Hessian) - see Chapter **??**. The range of scales are set by $FWHM_{min}/rescale$: $FWHM_{max}/rescale$.
- *'SOAGK'* : applies the second-order anisotropic Gaussian filter developed by Shui *et al.* 2012[3] and Lopez-Molina *et al* 2015[4]. This calls the *AGlineDetector* function from Carlos Lopez-Molina. The default values (which are editable using the *Edit* button in the **Param** panel) are 4 scales, with 4 values of sigma ranging from 1 to 2.8, and an anisotropy value of 1.2.

[1] K. He, J. Sun, and X. Tang. Guided image filtering. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35:1397–1409, 2013

[2] A. F. Frangi, W. J. Niessen, K.L. Vincken, and M.A. Viergever. *Multiscale vessel enhancement filtering*, pages 130–137. Springer Berlin Heidelberg, Berlin, Heidelberg, 1998

[3] P.-L. Shui and W.-C. Zhang. Noise-robust edge detector combining isotropic and anisotropic gaussian kernels. *Pattern Recognition*, 45:806 – 820, 2012

[4] C. Lopez-Molina, G. V. D. de Ulzurrun, J. M. Baetens, J. Van den Bulcke, and B. De Baets. Unsupervised ridge detection using second order anisotropic gaussian kernels. *Signal Processing*, 116:55–67, 2015

- *'Bowler Hat'* : An alternative morphology-based approach to enhance ridges using the difference between greyscale opening (erosion followed by dilation) with a radial set of line segments, and the equivalent opening using a disc[5]. This is designed to be more robust at junctions and less sensitive to interference from blob-like inclusions. The scale parameters are set as $1 + [2, 4, 8] * FWHM_{max}/rescale$, with 12 orientations, by default.

- *'Weiner'* : The Matlab *Weiner* filters use a rectangular neighbourhood of size $2 \times kernel + 1$ and provides some adaptive filtering based on the local image variance[6]. Where the variance is large, there is little smoothing, preserving the network edges. Where the variance is small, such as the background, there is more smoothing. Wiener filtering works best when the noise is constant-power ("white") additive noise, such as Gaussian noise.

- *'Wavelet'* : This calls the Matlab *wdenoise2* function which uses an empirical Bayesian method to remove noise. The 'bior4.4' wavelet is used with a posterior median threshold rule. Denoising is down to the minimum of floor(log2(min([M N]))) and wmaxlev([M N],'bior4.4') where M and N are the row and column sizes of the image.

- *'DNN'* : This calls the Matlab *'denoiseImage'* function which estimates a denoised image using a pre-trained denoising deep neural network. This works for Gaussian noise with a limited range of standard deviation. Calculations can take a considerable amount of time.

Several of these filtering operations have additional default parameters. Key values are calculated depending on the $FWHM_{min}$ and $FWHM_{max}$ values, although some of the parameters can also be edited using the **edit** button in the **Param** panel. for the *guided* and *Weiner* filter, the value of σ can be defined using the adjacent **kernel** textbox. examples of the results of the different filtering operations are shown in figure 6.5.

In addition, the *'CLAHE'* checkbox applies contrast-limited adaptive histogram equalisation[7] to the image to expand the contrast range over a set of rectangular regions tiled over the image. The size of the regions are automatically set to 200 pixels on the original image dimensions.

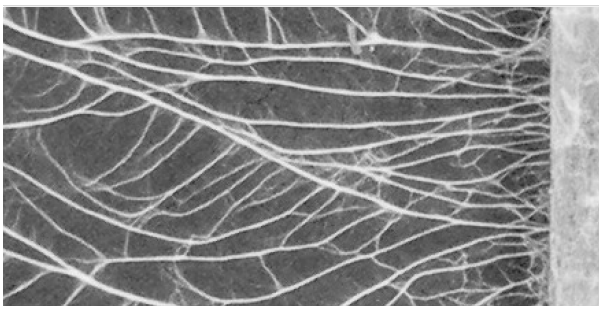
In each case, if the image is multi-dimensional, such as a time-series, the filtering is applied to each image separately.

[5] Sazak Ç., Nelson C.J., and B. Obara. The multiscale bowler-hat transform for blood vessel enhancement in retinal images. *Pattern Recognition*, 88:739–750, 2019

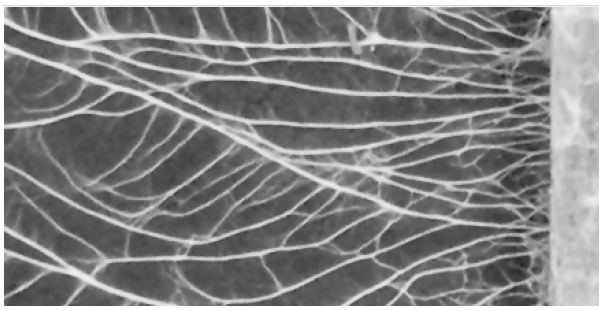
[6] Jae S. Lim. *Two-Dimensional Signal and Image Processing*, volume p. 548, equations 9.26, 9.27, and 9.29. Prentice Hall, Englewood Cliffs, NJ, 1990

[7] K. Zuiderveld. *Contrast limited adaptive histograph equalization*, pages 474–485. Graphic Gems IV. Academic Press Professional, San Diego:, 1994

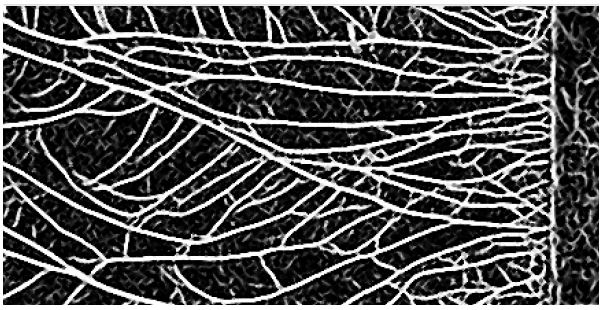
Figure 6.5: Examples of the different filtering algorithms



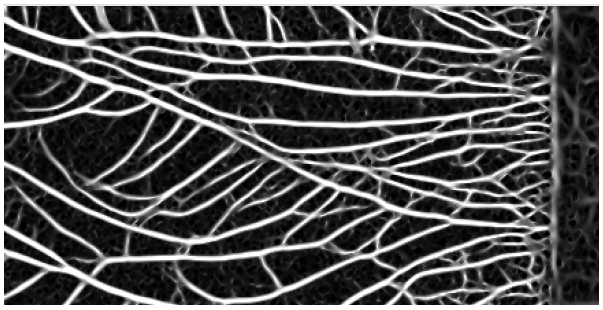
(a) Input image



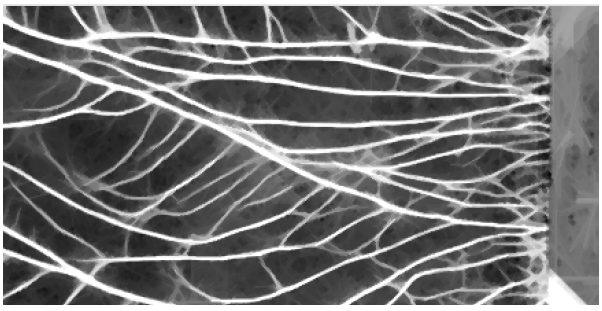
(b) Guided filter



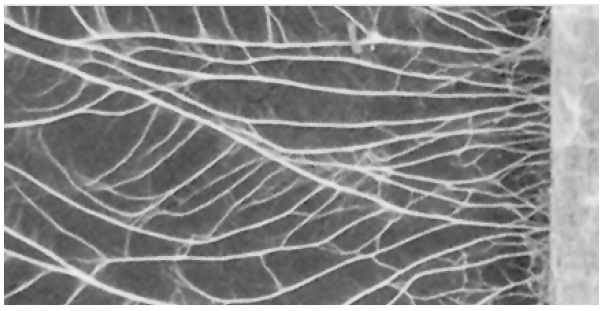
(c) Fibermetric filter



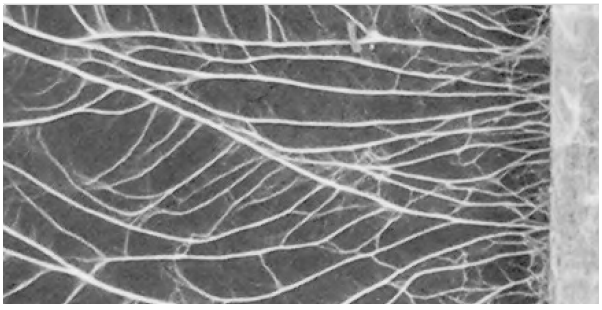
(d) SOAGK filter



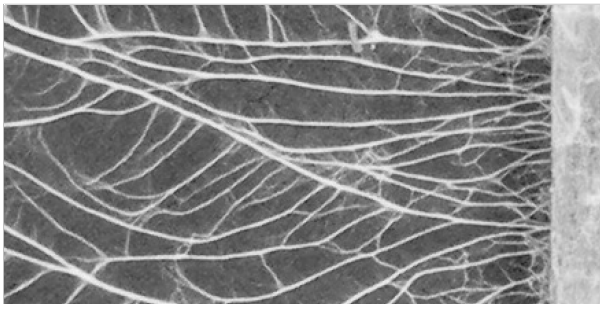
(e) BowlerHat filter



(f) Weiner filter



(g) Wavelet denoising



(h) Convolution neural network denoising

6.5 *Background correction*

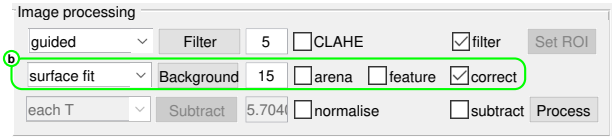


Figure 6.6: Background correction in the Image processing panel to compensate for uneven illumination

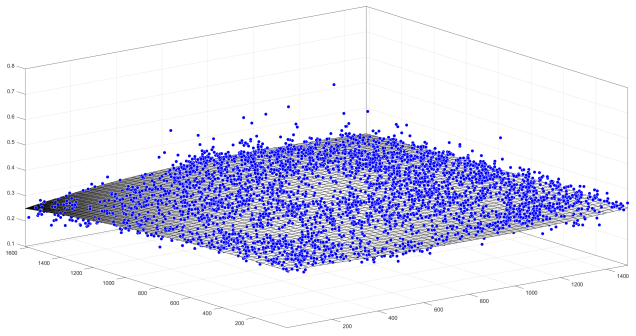
Once the image has been filtered, the **Background** controls (Fig. 6.6(b)) can be used to apply a shading correction and/or background subtraction to each image. Different methods can be selected from the *method* dropdown menu:

- *'tophat'* (default) : Performs morphological tophat filtering using a grayscale opening using disk-shaped kernel with the radius set by $(1.2 \times FWHM_{max})/(2 \times resample)$ (i.e. 20% larger than the radius of the largest hypha/cord expected). This removes any features smaller than $1.2 \times FWHM_{max}$ and provides an estimate of the local background around each pixel. The opened image is subtracted from the original to correct for the local background.

- *'Sub low pass'* : the image is filtered using a Gaussian kernel with a large radius sufficient to remove all the high-frequency information in the image. The standard deviation for the Gaussian kernel is calculated as $2 \times FWHM_{max} + 1$, or approximately twice the size of the largest hypha/cord diameter. The low pass image is subtracted from the original and the image re-normalised. This approach also suffers if there are large sheet-like regions

- *'div low pass'* : follows the same approach au *'Sub low pass'*, but then divides the image by the normalised background image to compensate for uneven illumination intensity across the field of view. This requires a subsequent background subtraction step.

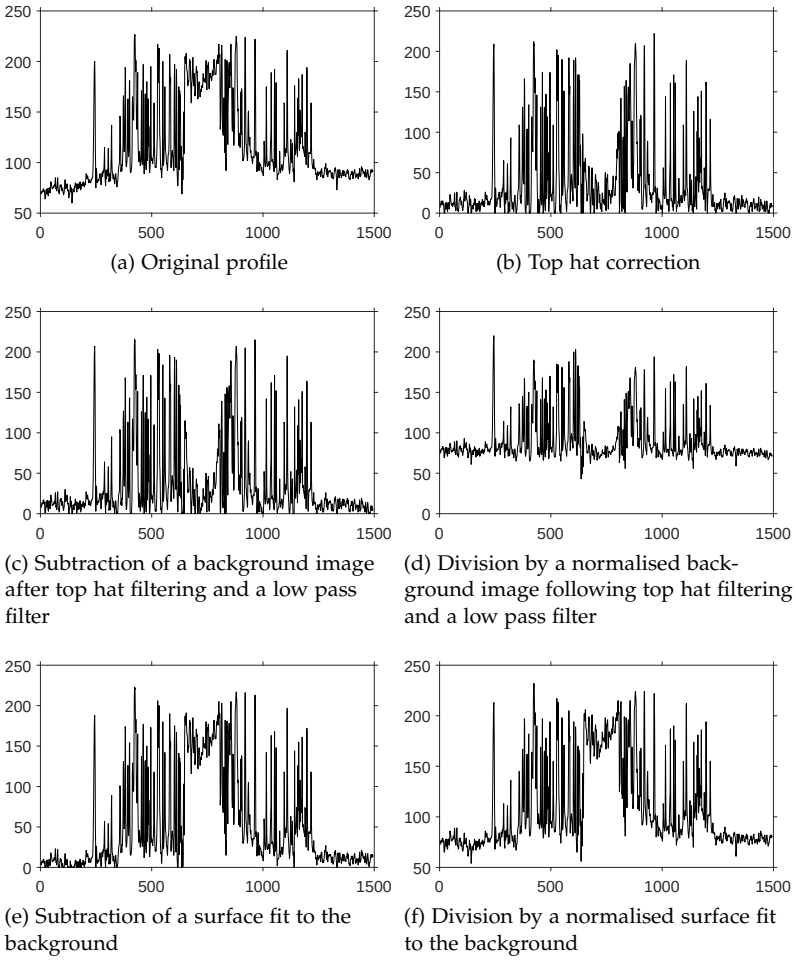
- *'Surface fit'* : this finds all the local minima across the image and fits a surface to points in the 10-90% interval using a cubic polynomial (*'poly33'*) (see below):



The surface is converted to an image and subtracted from the original. This method works if the network is sparse, giving plenty of local background estimates across the image. Different fitting models can be selected using the **edit** parameters button in the **param** panel from a list including *'linearinterp'*, *'cubicinterp'*, *'biharmonicinterp'*, *'thinplateinterp'*, *'lowess'*, *'loess'*, *'poly22'*, *'poly33'*, *'poly55'*

- *'surface div'* : follows the same approach as *'Surface fit'*, but then divides the image by the normalised background image to compensate for uneven illumination intensity across the field of view. This requires a subsequent background subtraction step.

A plot of the intensity along the diagonal of the image is automatically plotted to give a simple visual indication of the effects of background correction(Fig. 6.7). All the background correction algorithms perform reasonably well in terms of levelling the background compared to the original. The main differences are that the divisional corrections (d) & (f) require an additional background subtraction step, whilst the top hat and low pass filters also remove the inoculum. As the inoculum will be masked from later processing steps anyway, this does not make much difference to the results.



(a) Original profile

(b) Top hat correction

(c) Subtraction of a background image after top hat filtering and a low pass filter

(d) Division by a normalised background image following top hat filtering and a low pass filter

(e) Subtraction of a surface fit to the background

(f) Division by a normalised surface fit to the background

Figure 6.7: Examples of the different background correction algorithms for a profile drawn diagonally across the colony. Note that Top hat and low pass-filtering remove the inoculum from the image. Note also that the divisional corrections for uneven illumination (d) & (f) also require a subsequent background subtraction step.

6.6 *Background subtraction*

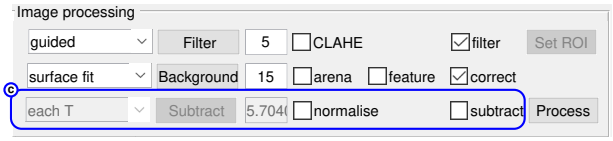


Figure 6.8: Background subtraction in the Image processing paneln

The **Subtract** button (Fig. 6.8(c)) is only active if the previous background correction step does not automatically set the background to zero. It applies a background subtraction from the corrected image using the average from a ROI defined by the **Set ROI** button. The average background is displayed in the **mean** textbox, which can be manually edited if appropriate. For time-series, the left-hand drop-down menu selects whterh to use the average background from the ROI for the whole time-series, or whether to use a value determined separately for each time point. The **normalise** checkbox ensures that the final image is rescaled to the full 8-bit range.

Note - if the ROI has not been set or falls outside the cropped image the program will return an error and wait for the ROI to be set

6.7 *Template*

The input image is then converted to a template for network extraction using the options in the **Template** panel (Fig. 6.9). This step is only relevant to time-series images - single images are not modified. The main decision is whether to construct a single network that covers all hyphae/cords whenever they appear in the time series, or whether to construct separate networks for each time point. The advantage of the single network is that every node and edge has a unique ID that can be followed as the hypha/cord changes over time. The disadvantage is that many of the edges in a growing network should have a value of zero untile hyphae grow into that region.

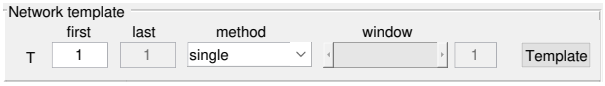


Figure 6.9: The Template control panel: The input image(s) are used to construct a template for the network extraction, with options to combine multi-dimensional images into a single template image

For time-series images, the template can be defined using the **method** dropdown menu for each frame separately (*individual*), or for a combination of all planes using either a maximum projection (*max proj*), average projection (*mean proj*), or median projection (*med proj*). A subset of the images can be selected for the projection using the **first** and **last** boxes. Alternatively, the images can be filtered over time with a mean or median filter, with the kernel size determined by the window slider, before taking the maximum projection (*max mean* or *max med*, respectively). These projection techniques only work if the input images are aligned with sub-pixel resolution (see Chapter 15). However, if applicable, the template image provides a single, complete network for all time-points that

can be used to interrogate the original image frames separately. This ensures that each node and edge retain the same unique identity at each time point. This allows time-series comparisons of changes in width as parts of the colony are strengthened or other parts regress for example.

6.8 Setting up a mask for the colony

Once the template has been defined, it may be useful to define a mask and associated colony boundary using the controls in the **Mask** panel (Fig. 6.11). The boundary serves to define the area occupied by the colony, and can also be used to exclude regions that should not be analysed, or to restrict the analysis to a specific region of the colony. The default setting is to automatically apply a boundary, but un-ticking the **use** checkbox will disable the boundary controls.

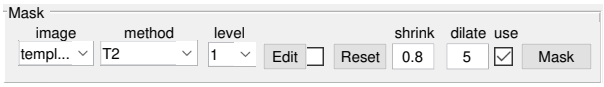


Figure 6.10: The Mask control panel that is used to define a binary image that includes the entire colony and also calculates the boundary using a 'shrunken' convex hull

Care is needed with this step as the thresholding operation may introduce breaks in some dim hyphae/cords, which are then not considered in subsequent processing steps. The method used to select the threshold adjacent drop-down menu provides a number of options including:

- *'Background'* : prompts the user to define a ROI in a background area of the image. The threshold is then calculated as the $background\ mean + 2 * SD\ units$.
- *'T1'* : this segments the main structures using an automatic threshold determined by Otsu's method[8] that minimizes the intraclass variance of the foreground and background distributions, but does not fill in any gaps or holes in the resulting binary image. Only the largest connected component is retained.
- *'T1 fill'* : fills in any holes in the *'T1'* mask. This helps to correct any small breaks in hyphae within any fully enclosed polygonal region, but masks out fewer background regions from the subsequent enhancement and segmentation steps.
- *'T1 boundary'* : This initially segments the image using the *T1* threshold, but then calculates the convex hull of the segmented outline. This boundary is then collapsed down onto the most prominent concave sections by an amount set using the **Shrink** textbox (typically 0.5) and filled completely.
- *'T2'*, *'T2 fill'* and *'T2 boundary'* : these operate in a similar manner to the *'T1'* settings, but use the selected value of a two-threshold partition of the image histogram (given by the adjacent drop down menu) and can be useful if there are a number of very

[8] N. Otsu. A threshold selection method from gray-level histograms. *IEEE Trans. Systems, Man, Cyber.*, 9:62–66, 1979

bright structures that distort a single threshold separation of background and object. Which threshold value is used depends on the adjacent drop-down menu.

- *'T3'*, *'T3 fill'* and *'T3 boundary'* : these operate in a similar manner, but use the selected value of a three-threshold partition of the image histogram.
- *'Holes'*, *'No holes'*, and *'erode'* : these are the same as the *'T1'* methods and are included for backward compatibility.
- *'manual'* : used if the mask is to be defined manually (see Chapter 16).

The image used to calculate the mask can be selected from the **image** drop-down menu from the following options:

- *Template* (default) : the mask is calculated from the template image and has the same dimensionality i.e. if a single template has been defined for a time-series, only a single boundary is calculated. Conversely, if the template is one per time-point a separate mask is also calculated for each time-point.
- *Background* : the mask is calculated from the filtered and background corrected image for a single image or each image in a time-series.
- *Accumulate* : the mask is calculated from the filtered and background corrected image for a single image or each image in a time-series, but in the latter case, the mask for each successive time-point also includes all information from the previous time-points.

To ensure that the mask is contiguous and captures all the significant parts of the colony, it can be 'dilated' by a set number of pixels given by the **dilate** text box.

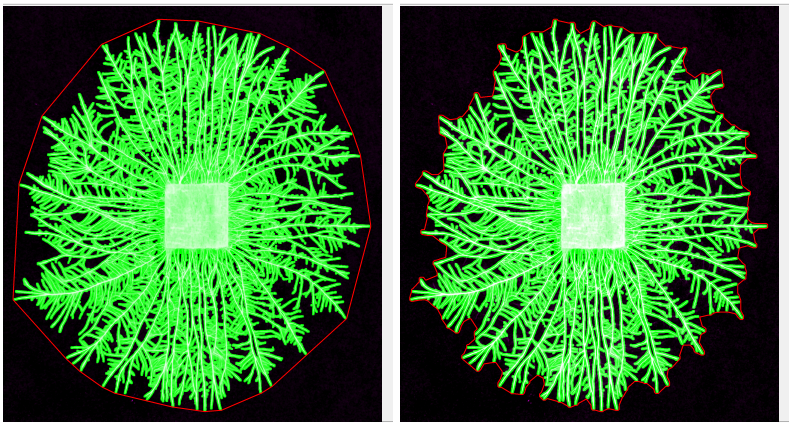
It is possible that the automatic boundary settings do not effectively select all parts of the colony, requiring the user to define the mask manually. The **edit** button will open the **Binary editing GUI** with a set of tools to allow manual adjustment of the binary image. Full details of the binary editing program are given in Chapter 16. If a mask image has been defined manually, the **use edit** check box is automatically activated and the manually defined mask will be used in subsequent processing steps. The manually-defined mask is saved in the parameter file and can be re-applied or edited further anytime the analysis is re-run.

The **Reset** button removes the stored version of the manually-edited boundary mask. Use with care as this option may delete a lot of careful editing!

6.9 *Defining the boundary of the colony*

Once the mask image has been set-up satisfactorily, it is used to define the boundary of the colony. For colonies with a very dense

even margin, this might reasonably be considered as the convex hull enclosing the outermost points of the mask.



(a) The binary mask (green) and colony boundary (red) defined as the convex hull

(b) The binary mask (green) and colony boundary (red) calculated using a convex hull with a shrink factor of 0.8

Figure 6.11: Estimation of the colony boundary and area covered

However, for colonies where growth is more sparse and the margin becomes separated into leading hyphae or cords with considerable gaps between them, the convex hull appears to be a progressively worse descriptor of the area covered by the colony (e.g. Fig. 6.11(a)). Thus, there is the option to 'shrink' the convex hull to give a progressively more concave hull that follows the outline of the colony more closely. The **shrink** text-box controls the amount of shrinkage from 0 (the convex hull) to 1 (a fully concave hull). This is not a precise operation for fungal colonies, but values from 0.5-0.9 seem to work quite well (Fig. 6.11(b)). As values approach 1, there is the risk that the boundary fragments, in which case the value needs to be reduced until a single region is obtained.

Clicking the **Mask** button runs the chosen method, displays the segmented binary image as a merge with the template image to aid comparison, and updates the mask thumbnail. The outermost boundary is also plotted in red, and can be toggled on or off using the **boundary** checkbox in the display controls panel.

The area enclosed by the outermost boundary controls the estimate of the area covered by the colony and impacts on any subsequent density estimates.

7
Width estimation

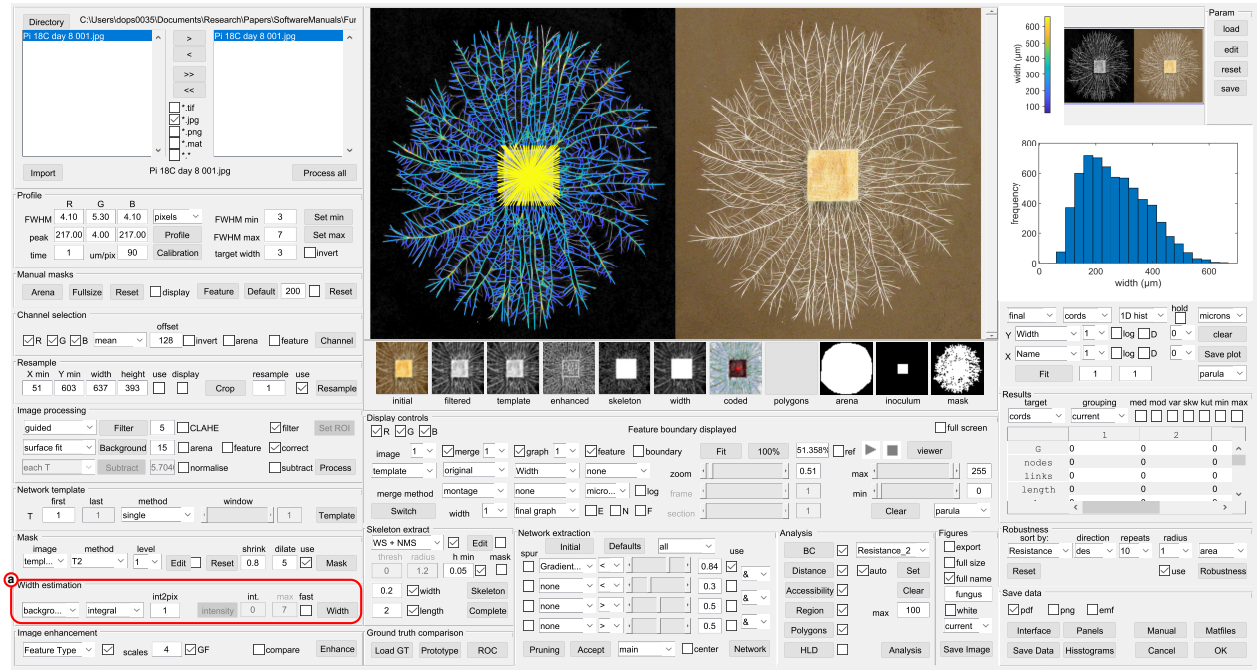


Figure 7.1: Controls to estimate the width of hyphae or cords

7.1 Estimation of the hyphal width

Once the pixel-skeleton has been segmented satisfactorily, the next step in network extraction is to estimate the hyphal width, before converting the pixel skeleton to a weighted graph representation, using the **Width estimation** panel (Fig. 7.2). The target image to base the width measurements on can be selected from the adjacent dropdown menu from the *resampled*, *filtered*, *background*, *template*, or *enhanced* image. The default is typically the background-subtracted image.

Figure 7.2: Controls to estimate the width of hyphae/cords

Unfortunately, the width of some hyphae in corded colonies falls

below the resolution of the image, so direct physical estimates of the width are challenging. A number of different approaches are available that all provide some information on the hyphal diameter including:

- *'Distance'* : This estimates the local FWHM of the hyphae/cords at each pixel in the skeleton using the original hyphal/cord intensity as a proxy for the amount of mycelium present. The peak intensity is estimated from the background-subtracted image, sampled for each pixel in the skeleton, whilst the distance is estimated from the distance transform of the pixel skeleton. The 50% threshold is estimated from where the pixel intensity falls below half the peak intensity, assuming a local background of zero. At the moment there is no interpolation, so there is likely to be considerable discretisation error with this approach, although this will be partially compensated by the averaging that takes place later along each hypha/cord during graph conversion to give the mean width of each hypha/cord.
- *'Maximum-gradient granulometry'* : The intensity image is subject to a series of image openings (erosion followed by dilation) that progressively remove structures as the size of the opening kernel exceeds the size of the underlying object. This results in an intermediate (x,y,s) image, where s increases with the size of the disk-shaped kernel. The intensity of each pixel in the skeleton initially decreases slowly with s as the kernel samples more of the object, but then reduces dramatically once the boundary of the object is reached, and the kernel only samples the background. The transition point for any pixel is determined from the maximum (negative) gradient of the granulometry curve. This approach constrains the width to integer pixels values, and also suffers from the digital approximation of small kernels to a true disk shaped kernel.
- *'Integrated intensity granulometry'* : This approach follows the same methodology as the maximum-gradient granulometry method, but rather than extract a specific size threshold, the integrated intensity under the granulometry curve is calculated. This provides a more nuanced interrogation of the local image intensity, but cannot be directly related to the physical hyphal width without additional assumptions about the relationship between reflection/transmission intensity and width. The output from integrated intensity measurements are typically linear with the width of the objects, but still requires a calibration factor to relate intensity to radius set by the **int2pix** textbox. This approach does help with estimation of relative hyphal/cord widths, even if they are sub-resolution objects.

An additional option is available to increase the speed of the granulometry by using decomposition of the disk-shaped structuring element into a set of linear elements using the **fast** checkbox,

at the expense of some accuracy in creating radially symmetric disc-shaped kernels.

7.2 *Comparison between automated width estimates and manual profiles*

To test whether the automated width measurements provide reasonable estimates for the hypha/cord widths, it is possible to compare the output with manually drawn profiles across well-defined hyphae/cords. In the **Image Display** panel, the width image (in pixels) can be merged with the background subtracted image (Fig. 7.3) using the *non-scaled* option for the **merge method**.

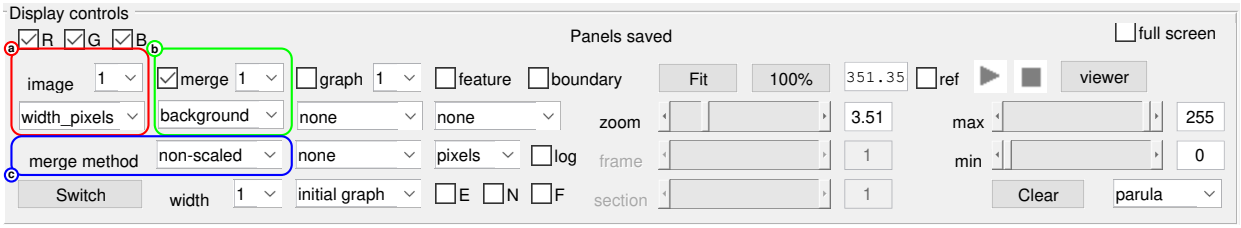
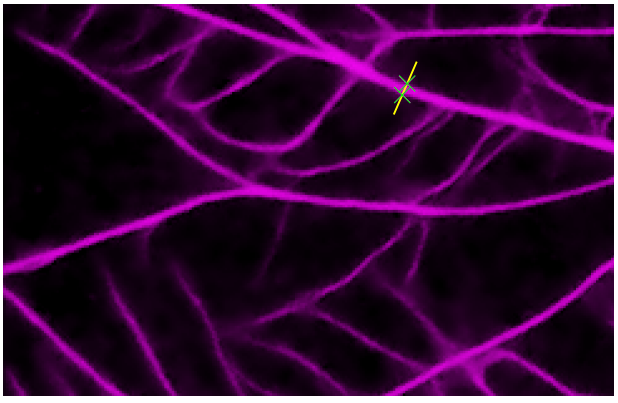
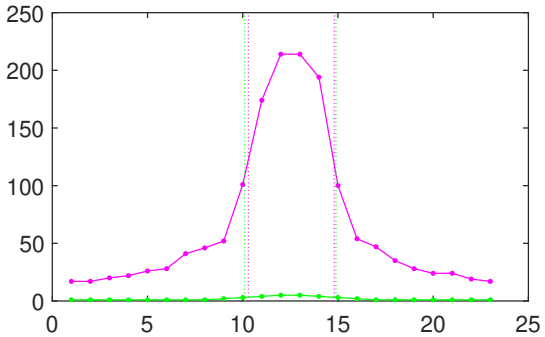


Figure 7.3: Merged display of the width image (in pixels) (a) and background (b) using the to allow comparison between automated and manual width estimates

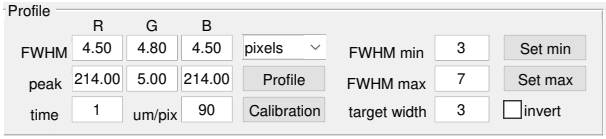
The image displayed will be dominated by the background image (in magenta) as the intensity of the non-scaled version of the width in pixels represents integer values of the width and will therefore be very dim (Fig. 7.4(a)).



(a) merged image



(b) manual profile



(c) profile results

Figure 7.4: Comparison between the automatic width estimate and a manual profile based on the full-width half-maximum width

However, a profile drawn across a cord (Fig. 7.4(b)) allows a direct comparison between the FWHM from the background subtracted image and the automatic width estimate from the peak

intensity in the green image, with the results displayed in the **Profile** panel (Fig. 7.4(c)).

8
Image enhancement

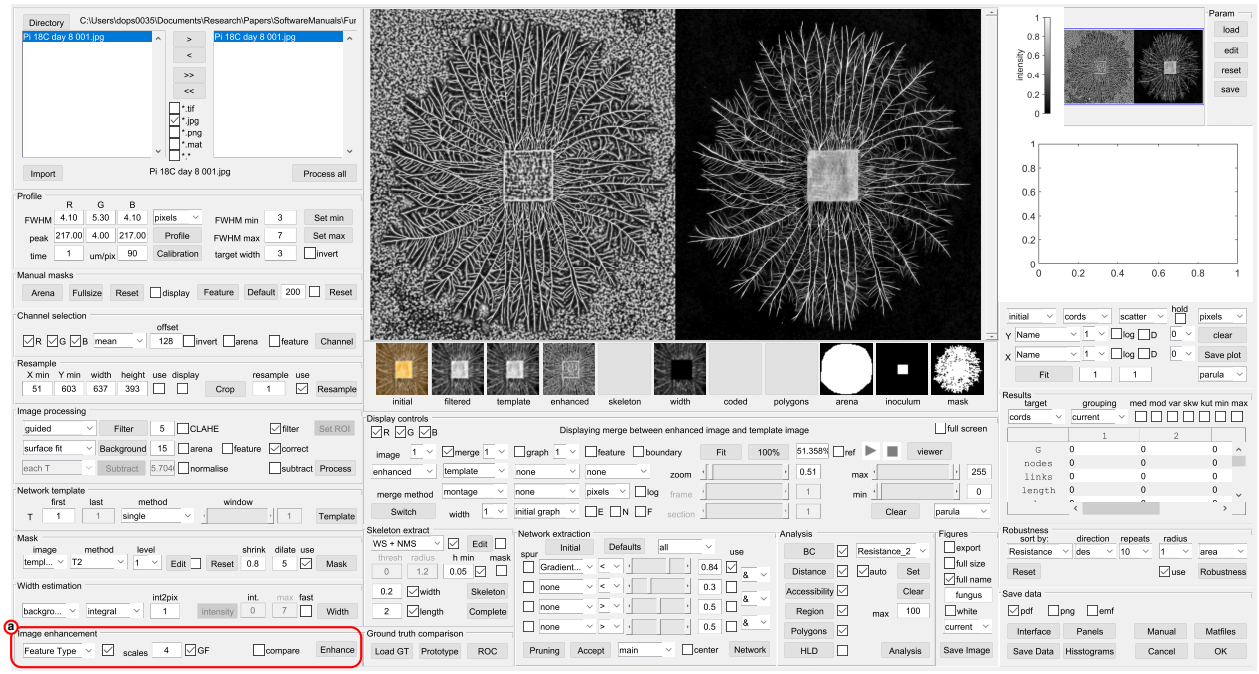


Figure 8.1: Controls for skeleton extraction

8.1 Introduction

As the objective of the overall programme is to be able to extract networks of hyphae or cords with different thickness and intensity from images that are also often contaminated with extraneous features or noise, it is often appropriate to try to enhance the template image prior to segmentation using *a priori* expectations of the structure of network edges in comparison to other image features. Fungal mycelia an instance of a general class of curvi-linear detection and enhancement problems that are common across many disciplines. Several different approaches have been proposed for fungal mycelia including intensity-independent phase congruency[1] or second-order anisotropic Gaussian kernels (SOAGK)[2], which has also been implemented in the Fungal Feature Tracker graphical user interface (GUI)[3]. In general terms, each method applies filters to

[1] B. Obara, V. Grau, and M. D. Fricker. A bioimage informatics approach to automatically extract complex fungal networks. *Bioinformatics*, 28:2374–81, 2012

[2] C. Lopez-Molina, G. V. D. de Ulzurrun, J. M. Baetens, J. Van den Bulcke, and B. De Baets. Unsupervised ridge detection using second order anisotropic gaussian kernels. *Signal Processing*, 116:55–67, 2015

[3] G. Vidal-Diez de Ulzurrun, T.-Y. Huang, C.-W. Chang, H.-C. Lin, and Y.-P. Hsueh. Fungal feature tracker (fft): A tool for quantitatively characterizing the morphology and growth of filamentous fungi. *PLoS computational biology*, 15:e1007428, 2019

emphasize ridges over a number of different scales and at a number of different orientations. Additional local properties characteristic of a true edge, such as the local shape asymmetry and alignment, can also be applied to aid subsequent segmentation. To maintain maximum flexibility we have include these algorithms, and other related enhancement methods in the Fungal Network Analysis GUI, although we currently prefer the phase-congruency approach as it handles soil-based microcosms much better than other algorithms.

Segmentation involves conversion of the enhanced image to a binary image, followed by thinning to give a single pixel wide skeleton. Both these operations can be implemented in a variety of different ways, and again a number of options are provided in the Fungal Network analysis GUI, each with different strengths and weaknesses. Often different binarization methods operate in concert with particular enhancement approaches, so it is best to consider them in tandem.

8.2 *Edge enhancement*

Figure 8.2: The Enhance controls (a) and the compare checkbox (b) in the Skeleton extraction panel

The first drop-down menu in the **Image enhancement** panel (Fig. 8.2) provides a number of different methods to enhance the mycelial network and facilitate subsequent segmentation. Several of these options duplicate the filtering functionality available in the image processing panel, effectively allowing different filtering approaches to be combined.

- *PC* : uses the intensity-independent phase congruency approach developed by Peter Kovesi[4] and implemented in the Matlab *phasecong3* function[5].
- *PCT* : This adds an additional Tensor evaluation to the phase congruency approach that helps to select ridge-like elements[6].
- *Feature Type* : uses the mean phase angle from the phase-congruency approach.
- *Vesselness* : applies the *FrangiFilter2D* Matlab implementation of the Frangi 'Vesselness' filter[7] written by Marc Schrijver and D. Kroon, that highlights ridge structures based on the eigenvectors of the local image gradient (Hessian). This is used here in preference to the Matlab *fibermetric.m* algorithm as it retains control of the coefficients used to select shape asymmetry and intensity.
- *'SOAGK'* : applies the *AGlineDetector* second-order anisotropic Gaussian filter developed by Lopez-Molina *et al* 2015[8].

[4] P. Kovesi. Image features from phase congruency. *Videre: Journal of Computer Vision Research*, 1:1–26, 1999

[5] P. D. Kovesi. MATLAB and Octave functions for computer vision and image processing, 2000. Available from: `http://www.peterkovesi.com/matlabfns/`

[6] B. Obara, V. Grau, and M. D. Fricker. A bioimage informatics approach to automatically extract complex fungal networks. *Bioinformatics*, 28:2374–81, 2012

[7] A. F. Frangi, W. J. Niessen, K.L. Vincken, and M.A. Viergever. *Multiscale vessel enhancement filtering*, pages 130–137. Springer Berlin Heidelberg, Berlin, Heidelberg, 1998

[8] C. Lopez-Molina, G. V. D. de Ulzurrun, J. M. Baetens, J. Van den Bulcke, and B. De Baets. Unsupervised ridge detection using second order anisotropic gaussian kernels. *Signal Processing*, 116:55–67, 2015

- *Neuriteness* : applies the *NeuritenessFilter* written by B. Obara that implements the 'Neuriteness' filter developed by Meijering et al.[9].

- *'Bowler Hat'* : An alternative morphology-based approach using the difference between greyscale opening (erosion followed by dilation) using a radial set of line segments, and the equivalent opening using a disc[10]. This is designed to be more robust at junctions and less sensitive to interference from blob-like inclusions. The scale parameters are set as $1 + [2, 4, 8] * FWHM_{max}/rescale$, with 12 orientations, by default.

- *MFAT* : improved Hessian-based enhancement techniques using Multiscale Fractional Anisotropy Tensors (MFAT), in both their eigenvalue-based (MFATλ) or probability-based (MFATp) form[11]

[9] E. Meijering, M. Jacob, J. Sarria, P. Steiner, H. Hirling, and M Unser. Design and validation of a tool for neurite tracing and analysis in fluorescence microscopy images. *Cytometry*, 58: 167 – 176, 2004

[10] Sazak Ç., Nelson C.J., and B. Obara. The multiscale bowler-hat transform for blood vessel enhancement in retinal images. *Pattern Recognition*, 88:739–750, 2019

[11] H.F. Alhasson, S.S. Alharbi, and B. Obara. 2d and 3d vascular structures enhancement via multiscale fractional anisotropy tensor. *Computer Vision - ECCV 2018 Workshops*, pages 365–374, 2019

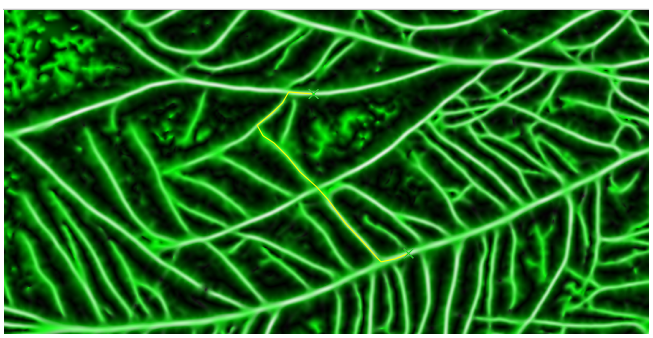
The number of scales can be set in the main interface using the **Scales** text box, whilst the other parameters are accessible using the **Edit** button in the **Param** panel. The theory behind each method is given in Chapter 9, along with the default values used for each of the parameters. Results for the different methods are shown in Figure 8.4.

The result of the image enhancement can be further smoothed using a *Guided* filter, if the **GF** checkbox is ticked, using the enhanced image as the template and a default neighbourhood of 5x5 pixels.

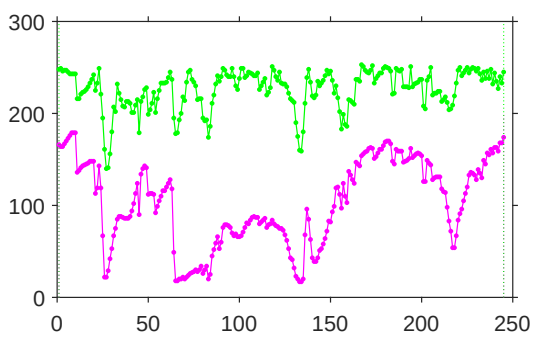
8.3 Visual comparison of methods

To examine the comparative effects of different filters in more detail, the **compare** checkbox can be used to save a copy of the result of the current operation in a *'compare'* image. The image can be processed again using different methods or parameters, and the result is automatically merged with the *'compare'* image. If the merged image is displayed in *falsecolor* then the images are superimposed in green and magenta, with each image scaled independently.

Figure 8.3: Using the **compare** checkbox to visualise the results of different enhancement algorithms



(a) Merged image from the feature Type enhancement (green) and SOAGK filter (magenta)

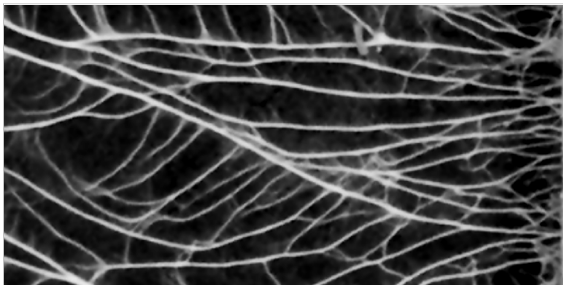


(b) Profile along two hyphal branches for Feature Type (green) and SOAGK (magenta) enhancement

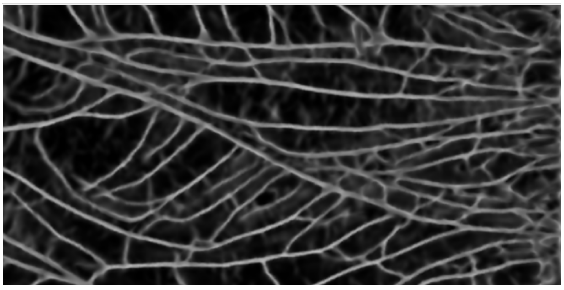
If the *non-scaled* **merge method** is selected, the images are displayed without scaling. The **Profile** button can be used to draw a transect across both images to provide a more quantitative evaluation of performance (Fig. 8.3).

In this case, the more consistent performance of the phase-congruency *Feature Type* method (green) is shown compared to the *SOAGK* method (magenta) for a profile running across a couple of cross-connecting cords in the centre of the image (Fig. 8.3(a), yellow). In essence, each cord in the *Feature Type* image has a similar intensity with relatively small dips at the junctions. In contrast, the *SOAGK* image shows much greated variation in intensity along the cords and shows significant dips at junctions. This makes it more challenging to achieve subsequent segmentation and skeletonisation that retains the full connectivity of the network.

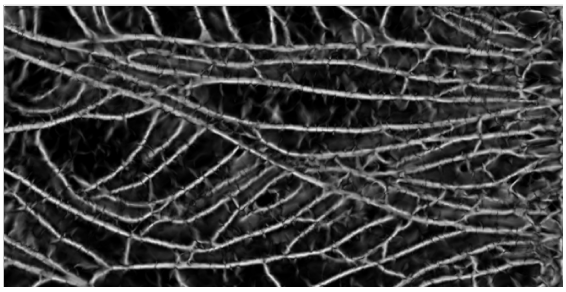
Figure 8.4: Examples of the different enhancement algorithms



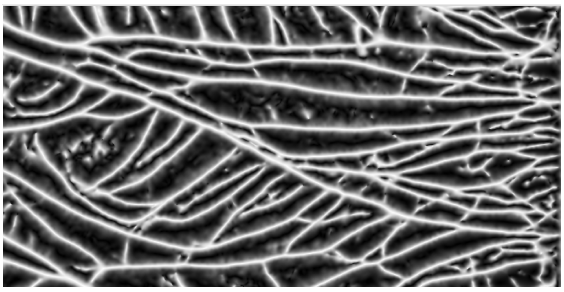
(a) Template image



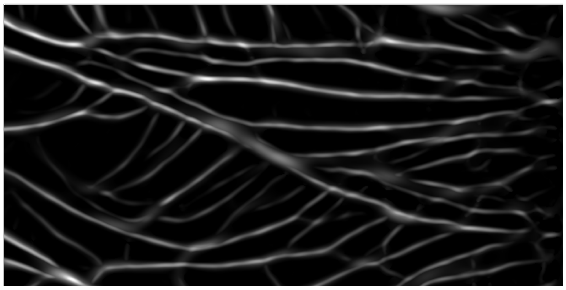
(b) Phase congruency



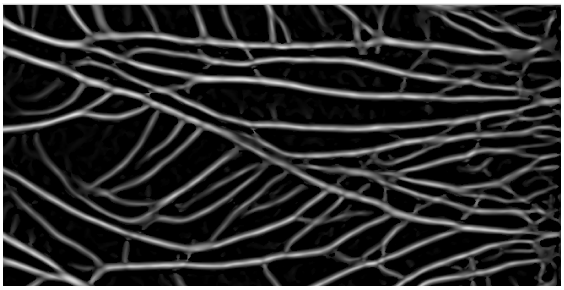
(c) Phase congruency Tensor



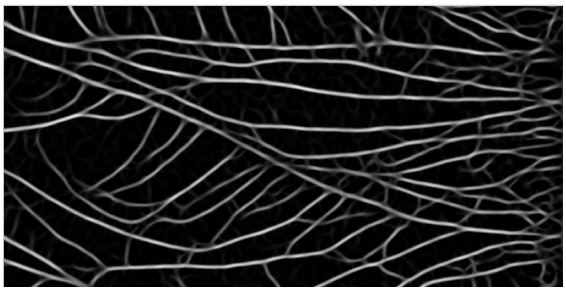
(d) Phase congruency Feature type



(e) 'Vesselness'



(f) 'Neuriteness'



(g) Second-order anisotropic Gaussians (SOAGK)

(h) Multiscale Fractional Anisotropy Tensor (MFAT) - lambda

(i) Multiscale Fractional Anisotropy Tensor (MFAT) - probability

(j) Bowler Hat

9
Overview of enhancement methods

9.1 Introduction

Whilst there are many techniques to detect edges in an image, these are not usually appropriate for detection of hyphae or cords, where the centerline of the structure is more relevant that the boundary edge of the object *per se*. The centerline can be considered as a ridge in the intensity landscape, and is characterised by a steep fall-off in intensity perpendicular to the direction of the ridge centerline, compared to relatively little change along the direction of the ridge. However, the situation is more complicated at junctions where ridges meet, as the intensity landscape no longer follows a simple well-defined pattern. There are a wide range of different approaches that could be used to enhance the ridge-like structures in fungal networks. This Chapter provides some of the theoretical background to the approaches implemented in the fungal network analysis package.

9.2 Frangi 'Vesselness'

One of the first methods to identify ridges exploited the local curvature of the intensity landscape as estimated from the Hessian (H_σ). This comprises second-order partial derivatives, D_{aa} along direction a, of the intensity image (I), where the value of the standard deviation of the Gaussian kernel (σ) is varied over a range of scales that span the sizes of the underlying features[1], see for example Fig. 9.1A-C:

[1] A. F. Frangi, W. J. Niessen, K.L. Vincken, and M.A. Viergever. *Multiscale vessel enhancement filtering*, pages 130–137. Springer Berlin Heidelberg, Berlin, Heidelberg, 1998

$$H_\sigma = \begin{bmatrix} D_{xx} & D_{xy} \\ D_{xy} & D_{yy} \end{bmatrix} = \begin{bmatrix} \frac{\delta^2 I}{\delta x^2} * G_\sigma & \frac{\delta^2 I}{\delta x \delta y} * G_\sigma \\ \frac{\delta^2 I}{\delta x \delta y} * G_\sigma & \frac{\delta^2 I}{\delta y^2} * G_\sigma \end{bmatrix} \tag{9.1}$$

where

$$G_\sigma = \frac{1}{2\pi\sigma^2} e^{-\frac{(x^2+y^2)}{2\sigma^2}} \tag{9.2}$$

In the resultant scale-space representation, further information on ridge-like features can be extracted from the eigenvalues and eigenvectors of the Hessian, which show characteristic behaviour for a filamentous structure. By ordering the eigenvalues in terms

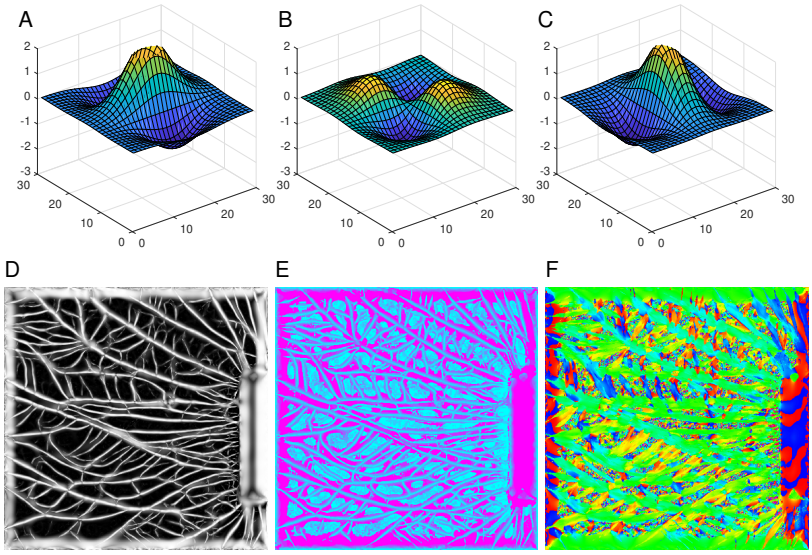


Figure 9.1: Frangi 'Vesselness' filter. (A-C) show surface plots of the D_{xx}, D_{xy} and D_{xy} filters at scale 7 that are used to calculate the second-order derivative of the image. The filters are shown inverted to highlight the shape of the ridge detector; (D) The 'Vesselness' output, calculated over 11 scales; (E) The scale at which the maximum response occurred; (F) the orientation of the maximum response.

of their absolute magnitude ($|\lambda_1| < |\lambda_2|$, for a 2D image), the smallest eigenvalue ($|\lambda_1|$) denotes the minimum change in intensity, with the corresponding eigenvector oriented along the centreline of the ridge, whilst the largest eigenvalue ($|\lambda_2|$) and eigenvector determine the orientation of the maximum curvature, normal to the ridge centreline. Prominent structures are distinguished from the background by relatively large values of the eigenvalues ($\sqrt{\lambda_1^2 + \lambda_2^2}$). In addition, the ratio $|\lambda_1|/|\lambda_2|$ gives an indication of how blob-like ($|\lambda_1| \approx |\lambda_2|$) or elongated and filament-like ($|\lambda_1| \ll |\lambda_2|$) the structure is at that point. Thus the 'Vesselness' (V_σ) measure (Frangi *et al.* 1998), defined by Equation (9.3), is large at those pixels that are part of a linear structure of scale (σ).

$$V_\sigma = e^{-\frac{\lambda_1^2}{2\beta^2\lambda_2^2}}\left(1 - e^{\frac{\lambda_1^2+\lambda_2^2}{2c^2}}\right) \tag{9.3}$$

Note that the relative contributions of the geometric ratio and the intensity components at a given scale (σ) are controlled by the coefficients β and c, respectively. Typically, β is set to 0.5 and c is set to half the maximum Hessian norm. Multi-scale 'Vesselness', for a given set of scales spanning the expected width of the vessels, can be computed as the maximum of the 'Vesselness' values calculated at each scale (Fig. 9.1D,E), and the eigenvectors at that scale used to define local orientation (Frangi *et al.* 1998).

Note: whilst contrast for the tubular regions is improved, only the edges of the cisternal regions are retained in the enhanced image.

9.3 *Meijering 'Neuriteness'*

An alternative weighting of the eigenvalues of the Hessian matrix was proposed by Meijering *et al.* (2004)[2]:

[2] E. Meijering, M. Jacob, J. Sarria, P. Steiner, H. Hirling, and M Unser. Design and validation of a tool for neurite tracing and analysis in fluorescence microscopy images. *Cytometry*, 58: 167 – 176, 2004

$$H' = \begin{bmatrix} D_{xx} + \alpha D_{yy} & (1-\alpha)D_{xy} \\ (1-\alpha)D_{xy} & D_{yy} + \alpha D_{xx} \end{bmatrix} \tag{9.4}$$

Where α is set to be $-1/3$ such that the filter used in the calculation of the Hessian matrix is maximally flat in its longitudinal direction, effectively generating an anisotropic second order Gaussian filter (Fig. 9.2A). Conveniently, these kernels can be implemented as steerable filters constructed from a set of basis kernels[3]. The 'Neuriteness' measure at scale σ, (N_σ) is determined from the modified eigenvalues as:

[3] W.T. Freeman and E.H. Adelson. The design and use of steerable filters. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 9:891–906, 1991

$$N_\sigma = \begin{cases} \frac{\lambda_\sigma}{\lambda_{\sigma,min}} & \text{if } \lambda_\sigma < 0 \\ 0 & \text{if } \lambda_\sigma \geq 0 \end{cases} \tag{9.5}$$

where λ_σ is the larger in absolute magnitude of the two modified eigenvalues, and $\lambda_{\sigma,min}$ is the smallest value of λ over all pixels such that:

$$\lambda_{\sigma,1}{}' = \lambda_{\sigma,1} + \alpha\lambda_{\sigma,2} \tag{9.6}$$

$$\lambda_{\sigma,2}{}' = \lambda_{\sigma,2} + \alpha\lambda_{\sigma,1} \tag{9.7}$$

$$\lambda_\sigma = \max\left(|\lambda_{\sigma,1}{}'|, |\lambda_{\sigma,2}{}'|\right) \tag{9.8}$$

$$\lambda_{\sigma,min} = \min_{\mathbf{p}\in I}(\lambda_\sigma) \tag{9.9}$$

$\lambda_{\sigma,1}, \lambda_{\sigma,2}$ are the eigenvalues of the Hessian matrix $H_\sigma(\mathbf{p})$, at pixel $\mathbf{p}$, for a given scale parameter σ. The maximum response across all scales gives the 'Neuriteness' image (Fig. 9.2C)

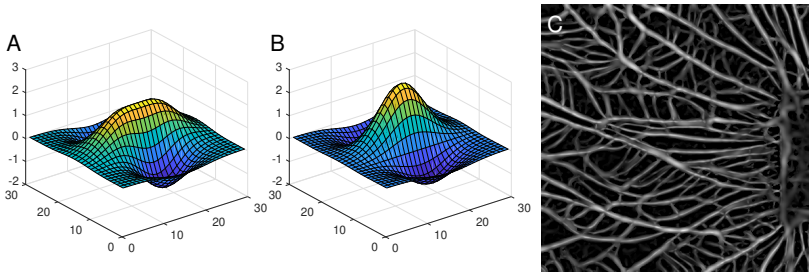


Figure 9.2: Meijering 'Neuriteness' filter. (A) A surface plot for the 'flattened' D_{xx} filter, shown inverted to highlight the shape of the ridge detector; (B) A normal D_{xx} filter for comparison; (C) The 'Neuriteness' output, calculated over 5 scales.

9.4 *Second-order anisotropic Gaussian kernels (SOAGK)*

The use of second-order derivatives of anisotropic Gaussian kernels (SOAGKs) was developed further by Shui *et al.* (2012)[4] and Lopez-Molina *et al.* (2015)[5] to improve detection of ridge like elements. The SOAGK are applied at a range of orientations to give anisotropic directional derivative (ANDD) filters at each scale (Fig. 9.3, A-C). On their own ANDD filters also generate extensions at the end of edge segments, termed edge-stretch, which has the benefit of improving local connectivity by filling in small gaps, particularly at junctions that occur in the 'Vesselness' filter for example,

[4] P.-L. Shui and W.-C. Zhang. Noise-robust edge detector combining isotropic and anisotropic gaussian kernels. *Pattern Recognition*, 45:806 – 820, 2012

[5] C. Lopez-Molina, G. V. D. de Ulzurrun, J. M. Baetens, J. Van den Bulcke, and B. De Baets. Unsupervised ridge detection using second order anisotropic gaussian kernels. *Signal Processing*, 116:55–67, 2015

but with the disadvantage of adding spurious features at the end of edge segments. The latter errors can be minimised by using a fused detector that combines the ANDD filter with a small isotropic Gaussian as a geometric mean (Shui and Zhang, 2012). The enhanced edge image is taken as the maximum response at any scale and orientation (Fig. 9.3, E). These filters give strong responses when aligned to the dominant ridge at each scale, and provide estimates of the ridge intensity and ridge orientation without calculation of the eigenvalues and eigenvectors. In addition, the response at junctions is not attenuated to the same degree as the 'Vesselness' response because of the edge-stretch phenomena. Conversion to a single-pixel wide skeleton then uses local non-maximal suppression to identify key pixels on the ridge centerline, followed by hysteresis thresholding to identify pixels that form the connected skeleton.

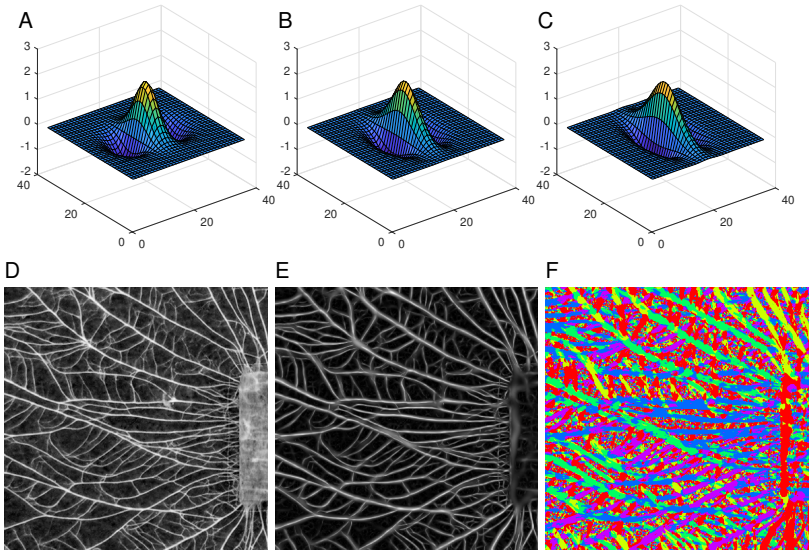


Figure 9.3: Second-order anisotropic Gaussian filter (SOAGK) after Lopez-Molina *et al* (2015). (A-C) Surface plots for the D_{xx} filter with 1,1.3 and 1.5 levels of anisotropy, shown inverted to highlight the shape of the ridge detector; (D) Original *Physarum* image (E) Output from the 'SOAGK' filter, calculated over 7 scales; (F) direction of the dominant ridge.

9.5 *Intensity-independent enhancement using phase-congruency*

While ridge enhancement can be built on purely intensity-based filters, such as the Hessian or SOAGKs, these have the downside of being sensitive to changes in image contrast, which often leads to loss of a few pixels from the skeleton during the subsequent thresholding step, effectively disconnecting these edges. This can be ameliorated to some extent by inclusion of a local contrast equalisation step prior to enhancement (Shui and Zhang, 2012), or by the use of adaptive or hysteresis thresholding during segmentation (Lopez-Molina *et al.* 2015). Nevertheless, in these approaches it is critical to establish a reliable, context-dependent threshold selection to achieve segmentation of a fully connected network.

Human observers face a similar challenge when trying to discriminate edges or ridges is a complex visual field. Morrone and Owens (1987)[6] proposed that human edge perception depends on

[6] M.C. Morrone and R.A. Owens. Feature detection from local energy. *Pattern Recognition Letters*, 6:303 – 313, 1987

the degree of phase congruency, which is independent of the image brightness. Phase congruency can be estimated from the local energy at each location, determined by convolution of the image with Gabor filters at varying scale and orientation[7]. The phase congruency approach has been developed further as a generic means to extract a variety of image features by Kovesi[8]. The approach exploits the knowledge that in a Fourier decomposition of an image, edges or ridges are defined by a high degree of phase congruency in the underlying sinusoidal components 9.4.

[7] S. Venkatesh and R. Owens. On the classification of image features. *Pattern Recognition Letters*, 11:339–349, 1990

[8] P. Kovesi. Image features from phase congruency. *Videre: Journal of Computer Vision Research*, 1:1–26, 1999

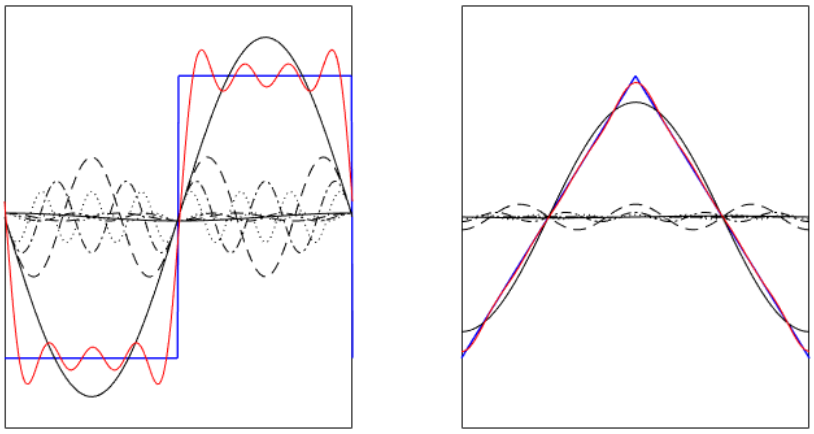


Figure 9.4: Construction of square and triangular waveforms from their Fourier series. In both diagrams, the first few terms of the respective Fourier series are plotted with broken lines; the sum of these terms is the solid line. Notice how the Fourier components are all in phase at the point of the step in the square wave, and at the peaks and troughs of the triangular wave. Based on Kovesi, 1999

Kovesi also introduced a range of improvements to the original measure to improve its overall utility, including log Gabor filters to increase the filter bandwidth, procedures for automated noise rejection, weighting to select against phase congruency of only a few frequencies, and improved spatial precision by including both the cosine and sine of the phase in the estimate. These provide good ridge enhancement, irrespective of image intensity, but also increase the number of parameters that can be tuned to achieve the best enhancement in any particular context.

A complete description of the Kovesi phase-congruency programme is available from:

`http://www.csse.uwa.edu.au/~pk/Research/MatlabFns/PhaseCongruency/Docs/convexpl.html`

Following Kovesi, the local energy for a one-dimensional profile, $I(x)$ is given by:

$$E(x) = \sqrt{F^2(x) + H^2(x)} \tag{9.10}$$

where $F(x)$ is the signal with its DC component removed, and $H(x)$ is the Hilbert transform of $F(x)$, obtained by convolving the signal with a quadrature pair of log Gabor filters at scale n, and summing the even filter convolutions $(e_n(x))$ to give $F(x)$, and the odd filter convolutions $(o_n(x))$ to give $H(x)$. The phase congruency $PC(x)$, is normalised by the sum of the Fourier amplitudes $\sum_n A_n(x) \simeq \sum_n \sqrt{e_n(x)^2 + o_n(x)^2}$, with the addition of a small

constant ε to improve stability at low Fourier amplitudes:

$$PC(x) = \frac{E(x)}{\sum_n A_n + \varepsilon} \tag{9.11}$$

The log Gabor filter bank is controlled by the minimum wavelength scale, the number of scales, the frequency bandwidth, and the number of filter orientations. Each of these requires some optimisation to achieve good enhancement for specific types of biological networks (Fig. 9.5B).

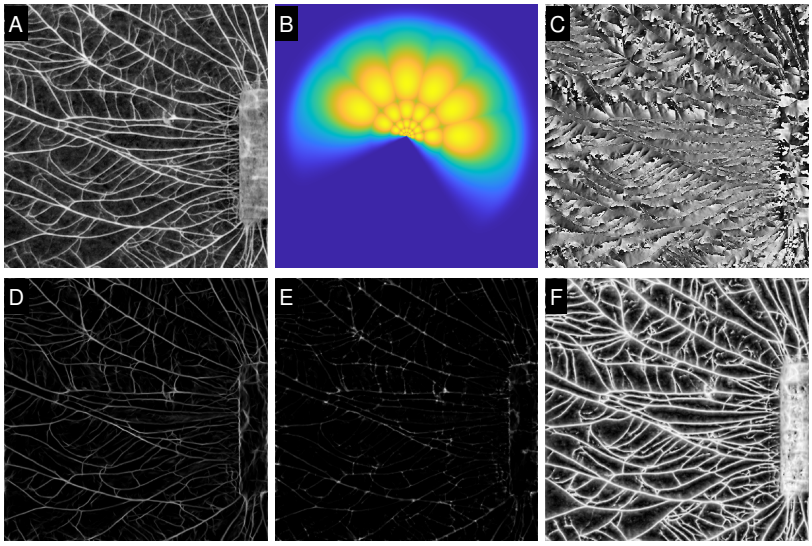


Figure 9.5: Phase Congruency filter. (A) Original image; (B) Set of log Gabor filters in the Fourier domain for 6 scales and 6 orientations; (C) Orientation image from the maximum response; (D) Maximum moment from the phase-congruency filter as a measure of edge strength of the tubules; (E) Minimum moment from the phase-congruency response, which highlights junctions; (F) Local weighted mean phase angle ('Feature Type'), which provides a robust output for segmentation.

The next tuneable parameter controls the amount of noise rejection. To estimate the amount of noise adaptively from the image, Kovesi used a measure of the mean (μ_R) and variance (σ_R^2) of the Rayleigh distribution (R) describing the noise distribution at the smallest scale, with the assumption that ridges are relatively sparsely distributed in the image, so the mean at this scale will be dominated by background noise. Thus the noise threshold (T), is given by the mean noise response plus some number, k, of deviation units:

$$T = \mu_R + k\sigma_R \tag{9.12}$$

The local energy term $E(x)$ is therefore modified by subtracting the estimated noise (and setting any values below zero to zero).

The second set of tuneable parameters relate to the minimum spread of frequencies required to constitute a useful estimate of phase congruency. In the case of the ER network we are concerned with the detection of ridges, rather than lines or step functions. The expected power spectrum of a ridge falls off at $1/\omega^4$, where ω is the centre frequency of the filter, which gives an expected distribution of frequency responses strongly skewed towards low-frequency end. The significance of $PC(x)$ can be down-weighted if the spread of frequencies is too narrow, however, in the case

of ridge detection, this criterion should not be too harsh. Kovesi provides an estimate of the frequency spread by considering the normalised ratio of the sum of the Fourier amplitudes divided by the maximum response:

$$s(x) = \frac{1}{N}\left(\frac{\sum_n A_n(x)}{A_{max}(x) + \varepsilon}\right) \tag{9.13}$$

Where N is the total number of scales, $A_{max}(x)$ is the maximum filter response at x, and ε prevents division by zero. To penalise regions with few frequency components, the weighting function is constructed as a sigmoidal function:

$$W(x) = \frac{1}{1 + e^{\gamma(c - s(x))}} \tag{9.14}$$

where c is the cut-off value below which phase congruency values are penalised, and γ is a gain factor that controls the sharpness of the cut-off

$$PC(x) = \frac{W(x)\lfloor E(x) - T \rfloor}{\sum_n A_n(x) + \varepsilon} \tag{9.15}$$

Where $\lfloor\ \rfloor$ denotes that $E(x) - T$ is equal to itself for positive values and zero otherwise.

The final amendment that Kovesi proposed is to include the information from both the cosine of the phase deviation, which should be large if phase congruency is high, and the absolute value of the sine of the phase deviation, which should be small (9.16):

$$\Delta\Phi_n(x) = \cos(\phi_n(x) - \bar{\phi}(x) - |\sin(\phi_n(x) - \bar{\phi}(x)| \tag{9.16}$$

This gives the complete estimate of phase congruency as:

$$PC(x) = \frac{\sum_n W(x)\lfloor A_n(x)\Delta\Phi_n(x) - T \rfloor}{\sum_n A_n(x) + \varepsilon} \tag{9.17}$$

The two dimensional extension of the phase congruency gives:

$$PC(x) = \frac{\sum_o \sum_n W_o(x)\lfloor A_{no}(x)\Delta\Phi_{no}(x) - T_o \rfloor}{\sum_o \sum_n A_{no}(x) + \varepsilon} \tag{9.18}$$

Where o is the index over orientations.

9.6 *Parameter settings for phase-congruency*

Our preferred enhancement method uses the phase congruency approach developed by Peter Kovesi[9], so we provide more detail on the parameter settings used. On the main Fungal Network Analysis GUI, only the number of scales can be altered interactively. Access to the other parameters is via the **Edit** button in the **Param** panel and affects the following parameters in the *phasecong3.m* program[10].

[9] P. Kovesi. Image features from phase congruency. *Videre: Journal of Computer Vision Research*, 1:1–26, 1999

[10] P. D. Kovesi. MATLAB and Octave functions for computer vision and image processing, 2000. Available from: http://www.peterkovesi.com/matlabfns/

9.6.1 *Selecting the size of the smallest object (maximum frequency response)*

The smallest edge that can be detected is related to the maximum frequency of the filter, and is set by the wavelength of the smallest scale filter (*enhance_PC_minWavelength*). The minimum value is 2 pixels, representing the Nyquist limit. However, this wavelength also gives considerable aliasing, so the recommended value is 3 pixels or more. The default is to set *minWavelength* to the **target width**.

9.6.2 *The filter bandwidth (sigmaOnf)*

The filter bandwidth is set by specifying the ratio of the standard deviation of the Gaussian describing the transfer function of the log Gabor filter in the log-frequency domain, to the filter center frequency. This is set by the *enhance_PC_sigmaOnf* slider. The smaller *sigmaOnf* is the larger the bandwidth of the filter. Peter Kovesi reports that empirically a *sigmaOnf* value of 0.75 will result in a filter with a bandwidth of approximately 1 octave and a value of 0.55 will result in a bandwidth of roughly 2 octaves. The default is 0.55.

9.6.3 *Scaling between centre frequencies (mult)*

Having set the filter bandwidth (*sigmaOnf*), the scaling between centre frequencies of successive filters can be set using the *enhance_PC_mult* to achieve the minimal overlap necessary to achieve fairly even spectral coverage. The default is 2.1.

Table 9.1 below gives the values determined experimentally by Peter Kovesi [11]

[11] P. Kovesi. Image features from phase congruency. *Videre: Journal of Computer Vision Research*, 1:1–26, 1999

sigmaOnf	Scale factor	bandwidth
0.85	1.3	-
0.75	1.6	1 octave
0.65	2.1	-
0.55	3.0	2 octaves

Table 9.1: Setting appropriate combinations of scale factor and filter bandwith

9.6.4 *Selecting the size of the largest feature (minimum frequency response)*

The minimum frequency is set by the wavelength of the largest scale filter. This is implicitly defined once the wavelength of the smallest scale filter (*enhance_PC_minWavelength*), the number of filter scales (*enhance_PC_nScales*), and the scaling between centre frequencies of successive filters (*enhance_PC_mult*) are set.

9.6.5 *The number of filter orientations (nOrient)*

The number of filter orientations is set by *enhance_PC_nOrient* and specifies the angular resolution of the filters in conjunction with the

angular spread of each filter. The default of six orientations seems to work well.

9.6.6 The angular spread of each filter

The angular interval between filter orientations is fixed by the number of filter orientations. In the frequency domain the spread of 2D log-Gabor filter in the angular direction is simply a Gaussian with respect to the polar angle around the centre. The angular overlap of the filter transfer functions is controlled by the ratio of the angular interval between filter orientations and the standard deviation of the angular Gaussian spreading function. Within the code this ratio is controlled by the parameter *dThetaOnSigma*. A value of *dThetaOnSigma* = 1.5 results in approximately the minimum overlap needed to get even spectral coverage. At present this parameter is fixed within the code and cannot be altered by the user.

9.6.7 Controlling for limited frequency spread

The *enhance_PC_cutOff* sets the fractional measure of the frequency spread below which phase congruency values get penalized. This is to ensure that sufficient frequencies are present to make a meaningful estimate of the phase congruency. The default is 0.5. The penalty function is further controlled by *enhance_PC_g*, which sets the sharpness of the transition in the sigmoid function used to weight phase congruency for frequency spread. The default value is 5.

9.6.8 Noise masking

The *enhance_PC_k* sets the number of standard deviations of the noise energy beyond the mean at which the noise threshold is set. The default value is 2, but can be increased to a value of 10-20 for noisy images. The metric used in the noise calculation is set by the *enhance_PC_noiseMethod* parameter from the following options:

- -1 - use median of smallest scale filter responses (default).
- -2 - use mode of smallest scale filter responses.
- 0 - use noiseMethod value as the fixed noise threshold.

10
Skeleton extraction

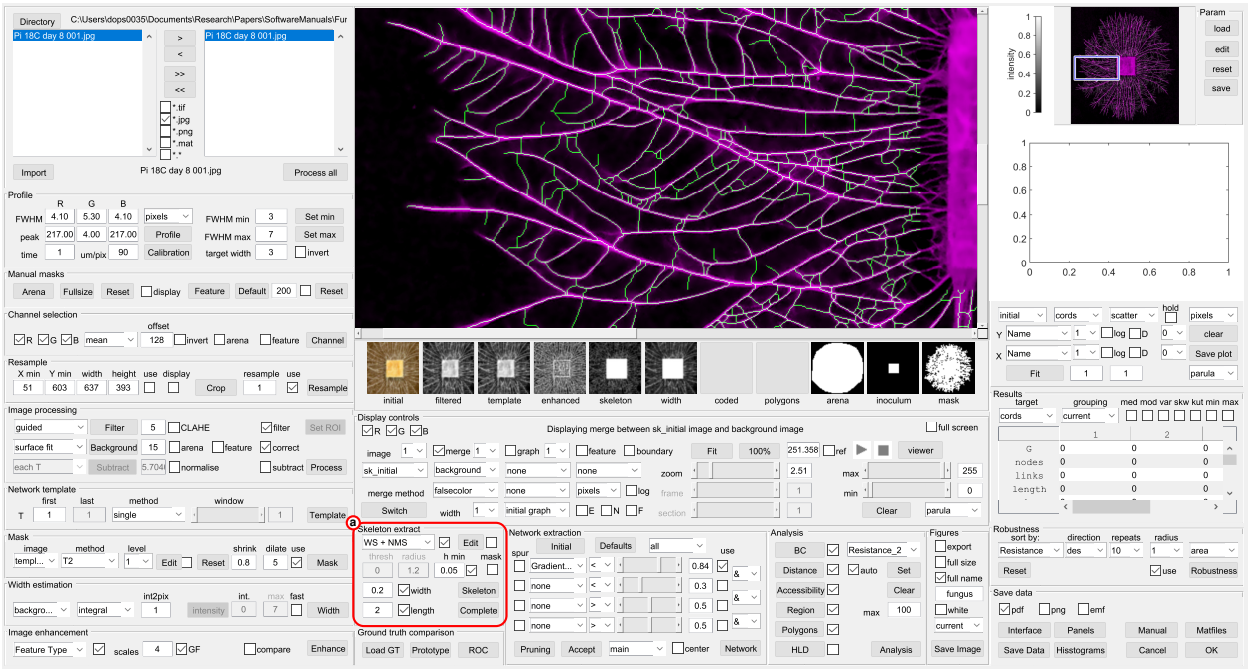


Figure 10.1: Controls for skeleton extraction

10.1 Introduction

Skeletonisation involves conversion of the enhanced image to a binary image, followed by thinning to give a single pixel wide skeleton. Both these operations can be implemented in a variety of different ways, and again a number of options are provided in the Fungal Network Analysis GUI, each with different strengths and weaknesses. Often different binarization methods operate in concert with particular enhancement approaches, so it is best to consider them in tandem.

10.2 Skeletonisation

The aim of the skeletonization step is to convert the enhanced image to a one-pixel wide skeleton along the centre-line of the

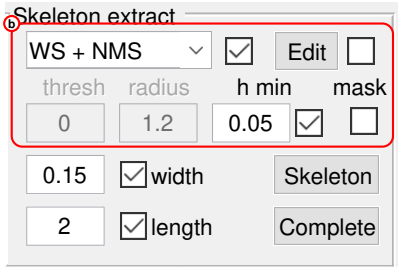


Figure 10.2: The initial skeletonization controls in the skeleton extraction panel. Note the labels may change depending on the method selected

cords or hyphae. This only gives an approximation to the true centre-line due to the pixel discretisation errors, but is one area where up-sampling the image during the rescale operation may be beneficial, but at the expense of computational time. A range of options are available from the left-hand drop down menu (Fig. 10.2), that exploit slightly different characteristics of the enhanced image, including:

- *'hysteresis'* : applies hysteresis thresholding using a lower threshold set by the **thresh** textbox. The upper threshold used to define the seed points is automatically set as *lower threshold* + 0.2. Hysteresis thresholding starts with seed pixels above the upper threshold, and then propagates the initial segmentation as long as pixels remain above the lower threshold. The resulting binary image is then thinned to give the single-pixel skeleton. The value of the lower threshold is critical - too high and the network becomes disconnected; too low and large blocks of the image are included in the resultant binary image that may fuse separate tubules into a single object. When this block is thinned, the skeleton does not map onto the ridge centre-lines. In addition, as the thinning process is not guided by the intensities in the enhanced image, the skeleton does not necessarily converge on the expected pattern at junction points.

- *'watershed'* : applies a watershed segmentation and then extracts the watershed lines as the pixel skeleton. The watershed is better at segmenting the centre-line of the ridges than other methods, and can handle variations in intensity well. However, it does not include any cords or hyphae that have a free end, and has a tendency to over-segment regions with noise. Over-segmentation can be avoided by including additional steps that suppress regions with small intensity fluctuations, such as the h-minimum transform (see Section 10.3).

- *'WS + NMS'* : Combines watershed segmentation and binarisation with non-maximal suppression (NMS) using an orientation image that gives the feature normal orientation angles in degrees (0-180). This uses the *featureorient.m* and *nonmaxsup.m* algorithm from Peter Kovesi[1]. The NMS method segments hyphal tips well, but often fails at junctions where the intensity drops. Conversely, the watershed algorithm ensures connectivity at junctions, but

[1] P. D. Kovesi. MATLAB and Octave functions for computer vision and image processing, 2000. Available from: http://www.peterkovesi.com/matlabfns/

fails at hyphal tips. The combination provides a robust approach to segmenting the network. Neverthless, the watershed algoithm still over-segments the background and requires additional use of the h-minimum transform (see Section 10.3) and pruning (Chapter **??**).

- *mean*, *median* or *Gaussian* : These apply a local adaptive threshold using the Matlab *adaptthresh.m* function. The local mean, median or Gaussian-weighted mean is calculated using a circular filter and used as a local threshold. The radius is set as a multiplier of the $factor \times FWHM_{max}/resample$ where the *factor* is given by the **radius** text box. The sensitivity is set between [0 1] using the **sens.** text box with higher values increasing the number of foreground pixels.

- *midgrey* : The mid-grey threshold is calculated as the average of the local maximum and minimum, calculated using morphological closing and opening operations, respectively, with the radius set as before. An additional offset k can be set using the **weight** text box to bias the segmentation to higher or lower values, with a default of 0.5.

$$T(x,y) = \frac{max + min}{2} - k \tag{10.1}$$

- *Niblack*[2]*:* The local threshold ($T(x,y)$) is calculated from the local mean ($m(x,y)$) and the local standard deviation ($s(x,y)$), with the radius set by the **radius** textbox, weighted by a factor (k) set using the **weight** textbox.

[2] W. Niblack. *An Introduction to Image Processing*. Prentice-Hall, 1986

$$T(x,y) = m(x,y) + k * s(x,y) \tag{10.2}$$

where k is typically -0.2.

- *Bernsen*[3]*:* The method calculates the local contrast and mid-grey values in a similar manner to the mid-grey algorithm from morphological operations. If the local contrast is above the contrast value k set by the **weight** textbox (typically 0.5), the threshold is set at the local mid-grey value. If the local contrast is below the contrast threshold, the pixel is classified as part of the object if the mid-grey value is above 0.5, or background if below.

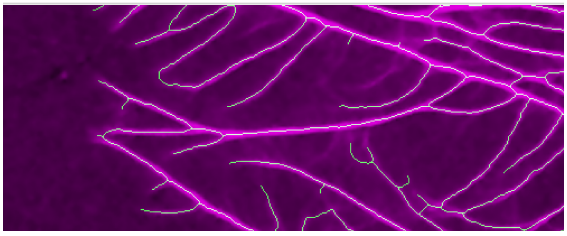
[3] J. Bernsen. Dynamic thresholding of grey-level images. In *International conference on pattern recognition*, volume 2, pages 1251–1255, 1986

- *Sauvola*[4]*:* The local threshold ($T(x,y)$) is calculated in a similar manner to the Niblack algorithm from the local mean ($m(x,y)$) and local standard deviation ($s(x,y)$), with the radius set as before, but the weighting is calculated as:

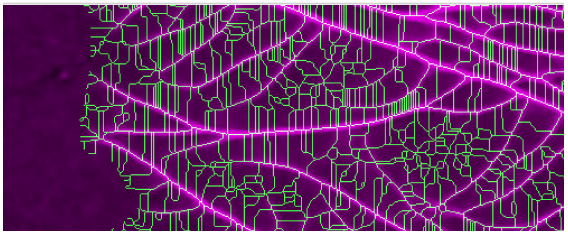
[4] J. Sauvola and M. Pietikäinen. Adaptive document image binarization. *Pattern recognition*, 33:225–236, 2000

$$T(x,y) = m(x,y)\left[1 + k\left(\frac{s(x,y)}{R} - 1\right)\right] \tag{10.3}$$

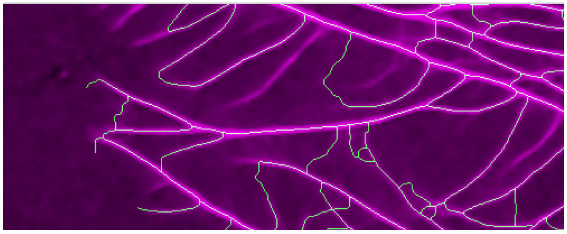
where $R = max(s)$ and k takes small positive values, typically in the range 0.2- 0.5, set by the **min** textbox.



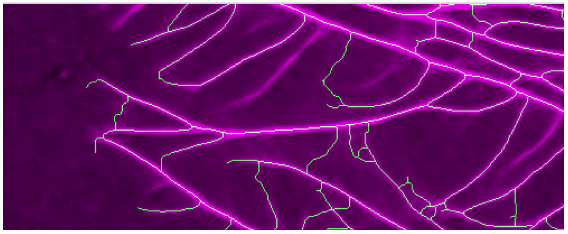
(a) Hysteresis threshold, min 0.3 max 0.5



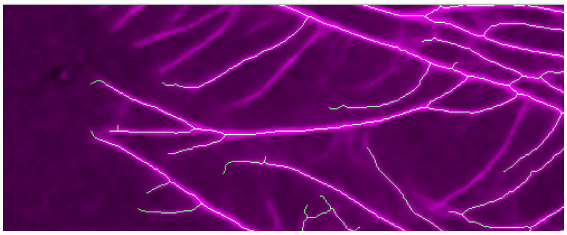
(b) Watershed segmentation



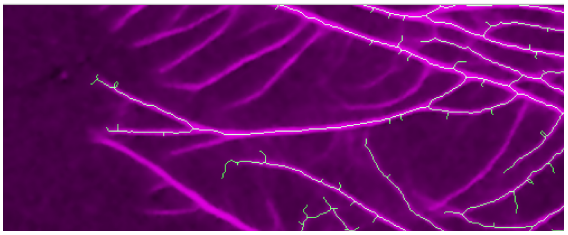
(c) Watershed with extended h-minimum transform



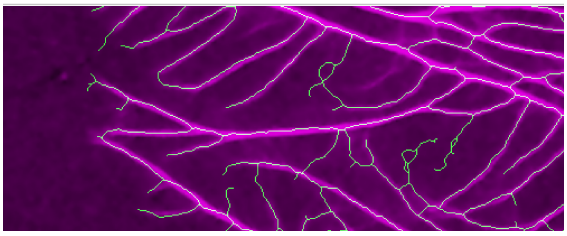
(d) Watershed combined with non-maximum suppression



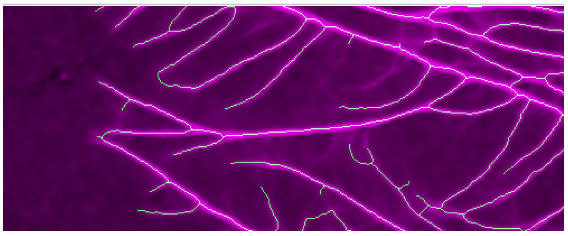
(e) adaptive threshold using the local mean



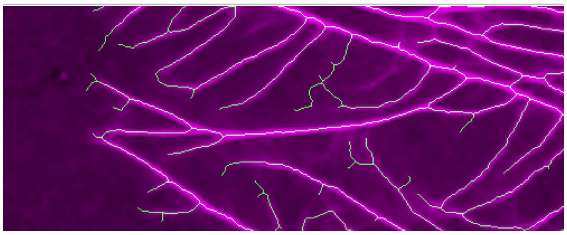
(f) adaptive threshold using the local median



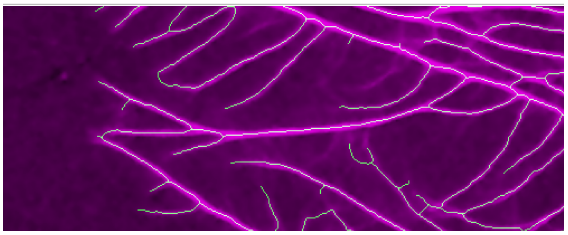
(g) adaptive threshold using a Gaussian-weighted local mean



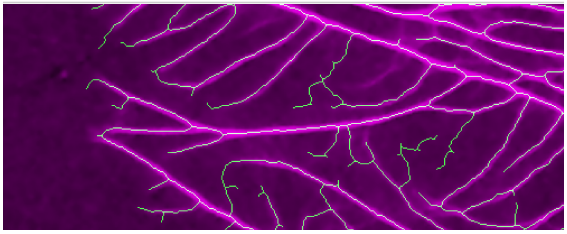
(h) midgrey



(i) Niblack



(j) Bernsen

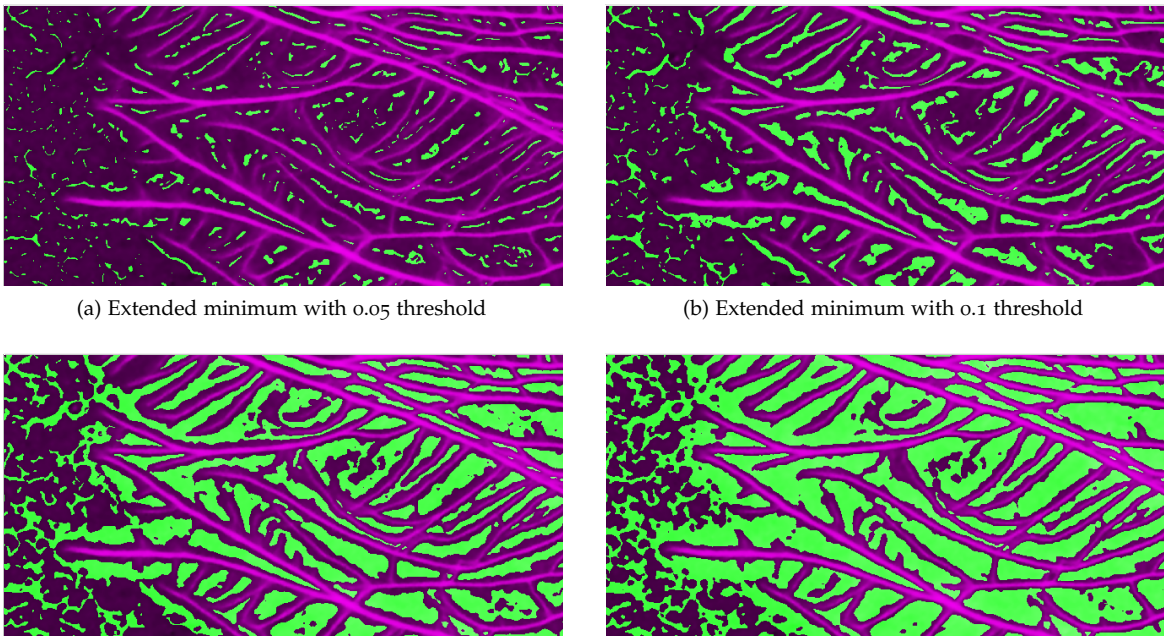


(k) Sauvola

Figure 10.3: Examples of the different skeletonisation algorithms

10.3 *Suppressing background using the extended h-minimum transform*

The extended h-minimum transform is calculated using the Matlab *imextendedmin.m* function and used to suppress all minima in the image whose depth is less than a threshold set by the **hmin** text box. These regions are then set to zero using the Matlab *imimposemin.m* function. This is useful to prevent the watershed algorithm in particular adding many spurious cross-connections over noisy background regions. The effect of the *h-minimum* transform can be toggled on and off using the adjacent checkbox. The extended h-minimum is also useful to force separation of closely aligned cords or hyphae as it will identify small dips between adjacent elements and set these to zero.



(a) Extended minimum with 0.05 threshold

(b) Extended minimum with 0.1 threshold

(c) Extended minimum with 0.2 threshold

(d) Extended minimum with 0.3 threshold

Figure 10.4: Impact of varying h-minimum thresholds used to mask out background regions (green)

Regions outside the boundary of the colony are automatically set to zero. In addition, the **mask** checkbox can also be toggled on and off to set regions outside the mask to zero. This may have the effect of breaking some links between cords/hyphae that traverse the background, but will leave behind short spurs that span the width of the mask to the centre-line. These additional elements can be pruned from the skeleton by truncation (see Section 10.4)

10.4 *Truncating the skeleton*

It is rare that the skeletonisation procedure yields an accurate skeleton for the entire network with no further intervention. A decision is usually required on the direction to bias the initial skeleton extraction to be cleaner, but potentially missing key connections, or fully-connected, but containing some spurious edges and con-

nections. Our preference is to over-connect initially, as capturing the entire network is regarded as most important for subsequent network analysis and flow-modelling, so our approach is to over-connect and then prune the skeleton.

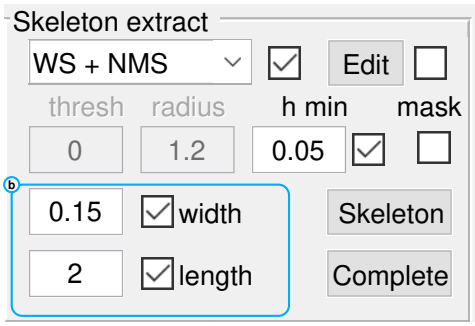


Figure 10.5: Controls to truncate the skeleton based on width or length

Two options are provided in the **skeleton extract** panel, to break the skeleton for pixels below a minimum width, and to truncate the skeleton based on the minimum length of spurs (Fig. 10.5). The minimum width is calculated as $factor * FWHM_{min}/rescale$, where the factor is set by the **width factor** text box (default 0.25). The adjacent checkbox toggles whether to apply the width constraint. The minimum length of spurs is also defined in terms of the size of the hyphae/cord as $factor * FWHM_{max}/rescale$, where the factor is set by the **length factor** textbox (default 2). The adjacent checkbox toggles the length constraint on and off.

10.5 Editing the skeleton

If the automated methods to delineate the pixel skeleton are still unsatisfactory, the skeleton can be edited manually using the **edit** button. This opens the binary editing program described in Chapter 16. The manually-edited skeleton is saved in the parameters file and will be re-used if the **use edit** checkbox is ticked.

11
Parameter selector

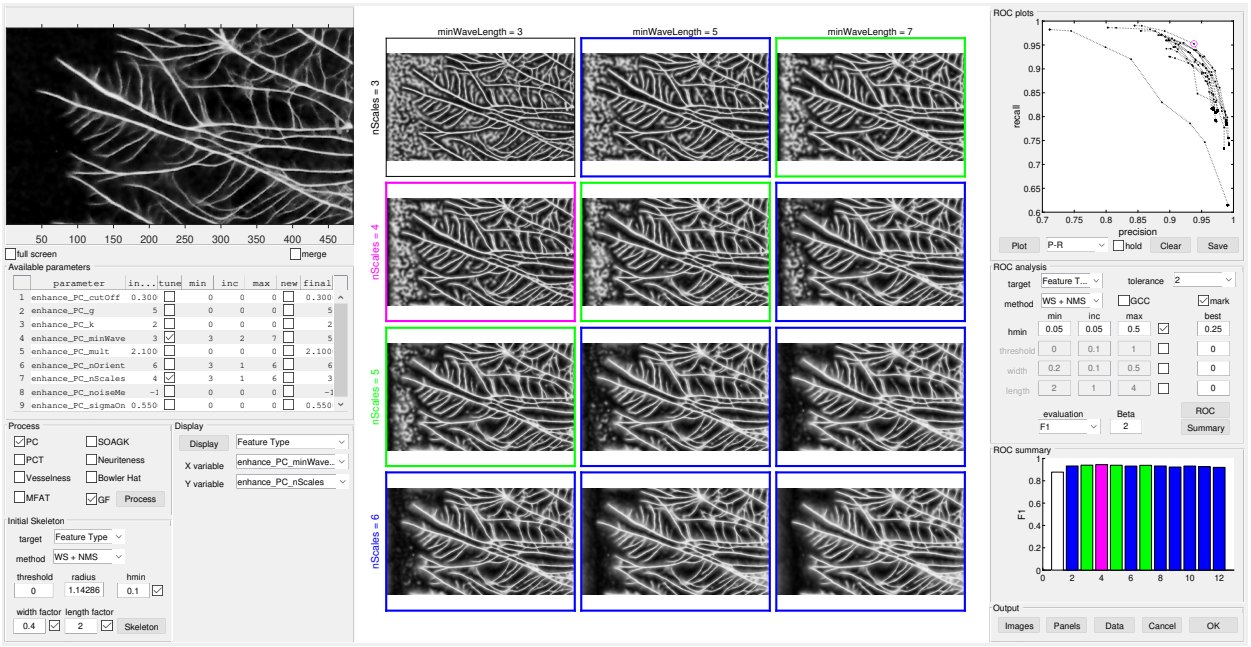


Figure 11.1: The GUI interface for the parameter selector program

11.1 Sensitivity analysis with factorial parameter combinations

The parameter selector panel allows a factorial combination of parameters to be applied to a region-of-interest (ROI) cropped from the image to compare with a manually-defined ground-truth. The program is called from the **Prototype** button in the *Skeleton extract* panel of the main interface (Fig. 11.2(a)).

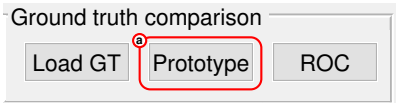


Figure 11.2: The **Prototype** button (a) in the Ground truth comparison panel on the main interface

The algorithm to apply to the ROI is chosen using the check-boxes in the *Process* panel (Fig. 11.3). At the moment, only one algorithm should be selected at a time using the appropriate check-box, in this case *PC* for phase-congruency. The other options match

the options available in the main interface. In a later version, multiple choices are planned to aid comparison between methods. Once a method has been selected, the parameter options specific to that method are displayed in the *Available parameters* panel (Fig. 11.4).

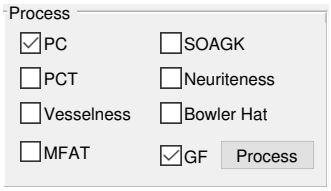
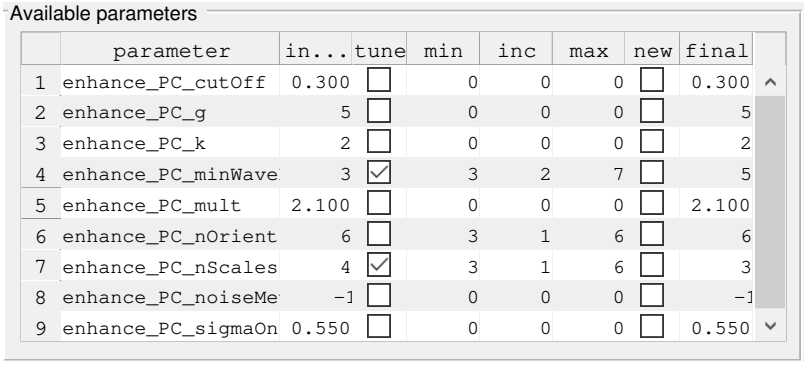


Figure 11.3: Check-boxes to select the enhancement algorithm



	parameter	in...	tune	min	inc	max	new	final
1	enhance_PC_cutOff	0.300	☐	0	0	0	☐	0.300
2	enhance_PC_g	5	☐	0	0	0	☐	5
3	enhance_PC_k	2	☐	0	0	0	☐	2
4	enhance_PC_minWave	3	☑	3	2	7	☐	5
5	enhance_PC_mult	2.100	☐	0	0	0	☐	2.100
6	enhance_PC_nOrient	6	☐	3	1	6	☐	6
7	enhance_PC_nScales	4	☑	3	1	6	☐	3
8	enhance_PC_noiseMe	-1	☐	0	0	0	☐	-1
9	enhance_PC_sigmaOn	0.550	☐	0	0	0	☐	0.550

Figure 11.4: Table showing the parameters that can be tuned for the selected algorithm

The current value for each parameter is shown in the **initial** column, and cannot be edited. A minimum of two parameters must be selected using the **tune** check-boxes, and the range to explore set using the minimum value (**min**), the increment (**inc**), and the maximum value (**max**). These two parameters are automatically set as the **X variable** and **Y variable** in the *Display* panel drop-down menus (Fig. 11.5).

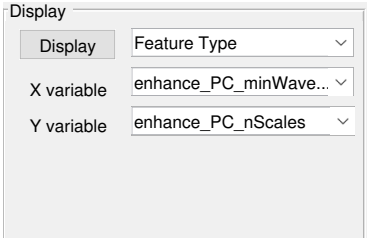


Figure 11.5: Controls to display a sub-set of the processed images

Each additional parameter selected for tuning, is displayed below the drop-down menus as a slider bar running between **Min** and **Max**. These sliders are used to scroll through the images for display. In principle every parameter can be tuned in this way, but the number of factorial combinations will increase very rapidly, so it is wise to explore a more limited sub-set of parameters and values in the first instance, to progressively hone in on the best combination.

If a value for a parameter has already been optimised, the **new** check-box can be ticked in the *Available parameters* table, and the value entered in the **final** column.

When the **Process** button is clicked, the ROI will be processed using the factorial combination of the parameter range set. Once the ROI has been processed, the results are displayed as a 2-D grid (Fig. 11.6), labelled with the values for the first two parameters selected for tuning in the *Display* panel. Different parameters can be selected for the display grid using the drop-down menus and sliders. The type of image displayed can be selected using the **image** drop-down menu in the *Display* panel.

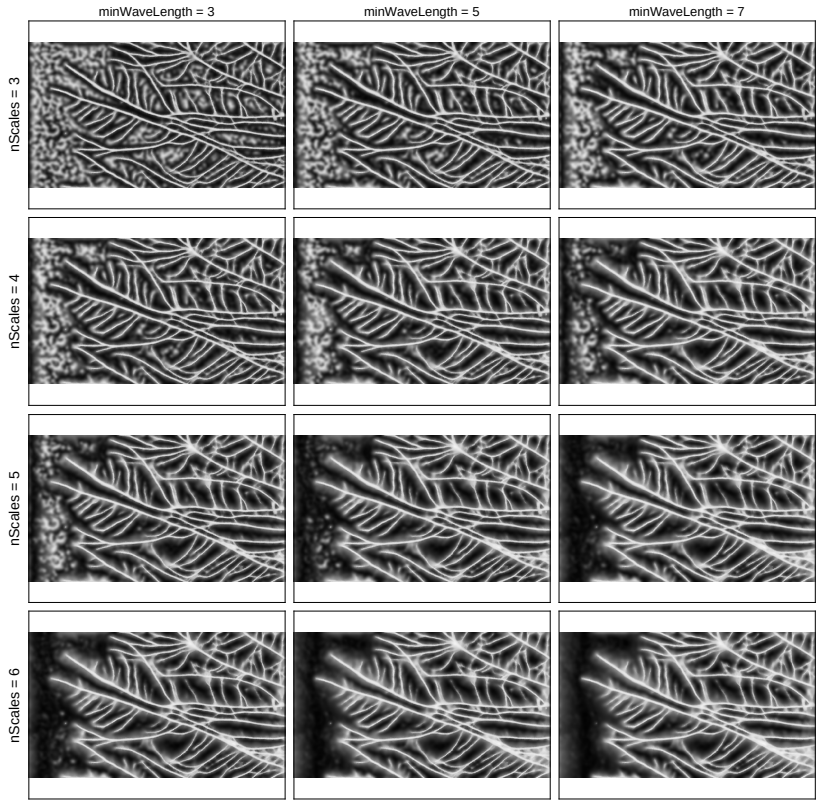


Figure 11.6: Grid showing results for the factorial sensitivity analysis for the feature type output from the phase congruency algorithm, where the minimum wavelength and the number of scales have been tuned

11.2 *Initial skeletonisation*

At the end of the processing step, the images are also automatically segmented using the values set in the *Initial skeleton* panel (Fig. 11.7).

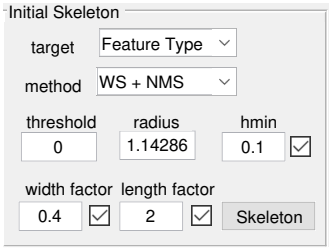


Figure 11.7: Initial segmentation controls

These follow the same options available in the main interface and process the enhanced **target** image to a single-pixel wide skeleton using the method selected using the **method** drop-down menu. Depending on the *method* selected, values for the **threshold**, **radius**, **hmin**, **width factor** and **length factor** are displayed based on the values currently in use in the main GUI (See Chapter 8). These can be edited individually if required, but can also be tuned for optimisation against the ground truth (Section 11.3). The skeleton can be displayed using the **image** drop-down menu. The skeletonisation parameters can be changed and just the skeletonisation step re-run on the set of enhanced images using the **Skeleton** button.

11.3 *Comparison with a ground-truth skeleton*

Pixel skeletons for each ridge enhancement-skeletonisation combination can be scored against a manually digitised ground-truth, within a chosen tolerance, typically around half the minimum hypha/cord width. Each pixel is classified as a true positive (TP), true negative (TN), false positive (FP), or false negative (FN). It is important that the ground-truth image is loaded into the main interface

using the **Load GT** button (Fig. 11.8) prior to calling the *Parameter selector* GUI, so that the same ROI is extracted for comparison.

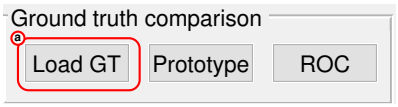


Figure 11.8: The **Load GT** button (b) in the Ground truth comparison panel on the main interface

The target enhanced image for comparison is selected using the **Target** drop-down menu in the **ROC analysis** panel, along with the segmentation **method**, and the **tolerance** (in pixels) that will still be considered as a 'true positive' relative to the Ground-Truth skeleton.

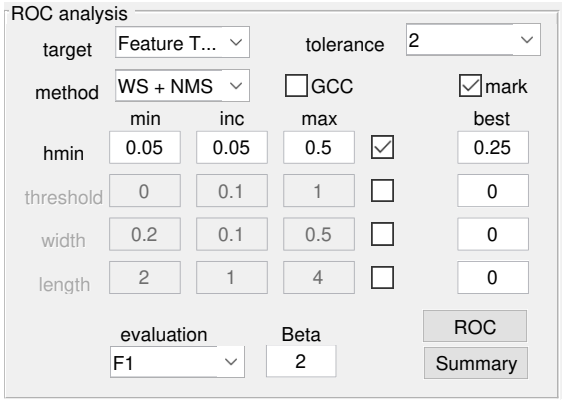


Figure 11.9: ROC analysis controls

If *hysteresis* thresholding is selected, the **threshold** options are enabled, and can also be set to run from **min** to **max**, with a given increment (**inc**). In a similar manner, the other skeletonisation parameters (**hmin**, **width factor** or **length factor**) can also be tuned to semi-automatically optimise against the ground-truth skeleton. The analysis can be restricted to the giant connected component (GCC), using the **GCC** check-box, as a crude test of connectivity in the resultant skeleton.

The **ROC** button initiates the analysis, and results are presented as a *ROC curve*, *P-R plot*, *z-score* or *histogram*, depending on the setting of the **ROC plot** drop-down menu in the **ROC plots** panel.

Precision-Recall (PR) analysis is used in preference to Receiver Operating Characteristic (ROC) plots, as the former are better suited to imbalanced datasets[1], when the number of true negatives (TNs) from the background is expected to be much greater than the true positives (TPs) from the skeleton. Precision was calculated as TP/(TP+FP), and Recall as TP/(TP+FN).

Results for each run of the *hmin* or *threshold* parameter are connected by a dotted line. It should be noted that *hmin* and *threshold* do not behave quite as a conventional tuneable threshold in ROC or PR analysis, as the output (a connected skeleton) does not necessarily correlate linearly with increasing or decreasing values of the parameter. For example, as the threshold is lowered, one might expect fewer FNs, more TPs, but also more FPs. However, if the

[1] T. Saito and M. Rehmsmeier. The precision-recall plot is more informative than the roc plot when evaluating binary classifiers on imbalanced datasets. *PLOS One*, 10:e0118432, 2015

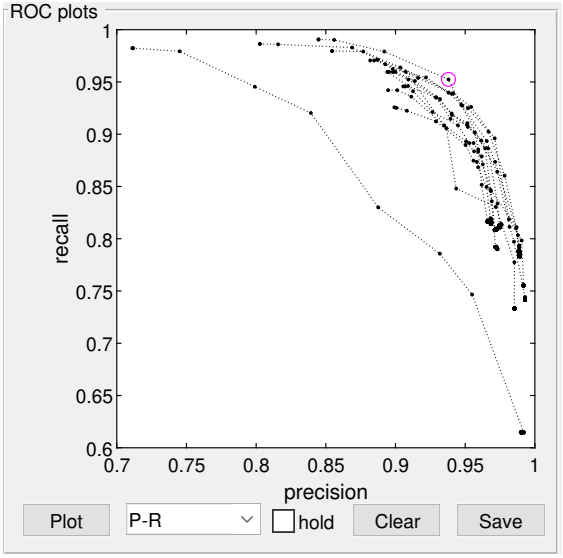


Figure 11.10: graphical output for a PRC curve

threshold merges two adjacent ridges, the resultant skeleton may be incorrectly positioned mid-way between both when the binary image is skeletonised. This give a decrease in TPs and and increase in FNs and FPs, and the overall performance may be worse than more stringent parameter settings

The **ROC clear** button erases the plot, whilst the **hold** check-box allows multiple analyses to be superimposed.

In addition to the PRC or ROC plot, the best performance can be estimated by a summary statistic selected from the *ROC summary* panel (Fig. 11.11).

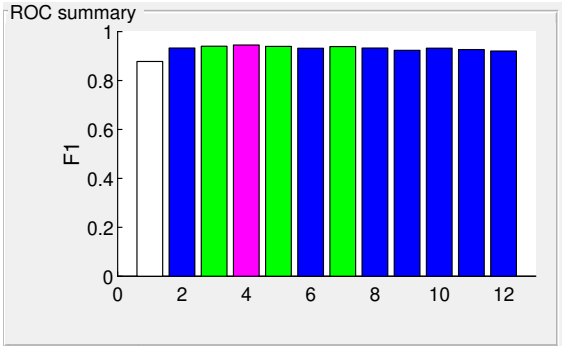


Figure 11.11: ROC summary controls and color-coded plot for the F_1 parameter

The best performance is assessed from the highest **evaluation** score, using a metric selected from the drop-down menu from *dprime*, F_1, *MCC*, F_2 or F_β. The F_β score represents a harmonic mean of recall and precision, where F_β, can be tuned to weight recall ($\beta = 2$) or precision ($\beta = 0.5$) more highly according to equation 11.1:

$$F_\beta = (1+\beta^2)\frac{\text{Precision} \times \text{Recall}}{(\beta^2 \times \text{Precision}) + \text{Recall}} \qquad (11.1)$$

The maximum score is colour-coded magenta on the histogram output, values within 98% coloured green and 95% coloured blue.

If the **mark** check-box is ticked, the same colour coding is applied to the border of the corresponding image in the image grid. In addition, the best score is highlighted on the PRC or ROC plot with a magenta circle. In practice, any of the combinations that score 99% or more would be appropriate, and does allow selection of parameters that may require less computation to achieve a comparable result.

If multiple parameters have been selected for the sensitivity analysis, the image with the best response may not be visible in the current grid. In this case, the image can be revealed by scrolling through the different parameter values using the sliders in the *Display* window. If the user is happy with the 'best' parameter combination, clicking on the preferred image will update the **final** parameter combination in the *Available parameters* table.

The *ROC* option in the *Display* panel, allows visualisation of the output of the ROC analysis in terms of TPs (green), FNs (magenta) and FPs (blue), for each of the parameter combinations visible (Fig. 11.12).

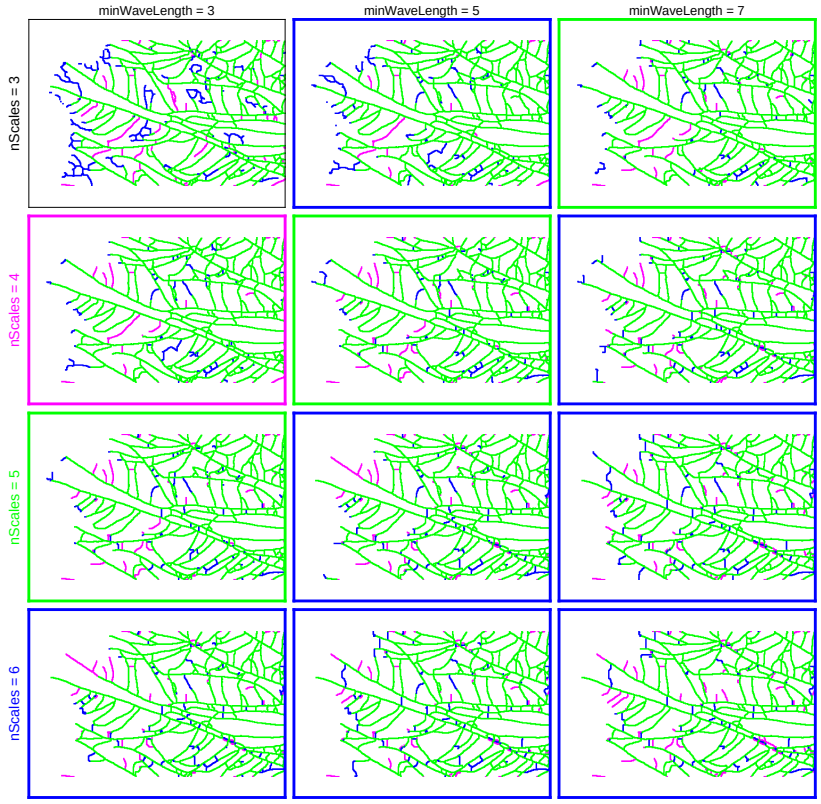


Figure 11.12: Grid showing TP (green), FN (red) and FP (blue) for skeletons derived from the first two parameters in the sensitivity analysis. The boundaries of each image indicate the best performance (magenta), and parameter combinations that give 99% (green) or 95% (blue) of the maximum

.

11.4 Output

The current image grid can be saved using the **Images** button in the *Output* panel, whilst the complete set of summary statistics for each parameter combination can be saved to an Excel spreadsheet

Figure 11.13: Output controls

using the **Data** button. The **Panels** button saves each panel as a *.pdf image for inclusion in the manual.

When the *Parameter selector* GUI is closed the optimised parameters are returned to the main program. At this stage they are not automatically used to update the parameter settings to avoid accidentally corrupting the current parameters, thus if changes have been made, they need to be explicitly changed using the **edit** button in the *Param* panel.

12
Network extraction

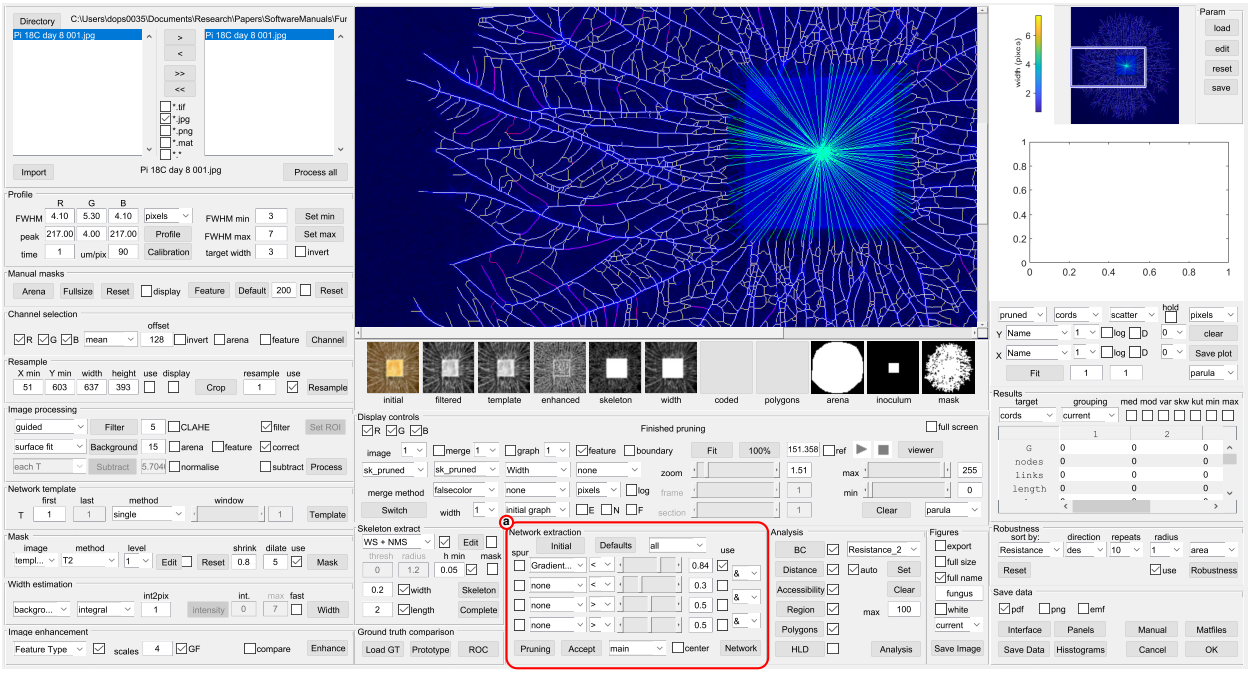


Figure 12.1: Controls to extract the network

12.1 *Overview*

Once the image has been reduced to a satisfactory single-pixel wide skeleton, it can be converted to an initial graph representation with nodes at junctions and hyphal tips, connected by edges that run along the hyphae/cords that are weighted by their length and width. If the graph has been over-segmented to ensure overall connectivity is retained, it may be necessary to prune some edges that are not required.

12.2 *Initial network extraction*

The **Initial** button in the **Network extraction** panel (Fig. 12.2) converts the pixel skeleton into a graph using the *edgelink.m* program written by Peter Kovesi[1].

[1] P. D. Kovesi. MATLAB and Octave functions for computer vision and image processing, 2000. Available from: http://www.peterkovesi.com/matlabfns/

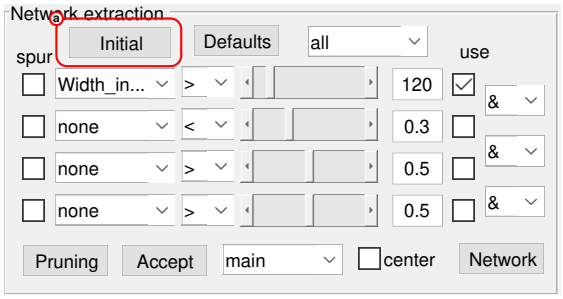


Figure 12.2: Network extraction panel to convert the pixel skeleton to an initial graph representation (a)

Any loops connecting the same two nodes, are split to create an extra node with degree 2 in the longer arm. This ensures that metrics that are critically dependent on parameters such as resistance to flow are correctly assigned to each edge separately, rather than to an average of the two edges. A provisional estimate of the average cord/hyphal width is made from the width image calculated earlier.

The resulting colour-coded network is displayed as an overlay on the template image (Fig. 12.3) for the metric selected from the **graph** drop down menu in the **Display controls** (Fig. 12.4(c)).

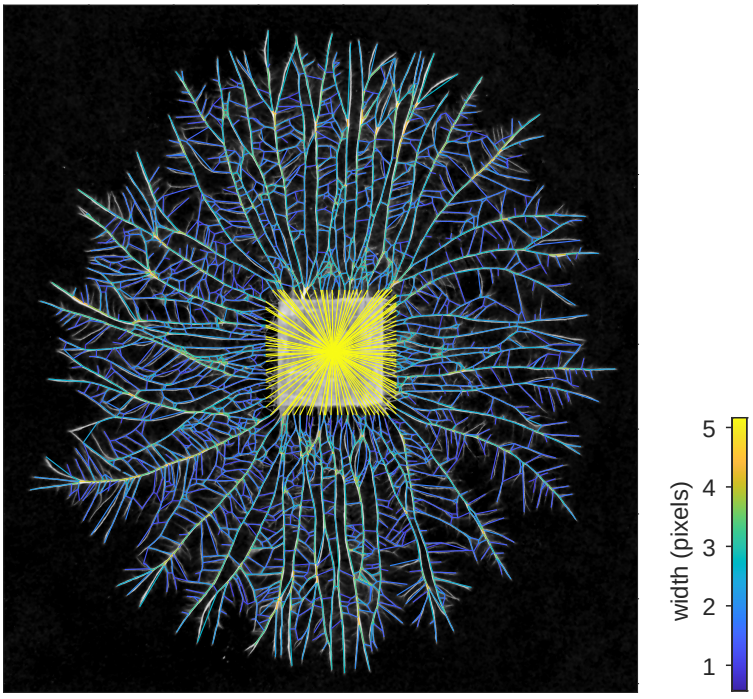


Figure 12.3: Conversion of the weighted pixel skeleton to a weighted graph. Junctions and free ends are represented as nodes linked by edges that have a vector of properties associated with them, such as the width (displayed here). The inoculum is represented as a 'super-node' connected to all the incident cords on the boundary

The colour-code set by the **colormap** dropdown menu (Fig. 12.4(c)) in the **Display controls** panel. The colorbar is updated to display the selected look-up table, and the scale in units selected from *pixels*, *microns*, *mm*, *cm* or *m* (Fig. 12.4(b)).

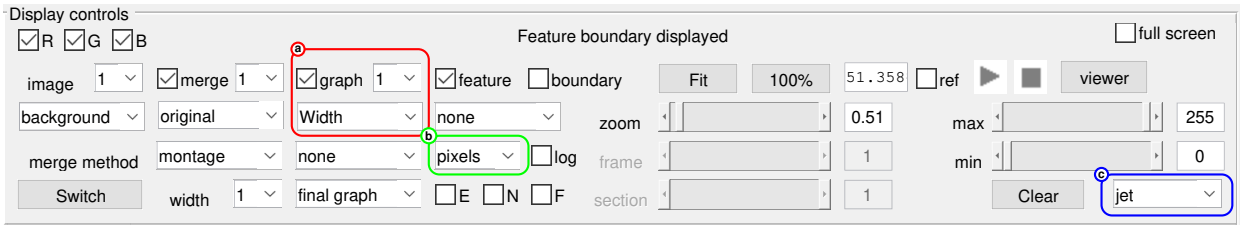


Figure 12.4: Dropdown menu to set the metric (a), units (b) and colour coding (c) for the graph overlay

12.3 *Network pruning*

Depending on the fidelity of the skeletonisation process, it is likely at this stage that the network may be over-connected and require additional pruning. Thus, the **Network extraction** panel has provision to set up to four different criteria that can be applied to restrict the edges that are included in the final graph (Fig. 12.5(a)).

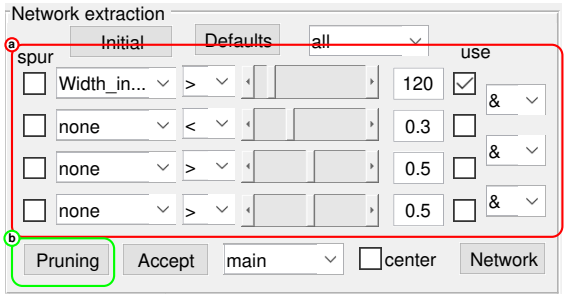


Figure 12.5: Controls to prune the network to include a subset of edges that meet the criteria selected

In each case, the metric is selected from the drop down menu from the list of metrics calculated for the initial graph (essentially based on the intensity, length or width of each edge). The operator can be selected from the adjacent drop down menu (>,<,==,>= or <=), and the value chosen from the slider and/or associated textbox. The min and max values of the slider are automatically calculated to span the range of the metric chosen. The **use** checkbox includes the criteria, whilst the **spur** checkbox restricts the application to hyphae/cords with a free end. The criteria can be combined in different ways, set by the top-most drop down menu: if *all* is selected all the active criteria must be met; *any* will include an edge if one or more criteria are met; *mean* requires a minimum of half the criteria to be true; whilst *combine* applies the boolean criteria set by the final drop down menus in a successive nested hierarchy. Thus for example, the rule might be set by combining ()(criteria 1 *and* criteria 2) *or* criteria 3).

The **Pruning** button (Fig. 12.5(b)) applies the criteria selected to the graph and displays the result as an overlay on the template image (blue), with the retained edges in yellow and the pruned edges in red (Fig. **??**). Edges within the inoculum are coded in green and are exempt from the pruning criteria.

It is useful to toggle on and off individual criteria to see what their effect is in isolation and in combination. It is also worth

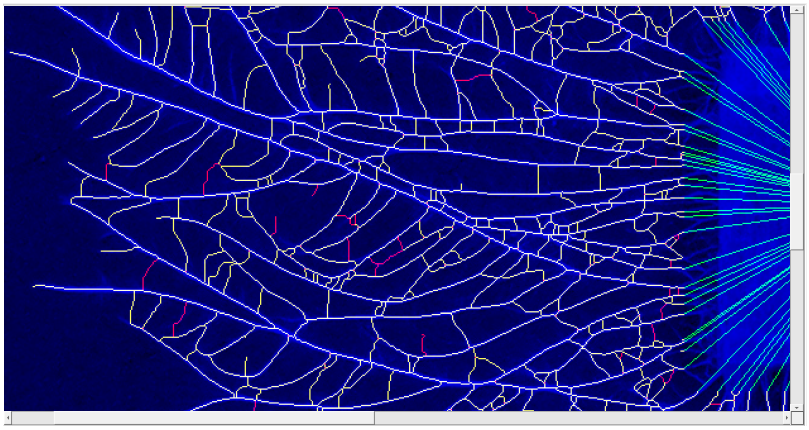


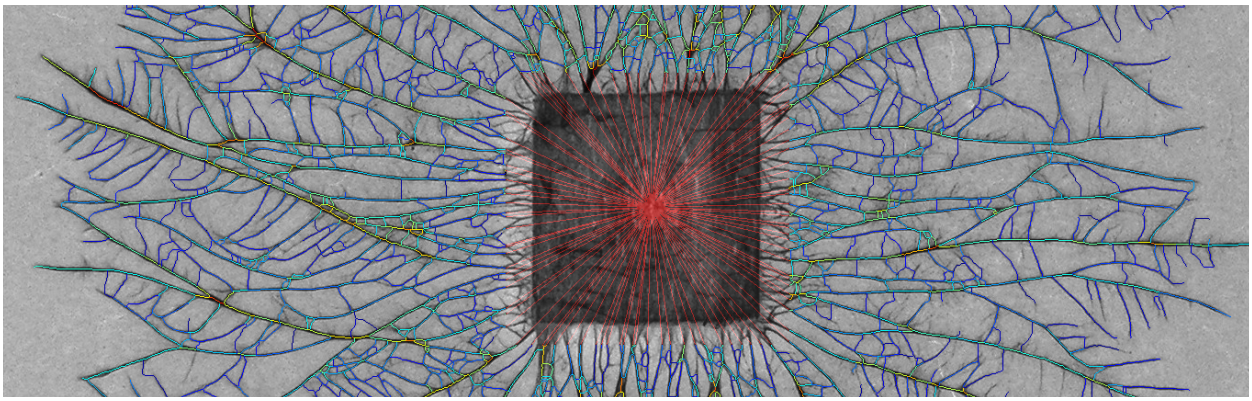
Figure 12.6: Effect of pruning criteria on the pixel skeleton with retained edges in yellow and rejected edges in red.

zooming in and panning around the image to check that other areas are processed correctly.

12.4 *Final graph representation*

If the pruning stages are satisfactory, the conditions are accepted using the **Accept** button. As pruning removes edges from the initial network, the final network is recalculated from the skeleton to ensure that the metrics correctly reflect the remaining edges. The **Network** button extracts the final network and also applies a more sophisticated estimate of the hyphal/cord widths that excludes the overlap region with the parent hypha/cord (termed the *'center width'*), which provides a more accurate measure of the cord diameter itself. The precise number of pixels to ignore in estimation of the width is set by the method drop down menu to be the average width of the *main* hyphae/cord, or the *average* of all edges incident at the node. A coded image of the average center width mapped back onto the pixel skeleton is displayed (and automatically saved for reference) (Fig. 12.7).

Figure 12.7: Coded image of the cord widths mapped back onto the pixel skeleton and superimposed on an inverted image of the background-subtracted colony. Edges in the inoculum are set to the maximum width and connect to a central inoculum node.

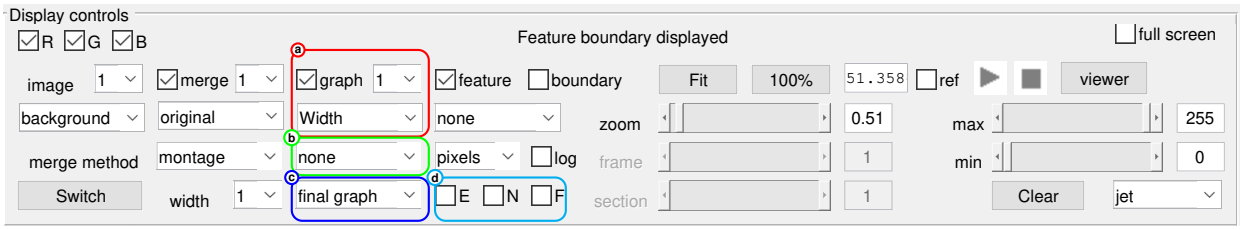


If the **center** checkbox is ticked, the coded image shows pixel in the overlap region in black.

12.5 *Graph display*

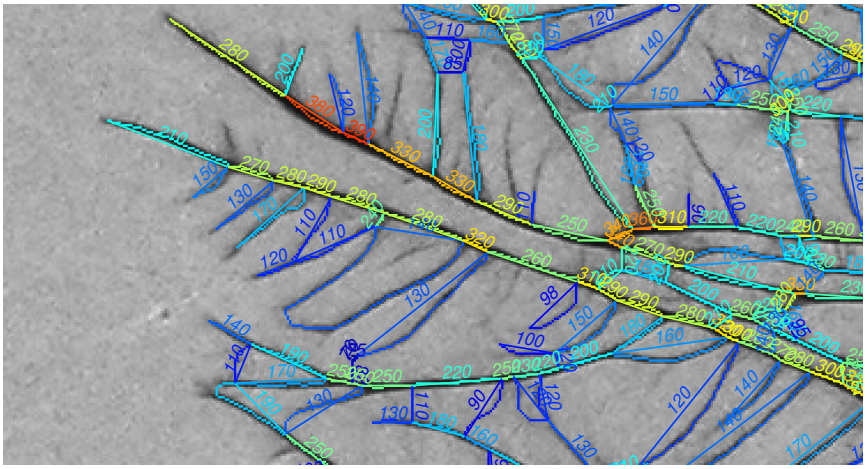
If the **graph** checkbox is ticked in the display control panel (Fig. 12.8(a)), the metric to be displayed is selected from a drop-down menu, and overlaid on the image as a graph, with the nodes at the junctions connected either by straight edges or mapped back onto the pixel skeleton depending on the output drop down menu (Fig. 12.8(c)), *final graph* or *final pixel*, respectively.

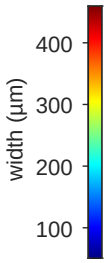
Figure 12.8: Controls to overlay the graph metric (a), node information (b), initial/final skeleton/graph (c), and annotation (d)



Various metrics for the nodes can also be displayed as colour-coded points using the (Fig. 12.8(b)) drop-down menu. The actual values for the edges and nodes can be shown using the **E** and **N** checkboxes, respectively, the scaling can be changed to logarithmic with the **log** checkbox (Fig. 12.8(d)).

Figure 12.9: Edge labels for the hyphal width (in microns).



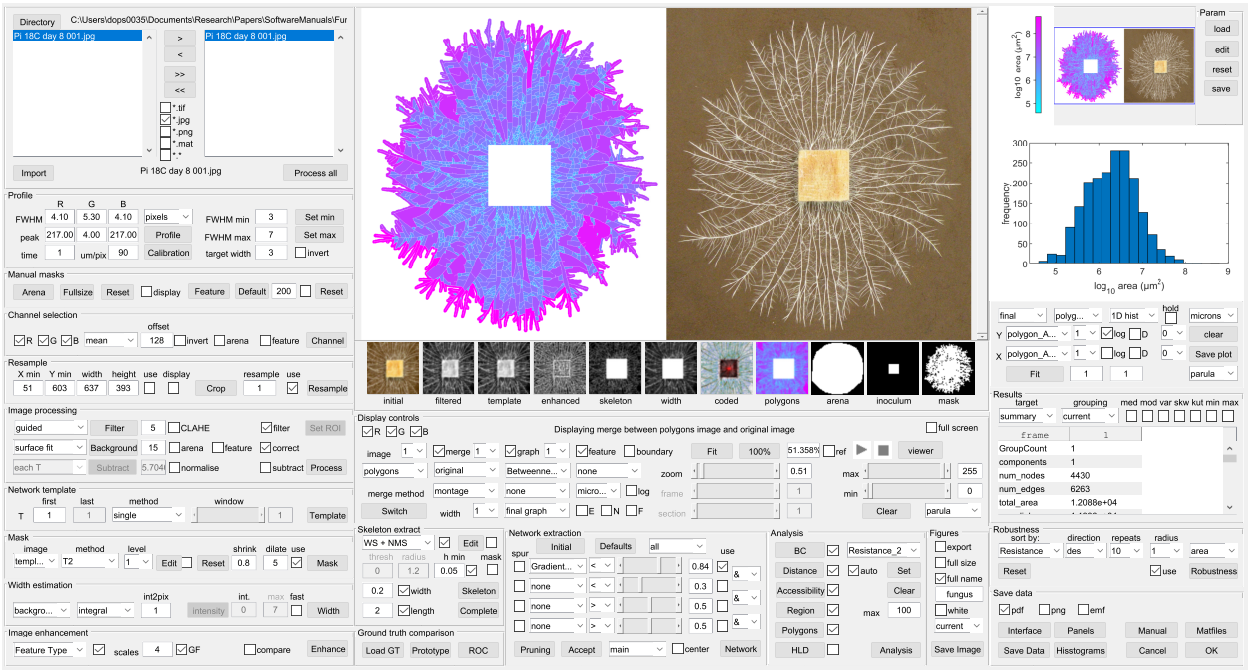


The colorbar is updated to reflect the calibrated metric (Fig. 12.9), and the color-map can be changed using the **colormap** drop-down menu (Fig. 12.8(f)).

The inoculum is represented as a 'super-node' positioned at the intensity-weighted centroid position that is connected to all the cords that are incident on the boundary. Each of these cords is given an arbitrary value equal to the maximum width value. The boundary of the inoculum can be highlighted using the **feature** checkbox (Fig. 12.8(e)).

The graph overlay can be removed by clicking the **Clear** button.

13
Analysis of network structure



13.1 Overview

Once the fully weighted network has been extracted, a number of additional metrics can be calculated using the controls in the **Analysis** panel including:

- Conventional graph theoretic measures, such as the edge and node betweenness centrality (**BC** button) using the metric selected in the adjacent drop down menu to weight the network. For corded networks this should be *Resistance_2* to represent the increase in the number of conduits as the cross-sectional area increases, whilst for mycelia with individual hyphae, it should be *Resistance_4* to represent the increase in predicted flow according to Poiseuille's law.

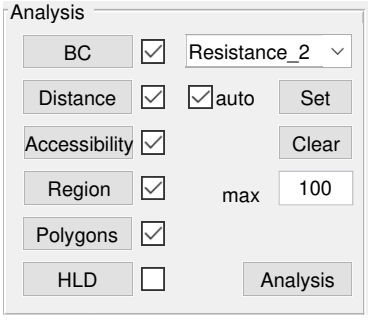


Figure 13.1: Options for analysis of the weighted graph

- The distance from the inoculum (**Distance** button). If the **auto** checkbox is ticked, the reference point is set to the inoculum node. The **Set** button allows the user to manually define a reference point on the image, which is saved with the experiment. The manual reference point can be removed using the **Clear** button. The distance is also used to calculate the *route factor*[1], defined as the average path length to the exit point through the network divided by the Euclidean distance:

$$q = \frac{1}{N-1} \sum_{i=1}^{N-1} \frac{l_{i0}}{d_{i0}} \tag{13.1}$$

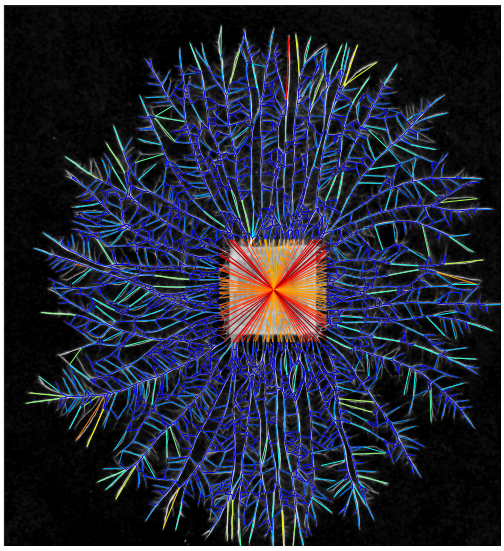
[1] M.T Gastner and MEJ Newman. Shape and efficiency in spatial distribution networks. *J. Stat. Mech.*, 2006: P01015, 2006

- The hydraulic accessibility (**Accessibility** button) serves as a prediction of how easy it is to reach any part of the network and is calculated as the inverse of the shortest path based on the resistance using the weighting metric selected.
- The **Region** button will allocate any signal intensity within a maximum distance in pixels (set by **max**) from the skeleton to the nearest element of the skeleton.
- The **Polygons** button calculates the properties of the regions enclosed by loops in the network, or bounded by the mask at the periphery
- The **HLD** button calculated a hierarchical loop decomposition for the dual-graph of the network.

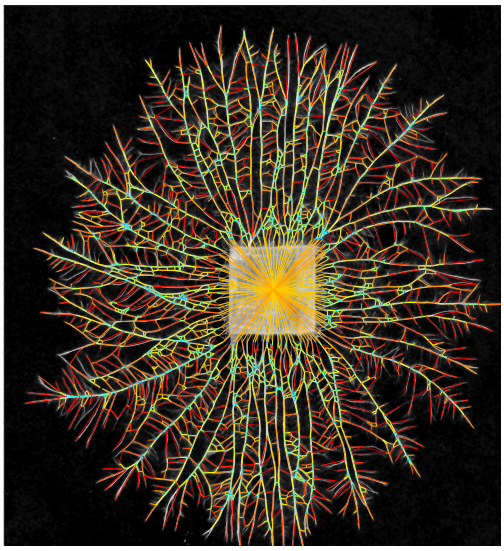
The **Analysis** button (Fig. 13.1(a)) calculates the complete set of metrics for options that are ticked, whilst the left-hand set of buttons calculate the metrics individually and can be toggled on and off with the adjacent checkbox.

The results for any of the metrics calculated can be overlaid on the image as before. Examples metric graphs are shown in Fig. 13.2.

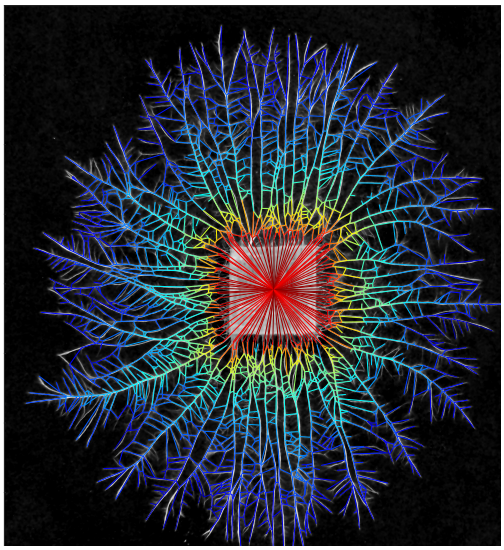
Figure 13.2: Examples of the different edge metrics



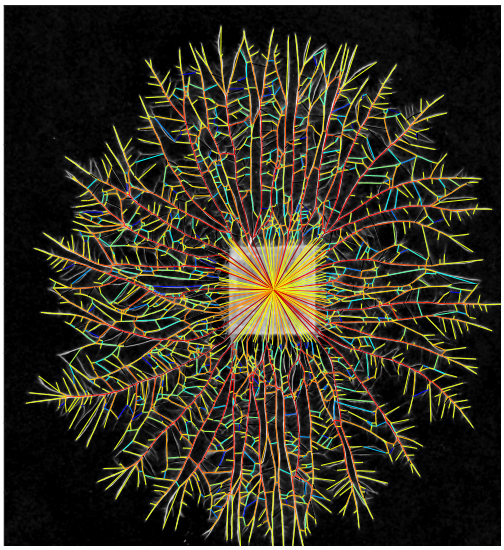
(a) Length



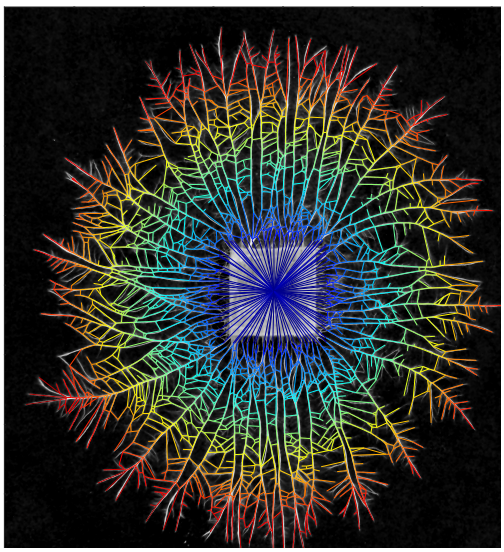
(b) Resistance



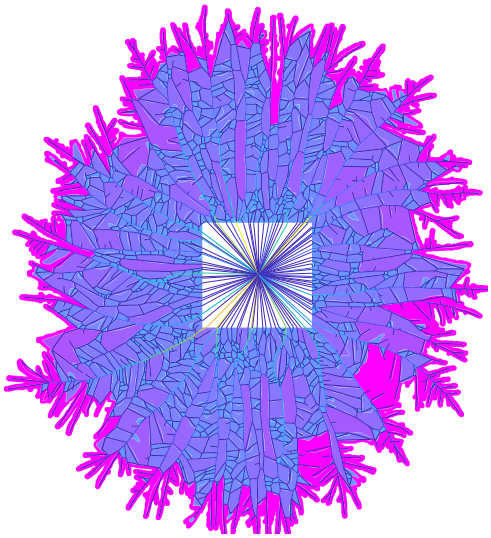
(c) Accessibility (log)



(d) Betweenness centrality (log)



(e) Distance



(f) polygon area (log)

13.2 *Plotting metrics*

Metrics for the edges and nodes can be plotted in pairwise combinations, individually as a histogram, or as an average value over time, depending on the settings in the plot controls panel (Fig. 13.3).

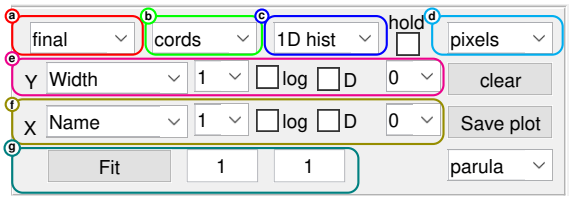


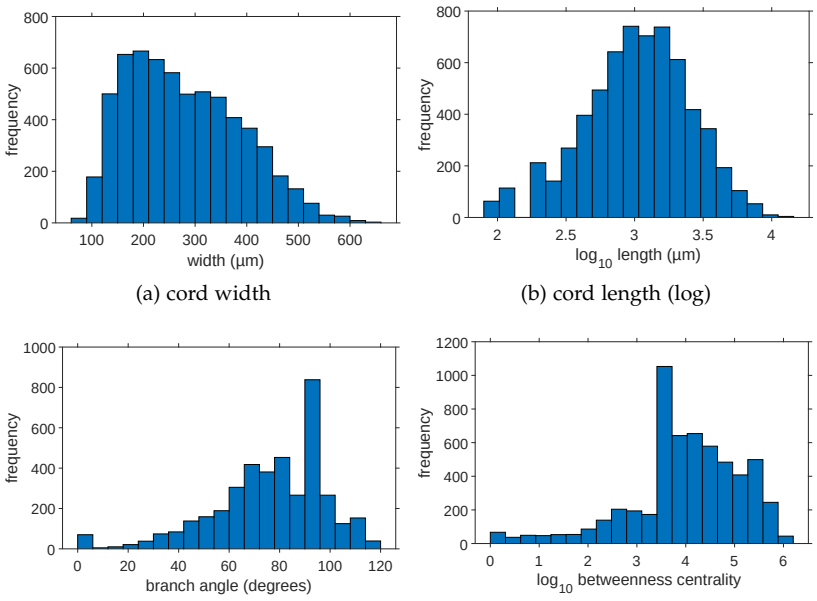
Figure 13.3: Plot controls for graphical output for the network metrics

The first drop down menu (Fig. 13.3(a)) selects whether to plot results from the *initial* or *final* graph; the second drop down menu (Fig. 13.3(b)) selects the object to plot (*cords, nodes, polygons etc.*); the third drop down menu (Fig. 13.3(c)) selects between a *scatter, 1D histogram, 2D histogram* or *time* plot, with the units set by the final drop down, typically in *pixels* or *microns* selected by Fig. 13.3(d).

The metric to be plotted on each axis is selected from the Y and X drop down menus (Fig. 13.3(e,f)), for a specific channel (default as channel 1). The data can be transformed using the adjacent **log** checkbox. For time-series, the **D** checkbox plots the difference between each time point. The adjacent drop-down menu allows an offset by an integer time amount forwards or backwards.

Scatter plot data can by fit with a linear function using the **Fit** button (Fig. 13.3(g)), with the coefficients returned in the adjacent text boxes. In the case of histograms for time-series, the colour is controlled by the look-up table chosen. Plots can be saved as images using the **Save plot** button, or cleared using the **Clear** button.

Typical results are shown in Fig. **??** for some edge and node metrics.



(a) cord width

(b) cord length (log)

(c) branch angles

(d) betweenness centrality (log)

13.3 *Summary of metrics calculated*

The results for all the metrics can also be displayed in the **Results** panel (Fig. 13.4).

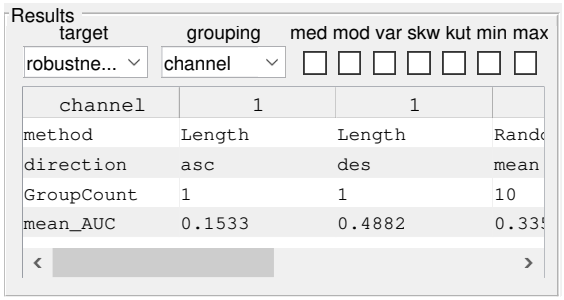


Figure 13.4: Summary results table

The **target** dropdown menu is used to select which results are shown from *cords*, *nodes*, *polygons*, *feature*, *robustness* or textitsummary, whilst the **grouping** drop down menu can be used to aggregate the data by just the *current* image, or by *channel*, *section*, *frame* or *all*. Normally results are summarised as the mean for the level of grouping chosen. However, additional metrics can be selected using the adjacent checkboxes, to include higher moments of the distribution including *median*, *variance*, *skewness*, *kurtosis*, *min* and *max*. These are added to the tabulated results and also included when the data is saved.

13.3.1 *Edge metrics*

Table 13.1: Metrics calculated for each hypha/cord

Metric	Explanatory notes
Original, cv	The mean intensity of pixels in the edge from the original image, along with the coefficient of variation (cv)
Intensity, cv	The mean intensity and cv after background subtraction
Width, Width cv	The estimated width ($2r$) and cv (μm)
Width center	The estimated width excluding overlap regions at the node
Length, length center	The total length and length excluding the overlap at nodes (l, μm)
Area, Volume	The cross-sectional area ($a = \pi r^2$, μm^2) and volume ($v = a * l$, μm^3)
Resistance	The predicted resistance to flow assuming Poiseuille flow (l/r^4, μm^{-3})
Number	The number of pixels in the edge
Tortuosity	The Euclidean distance between the nodes divided by the total length of the edge
Distance	The geodesic distance of each pixel from a manually defined reference point (μm)
Region	The integrated sum of intensities in the region closest to each edge
BC	The edge betweenness centrality
Route Factor	the average path length to the exit point through the network divided by the Euclidean distance

13.3.2 *Summary metrics*

A set of graph theoretic metrics and summary metrics are also provided, including the mycelial area, number of nodes and edges, and their length and width. The Global and Root efficiency is calculated[2], as the sum of the inverse of shortest paths between all nodes, or the inoculum node, respectively.

The global efficiency[3] is defined as the mean reciprocal of the shortest paths, weighted by resistance, with the reciprocal for disconnected nodes defined as zero (Equation 13.2). The Root efficiency (E_{root}) is calculated in a similar manner but just from the exit point to all other nodes, whilst the Root-Tip efficiency just considers the efficiency of transport from the inoculum to the tips.

$$E_{global} = \frac{1}{N(N-1)} \sum_{i \neq j \in G} \frac{1}{d_{ij}} \tag{13.2}$$

Each of these measures can be normalised by the same metric calculated for the edges present in the minimum spanning tree (MST).

The cyclomatic number, alpha, beta and gamma metrics derive from transportation geography for planar networks. The cyclomatic number gives a measure of the complexity of the network and is calculated as:

$$\text{cyclomatic number} = u = e - v + p \tag{13.3}$$

Where e is the number of edges, v is the number of vertices or nodes, and p is the number of components (typically set to 1 here as only the largest connected component is considered). The alpha

[2] V. Latora and M. Marchiori. Efficient behavior of small-world networks. *Phys. Rev. Lett.*, 87:198701, 2001

[3] V. Latora and M. Marchiori. Efficient behavior of small-world networks. *Phys. Rev. Lett.*, 87:198701, 2001

Metric	Explanatory notes
components	The number of sub-graphs in the network. This is typically one as only the largest connected component is selected
num_nodes	The number of nodes (excluding inoculum nodes)
num_edges	The number of edges (excluding inoculum edges)
node_degree	The mean node degree, excluding the inoculum
total_area	The area in mm^2, including the inoculum
mycelial_area	The area in mm^2, excluding the inoculum
node_density	The number of nodes divided by the mycelial area
edge_density	The number of edges divided by the mycelial area
total length	The total length of the network (mm)
length_mean, length_median, and length_sd	The average, median and standard deviation of lengths for the hyphae/cords, excluding the inoculum (μm)
length_density	The total length divided by the mycelial area (mm^{-1})
MST_length_ratio	The total length divided by the length of the minimum spanning tree (MST), calculated using shortest paths weighted by the edge resistance
MST_volume_ratio	The total volume divided by the volume of the minimum spanning tree (MST), calculated using shortest paths weighted by the edge resistance
width_mean, width_median, and width_sd	The average, median and standard deviation of widths for the hyphae/cords, excluding the inoculum (μm)
volume	The total volume of the network, excluding the inoculum (mm^3)
volume_density	The total volume of the network, excluding the inoculum, divided by the mycelial area (mm)
ntips	The number of tips
tip_ratio	The number of tips divided by the total number of nodes
tip_density	The number of tips divided by the mycelial area (mm^{-2})
k	The average node degree
G efficiency	The global efficiency of the network
R efficiency	The Root efficiency of the network calculated from the inoculum to all nodes
RT efficiency	The Root efficiency of the network calculated from the inoculum to just the tips
Gnorm	The global efficiency normalised to the global efficiency of the MST
Rnorm	The Root efficiency normalised to the global efficiency of the MST
RTnorm	The Root-Tip efficiency normalised to the global efficiency of the MST
alpha	The alpha coefficient or meshedness
beta	The beta coefficient
gamma	The gamma coefficient

Table 13.2: Graph-theoretic metrics

coefficient[4] or meshedness[5] estimates the fraction of cycles present compared to a fully connected planar network, and is calculated as:

$$\text{alpha coefficient} = \alpha = \frac{e - v + p}{2v - 5} \tag{13.4}$$

The beta coefficient measures the level of connectivity in a graph based solely on the number of edges and vertices. Simple branching networks with no loops have β value of less than one, for a single loop $\beta = 1$, whilst $\beta > 1$ for more complex networks with multiple loops.

$$\text{beta coefficient} = \beta = \frac{e}{v} \tag{13.5}$$

The gamma coefficient represents the number of links present compared to the maximum number expected for a planar graph with the same number of vertices. Gamma ranges from 0 to 1 for a fully connected network.

$$\text{gamma coefficient} = \gamma = \frac{e}{3(v - 2)} \tag{13.6}$$

13.3.3 *Node metrics*

A variety of metrics are also calculated for each node (Table 13.3)

Table 13.3: Metrics calculated for each node

Metric	**Explanatory notes**
node degree	The number of hyphae/cords connected to each node, excluding the inoculum node
total width	The sum of the hyphae/cord widths incident at each node (μm)
mean width	The average of the hyphae/cord widths incident at each node (μm)
intensity	The original intensity at the node
distance	The distance from the reference point (mm)
Maj, Mid and *Min*	The widths of the three main hyphae/cords incident at the node (μm). Only one value is given for a terminal node, whilst few node have a degree greater than three
Min_Maj, Mid_Maj and *Min_Mid*	The ratio of the hyphae/cord widths for the three main incident hyphae/-cords
Omaj, Omid and *Omin*	The orientation of the three main incident hyphae/cords, determined as a linear segment between the node and the midpoint of the hyphae/cords (degrees)
Omin_Omaj, Omid_Omaj and *Omin_Omid*	Branch angles between the three main incident hyphae/cords, determined from a linear segment from the node to the midpoint of each hyphae/cord (degrees)
branch_angle	The branch angle is calculated as the minimum of the angles around each node (degrees).
growth_direction	The growth direction is calculated as the maximum of the angles around each node (degrees).

The node degree is one for hyphae/cords with a free end, but typically 3 for all other nodes, excluding the inoculum node. Clas-

[4] P Haggett and RJ Chorley. *Network Analysis in Geography (pp 74-76) Edward Arnold Publishers Ltd.* London, 1969

[5] J. Buhl, J. Gautrais, R.V Solé, P. Kuntz, J.-L. Valverde, S.and Deneubourg, and G. Theraulaz. Efficiency and robustness in ant networks of galleries. *Eu. Phys. J. B*, 42:123–129, 2004

sifying the three main hypha/cords as major (*Maj*), middle (*Mid*) and minor (*Min*) allows calculation of additional metrics such as the ratio of incident hyphae/cord widths, the mean orientation, and particularly the branch angles. For the latter two measures, the hypha/cord is represented as a linear vector from the node to the mid-point of the hypha/cord. This ensures that a reasonable length is considered when estimating the direction, but will incur errors for long hyphae/cords with high tortuosity.

13.3.4 Polygon metrics

The polygon metrics are only well defined for dense networks with a high probability of fusion to created bounded areas. For sparse networks, the polygons are bounded by the mask and, in the limit, may be a single contiguous area running around the entire mycelium for a branching tree body plan. Nevertheless, with this caveat, a number of metrics can be calculated for the areas (Table 13.4)

Table 13.4: Metrics calculated for each polygon

Metric	**Explanatory notes**
polygon_area	The area of the polygons (which may be less than the mycelial area as, the outermost polygons are constrained by the mask rather than the shrunken convex hull)
MajorAxisLength	The length of the major axis of the ellipse that has the same normalized second central moments as the area (μm)
MinorAxisLength	The length of the minor axis of the ellipse that has the same normalized second central moments as the area (μm)
Orientation	The angle between the x-axis and the major axis of the ellipse that has the same second-moments as the area. The value ranges from -90 to 90 degrees
Solidity	The proportion of the pixels in the convex hull that are also in the areole. Computed as Area/ConvexArea
Perimeter	The distance around the boundary of the areole between each adjoining pair of pixels around the border (μm)
Circularity	The ratio of the perimeter of a circle with the same area to the actual perimeter
Elongation	The ratio of the major and minor axis lengths
Roughness	The ratio of the perimeter squared to the area
MaxDistance	The maximum distance from any pixel within the area to the bounding hyphae/cords (μm)
MeanDistance	The average distance from all pixels within the area to the bounding hyphae/-cords (μm)

13.3.5 Robustness metrics

Separate Excel sheets are saved for *Ordered Robustness*, *Random Robustness* and *Spatial robustness*. In each case, the data are grouped by the method, then the number of replicate runs (only one for ordered robustness). The absolute value of the volume removed

and the volume connected are given, followed by the percentage removed and percentage connected. The robustness column calculates the fraction connected, taking into account the volume that has already been removed. The AUC column gives the area under the curve (AUC) for each profile. The summary values for the AUC are given in a separate sheet (*Robustness AUC*) for each type of attack.

13.4 Saving images

Figures from the analysis can be saved using the controls in the **Figures** panel (Fig. 13.5). The **Save image** button prompts the user for a filename and saves a version of the image and colorbar currently displayed with all the annotations in '*.png', '*.pdf' or '.emf' format, depending on the checkboxes in the **Save data** panel (see below). If the **full size** checkbox is activated, the images will be saved in their native resolution from a separate window. The *current* dropdown menu controls the background for some graph plots, and can be set to white or black to aid visibility. Likewise, the *white* checkbox forces the background to white for some plots.

If the **full name** checkbox is ticked, the filename entered will be affixed with the original image filename. Alternatively, if left unchecked, the test in the **prefix** box will be appended instead.

The **export** checkbox will cause images to be saved at each step in the entire processing sequence. This is useful to generate a record of the analysis, but is slow as many large images will be written to disk.

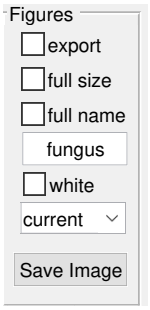


Figure 13.5: Output controls used to save images

13.5 Saving data and interface components

Data from the analysis can be saved from the **Save data** panel (Fig. 13.6).

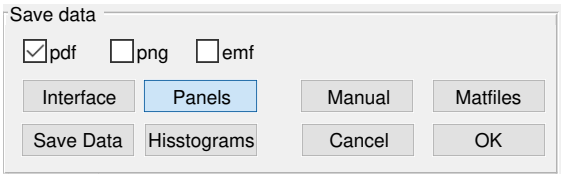


Figure 13.6: Output controls used to save data and interface components

The **Save data** button writes all the data for the edges, nodes, polygons, and robustness metrics, to separate sheets in an Excel file. Each sheet includes columns specifying the *filename*, *channel*, *section* and *frame*. Note: if any metric includes a NaN or Inf value, it appears to be replaced in Excel with a value of 65535.

The **Histogram** button generates a set of histograms for the edges and nodes and saves these in .pdf format.

The **Interface** button saves a copy of the the interface in '.png', '.pdf' or '.emf' format, depending on the checkboxes ticked, and can be used to tailor the illustrations in this manual to any specific application. Likewise, the **Panels** button saves a copy of the all

the panels in the interface. The **Manual** button runs a series of processing steps for robustness that automatically sets the controls in the interface for different robustness scenarios, and then outputs the graph overlays and plots accordingly. The **Matfiles** button saves a subset of images and graph objects in Matlab format, for later use in the Advection Diffusion Delivery (ADD) model interface.

14
Robustness measurements

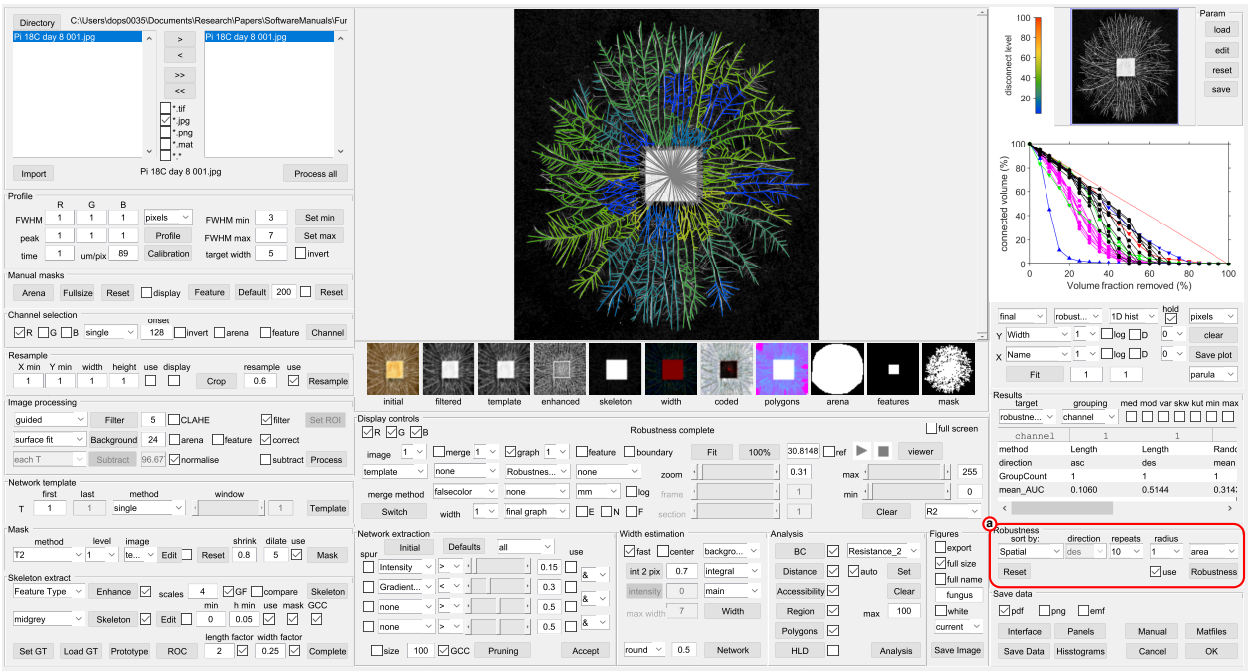


Figure 14.1: The main interface showing a spatial robustness measurement

14.1 Introduction

To simulate the affect of damage or fungivores on the mycelium, the digitised network can be subjected to a series of different types of attack to map out its instantaneous 'robustness'. Currently, there is no standardized measure of the robustness of the network to potential damage (i.e. removal of edges). In our case, we measured robustness as the percentage of the mycelium volume that was still connected to the root (inoculum) after fixed volume of edges were removed according to different morphological criteria, following spatial constraints, or at random[1]. The removal of edges was done in sequential steps of cumulative sums of 5% of the total volume of the original colony before damage, running from 5% volume removed to 95% volume removed. Using this *in silico* approach, the network is tested to destruction to generate a profile of the

[1] D.P. Bebber, J. Hynes, P.R. Darrah, L. Boddy, and M.D. Fricker. Biological solutions to transport network design. *Proc. Roy. Soc. B*, 274:2307–2315, 2007

connected volume versus the volume removed. To standardise between different experiments and species, it is convenient to normalise the axes to percent. In a simple attack profile, the y-axis value cannot exceed a 45° slope representing the amount of material removed. Thus a second normalisation to give the volume remaining as a fraction of the maximum that could be present is used to calculate the 'robustness'. In many cases, the profile can be summarised by a single metric, such as the volume removed at which 50% remains connected or the area under the curve (AUC).

In a real-life scenario, the network has the capacity to respond to grazing or damage by re-allocating resources away from the growing margin to strengthen existing links, or to form additional cross-links[2,3]. We term this dynamic response 'resilience'. At present, we can only document resilience experimentally by quantifying the change in network architecture following grazing or damage. If images are available for before and after, they can be aligned and used to extract a single network that contains all the links present at any period. This ensures that each edge has a unique ID that is retained throughout the time-series. The trajectory of each edge can therefore be evaluated following grazing.

[2] L. Boddy, J. Wood, E. Redman, J. Hynes, and M. D. Fricker. Fungal network responses to grazing. *Fungal Genetics and Biology*, 47:522–30, 2010

[3] T. D. Rotheray, T. H. Jones, M. D. Fricker, and L. Boddy. Grazing alters network architecture during interspecific mycelial interactions. *Fungal Ecology*, 1:124–132, 2008

14.2 *Ordered robustness*

During an ordered attack, the edges are sorted according to the metric chosen in the **sort by:** drop-down menu in the **Robustness** panel (Fig. 14.2). These include *Length, Width, Volume,* and *Resistance*. The sort direction is then chosen from the **direction** drop-down menu. In the case of ordered attacks, only one iteration is required as there is a unique ranking of edges according the criterion selected.

Figure 14.2: The Robustness control panel

Not all combinations will make biological sense, but the following may have some justification:

Length descending: Long hyphae or cords are removed first which simply reflects probability of encounter (i.e. longer hyphae are more likely to get attacked than shorter hyphae);

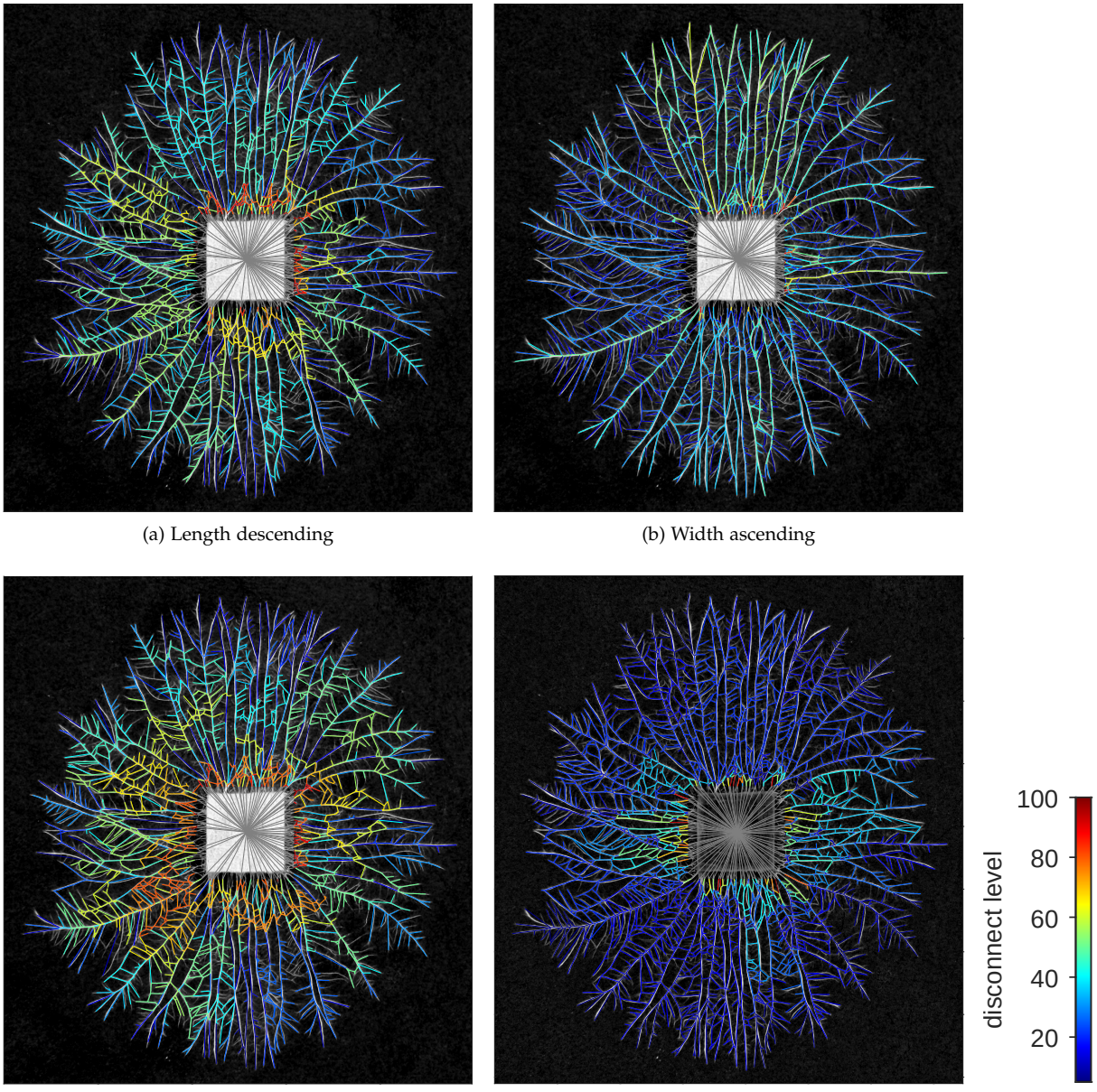
Resistance descending: Long thin hyphae are removed first, combining preferential attack by fungivores with small mouthparts and the probability of encounter;

Width ascending: Thin hyphae are removed first to mimic fungivores that preferentially attack smaller hyphae/cords first. These might include fungivorous bacteria that can only access digest the walls of thin and young hyphae or fungivorous animals with small mouthparts for cord-forming fungi;

Width descending: Hyphae with a large volume are removed first, representing selective feeding where each bite maximises the amount of material consumed per bite. This might include larger protists or nematodes for microscopic fungi; or fungivorous animals with larger mouth parts for cord-forming fungi;

Clicking the **Robustness** button will run the analysis and automatically overlay a colour-coded graph representation on the image (Fig. 14.3).

Figure 14.3: Colour-coded representation of the order that edges are removed in the different ordered attack scenarios. The disconnect level represents the volume removed before the edge is disconnected



(a) Length descending

(b) Width ascending

(c) Volume descending

(d) Resistance descending

The connected volume (as a percentage) is automatically plotted against the volume removed (as a percentage) (Fig. **??**).

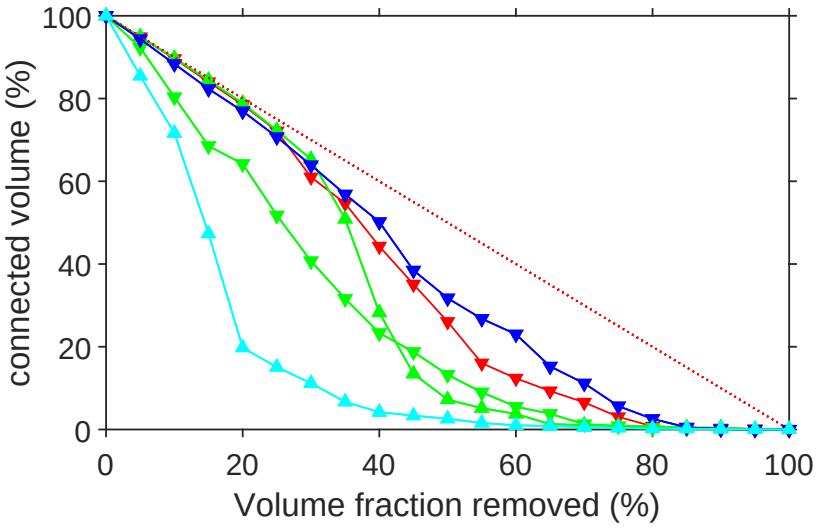
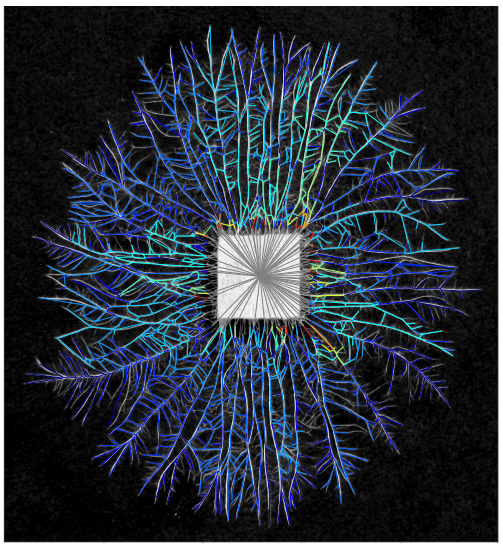


Figure 14.4: Profiles for different types of attack based on *Length* (red), *Width* (green), *Volume* (blue), or *Resistance* (cyan). Upward pointing triangles are ascending order, downward pointing triangles are descending order

14.3 *Random robustness*

We used random attacks as a standard graph-theoretic approach to test robustness. As the order is no longer unique, multiple runs are required to provide a representative view of random robustness, which can be set using the **repeats** drop-down menu (Fig. 14.5).

Figure 14.5: Random robustness with variable number of repeats



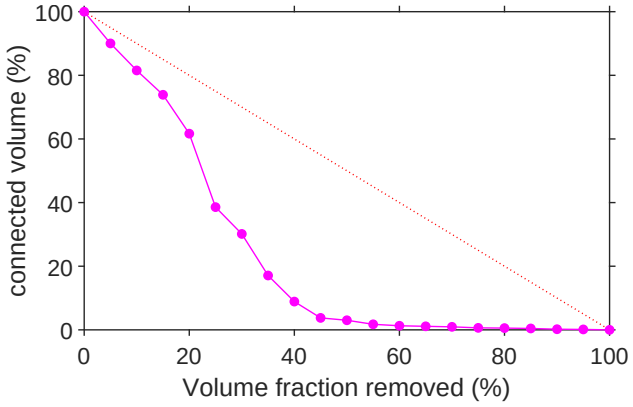
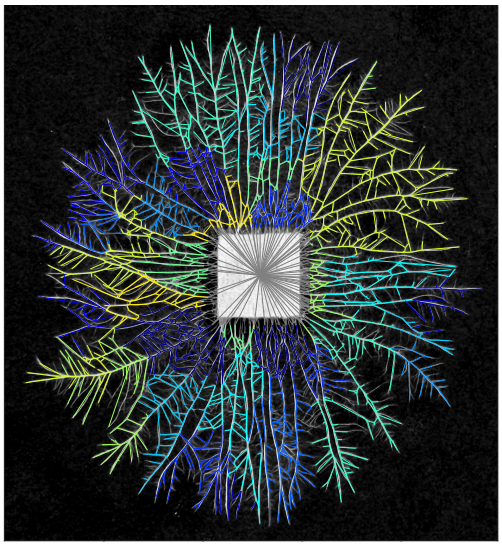


Figure 14.6: Left panel - Colour-coded representation of the order that edges are removed in a single random attack. Right panel - profiles from 10 random attacks

14.4 *Spatial robustness*

We also used spatial robustness when hyphae were removed in clusters or "chunks" of closely located hyphae, to mimic accidental damage or attacks by larger fungivores, such as woodlice. In this scenario, the radius of each attack is specified by the **radius** drop-down menu, eith are a percentage of the mycelial *area* or in *pixels* according to the adjacent drop-down menu.

Figure 14.7: Spatial robustness with variable number of repeats, and attacks of different radii based on the percentage of the mycelial area



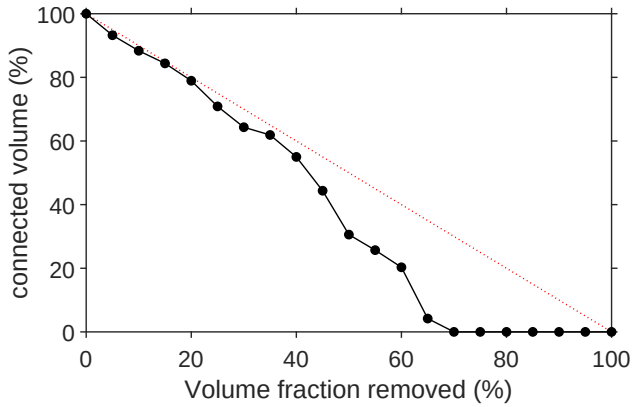


Figure 14.8: Left panel - Colour-coded representation of the order that edges are removed in a single spatial attack. Right panel - profiles from 10 spatial attacks

15
Importing Images

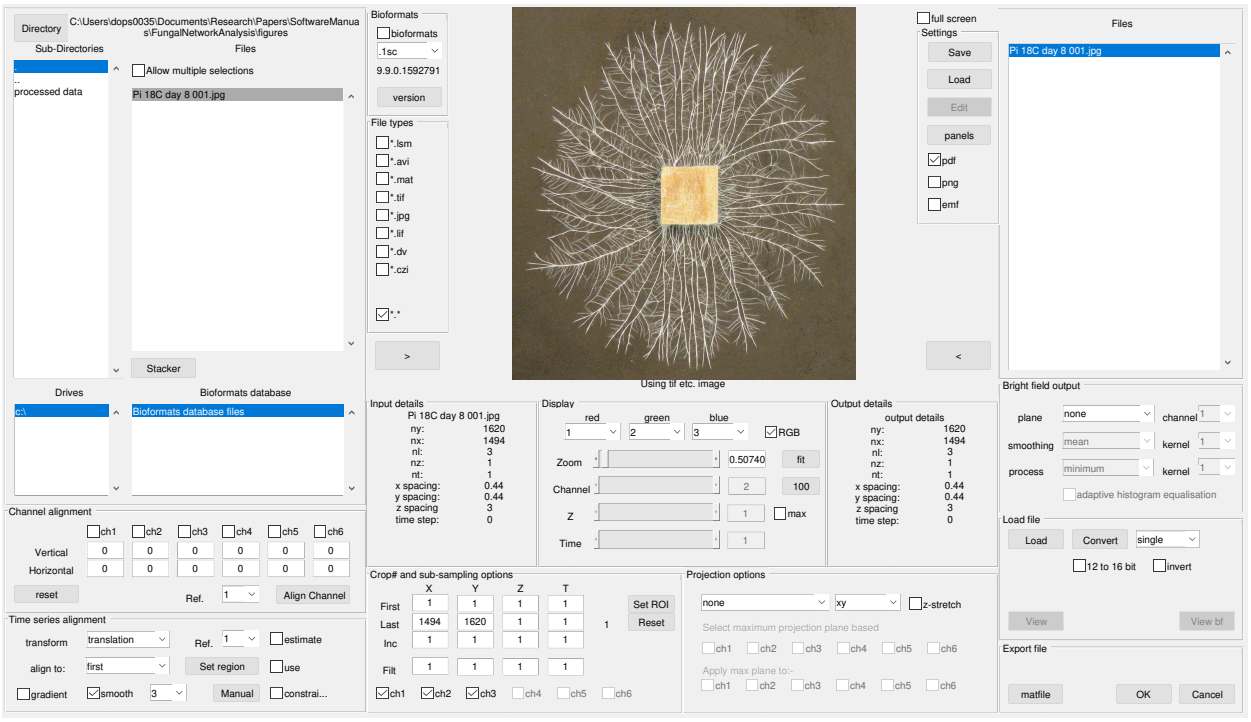


Figure 15.1: The interface to import images from a wide variety of formats, with additional controls to align, crop and project data

15.1 *Introduction*

This package was originally designed to import confocal image stacks into Matlab, particularly for ratio imaging applications. The import options allow some channel registration, time-series alignment, sub-sampling, smoothing and z-projection. Most file formats can now be handled using the Bio-Formats program (Linkert et al. 2010[1]) called by the import functions:

`http://www.openmicroscopy.org/site/support/bio-formats4/`
`http://loci.wisc.edu/software/bio-formats`

The latest versions of Java needs to be installed, and is available from:

`http://www.java.com/en/`

[1] M. Linkert, C. T. Rueden, C. Allan, J. M. Burel, W. Moore, A. Patterson, B. Loranger, J. Moore, C. Neves, D. Macdonald, A. Tarkowska, C. Sticco, E. Hill, M. Rossner, K. W. Eliceiri, and J. R. Swedlow. Metadata matters: access to image data in the real world. *J Cell Biol*, 189:777–82, 2010

15.2 File Selection

The file selection panel shows:

- a **Directory** button that opens a standard dialog box to select a different directory
- sub-directories of the current folder
- individual files in the current directory
- the currently selected file
- the available drives
- image files within a Leica "lif" database (if appropriate)

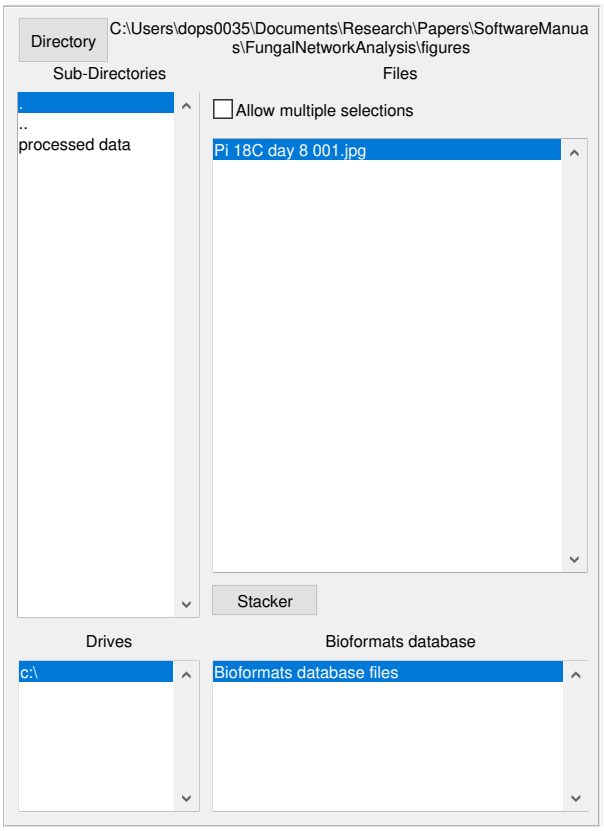


Figure 15.2: File selection panel

If the **Allow multiple selections** checkbox is ticked, a range of files can be loaded in one operation. This is useful to join a set of consecutive sequences from the same time-series experiment. The images must have the same dimensions in x and y, and have the same number of channels. Images are sorted in alphabetical order in the listbox and will be imported in this order. If a different order is required, each file has to be added in sequence manually.

A set of checkboxes (Fig. 15.3) are available to display a restricted set of file types or all files in the directory (*.*)

Figure 15.3: Setting the file extensions

Once the required file(s) have been selected, the information on each file can be loaded by double clicking or using the **right-arrow**.

If the system can read the file format information, The file name(s) will then appear in the **output list** box, and the image

details shown in both the **Input details** and **Output details** panels (see Fig. 15.4). If multiple files have been selected, only the details of the last file will be shown. Input and output details will be the same at this stage as no additional processing steps have happened.

If the system can not pick the pixel size information or time interval, a dialog box will appear with prompts for the user to enter the required information (see Fig. 15.5).

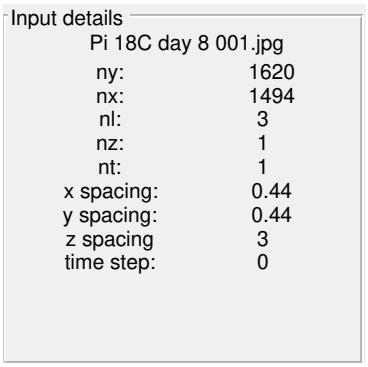


Figure 15.4: The input details panel: displays of the size and calibration values for the current image

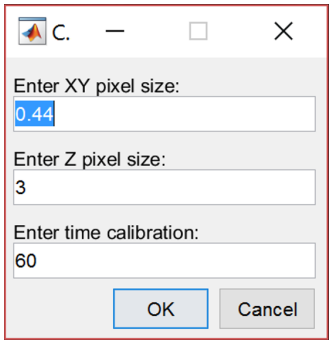


Figure 15.5: Dialog box prompting for the *x,y,z* spacing and the time-interval

15.3 *Combining separate channels*

The **Stacker** button is used to combine individual *.tif images from multiple channels and time-points into a single matlab file if they are numbered in sequence. The user is prompted for:

- the number of channels to combine and the pixel resolution (Fig. 15.6).
- an Excel file that contains the time stamp for each image as a single column. If this is not present, then the time interval defaults to 1;
- the first image for channel 1 (to get the core filename)
- the first image for channel 2 (to get the second core filename).

The images are then automatically loaded and concatenated into a single matlab file which is automatically saved. The matlab file can be loaded as normal.

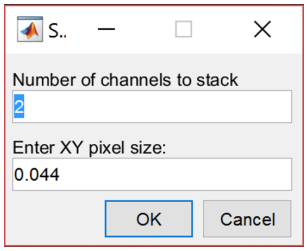


Figure 15.6: Setting the number of channels and pixel spacing when combining separate tif images

15.4 *Importing images from different formats (including Leica databases)*

The package can open most file formats using the Bio-Formats package. Leica images are stored in a single *.lif database and cannot be accessed directly. If a "lif" database is selected, or a format that cannot be read in directly, the Bio-Formats program reads in the list of image stacks or time-series and displays these in a separate pop-up window (see Fig. 15.7).

The file is selected by double clicking in the list box and the name appears in the box underneath. Clicking the **Load file** button

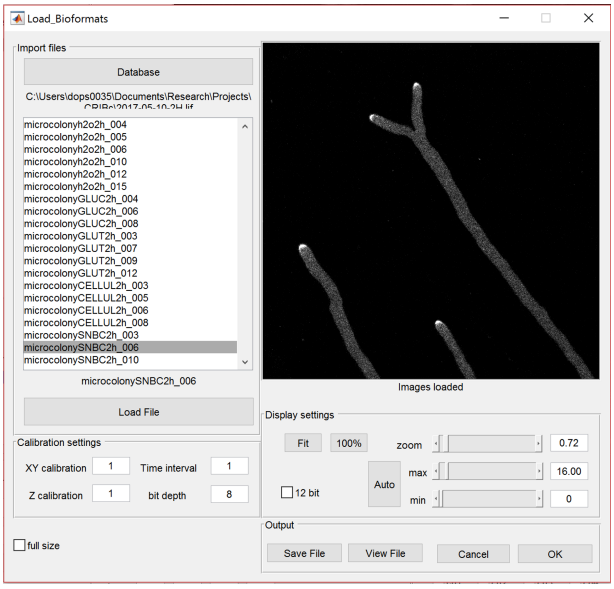


Figure 15.7: The Bio-Formats import window: Once a Leica database has been loaded, one of the image files can be selected and imported into the MatLab environment. The calibration parameters have to be set manually

will import the image and display the first three channels of the first time/z-plane in the window. The **x**, **y** and **z** pixel sizes have to be entered manually into the appropriate boxes, along with the **Time** interval and **bit depth**.

Clicking **OK**, will import the selected file into the main interface. It is also possible to view the full files at this stage using the **View file** button, or save the file as a matlab array using the **Save file** button.

15.5 Image display

Once the images have been loaded, the first image of a time series will be displayed in the image window (Fig. 15.8).

If the image is a *z*-stack, the median plane will be displayed. If it is a multi-channel image, the median channel image will be displayed. If multiple separate image files have been loaded, a different file can be displayed by selecting the appropriate file from the output list box. A number of controls are available to alter how the image is displayed (Fig. 15.8).

For multi-channel images, different channels can be assigned to the **R**, **G** and **B** image planes using the drop-down menus to construct a RGB image, which will be displayed if the **RGB** checkbox is active.

The image size can be increased using the **Zoom** slider. If the image is larger than the display window, horizontal and vertical scroll bars will appear. The **fit** button maximises the size of the image to fit within the display window. The **100%** button displays the image at a 1:1 image pixel to display pixel size.

The **Channel** slider displays a single channel image (if the RGB checkbox is un-ticked).

The **Z** slider scrolls through *z*-sections if the image contains 3-D

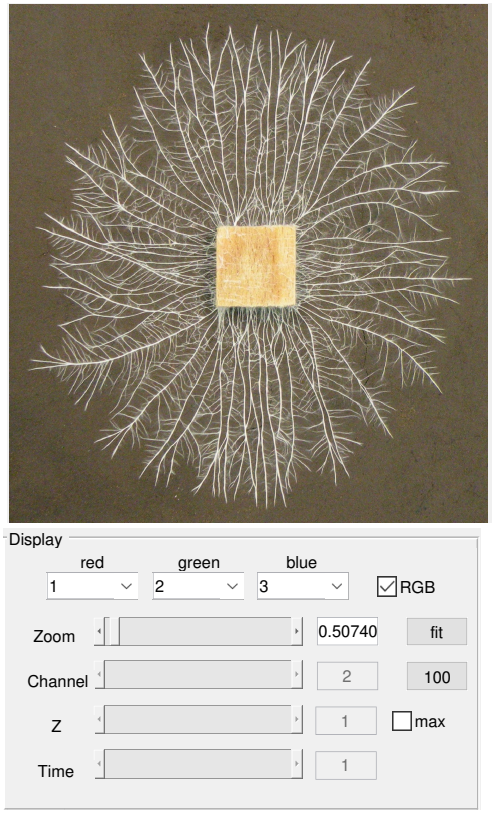


Figure 15.8: Image display panel and display controls: The image has been loaded in and an RGB version of the first three wavelength channels displayed. Controls are provided to allow the user to select which channel, section or time-point to display, with further options to combine channels into an RGB image, or display a maximum projection of the z-stack (if present)

(x,y,z) stacks. The **max** checkbox displays a maximum projection of the current z-stack.

The **Time** slider scrolls through each image in the time series.

15.6 *Image crop and sub-sampling options*

The image can be cropped by entering the **First** and **Last** pixel co-ordinates independently in the **X, Y, Z** or **T** text boxes (Fig. 15.9). Alternatively, a region-of-interest (ROI) can be selected in x and y using the **Set ROI** button. This prompts the user to draw a rectangular ROI on the image, which is then displayed in red. Completion of the ROI will update the values in the text boxes. Values can be reset using the **Reset** button if required.

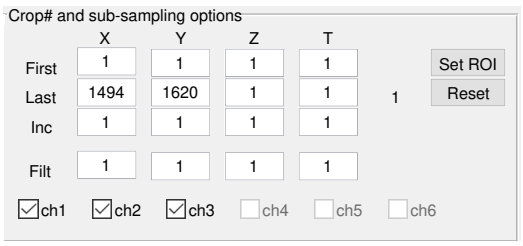


Figure 15.9: The crop panel: Provides controls to crop, sub-sample and filter images

The **Inc** text boxes allow (integer) sub-sampling independently in each image dimension.

The **Filt** text boxes allow spatial or temporal averaging over the designated number of pixels.

A series of checkboxes (**ch1**...**ch6**) are available to select which channels are to be included in the output image. Usually the bright-field channel is not included in this selection, as it is processed separately (see Section 15.10 - Bright-field image processing).

The file details in the **Output details** panel should update to reflect the cropping and sub-sampling chosen. If multiple images have been loaded, the crop and sub-sampling options apply to all the files.

15.7 *Alignment options between wavelength images*

Options are available to correct slight mis-registration in *x,y* between individual wavelength images using the **Channel alignment** controls (Fig. 15.10). A reference channel, typically containing the best image selected using the **Z** and **T** sliders, is chosen using the **Ref channel** drop-down list.

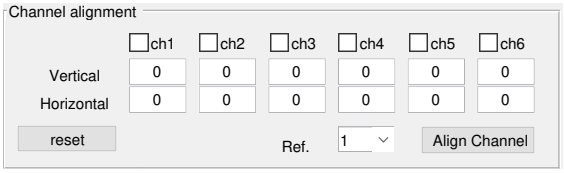


Figure 15.10: The channel alignment panel: Provides controls to allow sub-pixel registration between each channel and a selected reference channel

This acts as a template for cross correlation of any other channel, selected using the **ch1**...**ch6** checkboxes. The **Align Channel** button calculates the **Vertical** and **Horizontal** pixel offsets between the reference channel and the selected channels using cross-correlation across the whole image. These offsets are applied using bi-linear interpolation when the images are actually loaded.

The image display is modified during this process to display the before and after images in magenta and green.

15.8 *Alignment options over time*

Some level of correction for stage *x,y* drift or specimen movement can be achieved using the **Time series alignment** controls (Fig. 15.11). A reference channel (**Ref.**) is selected that has good contrast and features that are present in all images in the series, using the **Z** and **T** sliders, and, if necessary, a particular file if multiple files have been loaded simultaneously.

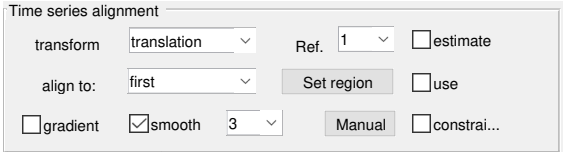


Figure 15.11: The Time series alignment panel: Provides controls to define a region for image alignment through the time-series, using a reference image

The **Set region** button prompts the user to select a ROI on the target image that will be used as a template to calculate the *x,y* pixel offsets, rotation and scaling for the corresponding image

in subsequent time-points using cross-correlation and bi-linear interpolation. The region used as a template must be within the red ROI outline, if this has been used to crop the image, and is highlighted in blue. The same offsets are applied to all channels and **z**-planes for each time point. The alignment only takes place when the images are actually loaded if the **use** checkbox is ticked.

A number of different transform types can be applied with increasing degrees of freedom, including *translation*, for (*x*,*y*) translation, *rigid*, for translation and rotation, *similarity* for translation, rotation, and scale and *affine*, for translation, rotation, scale, and shear. The **estimate** checkbox can improve registration for some time-series by providing an initial estimate of the transform using phase correlation.

In addition, registration can be achieved by matching Speeded-Up Robust Features (**SURF**) or maximally stable extremal regions (**MSER**) detected in each image. These are much faster than registration using cross-correlation and can accommodate translation, scale and rotation, but are also sensitive to relatively small changes in key features due to brownian motion of vesicles for example.

The **align to:** drop-down menu sets the reference image to the first in the series or the previous image. In addition, the **smooth** checkbox allows smoothing with a kernel set by the drop-down menu prior to registration. Note: this smoothing only applies to calculation of the alignment transfom, not the actual image data. The **gradient** checkbox can be used to align based on the local image gradients, which may be useful if there are large homogeneous features in the image.

15.9 Image projection options

It is possible to reduce the image dimensionality by projecting data along the *z*-axis using the drop-down menu in the **Projection options** panel (Fig: 15.12).

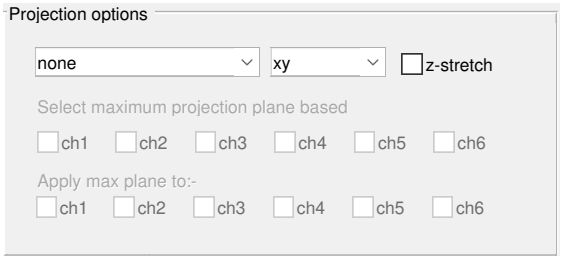


Figure 15.12: The projection panel: allows the user to reduce the image dimensionality by extracting a sub-set of the data in the *z*-dimension, or to re-orient the image stack to display *xz* or *yz* views. A number of different projection options can be selected and applied to different combinations of channels

- **Maximum:** displays the pixel with the maximum intensity in *z* for each channel (maximum intensity projection or MIP). This is a common approach to visualise data, but should not be used as a precursor to quantitative measurements, particularly when ratioing two channels, as it selects the 'noisiest' pixel at the extreme of the distribution along the *z*-axis, and pixels from

different positions in z for different channels will appear in the projected image, making a nonsense of the ratio image.

- **Minimum:** displays the pixel with the minimum intensity in z for each channel. This is rarely useful for fluorescence images, but can be helpful for bright-field processing. Nevertheless, it is recommended that bright-field processing is handles separately (see Section 15.10 - Bright-field image processing)
- **Average:** gives an average brightness projection in z, which provides good noise reduction and may be useful for simple objects that do not overlap in the z-direction.
- **Max plane:** this gives the user the option of selecting the z-position of the brightest pixel in one-or-more channels and then extracting the same (x,y,z) pixel (voxel) from the other channels. It is recommended that the imaged is smoothed in z (as well as x and y), before this operation to ensure that the brightest pixel is more likely to correspond to the centre of the object of interest. This approach is required if different channels are going to be ratioed later on to ensure that information from the same (averaged) voxels are compared.
- **Mx + mx plane:** This provides an option to calculate the maximum plane projection, based on specific selected channel(s) that extracts the appropriate (x,y,z) voxel from a second set of channels, selected by the second set of checkboxes. The remaining channels are processed using a simple maximum. This is useful for quantitative ratioing for the max plane images, which typically involve two wavelength channels and an autofluorescence channels for bleed through correction, and a morphological representation of the other channels from the (smoothed) maximum intensity projection.

15.10 *Bright-field image processing*

A bright-field image is often collected simultaneously with the fluorescence channels using a (non-confocal) transmission detector. Bright-field images can be processed separately to accentuate more useful information using the **Bright field output** controls (Fig: 15.13). The simplest form of processing is a **single plane**, selected by the position of the Z-slider from the bright field channel (**channel**). An amount of noise filtering can be applied using the process selected by the **smoothing** drop-down list (*mean*, *median* or *Wiener*) over a square x,y region specified by the **kernel** drop-down list.

If a projection option is chosen, the process required can be selected from the **process** drop-down menu. The algorithms available are designed to highlight pixels that might contain the most useful information within a local neighbourhood defined by the **kernel** drop-down menu.

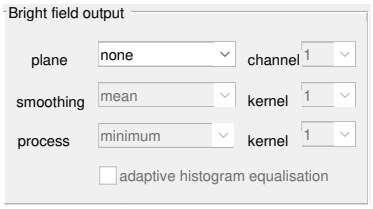


Figure 15.13: The bright field output panel: Provides controls to allow selection of single bright-field image planes or various algorithms to project the bright-field images, along with some contrast enhancement

Whether a projection of single plane option is chosen, the contrast of the resulting image can be improved using contrast-limited adaptive histogram equalisation (CLAHE) by checking the **adaptive histogram equalisation** box.

15.11 *Saving or loading the processing settings*

The **Save settings** button in the **Save options panel** saves the settings used for processing in a matlab file format, with the filename and a *settings* suffix (Fig: 15.14. The parameters can be re-loaded using the **Load settings** at a later stage. Note: when the settings are re-loaded, the user is prompted to re-set the alignment box for the time-series, and the cropping ROI is not available and must therefore be set-up again.

The **Save mat** button saves the processed fluorescence images and, if appropriate, bright-field images in a matlab file format. This can be re-loaded using the same interface at a later stage.

The **Save panels** button saves images of each of the panels in the image, using the format(s) selected from the *pdf*, *png* or *emf* checkboxes, and can be used to update this manual for any specific applications.

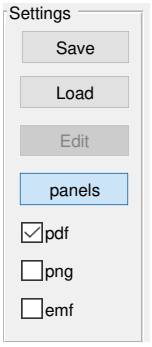


Figure 15.14: The Settings options panel: allows the user to save the loaded image and the processing settings

15.12 *Loading the selected files*

Once all the processing steps have been completed, the selected files can be loaded using the **Load** button (Fig: 15.15). If this is successful, the **View** button will be enabled and, if a separate bright field image has been processed, the **View bf** button. Clicking the **View** button will open a separate window with a video **Viewer** (see Chapter 17 - Viewer Program). There is an additional option to **Convert** the image format to *8-bit*, *16-bit*, *single* or *double* precision, using the adjacent drop-down menu. Images in 12-bit can be re-normalised to 16-bit using the **12 to 16 bit** checkbox. Images can also be inverted at this point using the **invert** checkbox.

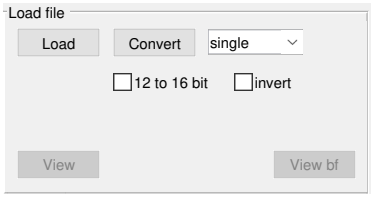


Figure 15.15: The Load file panel: The Load button imports the selected image(s) and applies all the alignment, sampling, smoothing and projection parameters chosen

15.13 *Saving the processed files*

If the image files have been loaded satisfactorily, clicking the **OK** button will return to the main program. The **Cancel** button will return to the main program, but without exporting the processed image file. The **matfile** button saves a copy of the processed image and information on the processing steps to a MATLAB file. This can be loaded later without having to re-process the original image.

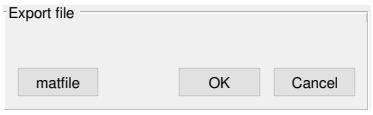


Figure 15.16: The export file panel to exit the program or save the processed image

16
Binary editing

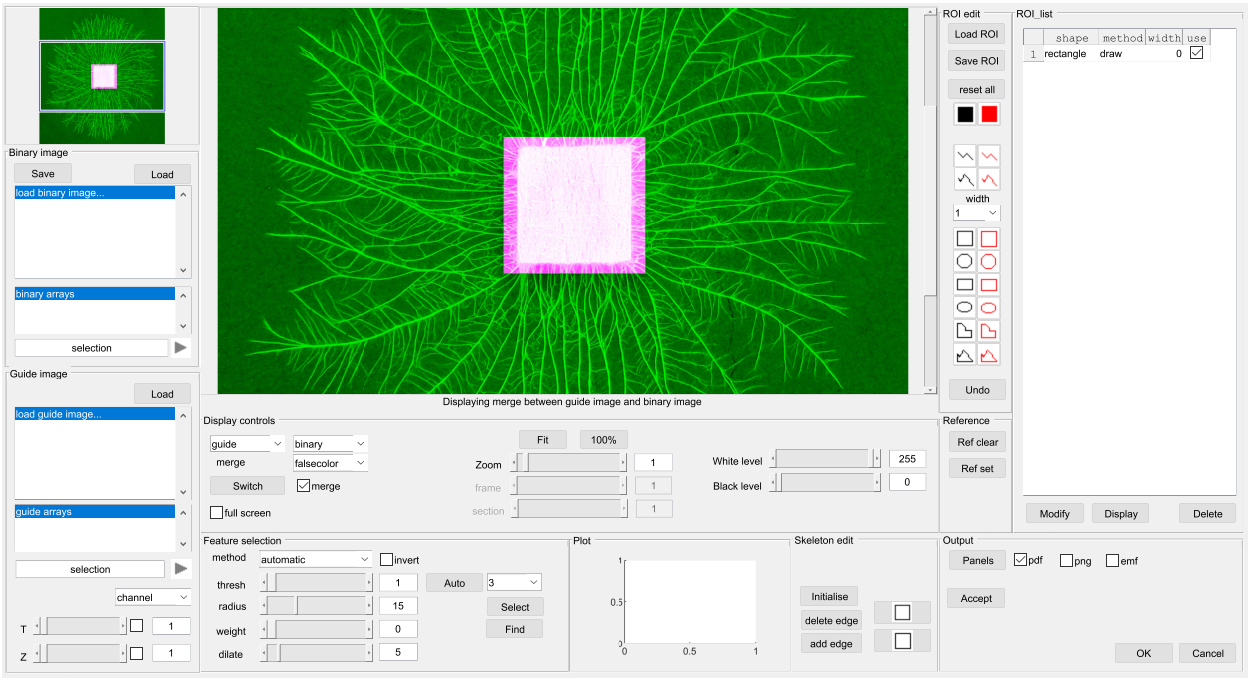


Figure 16.1: The GUI interface to edit binary images

16.1 *Loading the images*

Binary images are used to mask the boundary of the arena, delineate the food resources or other features, and represent the single-pixel wide skeleton. Whilst each of these can be generated automatically, it is often the case that they need to be edited manually. The *Binary editing* panel has tools to achieve this.

If the *Binary editing* window has been called from the main network interface, by clicking the **edit** mask button for example, the template image and the current mask (if any) will automatically be displayed as a green-magenta merge when the window opens (Fig. 16.2).

The order of the merged image can be changed using the **Switch** button, the type of merge using the drop-down menu, or removed

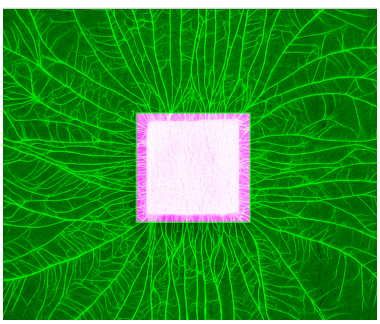


Figure 16.2: Main display window showing a merge between the guide image (green) and binary mask (magenta).

by un-checking the **merge** check-box (Fig. 16.3). The other display controls can be used to change the zoom, adjust the contrast, or, in multi-dimensional images, scroll through the sections and frames.

Figure 16.3: Main display window and controls to select the images to display, the type of merge, zoom and contrast.

In some instances, it may be desirable to load a previously constructed binary image, or a different guide image stored on disk. Available images (stored as MatLab arrays) can be loaded into the *Binary image* list-box (Fig. 16.4) using the **Load** button. The appropriate binary image from the arrays present in the matfile is selected by a single click, and displayed in the **selection** box. If the binary image is part of a structure within the matfile, it will be displayed in the **binary arrays** and has to be selected from this list-box. The adjacent right arrow is used to import it into the interface.

In a similar manner, a separate guide image can be loaded from a MatLab file on disk using the controls in the *Guide image* panel. If the guide image is part of a MatLab structure, the names of the available array are shown in the **guide arrays** list-box, and has to be selected from here.

The **channel** drop-down menu is used to choose the channel to import, whilst the **T** and **Z** sliders are used to select the frame and section, respectively. The check-boxes adjacent to the sliders give a maximum projection along that dimension. All changes are displayed in the main figure window. Once the most suitable combination has been chosen, the image is imported using the right arrow.

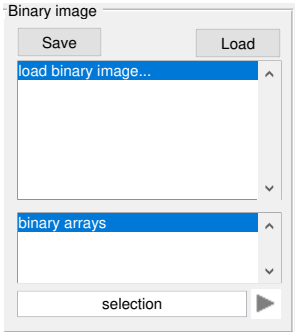


Figure 16.4: Controls to select and load a binary image (as a Matlab array).

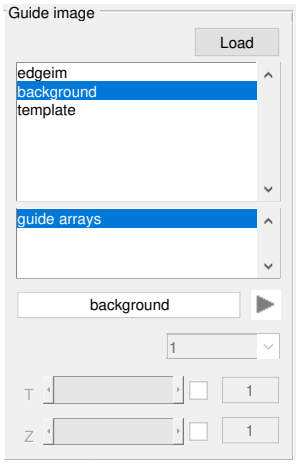


Figure 16.5: Controls to select and load a guide image (as a Matlab array).

16.2 *Automatic feature selection*

The *Feature selection* panel is used to try to automatically segment the features of interest (Fig. 16.6).

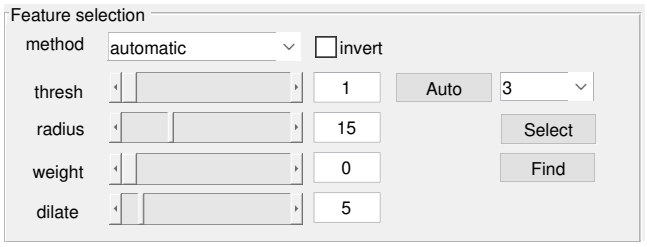


Figure 16.6: Controls to segment features in the image

The **method** drop-down menu provides access to a number of different thresholding strategies including:

- *manual:* The user chooses a threshold using the **threshold** slider.

- *automatic:* Automatically chooses a threshold when the **Auto** button is pressed using Otsu's criterion[1] for multiple thresholds, of which the lowest one is selected. The number of partitions is selected by the adjacent drop-down menu. A histogram of the image intensities is shown in the *plot* panel, with the position of the threshold(s) shown in green (Fig. 16.7).

[1] N. Otsu. A threshold selection method from gray-level histograms. *IEEE Trans. Systems, Man, Cyber.*, 9:62–66, 1979

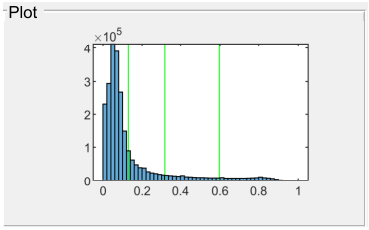


Figure 16.7: Histogram of piel intensities overlaid with the segmentation threshold

- *opening:* Applies a grey-scale opening to the image, with the radius determined by the **radius** slider. The image is segmented automatically using the lowest threshold from multi-level threshold menu;
- *tophat:* Applies a top-hat filter with the radius determined by the **radius** slider. The image is segmented automatically using the lowest threshold from multi-level threshold menu;
- *active contour:* Applies a grey-scale opening to the image, with the radius determined by the **radius** slider. The opened image is then matched to the underlying objects using an active contour algorithm.
- *local mean:* The local mean is calculated using a circular filter with the radius set by the **radius** slider and used as a local threshold. An additional constant can be subtracted from the threshold using the **weight** slider.
- *local median:* The local median is calculated using a circular filter with the radius set by the **radius** slider and used as a local threshold. An additional constant can be subtracted from the threshold using the **weight** slider.
- *midgrey:* The mid-grey threshold is calculated as the average of the local maximum and minimum, calculated using morphological closing and opening operations, respectively, with the radius set by the **radius** slider. An additional offset k can be set using the **weight** slider.

$$T(x,y) = \frac{max + min}{2} - k \tag{16.1}$$

- *Bernsen*[2]*:* The method calculates the local contrast and mid-grey values in a similar manner to the mid-grey algorithm from morphological operations. If the local contrast is above the contrast value k set by the **weight** slider, the threshold is set at the local mid-grey value. If the local contrast is below the contrast threshold, the pixel is classified as part of the object if the mid-grey value is above 0.5, or background if below.

[2] J. Bernsen. Dynamic thresholding of grey-level images. In *International conference on pattern recognition*, volume 2, pages 1251–1255, 1986

- *Niblack*[3]*:* The local threshold ($T(x,y)$) is calculated from the local mean ($m(x,y)$) and the local standard deviation ($s(x,y)$), with the radius set by the **radius** slider, weighted by a factor (k) set using the **weight** slider.

[3] W. Niblack. *An Introduction to Image Processing*. Prentice-Hall, 1986

$$T(x,y) = m(x,y) + k * s(x,y) \tag{16.2}$$

where k is typically -0.2.

- *Sauvola*[4]*:* The local threshold ($T(x,y)$) is calculated in a similar manner to the Niblack algorithm from the local mean ($m(x,y)$) and local standard deviation ($s(x,y)$), with the radius set by the **radius** slider, but the weighting is calculated as:

$$T(x,y) = m(x,y)\left[1 + k\left(\frac{s(x,y)}{R} - 1\right)\right] \tag{16.3}$$

where $R = max(s)$ and k takes small positive values, typically in the range 0.2- 0.5, set by the **weight** slider.

[4] J. Sauvola and M. Pietikäinen. Adaptive document image binarization. *Pattern recognition*, 33:225–236, 2000

16.3 Manual Editing

If automatic segmentation is not possible, or the results need further modification, The *ROI edit* controls can be used (Fig. 16.8). Each icon depicts the type of ROI that can be drawn in black, or erased in red. The sequence of ROIs can be saved (**Save ROI)**) and reloaded (Load ROI). In addition, any binary image created with the *Binary edit* called from another program is imported with the existing list of ROIs.

The **reset all** button clears all the ROIs and the segmented image. The solid black icon set the ROI to fill the image, whilst the solid red icon erases the full screen. These should only be used as the first ROI in the series. Each ROI is numbered and listed in the adjacent *ROI list* panel (Fig. **??**), which indicates the *shape* of the ROI, the *method* (*draw* or *erase*), the line width (only relevant for polylines and freehand lines), and whether the ROI should be used (**use** check-box).

Examples of the different ROI shapes are shown in Fig. 16.9.

Figure 16.8: Manual editing controls

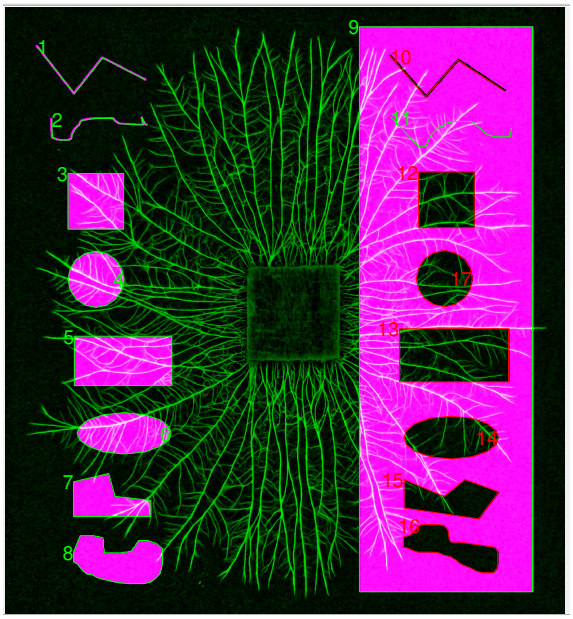


Figure 16.9: Drawing shapes (left) and erase shapes (right) available from the ROI tools menu

For all the geometric shapes, a vertex is added with each left mouse click, whilst a double-click closes the shape. each vertex

can be re-positioned by hovering over the vertex until a black circle appears and then dragging the circle to the new position. Additional vertices can be added along the lines using the **A** button on the keyboard. Once the shape is complete, it is plotted in green for an included region and red for an erased region, along with a unique label matching the row in the *ROI list*. In addition, the corresponding binary image is shown overlaid in magenta. For the polyline and freehand line buttons, the line width can be chosen in pixels from the drop-down menu, or adjusted in the *ROI list* box.

ROIs can be altered or deleted using the **Modify** button or **Delete** button, respectively, in the *ROI list* panel (Fig. 16.10). The **Display** button can be used to show the ROI overlay and binary image at any time.

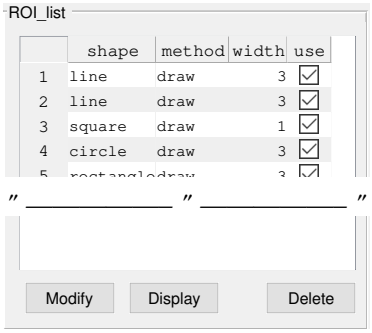


Figure 16.10: Modify, Display or Delete specific ROIs

16.4 *Output*

The **OK** button returns the adjusted binary iage to the calling program, along with the list of ROIs and their vertex co-ordinates. The **Cancel** button exits without saving any information. The **Panel** button saves a copy of each panel in *.png and *.pdf format for inclusion in the manual.

In addition, the binary image can be saved independently as a matlab file using the **Save** button in the *Load binary* panel (Fig. ref).

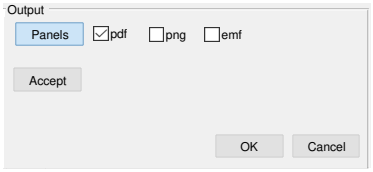


Figure 16.11: Output controls

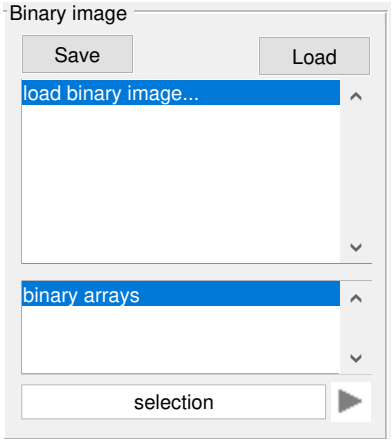
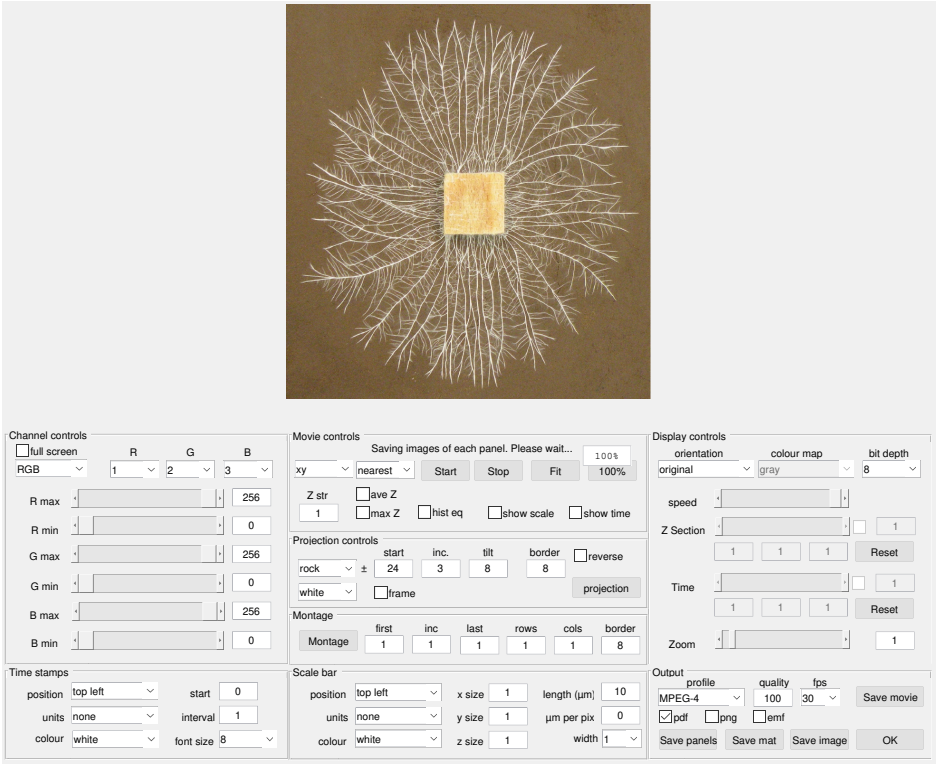


Figure 16.12: Option to save the adjusted binary image as a matfile

17
Viewer Program



17.1 Introduction

The video **Viewer** allows the output of any processing steps to be animated on screen or saved in standard movie formats format. Images can contain multiple channels, although a maximum of three can be shown in RGB format. The first image in the file is displayed when the viewer opens. If the image is too large for the display window, the central portion will be displayed with scroll bars to move around the image.

17.2 *Movie playback controls*

The **Movie controls** (Fig. 17.1) can be used to **Start** or **Stop** playback. The normal image orientation is viewed in the *xy* plane. If the data has multiple *z*-sections, the orientation can be changed to view animations in *xz* or *yz* orientation using the drop-down menu. The **Z str** textbox allows control of the amount of interpolation in the *z*-axis needed to correct for the asymmetry in *x,y* and *x* pixel spacing. The type of interpolation can be set using the drop-down menu from *nearest*, *linear* or *bicubic*.

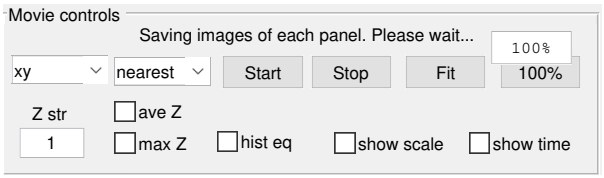


Figure 17.1: The movie controls panel: provides options to start/stop the animation or display max or average z-projections

The average (**ave Z**) or maximum (**max Z**) checkboxes generate the corresponding projection for the current time-point if the data has multiple *z*-sections.

The **hist eq** checkbox uses contrast-limited histogram equalisation (CLAHE) to improve the overall contrast in the image. If the data has more than one *z*-section, the maximum projection is calculated first before the CLAHE enhancement. The **show scale** and **show time** checkboxes toggle display of the time stamps set in the **Time stamps** panel, and the scale bar set in the **Scale bar** panel.

17.3 *Display controls*

The **Display controls** panel (Fig. 17.2) can be used to display images in their **original** orientation, or as a rotated image (*90 left*, *90 right* or *180*) using the drop-down menu.

If the image is initially a single intensity channel the **colour map** drop-down menu can be used to apply a colour look-up table (LUT) to the image.

The **bit depth** controls the number of bits to scale each channel in the display, and automatically adjusts the maximum scaling of the channel intensity controls.

Note, many microscopy images contain 12 bits of information (0-4095), but are stored in 16-bit format. This means the bit-depth may be automatically set to 16 (0-65535) when the image is loaded. As a result the screen will appear black until the bit-depth is set to 12. In other situations, the software cannot pick up the correct setting for the bit-depth and defaults to a value of 1. Images may appear completely saturated until the bit-depth is set to the appropriate value, usually 8.

The **speed** slider adjusts the speed of playback.

The **Z Section** or **Time** sliders select a specific *z*-section and time point and also have **check-boxes** to control whether images are

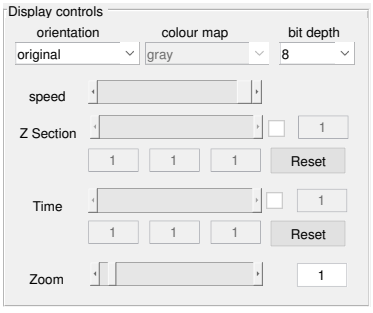


Figure 17.2: The display controls panel: set the image orientation and control the dimensions for animation

animated in this dimension during playback.

The **increment** slider controls the gap between images during playback.

The **Zoom** slider controls the scale of the image displayed. If the image is larger than the display window, horizontal and vertical scroll bars appear automatically.

17.4 *Image brightness controls*

Images are usually imported in RGB format where each channel corresponds to a different wavelength image. However, pseudo-colour coded ratio images are constructed in HSV colour space where the parameter of interest is coded as Hue, and the intensity and/or saturation are used to represent the strength of the original signals. Switching between RGB and HSV display is achieved with the **RGB** drop-down menu on the **Channel controls** panel (Fig. 17.3).

Figure 17.3: The channel controls: Select the channels to display and the minimum and maximum intensity for each channel

In RGB mode, the **R, G** and **B** drop-down menus can be used to select which channel should be displayed in which colour plane, and the sliders used to set the **minimum** and **maximum** values for each colour independently.

In **HSV** mode, the slider and drop-down labels change to represent the **Hue**, **Saturation** and intensity **Value**. This allows independent scaling of these parameters (but only yields a useful image if the data is in HSV format).

17.5 *Projection controls*

If the image contains an RGB z-stack or a series of *z*-stacks collected over time, the **Projection controls** panel (Fig. 17.4) can be used to construct a rocking or tilted animation of the maximum projection of each channel for each *z*-stack at different angles.

Figure 17.4: The projection controls: Allows calculation of tilting or rocking projections of z-stacks

The **start** text box sets the initial angle for the rotation, whilst the **inc.** text box controls the angle between projections. The **tilt** or **rock** is calculated symmetrically about zero between these limits.

The **Z str.** Sets a *z*-stretch to correct for the asymmetric sampling in *x,y* and *z*. This is calculated from the nominal z-pixel spacing divided by the *x,y* pixel spacing. However, additional correction may be required depending on the lens and immersion media using to collect the original data. Ideally, values are based on a calibrated sample, such as a 15 μm fluorescent sphere, to ensure correct geometric scaling.

If the original data is an HSV ratio image, the **HSV** setting needs to be selected in the channel controls panel, and the projection will

be calculated initially as a maximum projection of the intensity value (**V**). The *x,y,z*-pixel co-ordinate of this maximum brightness pixel is then used to extract the corresponding values from the Hue and Saturation channels. This approach allows projection of multi-dimensional pseudo-colour coded images.

17.6 Annotation controls

The **Time stamps** control panel (Fig. 17.5) allows the user to add a time-stamp to the display. The **position** controls the location of the time label (top left, top right, bottom left, bottom right), whilst the **start** and **interval** text boxes and **units** drop-down menu control the information to be displayed in a specific **font size** and **colour**.

Figure 17.5: The annotation panels: allows addition of a time-stamp to each image

The **Scale bar** control panel Fig. 17.6 adds a scale bar to the image. The **position** controls the location of the scale bar, whose **length** (in microns) and **width** are set by the appropriate text boxes. The calibration between pixels and microns is set by the μ**m per pix** textbox.

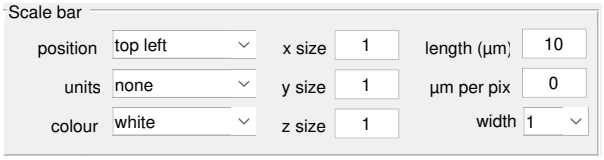


Figure 17.6: The annotation panels: allows addition of a scale bar to each image

17.7 Montage controls

A **Montage** of selected images (Fig. 17.7), defined by the **first**, **inc.** and **last** can be constructed in a matrix format defined by the **rows** and **cols** controls, with a gap between the images defined by the **border**. The montage appears in a new window and can be saved in a variety of formats.

Figure 17.7: The montage panel: Sets up a montage of images that can be subsequently saved in a grid format

17.8 Output controls

Processed images can be saved to a variety of video formats format using the **Movie** button in the **Output** panel (Fig. 17.8). This provides choice over the compression**profile** (default is MPEG-4), and,

if appropriate the **quality** and frames-per-second (**fps**) required. The image series can also be saved as a matlab array using the **Save mat** button, or as a single image using the **Save image** button.

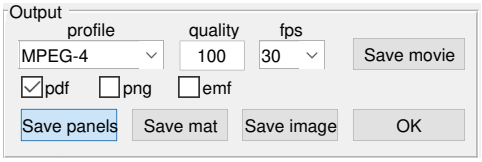


Figure 17.8: The output panel: Saves the final movie in *.MPEG-4 format

The **Save panels** button saves images of all the control panels to help in construction of an application specific manual.